Jian Pei Vincent S. Tseng
Longbing Cao Hiroshi Motoda
Guandong Xu (Eds.)

LNAI 7818

Advances in Knowledge Discovery and Data Mining

17th Pacific-Asia Conference, PAKDD 2013
Gold Coast, Australia, April 2013
Proceedings, Part I

1 Part I

Springer

Lecture Notes in Artificial Intelligence 7818

Subseries of Lecture Notes in Computer Science

Jian Pei Vincent S. Tseng Longbing Cao
Hiroshi Motoda Guandong Xu (Eds.)

Advances in Knowledge Discovery and Data Mining

17th Pacific-Asia Conference, PAKDD 2013
Gold Coast, Australia, April 14-17, 2013
Proceedings, Part I

Springer

Volume Editors

Jian Pei
Simon Fraser University, Burnaby, BC, Canada
E-mail: jpei@cs.sfu.ca

Vincent S. Tseng
National Cheng Kung University, Tainan, Taiwan
E-mail: tsengsm@mail.ncku.edu.tw

Longbing Cao
Guandong Xu
University of Technology Sydney, NSW, Australia
E-mail: {longbing.cao, guandong.xu}@uts.edu.au

Hiroshi Motoda
Osaka University, Japan
E-mail: motoda@ar.sanken.osaka-u.ac.jp

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-37452-4 e-ISBN 978-3-642-37453-1
DOI 10.1007/978-3-642-37453-1
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013934551

CR Subject Classification (1998): H.2.8, I.2, H.3, H.4, J.1, I.5

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

As the Program Committee Co-chairs, we welcome you to the proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2013), held at Gold Coast, Australia, during April 14-17, 2013.

The PAKDD conference series, since its inception in 1997, has been a leading international conference in the areas of data mining and knowledge discovery (KDD). It provides an inviting and inspiring forum for researchers and practitioners, from both academia and industry, to share new ideas, original research results, and practical experience. The 17th edition continued the great tradition, and had three world-class keynote speeches, a wonderful technical program, a handful of high-quality tutorials and workshops, as well as an interesting invited talk from industry.

The PAKDD 2013 conference received 363 submissions to the technical program, involving more than 1,000 authors in total. In the rigorous double-blind review process, each submission was reviewed by one senior Program Committee member and at least three Program Committee members. Many submissions were extensively and thoroughly discussed by the reviewers. Based on the detailed and critical discussion and reviews, the senior Program Committee members made recommendations. Overall, 98 papers from 341 authors were accepted in the technical program, yielding a 27% acceptance rate. Of these, 39 (10.7%) had long presentations (30 minutes) and 59 (16.3%) had short presentations (15 minutes). The technical program consisted of 22 sessions, covering the general fields of data mining and KDD extensively, including pattern mining, classification, graph mining, applications, machine learning, feature selection and dimensionality reduction, multiple information sources mining, social networks, clustering, text mining, text classification, imbalanced data, privacy-preserving data mining, recommendation, multimedia data mining, stream data mining, data preprocessing and representation.

We were lucky to have three world-class keynote speakers this year. Usama Fayyad, a renowned pioneer in big data entrepreneurship, addressed us on the big picture of big data. Huan Liu, a world-wide leader in social media mining, discussed this exciting new frontier of data mining. Qiang Yang, a famous expert on artificial intelligence and machine learning, talked on how machine learning can address the big data challenge. We were also pleased to have Alexandros Batsakis as an invited speaker from industry. He shared with us the latest developments on big data analytics infrastructure and enterprise applications.

The conference also included six workshops, covering a few exciting and fast-growing hot topics. We also had five very timely and educational tutorials, covering the hot topics of social networks, transfer learning, stream mining, outlier detection, and feature discovery.

In addition to the intellectually inspiring keynote speeches, technical program, workshops and tutorials, we had several dynamic social events to facilitate communication and informal interaction, including a welcome reception, a banquet, and an excursion.

Putting together a conference like PAKDD is never easy. It becomes possible only with tremendous contributions from the organizing team and many volunteers. We thank Jiuyong Li, Kay Chen Tan, and Bo Liu for organizing the workshop program. We also thank Tu Bao Ho and Mengjie Zhang for organizing the tutorial program. We are grateful to Chengqi Zhang for his leadership in the award selection. We owe a big thank-you to the 39 senior Program Committee members, 151 Program Committee members, and the external reviewers for their great contributions and collaboration. We thank Guandong Xu for assembling the proceedings. We also thank the General Chairs, Hiroshi Motoda and Longbing Cao, and the local organization team for their great support. Without the dedicated hard work of so many people, PAKDD 2013 would simply have been mission impossible.

February 2013

Jian Pei
Vincent S. Tseng

Organization

Honorary Co-chairs

Jiawei Han	University of Illinois at Urbana-Champaign, USA
Ramamohanarao Kotagiri	University of Melbourne, Australia
Graham Williams	Australia Taxation Office, Australia

Conference Co-chairs

Hiroshi Motoda	Osaka University, Japan
Longbing Cao	University of Technology, Sydney, Australia

Program Committee Co-chairs

Jian Pei	Simon Fraser University, Canada
Vincent S. Tseng	National Cheng Kung University, Taiwan

Local Arrangements Co-chairs

Vladimir Estivill-Castro	Griffith University (Gold Coast), Australia
Xue Li	University of Queensland, Australia
Richi Nayak	Queensland University of Technology, Australia
Xinhua Zhu	University of Technology, Sydney, Australia

Workshop Co-chairs

Jiuyong Li	University of South Australia, Australia
Kay Chen Tan	National University of Singapore, Singapore
Bo Liu	Guangdong University of Technology, China

Tutorial Co-chairs

Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Mengjie Zhang	Victoria University of Wellington, New Zealand

Awards Chair

Chengqi Zhang	University of Technology, Sydney, Australia

Sponsorship Co-chairs

Yue Xu	Queensland University of Technology, Australia
Eugene Dubossarsky	Analyst First
Suresh Sood	University of Technology, Sydney, Australia

Publicity Co-chairs

P.Krishna Reddy	The International Institute of Information Technology, Hyderabad, India
Yifeng Zeng	Aalborg University, Denmark
Xin Wang	University of Calgary, Canada
Zhihong Deng	Peking University, China

Proceedings Chair

Guandong Xu	Unverisity of Technology, Sydney, Australia

Registration Co-chairs

Qiang Wu	University of Technology, Sydney, Australia
Can Wang	University of Technology, Sydney, Australia

Big Data School Co-chairs

Jinyan Li	University of Technology, Sydney, Australia
Xinhua Zhu	University of Technology, Sydney, Australia

Steering Committee

Graham Williams	Australian Taxation Office, Australia
Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Hiroshi Motoda	Osaka University, Japan (Since 1997)
Rao Kotagiri	University of Melbourne, Australia (Since 1997)
Ning Zhong	Maebashi Institute of Technology, Japan (Since 1999)
Masaru Kitsuregawa	Tokyo University, Japan (Since 2000)
David Cheung	University of Hong Kong, China (Since 2001)
Graham Williams (Treasurer)	Australian National University, Australia (Since 2001)
Ming-Syan Chen	National Taiwan University, Taiwan, ROC (Since 2002)

Kyu-Young Whang	Korea Advanced Institute of Science & Technology, Korea (Since 2003)
Huan Liu	Arizona State University, USA (Since 1998)
Chengqi Zhang	University of Technology Sydney, Australia (Since 2004)
Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan(Since 2005)
Ee-Peng Lim	Singapore Management University, Singapore (Since 2006)
Jaideep Srivastava	University of Minnesota, USA (Since 2006)
Zhi-Hua Zhou	Nanjing University, China (Since 2007)
Takashi Washio	Institute of Scientific and Industrial Research, Osaka University, Japan (Since 2008)
Thanaruk Theeramunkong	Thammasat University, Thailand (Since 2009)
P. Krishna Reddy	International Institute of Information Technology, Hyderabad (IIIT-H), India (Since 2010)
Joshua Z. Huang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China (Since 2011)

Senior Program Committee Members

James Bailey	University of Melbourne, Australia
Michael Berthold	University of Konstanz, Germany
Nitesh Chawla	University of Notre Dame, USA
Sanjay Chawla	University of Sydney, Australia
Ming-Syan Chen	National Taiwan University, Taiwan
Arbee L. P. Chen	National Chengchi University, Taiwan
Peter Christen	The Australian National University, Australia
Diane Cook	Washington State University, USA
Ian Davidson	UC Davis, USA
Bart Goethals	University of Antwerp, Belgium
Tu-Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Xiaohua Hu	Drexel University, USA
Ming Hua	Facebook, USA
Joshua Huang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Hisashi Kashima	University of Tokyo, Japan
Jiuyong Li	University of South Australia, Australia
Ee-Peng Lim	Singapore Management University, Singapore
Chih-Jen Lin	National Taiwan University, Taiwan
Huan Liu	Arizona State University, USA
Nikos Mamoulis	University of Hong Kong, Hong Kong

Wee Keong Ng	Nanyang Technological University, Singapore
Wen-Chih Peng	National Chiao Tung University, Taiwan
P. Reddy	International Institute of Information Technology, Hyderabad (IIIT-H), India
Kyuseok Shim	Seoul National University, Korea
Masashi Sugiyama	Tokyo Institute of Technology, Japan
Pang-Ning Tan	Michigan State University, USA
Hanghang Tong	IBM T. J. Watson Research, USA
Shusaku Tsumoto	Shimane University, Japan
Wei Wang	University of North Carolina at Chapel Hill, USA
Haixun Wang	Microsoft Research Asia, China
Jianyong Wang	Tsinghua University, China
Ji-Rong Wen	Microsoft Research Asia, China
Xing Xie	Microsoft Research Asia, China
Hui Xiong	Rutgers University, USA
Xifeng Yan	UC Santa Barbara, USA
Jieping Ye	Arizona State University
Jeffrey Yu	The Chinese University of Hong Kong, Hong Kong
Chengqi Zhang	University of Technology, Australia
Zhi-Hua Zhou	Nanjing University, China

Program Committee Members

Hideo Bannai	Kyshu University, Japan
Marut Buranarach	National Electronics and Computer Technology Center, Thailand
Rui Camacho	Universidade do Porto, Portugal
Tru Cao	Ho Chi Minh City University of Technology, Vietnam
Keith Chan	The Hong Kong Polytechnic University, Hong Kong
Muhammad Cheema	University of New South Wales, Australia
Enhong Chen	University of Science and Technology of China, China
Jake Chen	Indiana University-Purdue University Indianapolis, USA
Jian Chen	Southern China University of Technology, China
Ling Chen	University of Technology Sydney, Australia
Shu-Ching Chen	Florida International University, USA
Songcan Chen	Nanjing University of Aeronautics and Astronautics, China
Yi-Ping Phoebe Chen	La Trobe University, Australia
Zheng Chen	Microsoft Research Asia, China

Hong Cheng	The Chinese University of Hong Kong, Hong Kong
Yiu-ming Cheung	Hong Kong Baptist University, Hong Kong
Bruno Cremilleux	Université de Caen, France
Bin Cui	Peking University, China
Alfredo Cuzzocrea	ICAR-CNR and University of Calabria, Italy
Dao-Qing Dai	Sun Yat-Sen University, China
Bolin Ding	Microsoft Research, USA
Wei Ding	University of Massachusetts Boston, USA
Guozhu Dong	Wright State University, USA
Wei Fan	IBM T. J. Watson Research Center, USA
Eibe Frank	University of Waikato, New Zealand
Joao Gama	Universidade do Porto, Portugal
Dragan Gamberger	Rudjer Boskovic Institute, Croatia
Cong Gao	Nanyang Technological University, Singapore
Jun Gao	Peking University, China
Junbin Gao	Charles Sturt University, Australia
Yong Guan	Iowa State University, USA
Ravi Gupta	Anna University, India
Sung-Ho Ha	Kyungpook National University, Korea
Yi Han	National Defence Technology University, China
Choochart Haruechaiy	National Electronics and Computer Technology Center, Thailand
Jingrui He	IBM Research, USA
Qi He	IBM Research, USA
Chin-Kuan Ho	Multimedia University, Malaysia
Kuo-Wei Hsu	National Chengchi University, Taiwan
Jun Huan	University of Kansas, USA
Jin Huang	Southern China Normal University, China
Daisuke Ikeda	Kyshu University, Japan
Akihiro Inokuchi	Osaka University, Japan
Sanjay Jain	National University of Singapore, Singapore
Shuiwang Ji	Old Dominion University, USA
Ruoming Jin	Kent State University, USA
Toshihiro Kamishima	National Institute of Advanced Industrial Science and Technology, Japan
Hung-Yu Kao	National Cheng Kung University, Taiwan
Panagiotis Karras	Rutgers University, USA
George Karypis	University of Minnesota, USA
Hiroyuki Kawano	Nanzan University, Japan
Latifur Khan	University of Texas at Dallas, USA
Hiroyuki Kitagawa	University of Tsukuba, Japan
Irena Koprinska	University of Sydney, Australia
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Wai Lam	The Chinese University of Hong Kong, Hong Kong

Yue-Shi Lee	Ming Chuan University, Taiwan
Carson K. Leung	University of Manitoba, Canada
Chengkai Li	The University of Texas at Arlington, USA
Chun-hung Li	Hong Kong Baptist University, Hong Kong
Feifei Li	Florida State University, USA
Jinyan Li	University of Technology Sydney, Australia
Ming Li	Nanjing University, China
Xiaoli Li	Institute for Infocomm Research, Singapore
Xue Li	The University of Queensland, Australia
Xuelong Li	University of London, UK
Zhenhui Li	Pennsylvania State University, USA
Kawuu W. Lin	National Kaohsiung University of Applied Sciences, Taiwan
Shou-de Lin	National Taiwan University, Taiwan
Fei Liu	Bosch Research, USA
Qingshan Liu	NLPR Institute of Automation Chinese Academy of Science, China
Shixia Liu	Microsoft Research Asia, China
Tie-Yan Liu	Microsoft Research Asia, China
David Lo	Singapore Management University, Singapore
Woong-Kee Loh	Sungkyul University, South Korea
Chang-Tien Lu	Virginia Polytechnic Institute and State University, USA
Hua Lu	Aalborg University, Denmark
Shuai Ma	Beihang University, China
Marco Maggini	Università degli Studi di Siena, Italy
Tao Mei	Microsoft Research Asia, China
Wagner Meira	Universidade Federal de Minas Gerais, Brazil
Toshiro Minami	Kyushu University Library, Japan
Pabitra Mitra	Indian Institute of Technology Kharagpur, India
Yang-Sae Moon	Kangwon National University, Korea
Yasuhiko Morimoto	Hiroshima University, Japan
Tsuyoshi Murata	Tokyo Institute of Technology, Japan
Richi Nayak	Queensland University of Technologies, Australia
See-Kiong Ng	Institute for Infocomm Research, A*STAR, Singapore
Wilfred Ng	Hong Kong University of Science and Technology, Hong Kong
Tadashi Nomoto	National Institute of Japanese Literature, Japan
Masayuki Numao	Osaka University, Japan
Manabu Okumura	Japan Advanced Institute of Science and Technology, Japan

Salvatore Orlando	University of Venice, Italy
Yonghong Peng	University of Bradford, UK
Jean-Marc Petit	Universite de Lyon, France
Vincenzo Piuri	Università degli Studi di Milano, Italy
Weining Qian	East China Normal University, China
Chotirat Ratanamatan	Chulalongkorn University, Thailand
Chandan Reddy	Wayne State University, USA
Patricia Riddle	University of Auckland, New Zealand
C. Sekhar	Indian Insitute of Technology, India
Hong Shen	Adelaide University, Australia
Jialie Shen	Singapore Management University, Singapore
Yi-Dong Shen	Chinese Academy of Sciences, China
Andrzej Skowron	University of Warsaw, Poland
Mingli Song	Zhejiang University, China
Yizhou Sun	Northeastern University, USA
Thepchai Supnithi	National Electronics and Computer Technology Center, Thailand
David Taniar	Monash University, Australia
Tamir Tassa	The Open University, Israel
Ivor Tsang	Nanyang Technological University, Singapore
Jeffrey Ullman	Stanford University, USA
Marian Vajtersic	University of Salzburg, Austria
Hui Wang	University of Ulster, UK
Jason Wang	New Jersey Science and Technology University, USA
Lipo Wang	Nanyang Technological University, Singapore
Xin Wang	University of Calgary, Canada
Raymond Wong	Hong Kong University of Science and Technology, Hong Kong
Jian Wu	Zhejiang University, China
Junjie Wu	Beihang University, China
Xindong Wu	University of Vermont, USA
Xintao Wu	University of North Carolina at Charlotte, USA
Xiaofeng Meng	Renmin University of China, China
Seiji Yamada	National Institute of Informatics, Japan
Min Yao	Zhejiang University, China
Mi-Yen Yeh	Academia Sinica, Taiwan
Dit-Yan Yeung	Hong Kong University of Science and Technology, China
Jian Yin	Sun Yat-Sen University, China
Jin Soung Yoo	IUPU, USA
Kennichi Yoshida	University of Tsukuba, Japan
Tetsuya Yoshida	Hokkaido University, Japan
Hwanjo Yu	Pohang University of Science and Technology, South Korea

Yifeng Zeng	Teesside University, UK
Bo Zhang	Tsinghua University, China
Daoqiang Zhang	Nanjing University of Aeronautics and Astronautics, China
Du Zhang	California State University, USA
Harry Zhang	University of New Brunswick, Canada
Junping Zhang	Fudan University, China
Mengjie Zhang	Victoria University of Wellington, New Zealand
Shichao Zhang	University of Technology, Australia
Wenjie Zhang	University of New South Wales, Australia
Ying Zhang	University of New South Wales, Australia
Zhongfei Zhang	Binghamton University, USA
Zili Zhang	Deakin University, Australia
Peixiang Zhao	Florida State University, USA
Yu Zheng	Microsoft Research Asia, China
Bin Zhou	University of Maryland Baltimore County, USA
Shuigeng Zhou	Fudan University, China
Wenjun Zhou	University of Tennessse - Knoxville, USA
Feida Zhu	Singapore Management University, Singapore
Xingquan Zhu	University of Technology Sydney, Australia

Table of Contents – Part I

Table of Contents – Part II

Erratum

Discovering Local Subgroups, with an Application to Fraud Detection

Rob M. Konijn, Wouter Duivesteijn, Wojtek Kowalczyk, and Arno Knobbe

LIACS, Leiden University, The Netherlands
{konijn,wouterd,wojtek,knobbe}@liacs.nl

Abstract. In Subgroup Discovery, one is interested in finding subgroups that behave differently from the 'average' behavior of the entire population. In many cases, such an approach works well because the general population is rather homogeneous, and the subgroup encompasses clear outliers. In more complex situations however, the investigated population is a mixture of various subpopulations, and reporting all of these as interesting subgroups is undesirable, as the variation in behavior is explainable. In these situations, one would be interested in finding subgroups that are unusual with respect to their neighborhood. In this paper, we present a novel method for discovering such *local subgroups*. Our work is motivated by an application in health care fraud detection. In this domain, one is dealing with various types of medical practitioners, who sometimes specialize in specific patient groups (elderly, disabled, etc.), such that unusual claiming behavior in itself is not cause for suspicion. However, unusual claims with respect to a reference group of similar patients do warrant further investigation into the suspect associated medical practitioner. We demonstrate experimentally how local subgroups can be used to capture interesting fraud patterns.

1 Introduction

In this paper, we present a method for discovering local patterns in data. Our method, called *Local Subgroup Discovery (LSD)*, is inspired by Subgroup Discovery (SD) techniques [4,8], which to some degree have a local focus, but the notion of locality plays a more important role here than in standard SD. When a dataset contains natural variations that are *explainable*, traditional SD methods will focus on these. By contrast, when relatively rare events such as fraud or terrorism are concerned, we aim to find *local deviations* within these natural fluctuations. Hence, the LSD method intends to provide interesting and actionable phenomena in the data, *within* the natural variations in the data that are not interesting to report. As we are interested in local deviations, we will use a neighborhood concept, in terms of a basic distance measure over the data.

In order to define the notion of locality, we work with a *reference group* that represents a subpopulation or neighborhood, in the context of which we want to evaluate subgroups. A *local subgroup* is a subgroup within this reference group. Such local subgroups will be judged in terms of their unusual distribution of the

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 1–12, 2013.

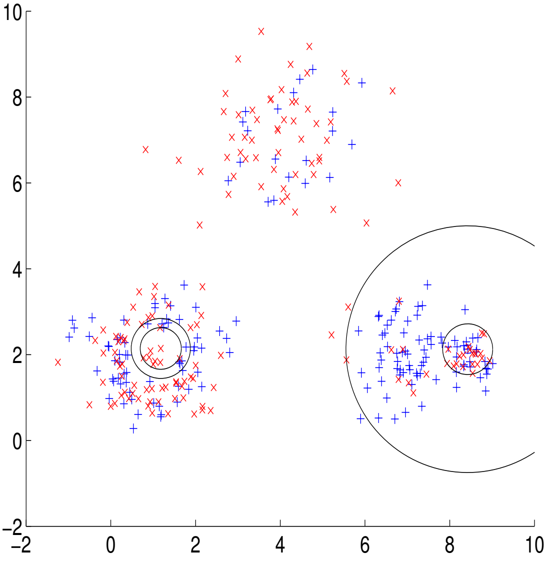


Fig. 1. Two local subgroups that are hard to find with traditional subgroup techniques. The smaller of the two concentric circles indicates the subgroup, the larger of the two circles indicates the reference group.

target attribute, compared with that of the reference group (rather than comparing with the entire population). Both the reference group and the subgroup will be defined in terms of distances from a common subgroup center. Standard quality measures from the SD literature are used to quantify how interesting a subgroup is with respect to its reference group. Figure 1 shows an example of two local subgroups within an artificial dataset. In SD, we are interested in finding groups that contain relatively many positive examples. The local subgroup consists of points within the smallest circle, the corresponding reference group are points within the larger circle. Both the reference group and the subgroup are defined in terms of distances from their subgroup center. Both subgroups contain relatively many red crosses, compared to their local neighborhood.

In this paper we are interested in finding the most interesting subgroups within reference groups in a dataset. The main contributions of this paper are: defining the Local Subgroup Discovery task, presenting a new algorithm that searches for local subgroups efficiently, and showing the usefulness of the local subgroup concept in a real-life fraud detection application.

1.1 Motivation: Health Insurance Fraud Detection

The motivation for LSD comes from the field of fraud detection in health insurance data, where we are interested in identifying suspicious claim behavior of medical practitioners. The problem is essentially an unsupervised one, since we are not presented with any examples of known fraud cases. The goal is to identify groups of claims that warrant further inspection by fraud investigators. We do this by comparing the claim distribution of one medical practitioner (typically, a high-cost claimer) with that of the remaining practitioners. The health insurance data we consider in our experiments contains information about patients and practitioners. A single record in this dataset summarizes the care (treatment, medication, costs, etc.) an individual has received over a selected period of time. Although the data is described in terms of patients, we are actually more concerned with an analysis of this data on the level of medical practitioners. After all, consistent 'inefficient' claim behavior (in the extreme case: fraud) committed by specific practitioners has a far more substantial commercial impact than such behavior committed by specific patients.

We will explain the setting with the help of Figure 1. Suppose that each point represents a patient. The dataset clearly consists of three clusters, which we can interpret as groups of patients with the same type of disease. Within these clusters the differences between patients are caused by treatment costs, variations in medication types, and so on. The approach we take in this paper is to single out an individual medical practitioner, and define a temporary binary column that identifies the patients of this practitioner. This column will then serve as the target in a Subgroup Discovery run, in order to find patterns in the claim behavior that distinguish this practitioner from the others. In Figure 1, patients of the marked practitioner are represented by red crosses. Patients of other practitioners appear as blue plusses. The two subgroups (smaller circles) indicate a difference in claim behavior within a reference group of patients having the same disease type (larger circles); the proportion of red plusses in the subgroup is much higher than the proportion of plusses in the reference group. If substantial local subgroups can be found for an individual practitioner, this is a strong indication of inefficient or fraudulent practice. By repeating this process, each time focusing on a different practitioner, we can produce a list of all suspicious practitioners, along with the necessary details about the unusual behavior.

2 Related Work

Describing distributional differences between the target and non-target set is usually referred to as Subgroup Discovery [8], but also as Contrast Set Mining [1], and Emerging Pattern Mining [2]. These methods find subgroups that are interesting compared to the whole dataset, whereas the method we propose finds locally interesting subgroups. Also, in Subgroup Discovery the subgroups are usually described by conditions on the non-target attributes, whereas in our application we use a distance measure to define subgroups. This is similar to epidemiology where a spatial scan statistic [6] is often used to identify regions where a certain type of disease (the target attribute) is more frequently present than in the rest of the country. Here a likelihood ratio statistic is often used to identify interesting regions. In our approach we also look for interesting regions in our data. Unlike with the spatial scan, our approach allows for any quality measure (instead of the likelihood ratio statistic) and finds reference groups together with subgroups. Calculating the quality measure on a subset of the data has been done before [7]. The subset of the data is obtained by using a distance measure. Luong et al. fix a nearest neighbor parameter k, calculate a quality measure on a part of the data based on this k, and then describe interesting regions in the dataset for which this quality measure is high. The difference with our approach is that we are interested in searching for interesting subgroups, while automatically finding relevant values of k.

To the best of our knowledge, this is the first approach to unsupervised fraud detection by using Subgroup Discovery to compare between entities (in this article these entities are medical practitioners, but they could be shops, cashiers, etc.). Other approaches to fraud detection are supervised methods (where fraud is labeled beforehand), and outlier detection methods. Since we do not have a labeled fraud set, and we are interested in differences on the aggregated level of claims of medical practitioners rather than finding single outliers, these methods are beyond the scope of the paper.

3 Preliminaries

Throughout this paper we assume a dataset D with N elements (*examples*) that are $(h+1)$-dimensional vectors of the form $x = \{a_1, .., a_h, t\}$. Hence, we can view our dataset as an $N \times (h+1)$ matrix, where each data point is stored as a row $x^i \in D$. We call $a^i = \{a^i_1, .., a^i_h\}$ the *attributes* of x^i, and t^i its *target.* We assume that each target is binary, and each vector of attributes comes from an undefined space $\mathcal{A}$ on which we have a distance measure $\delta : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R}$.

A *subgroup* can be any subset of the dataset $S \subseteq D$. A *quality measure* $q : 2^D \rightarrow \mathbb{R}$ is a function assigning a numeric value to any subgroup. Quality measures describe how interesting subgroups are, and usually take into account the size of a subgroup (larger is better) as well as its difference in target distribution (higher proportion of the target is better).

To deal with locality we propose a distance-based approach to find subgroups and reference groups, based on prototypes. A *prototype* can be any point in attribute space $x \in \mathcal{A}$. The *distance-based subgroup* S_σ based on x for parameter $\sigma \in \mathbb{N}$, consists of the σ nearest (according to δ) neighbors of x in D. The *reference group* R_ρ based on the same x for parameter $\rho \in \mathbb{N}$ s.t. $\rho \geq \sigma$, consists of the ρ nearest neighbors of x in D. The idea is that R_ρ forms a region in input space where the target distribution is different from that distribution over the whole dataset, and we strive to find subgroups $S_\sigma \subseteq R_\rho$ in which the target distribution is different from the distribution within R_ρ.

We write $S(x, \sigma, \rho)$ for the subgroup S_σ in a reference group R_ρ, which we call a *reference-based subgroup.* The prototype can be seen as the center of this subgroup, and as the center of the reference group encompassing the subgroup. A quality measure calculated for a reference-based subgroup considers only examples inside the reference group: the quality of the subgroup is calculated on a contingency table of the data, as if the reference group were the entire dataset.

$$ranking(x) = \{\ +, +, -, +, -, +, +, \underset{\sigma}{\underset{\uparrow}{+}}, -, -, -, +, -, \underset{\rho}{\underset{\uparrow}{-}}, \dots\ \}$$

Fig. 2. Ranking of the target vector for a prototype x. The target vector is sorted according to the distances to x. All examples from the leftmost observation (the closest point to x) to σ are in the subgroup. All examples from the leftmost observation to ρ are in the reference group.

Given a prototype x, a distance measure δ, and the target vector t, we can obtain a ranking of the target variable (see Figure 2). This ranking is a sorted list of targets, where the leftmost point is closest to x and the rightmost point is the point furthest from x. To find the optimal reference and subgroup for a given x, we have to set σ and ρ in such a way that the quality measure is maximized. For example, let us calculate the Weighted Relative Accuracy (WRAcc) quality measure for the subgroup and reference group in Figure 2, for the parameters $\sigma = 8$ and $\rho = 14$. The WRAcc of a subgroup S with respect to target t is defined as $P(St) - P(S)P(t)$, where $P(St)$ is the probability of the target and subgroup occurring together, $P(S)$ is the probability of a record being in the subgroup, and $P(t)$ is the probability of the target being true. These probabilities are all calculated given that a point belongs to the reference group. In this example the WRAcc is thus given by: ${}^{6}/_{14} -^{8}/_{14} \cdot^{7}/_{14} =^{1}/_{7} \approx 0.143$. If we would set ρ to 11 instead of 14 this would lead to a somewhat higher quality: ${}^{6}/_{11} -^{8}/_{11} \cdot^{6}/_{11} =^{18}/_{121} \approx 0.148$.

4 Finding Local Subgroups

In this section we explain how the optimal subgroups and their reference groups are found. First we explain how we search for the most interesting subgroups with the highest quality. We also explore our approach to two problems encountered when searching for local subgroups. The first problem is how to compare qualities found on different reference groups, with different reference group sizes and different numbers of positives. The second problem is concerned with the potential redundancy in the collection of reported subgroups.

4.1 Searching for the Optimal Values

In LSD, subgroups are described by optimal combinations of the prototype x and the parameters σ and ρ. Assume we are considering a candidate prototype x. We then loop over possible values of ρ, and for each value, try all possible values for σ and calculate the quality. Per value of ρ, the highest quality obtained in this way is called the *optimal quality*, and the value $\sigma(x, \rho)$ at which this maximum is obtained is called the *optimal value* for σ, given x and ρ.

If we were to search for optimal values of the quality measures in this way, we would find that for the ranking in Figure 2, the optimal value for WRAcc would be obtained at $\rho = 3$, and $\sigma = 2$, with a WRAcc value of $^2/_9 \approx 0.222$. Unfortunately, this subgroup is not that interesting because it is quite small. Nor is it very significant; in any dataset and for any prototype, we can typically construct such a tiny subgroup and a reference group that perfectly separates positive from negative examples. This behavior of favoring very small reference groups in which we can perfectly separate positive from negative cases does not only occur for the WRAcc measure; any quality measure will suffer from this problem. Hence, guiding the search solely by high quality will not lead to interesting results. The size of the reference group should also be big enough to ensure that a subgroup with a high quality is really interesting. Hence, we need to deal with the significance of the found quality.

4.2 Significance and Interestingness

To compute the significance of a candidate x, ρ, and associated $\sigma(x, \rho)$, we use a swap randomization technique, creating a baseline *distribution of false discoveries* (DFD). This method [3] was originally designed for SD where subgroups are defined by attribute-value descriptions. We modify the method for use with distance-based subgroups, as follows. First, the target variable in the reference group is permuted, while keeping the rest of the dataset intact. Within Figure 2 this corresponds to permuting all plusses and minusses up to ρ. Since we leave the attribute space intact, all distances between examples remain the same. Next we search for the optimal quality in this permuted reference group. The optimal quality found can be considered a false discovery, because it is a discovery made on data in which the attribute space is left intact, but its connections with the targets are randomized. We can repeat this procedure to obtain more false discoveries. Together, these qualities constitute a sample of the DFD. The DFD for a quality measure thus differs for each combination of reference group size, and the number of times the target is positive in the reference group. Using this DFD we can assign p-values to subgroups having a certain quality. As [3] describes, a normal distribution can be used to estimate p-values, corresponding to the null hypothesis that a subgroup with the given quality is a false discovery.

The p-value obtained gives us a fair measure to compare qualities found for different reference groups. A low p-value indicates a low probability of finding the quality by chance. Within our approach, we compare subgroups found on different reference groups by comparing their p-value. In Section 5 we show how to search for subgroups with the lowest p-values efficiently.

4.3 Choosing Prototypes and Removing Redundancy

In the previous section, we explained how to find optimal values for σ and ρ, for a given prototype x. Since we will find optimal values for each prototype, which can be each point in the dataset, this will lead to discovering many (redundant) subgroups. In this section, we describe how to find optimal (non-redundant) values for x, with the goal of presenting a concise list of subgroups to the user. To achieve this, we use a top-k approach: only the k most interesting subgroups are mined. Additionally we will use two techniques to remove redundancy from this top-k list. The first technique, based on the *quality neighborhood* of examples, will prevent redundant subgroups to enter the top-k. The second technique postprocesses the top-k to select a small group (generally 3 to 6 subgroups) as the least redundant subgroups from the top-k list.

Consider Only Local Maxima. Points that are close to each other in the dataset will generally have the same neighbors. When these examples are considered as prototypes, they will have similar optimal qualities, since their optimal subgroups and reference groups will strongly overlap. Given a subgroup size σ and a reference size ρ, we can compute the quality of the subgroup $S(x, \sigma, \rho)$ for each prototype x. In this way we can determine a *quality landscape* for our data. Within this quality landscape, we then look for local optima. To this end, we define the quality neighborhood $q_{\text{neighborhood}}(x, \sigma, \rho)$, of a point x. We do this by considering the set $X_x \subseteq D$ of the σ nearest neighbors of x. For each neighbor we determine the quality of its reference-based subgroup. These values form the quality neighborhood: $q_{\text{neighborhood}}(x, \sigma, \rho) = \{q\left(S(x', \sigma, \rho)\right) | x' \in X_x\}$, where $q()$ is the quality measure on the subgroup. A prototype x is a local maximum if its quality is maximal among its quality neighborhood: $q\left(S(x, \sigma, \rho\right) \geq \max q_{\text{neighborhood}}(x, \sigma, \rho)$. Prototypes that are no local maximum are considered not interesting, and their subgroups will therefore not be reported.

Post-processing the Top-k. There still may be some largely overlapping subgroups and reference groups of prototypes in each others neighborhood that have only a slightly different value for σ and ρ. Hence we employ redundancy removal techniques such as joint entropy and the exclusive coverage heuristic [5]. We select the combination of subgroups with the highest value for the heuristic.

5 Reduction of DFD Estimations

Distance-based methods can be computationally intensive. We use different pruning strategies to reduce the number of times the DFD has to be computed.

From all possible reference group sizes ρ for a prototype, we would like to report the optimal quality with the lowest p-value. To obtain p-values, we have to estimate the DFD. The estimation of all DFDs is computationally intensive (it requires $O(s\rho)$ calculations, where s is the sample size). In total there are n possible values of ρ so (if the DFD is not stored) it has to be recomputed n times for each prototype, where n is the number of examples. Computing the DFD is unnecessary for a subgroup that will not enter the top-k anyway. We present a user-set parameter and two pruning techniques to reduce the number of DFD calculations, and show the pseudocode.

Algorithm 1. Lowest p-Value for Prototype(db, x, $q_{optimal}$, $\sigma_{optimal}$)

```
input  : database $db$, prototype $x$, $q_{optimal}$, $\sigma_{optimal}$
output : $pval_{best}, q_{best}, \sigma_{best}, \rho_{best}$
$q_{threshold} = -\infty$;
$pval_{best} = \infty$;
foreach $\rho$ in decreasing order do
    if isLocalMaximum($x, \sigma_{optimal}, q_{optimal}(\rho)) \wedge (q_{optimal}(\rho) \geq q_{threshold})$ then
        compute $DFD_{x,\rho}$;
        p-value $\leftarrow DFD_{x,\rho}(q_{optimal}(\rho))$;
        if p-value $\leq pval_{best}$ then
            $pval_{best}$ $\leftarrow$p-value ;
            $\rho_{best} \leftarrow \rho$;
            $q_{best} \leftarrow q_{optimal}(\rho)$;
            $\sigma_{best} \leftarrow \sigma_{optimal}(\rho)$;
            $q_{threshold} \leftarrow q_{optimal}(\rho)$;
        else
            $q_{threshold} \leftarrow \Phi^{-1}_{DFD}(1 - pval_{best})$;
```

Maximum Value for ρ. To decrease computation time, and ensure locality of the patterns at the same time, the user can set a maximum value for the reference group size.

Consider Only Local Maxima. We can check whether a point is a local maximum before the DFD is estimated. If the point is not a local maximum, the DFD does not have to be estimated since there is a largely similar neighboring subgroup with a better quality.

Pruning the ρ-Search Space Based on Sample Size. To search efficiently we prune away parts of the search space where we know that the p-value can not be lower than the one already found. If for two different reference group sizes the same quality is obtained, the quality calculated on the largest reference group will be the most significant finding, and thus have the lowest p-value.

We explain how this pruning strategy works step by step, by using the pseudocode in algorithm 1. For each prototype we keep in memory a threshold on the quality, denoted by $q_{threshold}$. We also keep in memory the optimal p-value that is found so far for this prototype, $pval_{best}$. We are interested in finding the lowest p-value for this prototype. We start by calculating the p-value for a prototype for the maximal value for ρ in the first iteration.

Next, we observe the qualities found for the same prototype, for smaller values of ρ, in decreasing order. If the optimal quality found for such a smaller reference group, $q_{optimal}(\rho)$, is lower than the threshold, we skip the estimation of the DFD and the calculation of the p-value, because we know the p-value will be lower. For smaller values of ρ that have a higher optimal quality, we compute the DFD. We also obtain the cumulative distribution function of the DFD, Φ_{DFD}, and the inverse cumulative distribution function, Φ^{-1}_{DFD}. From the DFD we obtain the p-value of the quality found. If the newly obtained p-value is lower than $pval_{best}$, this subgroup is the best subgroup found so far. For this prototype x, the corresponding values for $\sigma_{optimal}$ and $\rho_{optimal}$ as well as its quality $q_{optimal}$, and the p-value are stored. The quality threshold is updated and is set to the quality corresponding to the new optimum. If the newly obtained p-value is higher than the one previously found on a bigger sample, these variables are not updated. The quality threshold is updated by inserting $1 - pval_{best}$ into the inverse cumulative distribution function of the DFD.

Pruning the ρ Search Space Using a Top-k Approach. A subgroup only enters the top-k if its p-value is lower than the current maximum p-value in the top-k. The key idea (again) is to update the threshold each time the DFD is estimated. The only difference with pruning-per-prototype is that we can update the quality threshold by inserting the current maximum p-value of the top-k list into Φ^{-1}_{DFD} instead of inserting the minimum p-value found so far for this prototype, to obtain the new threshold.

6 Experiments and Results

Artificial Data. We start by testing our method on the artificially generated data that already featured in the introduction (Figure 1). This two-dimensional data consists of 252 examples, with a roughly equal spread between positive and negative examples. 20 subgroups were obtained by considering each individual example in the data as a prototype, using Euclidean distance and WRAcc as quality measure. From these 20 subgroups, we select 3 non-redundant subgroups by using the exclusive coverage measure [5]. We compare the results with those of a 'traditional' SD algorithm which features in the generic SD tool Cortana[1]. The traditional SD algorithm describes subgroups in terms of attribute-value descriptions.

The first subgroup discovered by LSD is also found by Cortana. In Figure 1 this corresponds to the big cluster at the top. The second subgroup, found in the lower left cluster in Figure 1, is not found using traditional SD methods, because there are many subgroups from the big cluster with the same size that also have a high proportion of positive examples. The third subgroup is situated in the lower right corner. This subgroup is not detected with Cortana. This is because the density of the target variable in the subgroup is the same as the density of the target in the entire dataset, so the density of the target differs only locally.

In order to gauge the efficiency of our method and the different pruning options, we generated datasets of various sizes, while keeping the original distribution intact (all artificial datasets will be made available online). Figure 3 shows the influence of the different pruning strategies on the computation time. In general, a combination of pruning and local maxima offers the best performance, over an order of magnitude faster than either method alone. For comparison,

[1] `datamining.liacs.nl/cortana.html`

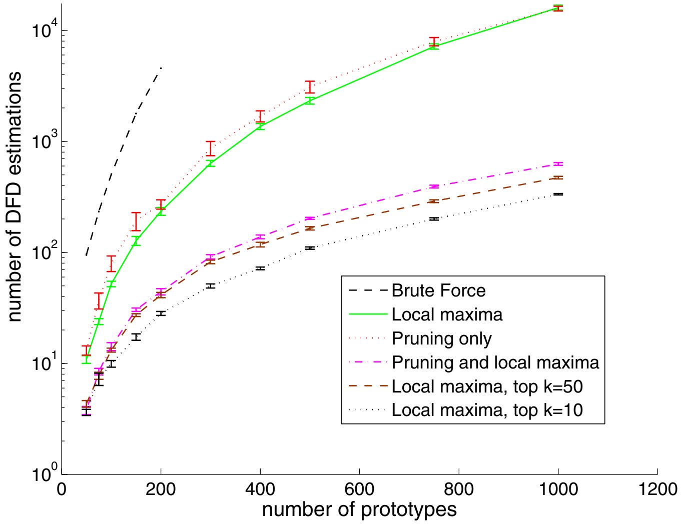


Fig. 3. The increase in computation time for different numbers of prototypes for the artificial dataset, using different pruning strategies. Because of the very long run time, the brute force approach is only performed up to 200 prototypes. The results are averaged over 10 runs, the bars around each point indicate the mean plus and minus the standard error of the mean.

the brute force approach at $n = 200$ takes 4,548 seconds, over two orders of magnitude slower (157.9 times) than the fastest result.

We also observed the effect the local maxima pruning strategy has on the Subgroup Discovery results. If the local maxima strategy is not used, this will lead to the presentation of many redundant subgroups from the big cluster in the top. Each point in this cluster is then presented as an interesting prototype.

Fraud Detection. Our health care application concerns fraud amongst dentists. Each patient is represented by a binary vector of treatments received during a year. The dataset contains 980,350 patients and 542 treatment codes. We use Hamming distance, and quality measure WRAcc. Note that because of the discrete nature of the data, there are many duplicate examples (many patients with identical treatments). Additionally, the distance of a point to different neighbors may be identical, which limits the number of values of σ and ρ that need to be tested. We restrict ρ to a maximum of 10% of the data. We select a dentist with a markedly high claiming profile, and define the target accordingly. 5,567 patients (0.57% of the population) visit this dentist.

Table 1 shows the local subgroups found by our proposed method, LSD. These results were obtained after mining the top 50 subgroups first, and then selecting for diversity using the exclusive coverage heuristic. The interpretation for subgroup S1 is that for patients receiving a *regular consult* (C11, C12) and *dental cleaning* (M50), the dentist often charges extra costs for a *plaque score* (M31)

Table 1. Prototypes and their support in the subgroup, and their support in the reference group excluding the subgroup. The codes indicate treatments that were charged for a patient, the supports indicate the fraction of patients receiving those treatments respectively. The columns # t and # $\neg t$ are the counts within those groups of positive and negative examples.

Subgroup	**Prototype and Supports**	# t	# $\neg t$	**WRAcc**	**p-value**
$S1$ prototype	{C11,C12,C22,M31,M50,X21}				
$S1$	{1.00,0.97,0.17,0.49,0.93,0.60}	54	78		
$R1 \setminus S1$	{1.00,0.94,0.03,0.12,0.95,0.13}	667	10,734	0.0042	< 0.0001
$S2$ prototype	{C11,M31}				
$S2$	{1.00,1.00}	30	2,189		
$R2 \setminus S2$	{1.00,0.11}	94	35,566	0.0006	< 0.0001
$S3$ prototype	{C13,X10,X21}				
$S3$	{0.38,0.71,0.18}	85	12,177		
$R3 \setminus S3$	{0.03,0.11,0.01}	55	30,417	0.0010	< 0.0001

and an *orthopantomogram x-ray picture* (X21). Also an *anamnese* (investigating the patients history, C22) is charged much more often for this group of patients. In subgroup S2, patients receiving a regular consult and a plaque score are occurring relatively more frequently than patients having only a regular consult without the plaque score (which are in the reference group outside the subgroup). In subgroup S3, codes X10 and X21 are *x-ray pictures*, and C13 means an *incidental consult.* Note that treatment C11 is not in the prototype, but still has a support of 77% in the subgroup and a support of 98% in the reference group outside the subgroup. We can conclude that this dentist charges relatively many x-ray pictures of type X10 with a regular or incidental consult. The qualities for the subgroups S1, S2 and S3 are 32, 15 and 14 standard deviations from the mean of the DFD, respectively, which results in p-values near zero.

As a baseline, we compare the results using traditional SD in Cortana, using WRAcc, a search depth of 3 and beam search with beam width 100. We obtain the top 50 subgroups first, and then select for diversity using the exclusive coverage heuristic [5]. There are two subgroups that cover the other subgroups in the top 50: X21 = 1, and C11 =1 $\wedge$ V21 = 1. Code X21 represents an *orthopantomogram* (x-ray photo), code C11 represents a *consult*, and code V21 is used for *polishing a sealing.* The subgroup sizes are 100,643 and 369,748, with 2,438 and 3,228 positive examples, respectively, which leads to a WRAcc of 0.0020, and 0.0014 respectively. The main difference between LSD and traditional SD is that the local approach presents locally deviating patient groups and provides information about the patient group's neighborhood. The resulting subgroups are easier to evaluate by domain experts, and detailed enough to be investigated further by fraud investigators. The traditional SD approach returns global patterns that are not interesting or specific enough to trigger an action.

With our LSD algorithm, we were able to mine interesting subgroups in 5.5 hours in this dataset containing almost a million examples and over 500 attributes. The Cortana traditional SD algorithm took 24 minutes. Both were run on the same machine with 32 GB of main memory, and 8 cores. Although the runtime of the LSD approach depends on the dataset, and the parameter for the maximum value of ρ, this shows that the LSD approach is scalable to fairly big datasets.

We applied our method to compare pharmacies as well as different type of dentists, also using other distance measures like standardized Euclidean distance (in this case, the emphasis is on costs per treatment rather than combinations of treatments only). The results were presented to the fraud investigation department of an insurance company, and were considered very interesting for further inspection. The absence of 'cheap' patients in the reference group as well as the presence of relatively many similar, but more expensive, patients in the subgroup is very useful for indicating inefficient claim behavior.

7 Conclusion and Further Research

In this paper, we present a new approach to find local subgroups in a database. These local subgroups are very relevant within a fraud detection application because systematically committed fraud leads to local distribution changes. Inspired by the fraud detection application, there are numerous directions to investigate further. One promising direction will be cost-based Subgroup Discovery to find even more interesting subgroups. Instead of the distance-based approach, we can also investigate a more traditional, descriptive approach to find local subgroups.

Acknowledgments. This research is financially supported by the Netherlands Organisation for Scientific Research (NWO) under project number 612.065.822 (Exceptional Model Mining).

References

1. Bay, S., Pazzani, M.: Detecting group differences: Mining contrast sets. Data Mining and Knowledge Discovery 5(3), 213–246 (2001)
2. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: Proceedings of KDD 1999, New York, NY, USA, pp. 43–52 (1999)
3. Duivesteijn, W., Knobbe, A.: Exploiting false discoveries - statistical validation of patterns and quality measures in subgroup discovery. In: proceedings ICDM (2011)
4. Klösgen, W.: Subgroup Discovery. In: Handbook of Data Mining and Knowledge Discovery, ch. 16.3, Oxford University Press, New York (2002)
5. Knobbe, A.J., Ho, E.K.Y.: Pattern teams. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 577–584. Springer, Heidelberg (2006), http://dx.doi.org/10.1007/11871637_58
6. Kulldorff, M.: A spatial scan statistic. Communications in Statistics - Theory and Methods 26(6), 1481–1496 (1997)
7. Luong, B.T., Ruggieri, S., Turini, F.: k-nn as an implementation of situation testing for discrimination discovery and prevention. In: Proceedings of KDD 2011, New York, NY, USA, pp. 502–510 (2011)
8. Wrobel, S.: An algorithm for multi-relational discovery of subgroups. In: Komorowski, J., Żytkow, J.M. (eds.) PKDD 1997. LNCS, vol. 1263, pp. 78–87. Springer, Heidelberg (1997)

PUF-Tree: A Compact Tree Structure for Frequent Pattern Mining of Uncertain Data

Carson Kai-Sang Leung[*] and Syed Khairuzzaman Tanbeer

Dept. of Computer Science, University of Manitoba, Canada
{kleung,tanbeer}@cs.umanitoba.ca

Abstract. Many existing algorithms mine frequent patterns from traditional databases of *precise* data. However, there are situations in which data are *uncertain*. In recent years, researchers have paid attention to frequent pattern mining from uncertain data. When handling uncertain data, UF-growth and UFP-growth are examples of well-known mining algorithms, which use the UF-tree and the UFP-tree respectively. However, these trees can be large, and thus degrade the mining performance. In this paper, we propose (i) a more compact tree structure to capture uncertain data and (ii) an algorithm for mining all frequent patterns from the tree. Experimental results show that (i) our tree is usually more compact than the UF-tree or UFP-tree, (ii) our tree can be as compact as the FP-tree, and (iii) our mining algorithm finds frequent patterns efficiently.

1 Introduction

Since the introduction of frequent pattern mining [1], there have been numerous studies on mining and visualizing *frequent patterns* (i.e., *frequent itemsets*) from *precise* data such as databases (DBs) of market basket transactions [8,11,12]. Users definitely know whether an item is present in, or is absent from, a transaction in these DBs of precise data. However, there are situations in which users are uncertain about the presence or absence of some items or events [3,4,5,10,14,16]. For example, a physician may highly suspect (but cannot guarantee) that a patient suffers from flu. The uncertainty of such suspicion can be expressed in terms of existential probability. For instance, a patient may have a 90% likelihood of having the flu, and a 20% likelihood of having a cold regardless of having the flu or not. With this notion, each item in a transaction t_j in DBs containing precise data can be viewed as an item with a 100% likelihood of being present in t_j.

To deal with uncertain data, the *U-Apriori algorithm* [6] was proposed in PAKDD 2007. As an Apriori-based algorithm, U-Apriori requires multiple scans of uncertain DBs. To reduce the number of DB scans (down to two), the tree-based *UF-growth algorithm* [13] was proposed in PAKDD 2008. In order to compute the expected support of each pattern *exactly*, paths in the corresponding UF-tree are shared only if tree nodes on the paths have the same item and same existential probability. Hence, the UF-tree may not be too compact. In

[*] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 13–25, 2013.

an attempt to make the tree compact, the *UFP-growth algorithm* [2] groups *similar* nodes (with the same item x and similar existential probability values) into a cluster. However, depending on the clustering parameter, the corresponding UFP-tree may be as large as the UF-tree (i.e., no reduction in tree size). Moreover, UFP-growth returns not only the frequent patterns but also some infrequent patterns (i.e., false positives). As alternatives to trees, hyper-structures were used by the *UH-Mine algorithm* [2], which was reported [15] to outperform UFP-growth.

Recently, we studied fast tree-based mining of frequent patterns from uncertain data [14]. In the current paper, we study the following questions: Can we further reduce the tree size (e.g., smaller than the UFP-tree)? Can the resulting tree be as compact as the FP-tree? How to mine frequent patterns from such a compact tree? Would such a mining algorithm be faster than UH-Mine? Our **key contributions** of this paper are as follows:

1. a **p**refix-capped **u**ncertain **f**requent pattern **tree (PUF-tree)**, which can be as compact as the original FP-tree; and
2. a mining algorithm (namely, **PUF-growth**), which is guaranteed to find *all and only those* frequent patterns (i.e., *no* false negatives and *no* false positives) from uncertain data.

The remainder of this paper is organized as follows. The next section presents background and related works. We then propose our PUF-tree structure and PUF-growth algorithm in Sections 3 and 4, respectively. Experimental results are shown in Section 5, and conclusions are given in Section 6.

2 Background and Related Works

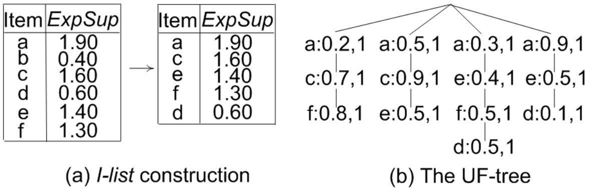
In this section, we first give some background information about frequent pattern mining of uncertain data (e.g., existential probability, expected support), and we then discuss some related works.

Let (i) `Item` be a set of m domain items and (ii) $X = \{x_1, x_2, \ldots, x_k\}$ be a k-itemset (i.e., a pattern consisting of k items), where $X \subseteq \texttt{Item}$ and $1 \leq k \leq m$. Then, a transactional DB $= \{t_1, t_2, \ldots, t_n\}$ is the set of n transactions, where each transaction $t_j \subseteq \texttt{Item}$. The projected DB of X is the set of all transactions containing X.

Unlike precise DBs, each item x_i in a transaction $t_j = \{x_1, x_2, \ldots, x_h\}$ in an uncertain DB is associated with an ***existential probability value*** $\boldsymbol{P(x_i, t_j)}$, which represents the likelihood of the presence of x_i in t_j [9]. Note that $0 < P(x_i, t_j) \leq 1$. The ***existential probability*** $\boldsymbol{P(X, t_j)}$ **of a pattern** $\boldsymbol{X}$ **in** $\boldsymbol{t_j}$ is then the product of the corresponding existential probability values of items within X when these items are independent [9]: $P(X, t_j) = \prod_{x \in X} P(x, t_j)$. The ***expected support*** $\boldsymbol{expSup(X)}$ of X in the DB is the sum of $P(X, t_j)$ over all n transactions in the DB: $expSup(X) = \sum_{j=1}^{n} P(X, t_j)$. A pattern X is ***frequent*** in an uncertain DB if $expSup(X) \geq$ a user-specified minimum support threshold *minsup*. Given a DB and *minsup*, the research problem of **frequent pattern**

Table 1. A transactional DB (*minsup*=0.5)

TID	Transactions	Sorted transactions (with infrequent items removed)
t_1	$\{a{:}0.2,\ b{:}0.2,\ c{:}0.7,\ f{:}0.8\}$	$\{a{:}0.2,\ c{:}0.7,\ f{:}0.8\}$
t_2	$\{a{:}0.5,\ c{:}0.9,\ e{:}0.5\}$	$\{a{:}0.5,\ c{:}0.9,\ e{:}0.5\}$
t_3	$\{a{:}0.3,\ d{:}0.5,\ e{:}0.4,\ f{:}0.5\}$	$\{a{:}0.3,\ e{:}0.4,\ f{:}0.5,\ d{:}0.5\}$
t_4	$\{a{:}0.9,\ b{:}0.2,\ d{:}0.1,\ e{:}0.5\}$	$\{a{:}0.9,\ e{:}0.5,\ d{:}0.1\}$



(a) *I-list* construction (b) The UF-tree

Fig. 1. The UF-tree for the DB shown in Table 1 when *minsup*=0.5

mining from uncertain data is to discover from the DB a complete set of frequent patterns having expected support ≥ *minsup*.

Recall from Section 1 that the tree-based **UF-growth algorithm** [13] uses **UF-trees** to mine frequent patterns from uncertain DBs in two DB scans. Each node in a UF-tree captures (i) an item x, (ii) its existential probability, and (iii) its occurrence count. Tree paths are shared if the nodes on these paths share the same ⟨item, existential probability⟩-value. In general, when dealing with uncertain data, it is not uncommon that the existential probability values of the same item vary from one transaction to another. As such, the resulting UF-tree may not be as compact as the FP-tree. See Fig. 1, which shows a UF-tree for the DB presented in Table 1 when *minsup*=0.5. The UF-tree contains four nodes for item a with different probability values as children of the root. Efficiency of the corresponding UF-growth algorithm, which finds all and *only those* frequent patterns, partially relies on the compactness of the UF-tree.

In an attempt to make the tree more compact, the **UFP-growth algorithm** [2] was proposed. Like UF-growth, the UFP-growth algorithm also scans the DB twice and builds a **UFP-tree**. As nodes for item x having similar existential probability values are clustered into a mega-node, the resulting mega-node in the UFP-tree captures (i) an item x, (ii) the *maximum* existential probability value (among all nodes within the cluster), and (iii) its occurrence count (i.e., the number of nodes within the cluster). Tree paths are shared if the nodes on these paths share the same item but *similar* existential probability values. In other words, the path sharing condition is less restrictive than that of the UF-tree. By extracting appropriate tree paths and constructing UFP-trees for subsequent projected DBs, UFP-growth finds all frequent patterns and some *false positives* at the end of the second DB scan. The third DB scan is then required to remove those false positive.

The **UH-Mine algorithm** [2] stores all frequent items in each DB transaction in a hyper-structure called **UH-struct**. As UH-Mine does not take advantage of prefix-sharing, the size of the resulting UH-struct is always as large as that of the DB for the frequent items. However, UH-Mine was reported [2,15] to be faster than UFP-growth.

3 Our PUF-Tree Structure

To reduce the size of the UF-tree and UFP-tree, we propose the **prefix-capped uncertain frequent pattern tree (PUF-tree)** structure, in which important information about uncertain data is captured so that frequent patterns can be mined from the tree. The PUF-tree is constructed by considering an upper bound of existential probability value for each item when generating a k-itemset (where $k > 1$). We call the upper bound of an item x_r in a transaction t_j the **(prefixed) item cap** of x_r in t_j, as defined below.

Definition 1. The **(prefixed) item cap** $\boldsymbol{I^{Cap}(x_r, t_j)}$ of an item x_r in a transaction $t_j = \{x_1, \ldots, x_r, \ldots, x_h\}$, where $1 \leq r \leq h$, is defined as the product of $P(x_r, t_j)$ and the highest existential probability value M of items from x_1 to x_{r-1} in t_j (i.e., in the *proper prefix* of x_r in t_j):
$I^{Cap}(x_r, t_j) = \begin{cases} P(x_r, t_j) \times M & \text{if } h > 1 \\ P(x_1, t_j) & \text{if } h = 1 \end{cases}$, where $M = \max_{1 \leq q \leq r-1} P(x_q, t_j)$. □

Example 1. Consider an uncertain DB with four transactions as presented in the second column in Table 1. The item cap of c in t_1 can be computed as $I^{Cap}(c, t_1) = 0.7 \times \max\{P(a, t_1), P(b, t_1)\} = 0.7 \times \max\{0.2, 0.2\} = 0.7 \times 0.2 = 0.14$. Similarly, the item cap of f in t_1 is $I^{Cap}(f, t_1) = 0.8 \times \max\{P(a, t_1),\ P(b, t_1), P(c, t_1)\} = 0.8 \times max\{0.2, 0.2, 0.7\} = 0.8 \times 0.7 = 0.56$. □

Note that $I^{Cap}(x_r, t_j)$ provides us with an important property of covering the existential probabilities of all possible patterns containing x_r and its prefix in t_j, as stated in the following theorem.

Theorem 1. *Let X be a k-itemset in transaction t_j (where $k > 1$). The existential probability of any of its non-empty proper subset Y that shares the same suffix x_r as X (i.e., $Y \subset X \subseteq t_j$) is always less than or equal to $I^{Cap}(x_r, t_j)$.*

Proof. Let (i) $t_j = \{x_1, \ldots, x_r, \ldots, x_h\}$, (ii) $X = \{x_s, \ldots, x_r\}$ be a k-itemset in t_j, and (iii) $Y = \{x_t, \ldots, x_r\}$ be a k'-itemset (where $k' < k$) and a proper subset of X (i.e., $Y \subset X \subseteq t_j$). Then, recall from Section 2 that the existential probability $P(Y, t_j)$ of Y in t_j is defined as follows: $P(Y, t_j) = \prod_{x \in Y} P(x, t_j)$, which is equivalent to $P(x_r, t_j) \times \prod_{(x \in Y) \wedge (x \neq x_r)} P(x, t_j)$. Note that $0 < P(x, t_j) \leq 1$. So, $\prod_{(x \in Y) \wedge (x \neq x_r)} P(x, t_j) \leq \max_{1 \leq q \leq r-1} P(x_q, t_j)$. Hence, $P(Y, t_j) = P(x_r, t_j) \times \prod_{(x \in Y) \wedge (x \neq x_r)} P(x, t_j) \leq P(x_r, t_j) \times \max_{1 \leq q \leq r-1} P(x_q, t_j) = I^{Cap}(x_r, t_j)$. □

Since the expected support of X is the sum of all existential probabilities of X over all the transactions containing X, the *cap* of expected support of X can then be defined as follows.

Definition 2. The **cap of expected support** $\boldsymbol{expSup^{Cap}(X)}$ of a pattern $X = \{x_1, \ldots, x_k\}$ (where $k > 1$) is defined as the sum (over all n transactions in a DB) of all item caps of x_k in all the transactions that contain X: $expSup^{Cap}(X) = \sum_{j=1}^{n}\{I^{Cap}(x_k, t_j) \mid X \subseteq t_j\}$. □

Based on Definition 2, $expSup^{Cap}(X)$ for any k-itemset X=$\{x_1, \ldots, x_k\}$ can be considered as an *upper bound* to the expected support of X, i.e., $expSup(X) \leq expSup^{Cap}(X)$. So, if $expSup^{Cap}(X) < minsup$, then X cannot be frequent. Conversely, if X is a frequent pattern, then $expSup^{Cap}(X)$ must be $\geq$ *minsup*. Hence, we obtain a *safe condition* with respect to $expSup^{Cap}(X)$ and *minsup*. This condition, which helps us to obtain an upper bound of the expected support for each pattern, could be safely applied for mining all frequent patterns.

As the *expected support* satisfies the downward closure property [1], all non-empty subsets of a frequent pattern are also frequent. Conversely, if a pattern is infrequent, then none of its supersets can be frequent. In other words, for $Y \subset X$, (i) $expSup(X) \geq minsup$ implies $expSup(Y) \geq minsup$ and (ii) $expSup(Y) < minsup$ implies $expSup(X) < minsup$.

Example 2. Consider two patterns X & Y. Then, $P(Y, t_j) \geq P(X, t_j)$ for any transaction t_j such that $Y \subset X \subseteq t_j$. Moreover, Y must be present in any transaction whenever X is present, but *not* vice versa. So, the number of transactions containing Y is at least the number of transactions containing X. Hence, $expSup(Y)$=$\sum_{j=1}^{n} P(Y, t_j) \geq \sum_{j=1}^{n} P(X, t_j)$=$expSup(X)$ for all n transactions in a DB. If X is frequent (i.e., its expected support $\geq$ *minsup*), then Y is also frequent because $expSup(Y) \geq expSup(X) \geq minsup$. Conversely if Y is infrequent (i.e., its expected support $<$ *minsup*), then X cannot be frequent because $expSup(X) \leq expSup(Y) < minsup$. □

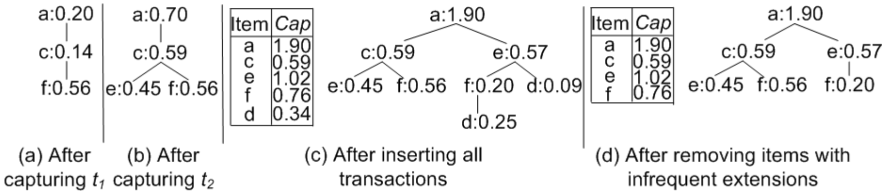
In contrast, the *cap of expected support* generally does *not* satisfy the downward closure property because $expSup^{Cap}(Y)$ can be less than $expSup^{Cap}(X)$ for some proper subset Y of X. See Example 3.

Example 3. Let t_3=$\{a$:0.3, e:0.4, f:0.5, d:0.5$\}$ be the only transaction in the DB containing X=$\{a, e, f, d\}$ and its subset Y=$\{e, f\}$. Then, $expSup^{Cap}(Y) = P(f, t_3) \times \max\{P(a, t_3), P(e, t_3)\}$=0.5×0.4=0.20 and $expSup^{Cap}(X) = P(d, t_3) \times \max\{P(a, t_3), P(e, t_3), P(f, t_3)\} = 0.5 \times 0.5 = 0.25$. This shows that $expSup^{Cap}(Y)$ can be $<$ $expSup^{Cap}(X)$. □

However, for some *specific* cases (e.g., when X & Y share the same suffix item x_r), the cap of expected support satisfies the downward closure property, as proved in Lemma 1. We call such property the ***partial downward closure property***: For any non-empty subset Y of X such that both X & Y ends with x_r, (i) $expSup^{Cap}(X) \geq minsup$ implies $expSup^{Cap}(Y) \geq minsup$ and (ii) $expSup^{Cap}(Y) < minsup$ implies $expSup^{Cap}(X) < minsup$.

Lemma 1. The cap of expected support of a pattern X satisfies the *partial* downward closure property.

Proof. Let (i) X & Y be two itemsets such that $Y \subset X$, and (ii) X & Y share the same suffix item x_r. Then, Y must be present in any transaction whenever X is present, but *not* vice versa. The number of transactions containing



(a) After capturing t_1 (b) After capturing t_2 (c) After inserting all transactions (d) After removing items with infrequent extensions

Fig. 2. Our PUF-tree for the DB in Table 1 when *minsup*=0.5

Y is higher than or equal to the number of transactions containing X. For any transaction t_j in which X & Y are present, they share the same suffix item x_r and thus share the same $I^{Cap}(x_r, t_j)$. So, $expSup^{Cap}(Y) = \sum_{j=1}^{n} I^{Cap}(x_r, t_j)| Y \subseteq t_j\} \geq \sum_{j=1}^{n} I^{Cap}(x_r, t_j)|X \subseteq t_j\} = expSup^{Cap}(X)$. As a result, if $expSup^{Cap}(X) \geq minsup$, then $expSup^{Cap}(Y) \geq expSup^{Cap}(X) \geq minsup$. Conversely, if $expSup^{Cap}(Y) < minsup$, then $expSup^{Cap}(X) \leq expSup^{Cap}(Y) < minsup$. In other words, the cap of expected support of a pattern satisfies the *partial* downward closure property. □

To address the limitation of path sharing of UF-tree, we avoid keeping multiple nodes for the same item having different existential probability values. The basic idea is that, instead of storing the *exact* existential probability value for a node in the transaction, we store in the PUF-tree node the *maximum* existential probability value of the prefix from that node up to the root (i.e., the item cap of the item). For any node in a PUF-tree, we maintain (i) an *item* and (ii) its prefixed *item cap* (i.e., the sum of all item caps for transactions that pass through or end at the node).

How to construct a PUF-tree? With the first scan of the DB, we find distinct frequent items in DB and construct a header table called *I-list* to store only frequent items in some consistent order (e.g., canonical order) to facilitate tree construction. Then, the actual PUF-tree is constructed with the second DB scan in a fashion similar to that of the FP-tree [7]. A key difference is that, when inserting a transaction item, we first compute its item cap and then insert it into the PUF-tree according to the *I-list* order. If that node already exists in the path, we update its item cap by adding the computed item cap to the existing item cap. Otherwise, we create a new node with this item cap value. For better understanding of the PUF-tree construction, see Example 4.

Example 4. Consider the DB in Table 1, and let the user-specified support threshold *minsup* be set to 0.5. Let the *I-list* follow the descending order of expected supports of items. After the first DB scan, the contents of the *I-list* after computing the expected supports of all items and after removing infrequent items (e.g., item b) are $\langle a$:1.9, c:1.6, d:0.6, e:1.4, f:1.3$\rangle$. After sorting, the *I-list* becomes $\langle a$:1.9, c:1.6, e:1.4, f:1.3, d:0.6$\rangle$, as shown in Fig. 1.

With the second DB scan, we insert only the frequent items of each transaction (with their respective item cap values) in the *I-list* order. For instance, when inserting transaction t_1=$\{a$:0.2, c:0.7, f:0.8$\}$, items a, c and f (with their respective item cap values 0.2, 0.7×0.2=0.14 and 0.8×0.7=0.56) are inserted in the PUF-tree as shown in Fig. 2(a). Fig. 2(b) shows the status of the PUF-tree after inserting t_2={a:0.5, c:0.9, e:0.5} with item cap values 0.5, 0.9×0.5=0.45 and 0.5×0.9=0.45 for items a, c and e. As t_2 shares a common prefix $\langle a, c\rangle$ with an existing path in the PUF-tree, (i) the item cap values of those items in the common prefix (i.e., a and c) are added to the corresponding nodes and (ii) the remainder of the transaction (i.e., a new branch for item e) is inserted as a child of the last node of the prefix (i.e., as a child of c). After capturing all transactions in the DB in Table 1, we obtain the PUF-tree shown in Fig. 2(c). Similar to the FP-tree, our PUF-tree maintains horizontal node traversal pointers, which are not shown in the figures for simplicity. □

Note that we are *not confined* to sorting and storing items in descending order of expected support. We could use other orderings such as descending order of item caps or of occurrence counts. An interesting observation is that, if we were to store items in descending order of occurrence counts, then *the number of nodes in the resulting PUF-tree would be the same as that of the FP-tree.*

Note that the sum of all item cap values (i.e., ***total item cap***) for all nodes of an item in a PUF-tree (which is computed when inserting each transaction) is maintained in the *I-list*. In other words, an item x_r in the *I-list* of the PUF-tree maintains $expSup^{Cap}(X)$ for $X = \{x_1, x_2, \ldots, x_r\}$.

Lemma 2. For a k-itemset $X = \{x_s, \ldots, x_r\}$ (where $k > 1$) in a DB, if $expSup^{Cap}(X) < minsup$, then any k''-itemset $Z = \{x_t, \ldots, x_r\}$ (where $k'' > k$) in the DB that contains X (i.e., $Z \supset X$) cannot be frequent.

Proof. Let X & Z be two itemsets such that (i) $X \subset Z$ and (ii) they share the same suffix item x_r. Following Lemma 1, if $expSup^{Cap}(X) < minsup$, then $expSup^{Cap}(Z) \leq expSup^{Cap}(X) < minsup$. As $expSup^{Cap}(Z)$ serves as an upper bound to $expSup(Z)$, we get $expSup(Z) \leq expSup^{Cap}(Z)$. Hence, if $expSup^{Cap}(X) < minsup$, then Z cannot be frequent because $expSup(Z) \leq expSup^{Cap}(Z) < minsup$. □

The above lemma allows us to prune the constructed PUF-tree further by removing any item having a total item cap (in the *I-list*) less than *minsup*. (The horizontal node traversal pointers allow us to visit such nodes in the PUF-tree in an efficient manner.) Hence, we can remove item d from the PUF-tree in Fig. 2(c) because $expSup^{Cap}(d) < minsup$. This results in a more compact PUF-tree, as shown in Fig. 2(d). This tree-pruning technique can save the mining time as it skips those items.

Let $F(t_j)$ be the set of frequent items in transaction t_j. Based on the aforementioned tree construction mechanism, the item cap in a node x in a PUF-tree maintains the sum of item caps (i.e., total item cap) of an item x for all transactions that pass through or end at x. Because common prefixes are shared, the PUF-tree becomes more compact and avoids having siblings containing the same item but having different existential probability values. However, as the item caps of different items in a transaction can be different, the item cap of a node in a PUF-tree does not necessarily need to be greater than or equal to that of all of its child nodes.

It is interesting to note that, although item caps of a parent and child(ren) are not related, a PUF-tree can be a highly compact tree structure. The number of tree nodes in a PUF-tree (i) can be the same to that of an FP-tree [7] (when the PUF-tree is constructed using the frequency-descending order of items) and (ii) is bounded above by $\sum_{t_j \in \text{ DB}} |F(t_j)|$. In addition, the complete set of mining results can be generated because a PUF-tree contains $F(t_j)$ for all transactions and it stores the total item cap for a node. Mining based on this item cap value ensures that no frequent k-itemset ($k > 1$) will be missed.

Furthermore, recall that the expected support of $X = \{x_1, \ldots, x_k\}$ is computed by summing the products of the existential probability value of x_k with those of all items in the proper prefix of X over all n transactions in the DB, i.e., $expSup(X) = \sum_{j=1}^{n}(P(x_k, t_j) \times (\prod_{i=1}^{k-1} P(x_i, t_j)))$. Hence, the item cap for X computed based on the existential probability value of x_k and the highest existential probability value in its prefix provides a *tighter* upper bound (than that based on highest existential probability value in transactions containing X) because the former involves only the existential probability values of items that are in X whereas the latter may involve existential probability values of items that are not even in X.

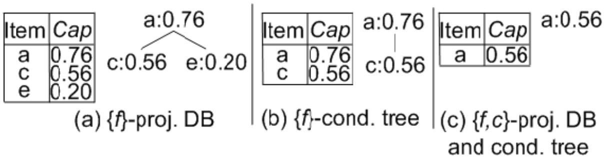
4 Our PUF-Growth Algorithm for Mining Frequent Patterns from PUF-Trees

Here, we propose a pattern-growth mining algorithm called **PUF-growth**, which mines frequent patterns from our PUF-tree structure. Recall that the construction of a PUF-tree is similar to that of the construction of an FP-tree, except that item caps (instead of occurrence frequencies) are stored. Thus, the basic operation in PUF-growth for mining frequent patterns is to construct a projected DB for each potential frequent pattern and recursively mine its potential frequent extensions.

Once an item x is found to be potentially frequent, the existential probability of x must contribute to the expected support computation for every pattern X constructed from $\{x\}$-projected DB (denoted as DB_x). Hence, the cap of expected support of x is guaranteed to be the upper bound of the expected support of the pattern. This implies that the complete set of patterns with suffix x can be mined based on the partial downward closure property stated in Lemma 1. Note that $expSup^{Cap}(X)$ is the upper bound of $expSup(X)$, and it satisfies the partial downward closure property. So, we can directly proceed to generate all potential frequent patterns from the PUF-tree based on the following corollary.

Corollary 1. Let (i) X be a k-itemset (where $k > 1$) with $expSup^{Cap}(X) \geq$ *minsup* in the DB and (ii) Y be an itemset in the X-projected DB (denoted as DB_X). Then, $expSup^{Cap}(Y \cup X)$ in the DB $\geq$ *minsup* if and only if $expSup^{Cap}(Y)$ in all the transactions in $DB_X \geq$ *minsup*.

Proof. Let (i) X be a k-itemset (where $k > 1$) with $expSup^{Cap}(X) \geq minsup$ in the DB and (ii) Y be an itemset in the X-projected DB (i.e., $Y \in DB_X$).



Item	*Cap*
a	0.76
c	0.56
e	0.20

(a) {*f*}-proj. DB

Item	*Cap*
a	0.76
c	0.56

(b) {*f*}-cond. tree

Item	*Cap*
a	0.56

(c) {*f,c*}-proj. DB and cond. tree

Fig. 3. Our PUF-growth mines frequent patterns from the PUF-tree in Fig. 2(d)

Then, due to the mining process (especially, the construction of projected DBs) in the PUF-growth algorithm, itemset $(Y \cup X)$ in the DB shares the same suffix (i.e., X) as itemset Y in DB_X. Moreover, due to the definition of projected DBs, the transactions that contain $(Y \cup X)$ in the DB are identical to those transactions that contain Y in DB_X. Hence, $expSup^{Cap}(Y \cup X)$ in the DB equals to $expSup^{Cap}(Y)$ in DB_X. Consequently, if $expSup^{Cap}(Y \cup X) \geq minsup$ in the DB, then $expSup^{Cap}(Y) \geq minsup$ in DB_X, and vice versa. □

Based on Lemma 1 and Corollary 1, we apply the PUF-growth algorithm to our PUF-tree for generating only those k-itemsets (where $k > 1$) with caps of expected support $\geq$ *minsup*. Similar to UFP-growth, this mining process may also lead to some false positives in the resulting set of frequent patterns at the end of the second DB scan, and all these false positives will be filtered out with the third DB scan. Hence, our PUF-growth is guaranteed to return the *exact* set of frequent patterns (i.e., *all* and *only those* frequent patterns with *neither* false positives *nor* false negatives).

Example 5. The PUF-growth algorithm starts mining from the bottom of the *I-list* (i.e., item f with $expSup^{Cap}(f) = 0.76$). The $\{f\}$-projected DB, as shown in Fig. 3(a), is constructed by accumulating tree paths $\langle a{:}0.56,\ c{:}0.56\rangle$ and $\langle a{:}0.20,\ e{:}0.20\rangle$. Note that the total item cap of f is the sum of item caps in each respective path. When projecting these paths, PUF-growth also calculates the cap of the expected support of each item in the projected DB (as shown in the *I-list* in the figure). Based on Lemma 2, PUF-growth then removes infrequent item e from the $\{f\}$-projected DB—because $expSup^{Cap}(e) = 0.20 < minsup$—and results in the $\{f\}$-conditional tree as shown in Fig. 3(b).

This $\{f\}$-conditional tree is then used to generate (i) all 2-itemsets containing item f and (ii) their further extensions by recursively constructing projected DBs from them. Consequently, pattern $\{f, c\}$:0.56 is generated first, and the $\{f, c\}$-projected DB is then constructed as shown in Fig. 3(c). Pattern $\{f, c, a\}$:0.56 is generated from the $\{f, c\}$-conditional tree. Pattern $\{f, a\}$:0.76 is then generated, which terminates the mining process for $\{f\}$ because no further extension of the projected DB can be generated. Patterns containing items such as e, c and a can then be mined in a similar fashion. The complete set of patterns generated by PUF-growth includes $\{f, c\}$:0.56, $\{f, c, a\}$:0.56, $\{f, a\}$:0.76, $\{e, a\}$:1.02 and $\{c, a\}$:0.59. □

As shown in Example 5, PUF-growth finds a complete set of patterns from a PUF-tree without any false negatives.

5 Experimental Results

We compared the performances of our PUF-growth algorithm with existing algorithms (e.g., UF-growth [13], UFP-growth [2] and UH-Mine [2]) on both real and synthetic datasets. The synthetic datasets, which are generally sparse, are generated within a domain of 1000 items by the data generator developed at IBM Almaden Research Center [1]. We also considered several real datasets such as mushroom, retail and connect4. We assigned a (randomly generated) existential probability value from the range (0,1] to each item in every transaction in the dataset. The name of each dataset indicates some characteristics of the dataset. For example, the dataset u100k10L_10_100 contains 100K transactions with average transaction length of 10, and each item in a transaction is associated with an existential probability value that lies within a range of [10%, 100%]. Due to space constraints, we present here the results on some of the above datasets.

All programs were written in C and run with UNIX on a quad-core processor with 1.3GHz. Unless otherwise specified, runtime includes CPU and I/Os for *I-list* construction, tree construction, mining, and false-positive removal (of PUF-growth). The results shown in this section are based on the average of multiple runs for each case. In all experiments, *minsup* was expressed in terms of the percentage of DB size, and the PUF-trees were constructed using the descending order of expected support of items.

5.1 Compactness of PUF-Trees

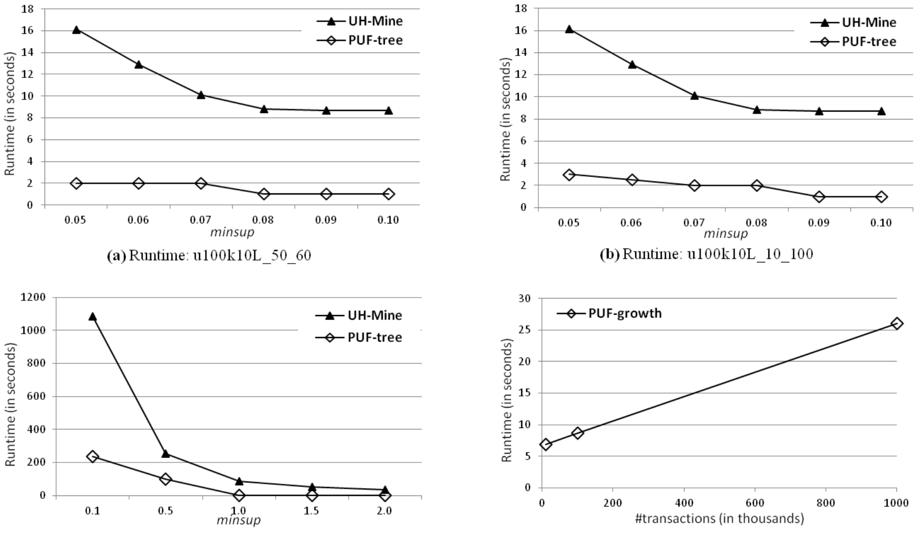
The UF-tree, UFP-tree and PUF-tree all arranged items in the same order (e.g., descending order of expected support). Hence, depending on the clustering parameter, the number of nodes of a UFP-tree (in its best case) could be similar to that of a PUF-tree. Note that the size of UFP-tree was larger than that of PUF-tree because the UFP-tree stored extra cluster information in nodes. The size of UF-tree was larger than that of the other three because the UF-tree may contain multiple nodes for the same item (under the same parent). So, in the first experiment, we compared the compactness of our PUF-trees with the UF-tree[1].

The node counts between PUF-trees and UF-trees, as presented in Table 2, show that PUF-trees were more compact than UF-trees for both sparse and dense datasets and for high and low *minsup* thresholds. The ratio $\frac{\#\text{nodes in PUF-tree}}{\#\text{nodes in UF-tree}}$ represents the gain of PUF-trees over UF-trees (e.g., for mushroom_50_60, the PUF-tree contained 8108 nodes, which was only 6.68% of the 121205 nodes in the UF-tree). The gain of PUF-trees was much promising in dense datasets (e.g., mushroom_50_60) than sparse datasets (e.g., u100k10L_10_100) because the PUF-tree is more likely to share paths for common prefixes in *dense* datasets. In contrast, the UF-tree contains a distinct tree path for each distinct ⟨item, existential probability⟩ pair, and thus *not* as compact as the PUF-tree.

[1] Because UH-Mine stores all transactions in the UH-struct, it is usually less compact than the PUF-tree. So, we avoid showing the comparison with UH-Mine.

Table 2. Compactness of PUF-trees

Dataset	*minsup*	$\frac{\#\text{PUF-tree nodes}}{\#\text{UF-tree nodes [13]}}$	*minsup*	$\frac{\#\text{PUF-tree nodes}}{\#\text{UF-tree nodes [13]}}$
retail_50_60	0.2	$\frac{159{,}504}{672{,}670} \approx 23.71\%$	2	$\frac{208}{17{,}708} \approx 1.17\%$
mushroom_50_60	0.1	$\frac{8{,}108}{121{,}205} \approx 6.68\%$	7	$\frac{2{,}982}{101{,}985} \approx 2.92\%$
u100k10L_50_60	0.05	$\frac{90{,}069}{807{,}442} \approx 11.15\%$	0.1	$\frac{90{,}007}{798{,}073} \approx 11.28\%$
u100k10L_10_100	0.05	$\frac{90{,}069}{881{,}010} \approx 10.22\%$	0.1	$\frac{90{,}013}{872{,}062} \approx 10.32\%$



(a) Runtime: u100k10L_50_60

(b) Runtime: u100k10L_10_100

(c) Runtime: mushroom_50_60

(d) Scalability

Fig. 4. Experimental results

5.2 Runtime

Recall that UH-Mine was shown to outperform the UFP-growth [2,15]. So, we also compared our PUF-growth with UH-Mine. Figs. 4(a)–(c) show that PUF-growth took shorter runtime than UH-Mine for datasets u100k10L_50_60, u100k10L_10_100 and mushroom_50_60. The primary reason is that, even though the UH-Mine finds the exact set of frequent patterns when mining an extension of X, it may suffer from the high computation cost of calculating the expected support of X on-the-fly for all transactions containing X. Such computation may become more costly when mining a large number of patterns (e.g., in mushroom_50_60) and long patterns (e.g., in u100k10L_50_60, u100k10L_10_100, retail_50_60).

5.3 Number of False Positives

In practice, although both UFP-tree and PUF-trees are compact, their corresponding algorithms generate some false positives. Hence, their overall

Table 3. Comparison on false positives (in terms of % of total #patterns)

Dataset	***minsup***	**UFP-growth [2]**	**PUF-growth**
u10k5L_80_90	0.1	61.20%	22.55%
u100k10L_50_60	0.07	89.32%	25.99%

performances depend on the number of false positives generated. In this experiment, we measured the number of false positives generated by UFP-growth and PUF-growth. Due to space constraints, we present results (in percentage) using one *minsup* value for each of the two datasets (i.e., u10k5L_80_90 and u100k10L_50_60) in Table 3. In general, PUF-growth was observed to remarkably reduce the number of false positives when compared with UFP-growth. The primary reason of this improvement is that upper bounds of expected support of patterns in clusters are *not* as tight as the upper bounds provided by PUF-growth. In a UFP-tree, if a parent has several children, then each child will use higher cluster values in the parent to generate the total expected support. If the total number of existential probability values of that child is still lower than that of the parent's highest cluster value, then the expected support of the path with this parent and child will be high. This results in more false positives in long run.

5.4 Scalability

To test the scalability of PUF-growth, we applied the algorithm to mine frequent patterns from datasets with increasing size. The experimental result presented in Fig. 4(d) indicates that our PUF-growth algorithm (i) is scalable with respect to the number of transactions and (ii) can mine large volumes of uncertain data within a reasonable amount of time.

The above experimental results show that our PUF-growth algorithm effectively mines frequent patterns from uncertain data irrespective of distribution of existential probability values (whether most of them have low or high values, whether they are distributed into a narrow or wide range of values).

6 Conclusions

In this paper, we proposed the **PUF-tree structure** for capturing important information of uncertain data. In addition, we presented the **PUF-growth algorithm** for mining frequent patterns. The algorithm uses the PUF-tree to obtain upper bounds to the expected support of frequent patterns (i.e., *item caps*, which are computed based on the highest existential probability of an item in the prefix); it guarantees to find *all* frequent patterns (with *no* false negatives). Experimental results show the effectiveness of our PUF-growth algorithm in mining frequent patterns from our PUF-tree structure.

Acknowledgements. This project is partially supported by NSERC (Canada) and University of Manitoba.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: VLDB 1994, pp. 487–499 (1994)
2. Aggarwal, C.C., Li, Y., Wang, J., Wang, J.: Frequent pattern mining with uncertain data. In: ACM KDD 2009, pp. 29–37 (2009)
3. Bernecker, T., Kriegel, H.-P., Renz, M., Verhein, F., Zuefle, A.: Probabilistic frequent itemset mining in uncertain databases. In: ACM KDD 2009, pp. 119–127 (2009)
4. Calders, T., Garboni, C., Goethals, B.: Approximation of frequentness probability of itemsets in uncertain data. In: IEEE ICDM 2010, pp. 749–754 (2010)
5. Calders, T., Garboni, C., Goethals, B.: Efficient pattern mining of uncertain data with sampling. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS (LNAI), vol. 6118, pp. 480–487. Springer, Heidelberg (2010)
6. Chui, C.-K., Kao, B., Hung, E.: Mining frequent itemsets from uncertain data. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 47–58. Springer, Heidelberg (2007)
7. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM SIGMOD 2000, pp. 1–12 (2000)
8. Lakshmanan, L.V.S., Leung, C.K.-S., Ng, R.T.: Efficient dynamic mining of constrained frequent sets. ACM TODS 28(4), 337–389 (2003)
9. Leung, C.K.-S.: Mining uncertain data. WIREs Data Mining and Knowledge Discovery 1(4), 316–329 (2011)
10. Leung, C.K.-S., Hao, B.: Mining of frequent itemsets from streams of uncertain data. In: IEEE ICDE 2009, pp. 1663–1670 (2009)
11. Leung, C.K.-S., Irani, P.P., Carmichael, C.L.: FIsViz: a frequent itemset visualizer. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 644–652. Springer, Heidelberg (2008)
12. Leung, C.K.-S., Jiang, F.: RadialViz: an orientation-free frequent pattern visualizer. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part II. LNCS (LNAI), vol. 7302, pp. 322–334. Springer, Heidelberg (2012)
13. Leung, C.K.-S., Mateo, M.A.F., Brajczuk, D.A.: A tree-based approach for frequent pattern mining from uncertain data. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 653–661. Springer, Heidelberg (2008)
14. Leung, C.K.-S., Tanbeer, S.K.: Fast tree-based mining of frequent itemsets from uncertain data. In: Lee, S.-g., Peng, Z., Zhou, X., Moon, Y.-S., Unland, R., Yoo, J. (eds.) DASFAA 2012, Part I. LNCS, vol. 7238, pp. 272–287. Springer, Heidelberg (2012)
15. Tong, Y., Chen, L., Cheng, Y., Yu, P.S.: Mining frequent itemsets over uncertain databases. PVLDB 5(11), 1650–1661 (2012)
16. Zhang, Q., Li, F., Yi, K.: Finding frequent items in probabilistic data. In: ACM SIGMOD 2008, pp. 819–832 (2008)

Frequent Pattern Mining in Attributed Trees

Claude Pasquier[1,2], Jérémy Sanhes[1], Frédéric Flouvat[1],
and Nazha Selmaoui-Folcher[1]

[1] University of New Caledonia, PPME, BP R4, F-98851 Nouméa, New Caledonia
{jeremy.sanhes,frederic.flouvat,nazha.selmaoui}@univ-nc.nc
[2] Institute of Biology Valrose (IBV)
UNS - CNRS UMR7277 - INSERM U1091, F-06108 Nice cedex 2
claude.pasquier@unice.fr

Abstract. Frequent pattern mining is an important data mining task with a broad range of applications. Initially focused on the discovery of frequent itemsets, studies were extended to mine structural forms like sequences, trees or graphs. In this paper, we introduce a new data mining method that consists in mining new kind of patterns in a collection of attributed trees (atrees). Attributed trees are trees in which vertices are associated with itemsets. Mining this type of patterns (called asubtrees), which combines tree mining and itemset mining, requires the exploration of a huge search space. We present several new algorithms for attributed trees mining and show that their implementations can efficiently list frequent patterns in a database of several thousand of attributed trees.

Keywords: tree mining, frequent pattern mining, attributed tree.

1 Introduction

Frequent pattern mining is an important problem in data mining research. Initially focused on the discovery of frequent itemsets [1], studies were extended to mine structural forms like sequences [2], trees [7] or graphs [22]. While itemset mining seeks frequent combinations of items in a set of transactions, structural mining seeks frequent substructures. Most existing studies focus only on one kind of problem (itemset mining or structural mining). However, in order to represent richer information, it seems natural to consider itemsets that are organized in complex structures. In this paper, we introduce the problem of mining attributed trees that are tree structures in which each vertex is associated with an itemset.

In web log analysis, for example, it is common to represent user browsing in tree-like data where each page is identified with an unique id. However, one can more pertinently characterize browsed pages with lists of keywords associated with their content. This approach allows to capture the browsing habits of users even when the web site is reshuffled. Other applications can be imagined in various area such as retweet trees mining, spatio-temporal data mining, phylogenetic tree mining and XML document mining.

The key contributions of our work are the following: 1) We present the problem of mining ordered and unordered substructures in a collection of attributed trees.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 26–37, 2013.

2) We define canonical forms for attributed trees. 3) We propose a method for attributed trees enumeration that is based on two operations: itemset extension and tree extension. 4) We present an efficient algorithm IMIT for extracting frequent substructures in a set of attributed trees. 5) We perform extensive experiments on several synthetic datasets and a real weblogs dataset.

The rest of this paper is organised as follows. Section 2 presents basic concepts and defines the problem. Section 3 proposes a brief overview of related works, particularly few studies that mix itemset mining and structure mining. Section 4 describes the method including the search space exploration, the frequency computation and the candidates pruning method. Section 5 reports several applications of the algorithms to mine both synthetic and real datasets. Finally, section 6 concludes the paper and presents possible extensions of the current work.

2 Basic Concepts and Problem Statement

In this section, we give basic definitions and concepts and then introduce the problem of attributed tree mining.

2.1 Preliminaries

Let $\mathcal{I} = \{i_1, i_2, .., i_n\}$ be a set of items. An **itemset** is a set $\mathcal{P} \subseteq \mathcal{I}$. The size of an itemset is the number of items. The set $\mathcal{D}$ of itemsets presents in a database is denoted by $\{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_m\}$ where $\forall \mathcal{P} \in \mathcal{D}, \mathcal{P} \subseteq \mathcal{I}$. $\mathcal{D}$ is a transaction database.

A **tree** $S = (V, E)$ is a directed, acyclic and connected graph where V is a set of vertices (nodes) and $E = \{(u, v) | u, v \in V\}$ is a set of edges. A distinguished node $r \in V$ is considered as the root, and for any other node $x \in V$, there is a unique path from r to x. If there is a path from a vertex u to v in $S = (V, E)$, then u is an *ancestor* of v (v is a *descendant* of u). If $(u, v) \in E$ (i.e. u is an immediate ancestor of v), then u is the *parent* of v (v is a *child* of u). An ordered tree has a left-to-right ordering among the siblings. In this paper, unless otherwise specified, all trees we consider are unordered.

An **attributed tree**, or (**atree**) is a triple $T = (V, E, \lambda)$ where (V, E) is the underlying tree and $\lambda : V \rightarrow \mathcal{D}$ is a function which associates an itemset $\lambda(u) \in \mathcal{I}$ to each vertex $u \in V$. The size of an attributed tree is the number of items associated with its vertices.

In this paper, we use a string representation for an atree based on that defined for labeled trees by Zaki [24]. This representation is only intended to provide a readable form for atrees. The string representation for an atree T is generated by adding a representation of the nodes found in T in a depth-first preorder traversal of T and adding a special symbol \$ when a backtracking from a child to its direct parent occurs. In the paper, for simplicity, we omit the trailing \$s. A string representation of a node is generated by listing all the items present in the associated itemset in a lexicographical order. For example, the string representation of atree $T2$ from Fig. 1 is ”*a c* \$ *cde ab* \$ *a*”.

Attributed trees can be understood as itemsets organized in a tree structure. As such, **attributed tree inclusion** can be defined with respect to itemsets inclusion or structural inclusion. For itemset inclusion, we say that atree T_1 is contained in another atree T_2 if both atrees have the same structure and for each vertex of T_1, the associated itemset is contained in the itemset of the coresponding vertex in T_2. More formally, $T_1 = (V_1, E_1, \lambda_1)$ is **contained in** $T_2 = (V_2, E_2, \lambda_2)$, and is denoted by $T_1 \sqsubset_I T_2$, *if* $V_1 = V_2$ and $E_1 = E_2$ and $\forall x \in V_1, \lambda_1(x) \subseteq \lambda_2(x)$. Structural inclusion is represented by the classical concept of subtree [5,7,12,17,19,23,24].

From the previous definition, we generalize the notion of asubtree in the following way. $T_1 = (V_1, E_1, \lambda_1)$ is a **asubtree** of a atree $T_2 = (V_2, E_2, \lambda_2)$ and is denoted $T_1 \sqsubset T_2$ *if* T_1 is an isomorphic asubtree of T_2, i.e. there exists a mapping $\varphi : V_1 \to V_2$ such that $T_1 \neq T_2$ and $(u, v) \in E_1$ if $(\varphi(u), \varphi(v)) \in E_2$ and $\forall x \in V_1, \lambda_1(x) \subseteq \lambda_2(\varphi(x))$. If T_1 is an asubtree of T_2, we say that T_2 is an asupertree of T_1. T_1 is called an **induced asubtree** of T_2 *iff* T_1 is an isomorphic asubtree of T_2 and φ preserves the parent-child relationships. T_1 is called an **embedded asubtree** of T_2 *iff* T_1 is an isomorphic asubtree of T_2 and φ preserves the ancestor-descendant relationships. $T_1 = (V_1, E_1, \lambda_1)$ is called a **gap-i asubtree** of $T_2 = (V_2, E_2, \lambda_2)$ *iff* T_1 is an isomorphic asubtree of T_2 and φ preserves the ancestor-descendant relationships with the following constraint: $\forall u \forall v \in E_1$ such that u is an ancestor of v and $d(\varphi(u), \varphi(v)) = 1$, $d(u, v) \leq i$ where $d(x, y)$ represents the number of edges between x and y in the atree.

Fig. 1 shows an example of an atree database composed of three different atrees with two (incomplete) sets of common asubtrees using a maximum gap of 0 and 1.

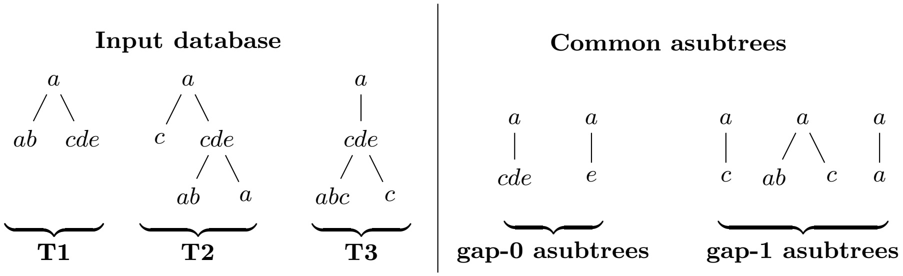


Fig. 1. Example of an atrees database with some common asubtrees

All tree mining algorithms dealing with unordered trees have to face the isomorphism problem. To avoid the redundant generation of equivalent solutions, one tree is chosen as the canonical form and other alternative forms are discarded [3,8,17,23,25]. In previous works, canonical forms are based on a lexicographical ordering on node's labels. In our work, we define an ordering based on node's associated itemsets. Given two itemsets $\mathcal{P}$ and $\mathcal{Q}$ ($\mathcal{P} \neq \mathcal{Q}$), we say that $\mathcal{P} < \mathcal{Q}$ iff 1) $\forall i \in [1, \min(|\mathcal{P}|, |\mathcal{Q}|)] : \mathcal{P}_i \leq \mathcal{Q}_i$ and 2) if $\forall i \in [1, \min(|\mathcal{P}|, |\mathcal{Q}|)] : \mathcal{P}_i = \mathcal{Q}_i$, then $|\mathcal{P}| > |\mathcal{Q}|$. From the definition above, an ordering, $\prec$, among atrees can be defined. From this, a **canonical form of isomorphic atrees** is easily determined using the method presented by Chi et al. [7].

The problem with frequent atrees mining is that the number of frequent patterns is often large. In real applications, generating all solutions can be very expensive or even impossible. Moreover, lots of these frequent atrees contain redundant information. In Fig. 1, for example, atree "$a\ e$" is present in all transactions but the pattern is already encoded in atree "$a\ cde$" because "$a\ e$" is contained in atree "$a\ cde$". This is the same for atree "$a\ a$" which is an asubtree of "$a\ ab\ \$\ c$".

Since the proposal of Manilla et al. [13] huge efforts have been made to design condensed representations that are able to summarize solutions in smaller sets. Set of closed patterns is an example of such a condensed representation [18]. We say that an atree T is a **closed atree** if none of its proper asupertrees has the same support as T. In this paper, we introduce another condensed representation which is defined with respect to the **contained in** relationship only. We say that an atree T is a **c-closed atree** (content closed) if it is not contained (as defined above) in another atree with the same support as T.

2.2 Problem Statement

Given a database $\mathcal{B}$ of atrees and an atree T, the **per-tree frequency** of T is defined as the number of atrees in $\mathcal{B}$ for which T is an asubtree. An atree is frequent if its per-tree support is greater than or equal to a minimum threshold value. The problem consists in enumerating all frequent patterns in a given forest of atrees.

3 Related Works

Most of the earlier frequent tree mining algorithms are derived from the well-known Apriori strategy [1]: a succession of candidates generation phase followed by a support counting phase in which infrequent candidates are filtered out. Two strategies are possible for candidate generation: extension and join. With extension, a new candidate tree is generated by adding a node to a frequent tree [3,17]. With join, a new candidate is created by combining two frequent trees [12,25]. Combination of the two principles has also been studied [8].

Extension principle is a simple method suitable to mine implied trees because the number of nodes that can be used to extend a given subtree is often lower than the number of frequent subtrees.

Other tree mining algorithms are derived from FP-growth approach [11]. These algorithms, which adopt the divide-and-conquer pattern-growth principle avoid the costly process of candidate generation. However, pattern-growth approach cannot be extended simply to tackle the frequent tree pattern mining problem. Existing implementations are limited in the type of trees they can handle: induced unordered trees with no duplicate labels in each node's childs [23], ordered trees [21] or embedded ordered trees [26] are some kind of trees that were successfully mined with pattern-growth approach.

Finding condensed representations of frequent patterns is a natural extension of pattern mining. For itemset mining, the notion of closure is formally defined

[18]. Several works explored this topic in the context of tree mining and proposed mining methods as well as various implementations [9,19,20]. To the best of our knowledge, no method has been proposed for the general case of attributed trees.

Recently we saw growing interest in mining itemsets organized in structures. Miyoshi et al. [14] consider labeled graphs with quantitative attributes associated with vertices. This kind of structure allows to solve the problem by combining a "classical" subgraph mining algorithm for the labeled graph, and an existing itemset mining algorithm for quantitative itemsets in each vertex. Mining attributed subgraphs independently of labels of vertices is impossible with this approach. Several studies [10,15,16] deal with attributed graphs but are looking for frequent subgraphs sharing common sets of attributes. Our work differs from these studies in the sense that itemsets associated with the vertices of a given frequent substructures are not necessarily identical.

4 Mining Frequent Atrees

We are mainly interested in identifying induced ordered and unordered asubtrees. Depending on applications, some patterns including gaps in the ancestor-descendant relationship can also be considered. However, in order to collect only interesting patterns, the gap used should remain small. Otherwise, the relationship between a node and its descendants is not really tangible. Although we focus on induced asubtrees, we designed a general method that is able to mine asubtrees with any gap value, including embedded asubtrees. However, because of the primary objective, our method works better for induced asubtree mining and performances decrease as gap parameter increases.

4.1 Atrees Enumeration

Using the operator $\prec$, it is possible to construct a candidate tree Q representing the complete search space [4] in the following way. The root node of the tree is at the top level and labeled with $\emptyset$. Recursively, for each leaf node $n \in Q$, children n' are added such that $n \prec n'$. Children of a node $n \in Q$, are generated either by tree extension or by itemset extension.

Tree Extension. For tree extension, we use a variation of the well-known rightmost path extension method [3,17]. Let T be an atree of size k. T can be extended to generate new atrees in two different ways. In the first way, a new child N is added to the rightmost node of T (right node extension). In the second way, a new sibling N is added to a node in the rightmost path of T (right path extension) [6].

In the classical approach, N represents every valid node from the input database. In our approach, new nodes N are created from every valid node Q from the input database. In fact, each node Q, associated with an itemset of size k, generates a set of k nodes $\mathcal{N} = \{N_1, .., N_k\}$ used for tree extension. Each N_i is associated with an itemset of size 1; the only item being the ith item of $\lambda(Q)$.

For example, in Fig. 1, the nodes that can be used for right node extension of pattern "*a cde*" are "*ab*", "*a*" (from atree $T2$), "*abc*" and "*c*" (from atree $T3$). From node "*abc*", three extensions are generated ("*a*", "*b*" and "*c*") while node "*ab*" generates "*a*" and "*b*". Nodes "*a*" and "*c*" generate extensions "*a*" and "*c*" respectively. Three different candidates are then obtained by adding each of these extension to the candidate pattern: "*a cde a*", "*a cde b*" and "*a cde c*"

For ordered trees, this method of candidate generation has been shown to be complete as well as non-redundant [3]. However, for unordered trees, it might generate redundant patterns in the form of isomorphic trees. Duplicate candidates are detected and discarded before the candidate extension process by performing a canonical check.

Itemset Extension. For itemset extension, we use a variation of the method presented by Ayres et al. [4]. With this variation, a new item I is added to the itemset associated with the rightmost node of the candidate atree T. Items used for itemset extension are derived from the itemset associated with this node in the input database. The constraint is that the new item must be greater than any item associated with the rightmost node of T.

4.2 Frequency Computation

We organize our data in a structure storing all information needed for the mining process. Our structure is an extension of the vertical representation of trees introduced by Zaki [24,25]. Briefly, each candidate asubtree is associated with its pattern and several data allowing to pinpoint all its occurrences in the database. The first candidates, composed of a unique node associated with one item, are generated by scanning the input database. Using only this unique structure, it is easy to compute the number of occurrences of each pattern. In addition, this same structure is sufficient to generate all possible extensions of a given pattern. When a pattern of size k is processed, all occurrences are extended with tree extension and itemset extension methods described before to generate new $(k + 1)$-candidates that are themselves stored in the structure.

4.3 Search Space Exploration

Several techniques can be used to prune the search tree.

Candidate Pruning. The same rules specified by Agrawal and Srikant twenty years ago [1], can be applied to the case of atrees: i) any sub-pattern of a frequent pattern is frequent, and ii) any super-pattern of a non frequent pattern is non frequent. As the frequency count is an anti-monotonic function (extending a pattern cannot lead to a new pattern with a greater frequency), is it possible to stop the exploration of a branch when the frequency of a candidate is less than the minimum support. For example, in Fig. 1, during the mining of atrees, when we examine pattern "*a c a*" and found that its frequency is lower than the minimum support, we do not generate candidates obtained by extending "*a c a*" (e.g. "*a c ab*", "*a c a \$ b*", "*a c a \$ \$ c*").

In addition, in the case of unordered tree mining, extension of a candidate is stopped if it is not in canonical form.

C-closed Atrees Enumeration. By enumerating only atrees that are not **contained** in another atree with the same support, the search space can be considerably reduced. Enumerating c-closed atrees involves the storage of every frequent pattern found with their associated per-tree frequency and their total number of occurrences in the database (the **occurrence-match frequency**).

Let T be a candidate atree currently processed, $\mathcal{T}$ be the set of all previously identified frequent atrees and $\mathcal{X}$ be the set of candidates generated by extension of T. We distinguish two subsets of $\mathcal{X}$. $\mathcal{X}_I$ is the set of atrees generated by itemset extension of T and $\mathcal{X}_T$ is composed of tree extensions of T. We define two functions: f_t which gives the per-tree frequency of an atree and f_o which returns its occurence-match frequency.

We say that T is a c-closed atree if $\nexists T' \in \mathcal{T} U \mathcal{X}_I$ such that $T \sqsubset_I T'$ and $f_t(T') = f_t(T)$. However, finding an itemset extension of T with the same per-tree frequency as T does not allows to stop the exploration of other candidates in $\mathcal{X}$. The following additional conditions must also be satisfied: $\exists T' \in \mathcal{T} U \mathcal{X}_I$: $T \sqsubset_I T'$ and $f_o(T') = f_o(T)$.

In Fig. 1, for example, the first candidate to be examined is ”a” with a per-tree frequency of 3. By itemset extension, we build $\mathcal{X}_I = \{$”ab”, ”ac”$\}$. Candidate ”ab” has a per-tree frequency of 3, therefore, candidate ”a” is not c-closed as ”a” $\sqsubset_I$ ”ab”. However, pattern ”a” appears 7 times in the database while the total occurrence of candidate ”ab” is 3. The 4 times where ”a” occurs in an itemset which does not contain ”b” may lead to the generation of other patterns that are c-closed. This is the case in Fig. 1 where a right node extension of pattern ”a” generates candidate ”$a\ e$” with a per-tree frequency of 3.

Closed Atrees Enumeration. We say that T is a a closed atree if $\nexists T' \in \mathcal{T} U \mathcal{X}$ such that $T \sqsubset T'$ and $f_t(T') = f_t(T)$. The extension of T can be stopped if $\exists T' \in \mathcal{T} U \mathcal{X} : T \sqsubset T'$ and $f_o(T') = f_o(T)$. In addition, one has also to remove non closed trees from $\mathcal{T}$, i.e. all atrees that are asubtrees of T with a same per-tree frequency. The check for closure requires to perform several subtree isomorphism checks that are costly operations.

4.4 Mining Algorithms

Fig. 2 shows the high level structure of the IMIT algorithm. First, a set with all asubtree of size 1 is built by scanning the input database. Then, a loop allows to process every candidate in the set. The function *GetFirst* return the smallest candidate in the set according to the $\prec$ operator. The processing involves a canonical test and a frequency test. A frequent candidate which is in canonical form is added to the list of solutions and all of its extensions are added to the list of candidates. The processing of a candidate finishes by removing it from the candidates' list.

```
$IMIT(\mathcal{D}, minSup)$
$\mathcal{C} \leftarrow \{all\ asubtrees\ of\ size\ 1\ in\ \mathcal{D}\}$
while $\mathcal{C} \neq \emptyset$ do
    $T \leftarrow getFirst(\mathcal{C})$
    if $isCanonical(T)$ and $f_t(T) \geq minSup$ then
        $\mathcal{T} \leftarrow \mathcal{T} \cup \{T\}$
        $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{X}$
    end if
    $\mathcal{C} \leftarrow \mathcal{C} \setminus \{T\}$
end while
$printSolutions(\mathcal{T})$
```

Fig. 2. IMIT Algorithm

This algorithm is sufficient to enumerate all solutions but it has a huge search space. To limit the redundancies in the set of solutions, we developed IMIT_CLOSED, an algorithm extracting closed asubtrees (Fig. 3). As illustrated in section 5, the algorithm is costly and is not usable to mine large input databases.

We designed IMIT_CONTENT_CLOSED, a third algorithm extracting c-closed asubtrees. This new algorithm (not shown in this paper) can be easily deduced from the IMIT_CLOSED algorithm (Fig. 3) by replacing $\sqsubset$ by $\sqsubset_I$, replacing $\mathcal{X}$ by $\mathcal{X}_I$ in line 5 and removing line 11 to 13. The use of $\sqsubset_I$ instead of $\sqsubset$ allows to only perform itemsets inclusion tests that are less costly than subtree isomorphism checks. Lines 11 to 13 remove from the set of solutions those that are asubtree of the current candidate. This test is not needed for the

```
$IMIT\_CLOSED(\mathcal{D}, minSup)$
 1: $\mathcal{C} \leftarrow \{all\ asubtrees\ of\ size\ 1\ in\ \mathcal{D}\}$
 2: while $\mathcal{C} \neq \emptyset$ do
 3:     $T \leftarrow getFirst(\mathcal{C})$
 4:     if $isCanonical(T)$ and $f_t(T) \geq minSup$ then
 5:         if $\nexists T' \in \mathcal{T} \cup \mathcal{X} : T \sqsubset T'\ and\ f_t(T') = f_t(T)$ then
 6:             $\mathcal{T} \leftarrow \mathcal{T} \cup \{T\}$
 7:         end if
 8:         if $\nexists T' \in \mathcal{T} : T \sqsubset T'\ and\ f_o(T') = f_o(T)$ then
 9:             $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{X}$
10:         end if
11:         for all $T' \in \mathcal{T}\ such\ that\ T' \sqsubset T\ and\ f_o(T') = f_o(T')$ do
12:             $\mathcal{T} \leftarrow \mathcal{T} \setminus \{T'\}$
13:         end for
14:     end if
15:     $\mathcal{C} \leftarrow \mathcal{C} \setminus \{T\}$
16: end while
17: $printSolutions(\mathcal{T})$
```

Fig. 3. IMIT_CLOSED Algorithm

extraction of c-closed patterns. Experiments show that this third algorithm is the best compromise between non redundancy of solutions and execution time.

5 Experimental Results

All algorithms are implemented in C++ using STL. Experiments were performed on a computer running Ubuntu 12.04 LTS and based on a Intel©Core$^{\text{TM}}$i5-2400 @ 3.10GHz with 8 Gb main memory. All timings are based on total execution time, including all preprocessing and results output.

5.1 Synthetic Datasets

We modified the synthetic data generation program proposed by Zaki [24] in order to be able to generate atrees with different size of itemsets. We added two new parameters controlling the minimum and maximum itemset's size. This allows to generate atree with fixed itemset's size or with a size randomly chosen in a range.

We used the default parameters as in [24] except for the number of subtrees T that is set to 10,000. We build five datasets by varying the size of itemsets. In T10K, all vertices are associated with itemsets of size 1. This allows us to compare our implementation with SLEUTH [25]. In T10K-3 and T10K-5, vertices are associated with itemsets of size 3 and 5 respectively. In T10K-1/10, vertices are associated with itemsets of size randomly selected between 1 and 10, while in T10K-1/20, itemsets' size vary from 1 to 20.

5.2 Web Logs Datasets

We built a dataset on logs given by our university following the method described by Zaki [24]. However, instead of labeling nodes with URLs of the browsed pages, we associated them with itemsets representing keywords of their content. The dataset is composed of 126,396 attributed trees with itemsets of size 10 (10 keywords by page).

5.3 Performance Evaluation

Fig. 4 shows the execution time for mining c-closed sets of induced unordered patterns using IMIT_CONTENT_CLOSED on our five synthetic datasets. For comparison, we added in the figure the execution time of SLEUTH, a reference implementation of equivalence class extension paradigm [25], on the T10K dataset. IMIT_CONTENT_CLOSED is about two times slower than SLEUTH for all support values except the smallest ones where SLEUTH is penalized by the cost of joining millions of frequent patterns.

Although IMIT_CONTENT_CLOSED is slower than SLEUTH, these results are satisfactory because our algorithm is designed to mine attributed trees.

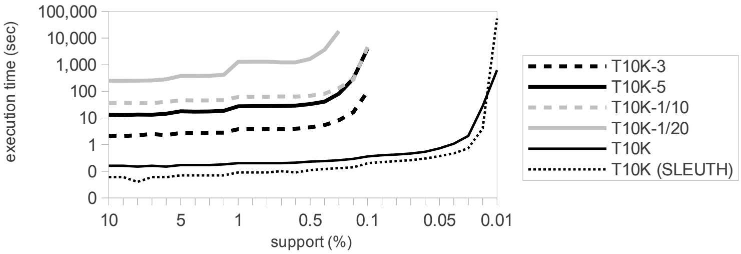


Fig. 4. Execution time of IMIT_CONTENT_CLOSED for mining c-closed sets of induced unordered patterns on 5 synthetic datasets

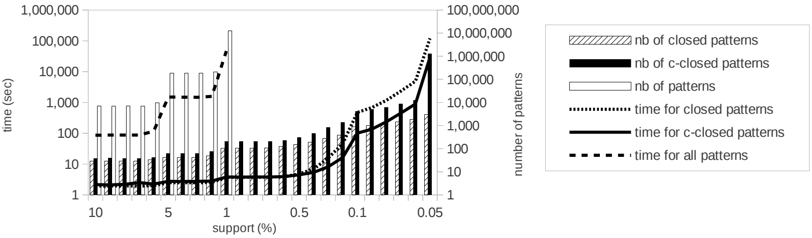


Fig. 5. Induced unordered mining with 3 versions of IMIT on T10K-3 dataset

As such, it is normal to perform worst on mining labeled trees than dedicated implementations. The memory footprint of our algorithm is twice as SLEUTH's one.

The figure also shows that mining attributed trees is extremely more computing intensive than mining labeled trees; and the difference is largely underestimated because only c-closed patterns were mined. Mining all patterns generates a huge number of solutions and takes a long time. To give an idea, mining the T10K-3 dataset with a minimum support of 1% outputs 12 millions patterns in 15 hours (Fig. 5). Mining c-closed atrees allows to reduce both the number of patterns and the execution time. Thus, at 1% minimum support, 200 c-closed patterns are found in 4 seconds.

As shown in the same figure, the search for closed patterns allows to reduce further the number of patterns. At 1% minimum support, for example, the number of patterns drops to 103. However, because of the costly subtree isomorphism checks, in return, performances collapse when patterns become numerous. The result is that the difference in computation time increases as the minimum support decreases.

Fig. 6 show the execution time and number of c-closed patterns in the weblogs dataset. This dataset is much larger than synthetic datasets used before and its mining cannot be performed with a minimum support of less than 10% in a reasonable amount of time. Mining the weblog dataset with a minimum support of 6% lasts 6 hours and returns 360 patterns.

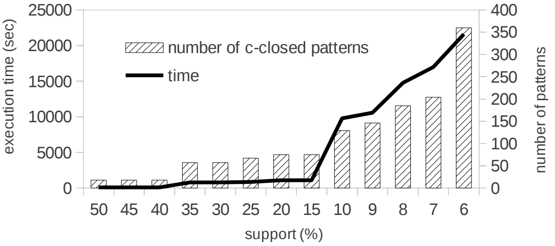


Fig. 6. Performances of IMIT_CONTENT_CLOSED for mining c-closed sets of induced unordered patterns on weblogs datasets

6 Conclusion and Perspectives

In this paper, we introduce the problem of mining attributed trees. We investigate methods enumerating all frequent patterns or only closed ones, but these methods proved inefficient because of, in the first case, the huge number of patterns returned, and in the second case, the cost of subtree isomorphism checks. Finally, we propose a condensed representation of frequent atrees that is defined with respect to itemset inclusion. This representation allows to drastically reduce both the number of patterns and the execution time. We evaluate the efficiency of the proposed algorithm, IMIT_CONTENT_CLOSED, and show that it successfully extract frequent patterns in large datasets. One future work is to extend the proposed algorithm to effectively mine frequent closed patterns. Another future work consists in developing similar methods for mining more complex structures such as attributed graphs.

Acknowledgments. This work was funded by French contract ANR-2010-COSI-012 FOSTER.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. SIGMOD Rec. 22(2), 207–216 (1993)
2. Agrawal, R., Srikant, R.: Mining sequential patterns. In: ICDE, pp. 3–14 (1995)
3. Asai, T., Abe, K., Kawasoe, S., Arimura, H., Sakamoto, H., Arikawa, S.: Efficient substructure discovery from large semi-structured data. In: SDM (2002)
4. Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: KDD, pp. 429–435 (2002)
5. Balcázar, J.L., Bifet, A., Lozano, A.: Mining frequent closed rooted trees. Mach. Learn. 78(1-2), 1–33 (2010)
6. Chehreghani, M.H.: Efficiently mining unordered trees. In: ICDM, pp. 111–120 (2011)
7. Chi, Y., Muntz, R.R., Nijssen, S., Kok, J.N.: Frequent subtree mining - an overview. Fundam. Inf. 66(1-2), 161–198 (2004)

8. Chi, Y., Yang, Y., Muntz, R.R.: Hybridtreeminer: An efficient algorithm for mining frequent rooted trees and free trees using canonical form. In: SSDBM, pp. 11–20 (2004)
9. Chi, Y., Yang, Y., Xia, Y., Muntz, R.R.: Cmtreeminer: Mining both closed and maximal frequent subtrees. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 63–73. Springer, Heidelberg (2004)
10. Fukuzaki, M., Seki, M., Kashima, H., Sese, J.: Finding itemset-sharing patterns in a large itemset-associated graph. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part II. LNCS (LNAI), vol. 6119, pp. 147–159. Springer, Heidelberg (2010)
11. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. SIGMOD Rec. 29(2), 1–12 (2000)
12. Hido, S., Kawano, H.: Amiot: Induced ordered tree mining in tree-structured databases. In: ICDM, pp. 170–177 (2005)
13. Mannila, H., Toivonen, H.: Multiple uses of frequent sets and condensed representations. In: KDD, pp. 189–194 (2005)
14. Miyoshi, Y., Ozaki, T., Ohkawa, T.: Frequent pattern discovery from a single graph with quantitative itemsets. In: ICDM Workshops, pp. 527–532 (2009)
15. Moser, F., Colak, R., Rafiey, A., Ester, M.: Mining cohesive patterns from graphs with feature vectors. In: SDM, pp. 593–604 (2009)
16. Mougel, P.-N., Rigotti, C., Gandrillon, O.: Finding collections of k-clique percolated components in attributed graphs. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part II. LNCS (LNAI), vol. 7302, pp. 181–192. Springer, Heidelberg (2012)
17. Nijssen, S., Kok, J.N.: Efficient discovery of frequent unordered trees. In: First International Workshop on Mining Graphs, Trees and Sequences (MGTS) (2003)
18. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)
19. Termier, A., Rousset, M.C., Sebag, M.: Dryade: A new approach for discovering closed frequent trees in heterogeneous tree databases. In: ICDM, pp. 543–546 (2004)
20. Termier, A., Rousset, M.C., Sebag, M., Ohara, K., Washio, T., Motoda, H.: Dryadeparent, an efficient and robust closed attribute tree mining algorithm. IEEE Trans. on Knowl. and Data Eng. 20(3), 300–320 (2008)
21. Wang, C., Hong, M., Pei, J., Zhou, H., Wang, W., Shi, B.: Efficient pattern-growth methods for frequent tree pattern mining. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 441–451. Springer, Heidelberg (2004)
22. Washio, T., Motoda, H.: State of the art of graph-based data mining. SIGKDD Explor. Newsl. 5(1), 59–68 (2003)
23. Xiao, Y., Yao, J.F., Li, Z., Dunham, M.H.: Efficient data mining for maximal frequent subtrees. In: ICDM, pp. 379–386 (2003)
24. Zaki, M.J.: Efficiently mining frequent trees in a forest. In: KDD, pp. 71–80 (2002)
25. Zaki, M.J.: Efficiently mining frequent embedded unordered trees. Fundam. Inf. 66(1-2), 33–52 (2004)
26. Zou, L., Lu, Y., Zhang, H., Hu, R.: Prefixtreeespan: a pattern growth algorithm for mining embedded subtrees. In: Aberer, K., Peng, Z., Rundensteiner, E.A., Zhang, Y., Li, X. (eds.) WISE 2006. LNCS, vol. 4255, pp. 499–505. Springer, Heidelberg (2006)

Mining Frequent Patterns from Human Interactions in Meetings Using Directed Acyclic Graphs

Anna Fariha[1], Chowdhury Farhan Ahmed[1], Carson Kai-Sang Leung[2], S M Abdullah[3], and Longbing Cao[4]

[1] Dept. of Computer Science and Engineering,
University of Dhaka, Bangladesh
purpleblueanna@gmail.com, farhan@khu.ac.kr
[2] Dept. of Computer Science, University of Manitoba, Canada
kleung@cs.umanitoba.ca
[3] Dept. of Computer Science and Engineering,
United International University, Bangladesh
smab@cse.uiu.ac.bd
[4] Faculty of Engineering and IT, University of Technology - Sydney, Australia
longbing.cao@uts.edu.au

Abstract. In modern life, interactions between human beings frequently occur in meetings, where topics are discussed. Semantic knowledge of meetings can be revealed by discovering interaction patterns from these meetings. An existing method mines interaction patterns from meetings using tree structures. However, such a tree-based method may not capture all kinds of triggering relations between interactions, and it may not distinguish a participant of a certain rank from another participant of a different rank in a meeting. Hence, the tree-based method may not be able to find all interaction patterns such as those about correlated interaction. In this paper, we propose to mine interaction patterns from meetings using an alternative data structure—namely, a *directed acyclic graph* (*DAG*). Specifically, a DAG captures both temporal and triggering relations between interactions in meetings. Moreover, to distinguish one participant of a certain rank from another, we assign weights to nodes in the DAG. As such, a meeting can be modeled as a weighted DAG, from which weighted frequent interaction patterns can be discovered. Experimental results showed the effectiveness of our proposed DAG-based method for mining interaction patterns from meetings.

Keywords: Data mining, Frequent patterns, Human interaction, Modeling meetings, Directed Acyclic Graphs.

1 Introduction and Related Works

Meetings and human interactions are integral parts of workplace dynamics for communicating between members participating in a meeting. During a meeting, several kinds of human interactions may occur. Examples include (i) proposing

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 38–49, 2013.

an idea, (ii) positively or negatively reacting to a proposal, (iii) acceptance of a proposal. To gather significant information regarding the success rate of the decision made in a meeting, one can mine patterns from human interactions occurred in the meeting.

Data mining is useful in discovering implicit, previously unknown, and potentially valuable information or knowledge from large datasets. For instance, frequent pattern mining [1,4,7] is helpful in finding frequently occurring patterns, such as *interaction patterns* from meetings. The discovered interaction patterns help to (i) estimate the effectiveness of decisions made in meetings, (ii) designate whether a meeting discussion is fruitful, (iii) compare two meeting discussions using interaction flow as a key feature [12] and (iv) index meetings for further ease of access in database.

To acquire the semantic information from a meeting, researchers extracted the meeting contents and represented them in a machine readable format. For instance, Waibel et al. [9] presented a meeting browser that describes the dynamics of human interactions. McCowan et al. [5] recognized group actions in meetings by modeling the joint behavior of participants and expressed group actions as a two-layer process by a hidden Markov model framework. Otsuka et al. [6] used gaze, head gestures, and utterances to determine who responds to whom in multiparty face-to-face conversations. Yu et al. [11] proposed a multimodal approach for interaction recognition; they [12] also used a tree-based mining method to discover frequent patterns from human interactions occurred in meetings. Such a method focuses mostly on capturing direct parent-child relations. However, there are other triggering relations in meetings as illustrated in Example 1.

Example 1. Let us consider a scenario about a meeting of four persons (e.g., professor A, assistant professor B, and two lecturers C & D) with different weights/ranks. At the beginning of the meeting, B proposes an idea which triggers three interactions: (i) C expresses his negative opinion towards the proposed idea, (ii) C asks D for opinion on the idea, and (iii) A expresses some positive opinion towards the idea. Now, the interaction of C's request for D's opinion triggers a single interaction performed by D. Although the response of D is triggered by C's request of opinion, such a response is generally influenced by A's positive opinion. To elaborate, D may initially feel negatively regarding B's proposed idea. But, after listening to A's positive comments, B may change his mind and lean towards a neutral or even positive opinion. □

Example 1 reveals that (i) an interaction can be triggered or influenced by multiple interactions and (ii) the extent of influence can be significantly dependent on the weight/rank of the person triggering that interaction. However, the aforementioned tree-based method [12] does not capture these triggering relations. Moreover, as this method does not associate the actions with the rank of the person causing the actions, it does not distinguish the same kinds of actions performed by two persons having different weights/ranks.

Observing that (i) directed acyclic graphs (DAGs) and trees are both specializations of graphs and (ii) trees may not capture all triggering relations or person's weight/rank, we explore the use of DAGs (as alternatives to trees) for modeling meetings. As interactions occur in meetings flow in only one direction with respect to time (i.e., no cycle), DAGs would be a logical choice for modeling meetings. A *key contribution* of this paper is our DAG-based modeling of interactions occurred in meetings. In particular, *DAGs* capture two kinds of relations: (i) temporal relations and (ii) triggering relations (cf. *trees* capture temporal relations but not all triggering relations). By doing so, each interaction is represented by a node in a DAG, and the label of the node indicates the class of interaction. Moreover, every node is associated with a *weight*, indicating the rank of the person who initiates the interaction.

Another *key contribution* of this paper is our DAG-based mining of weighted frequent interaction patterns from meetings. Note that, Chen et al. [3] mined DAG patterns from DAG databases. Termier et al. [8] presented DigDag as the first algorithm to mine closed frequent embedded sub-DAGs. Werth et al. [10] designed and implemented a DAG miner for mining DAGs from DAG databases. However, these related works do not consider weights of nodes in DAGs, let alone mining *weighted* frequent interaction patterns. Furthermore, there exist related works [2] that mine weighted frequent patterns from *transactional databases* (cf. *DAGs*). Inspired by these works, we integrate DAG-based mining and interaction pattern mining [10] with weighted frequent pattern mining [2] to form our **WDAGmeet** algorithm for **W**eighted **DAG**-based **meet**ing mining.

The rest of this paper is organized as follows. Section 2 introduces our DAG-based representation of interaction flow in meetings. Section 3 presents our weighted DAG-based frequent interaction pattern mining. Evaluation results are shown in Section 4, and conclusions are given in Section 5.

2 DAG-Based Representation of Interaction Flow

In this section, we introduce a DAG-based representation of interaction flow in decision-making meetings. Human interactions occurred in these meetings can be mainly categorized into the following nine classes:

1. **PRO:** A participant proposes an idea.
2. **ASK:** A participant asks for opinion regarding a proposal.
3. **POS:** A participant expresses positive attitude towards a proposal.
4. **NEG:** A participant expresses negative attitude towards a proposal.
5. **ACK:** A participant agrees on some other's comment, decision, or attitude.
6. **COM:** A participant comments on another action (PRO, ACK, POS, etc.).
7. **REQ:** A participant requests information regarding an issue.
8. **ACC:** A participant accepts the proposed idea.
9. **REJ:** A participant rejects the proposed idea.

When building a DAG to model the interaction flow occurred in meetings, we label each node in the DAG with one of the above nine classes of human interactions.

To further specify the rank of the person who initiated the interaction, we assign a weight with value ranging from 1 to n inclusive (e.g., n=3, 4 or 5). Although the same response can be made from different persons of different ranks,

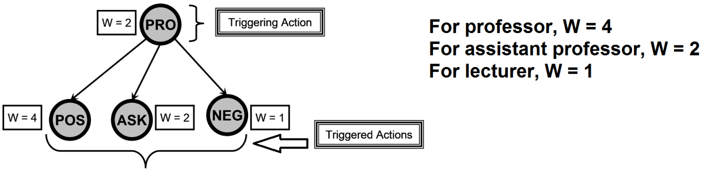


Fig. 1. A DAG-based representation of interaction flow showing triggering relations

a response from a person having a heavier weight usually strongly influences the decision making process than that from a person of a lighter weight. Each node in the weighted DAG in Fig. 1 denotes an instance of human interaction occurred in a meeting. The label and weight of the nodes indicate the class of the interaction and the corresponding impact factor, respectively.

So far, we have categorized interactions into nine classes (e.g., PRO, ASK) based on the activities in a meeting. From the perspective of spontaneity, these interactions can also be categorized into two types of interactions: (i) *triggering interaction* and (ii) *triggered interaction*. For example, the PRO node in Fig. 1 reflects a triggering interaction, which represents an assistant professor proposes an idea spontaneously. The remaining nodes (POS, ASK & NEG) reflect three triggered interactions, which occur in response to the triggering interaction. Directed edges between nodes indicate **triggering relations** between the nodes, and the arrows point from the triggering interaction to the triggered one. Consequently, we generate a DAG-based interaction flow diagram for modeling meetings.

Besides those triggering relations, our proposed DAG-based representation of interaction flow also captures **temporal relations**. To elaborate, a DAG represents the temporal relations by topological level. Nodes of a certain topological level appear temporally before nodes of the next/lower level. Within the same topological level, the node on the left appears temporally before the node on the right. See Example 2.

Example 2. Fig. 2 shows a sample session of a meeting, in which an assistant professor A proposes an idea. Triggered by A's proposed idea, one of his colleagues B first expresses her negative opinion and then asks others' opinions.

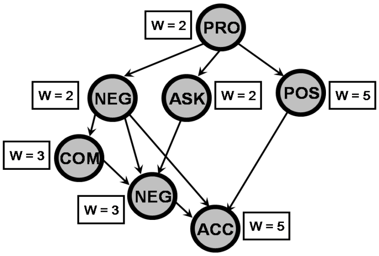


Fig. 2. A DAG-based representation of interaction flow showing both triggering relations and temporal relations in a meeting

On the other hand, a professor C (with heavier weight) expresses his positive opinion on A's idea. An associate professor D first comments on B's negative opinions. Based on both his comments and B's negative opinion, D then expresses his negative opinion in response to B's asking of opinion. Finally, based on two negative opinion from assistant professor B and associate professor D as well as the positive opinion from professor C, professor E accepts A's idea, biased to the interaction performed by person of higher rank. Note that, Fig. 2 captures not only single triggering relations but also interactions triggered by multiple triggering interactions. □

3 DAG-Based Frequent Pattern Mining from Interaction Flow DAGs

Once the DAG-based interaction flow diagram is generated, we can mine frequent interaction patterns (sub-DAG patterns) from the diagram. Before describing the key steps in this interaction pattern mining process, let us consider the following definitions.

Definition 1 (DAG-Based Interaction Flow). One single meeting may consist of several sessions. Interaction flow within each session can be represented by a DAG $D = (V, E)$, where $V = \{v_1, v_2, ..., v_n\}$ is a set of n vertices and $E = \{e_1, e_2, ..., e_m\}$ is a set of m directed edges. All DAGs are connected acyclic graphs and no two DAGs, representing sessions of the same meeting, are connected to each other. Each node v_i is assigned a class label $L(v_i)$, where $L(v_i) \in$ {PRO, ASK, POS, NEG, ACK, COM, REQ, ACC, REJ}. Each node is associated with a weight $W(v_i)$ that carries information regarding the (absolute or relative) rank of a participant who initiates an interaction in a meeting. Each edge is a directed connection between two vertices, i.e., $E = \{(v_i, v_j)|1 \leq i, j \leq n; v_i, v_j \in V\}$. Here, v_i denotes the source/origin of the directed edge and v_j denotes the destination of that edge. An edge from v_i to v_j implies that v_i (completely or partially) triggers v_j. The levels of interactions can be determined according to their topological orders. Interactions of higher levels occur earlier than those of lower levels. Within the same level, interactions on the left occur earlier than those on the right. □

Definition 2 (Sub-DAG and Super-DAG). Consider two DAGs $D = (V, E)$ and $D' = (V', E')$ such that (i) D' is a connected DAG; (ii) $V' \subseteq V$; (iii) $E' \subseteq E$; (iv) for each $v'_i \in V', L(v'_i) = L(v_i)$ and $W(v'_i) = W(v_i)$ for a $v_i \in V$; (v) for each $e = (v'_i, v'_j) \in E'$, these v'_i & $v'_j \in V'$ are mapped to the corresponding v_i & $v_j \in V$. Then, D' is a **sub-DAG** of D. Equivalently, D is a **super-DAG** of D'. □

Definition 3 (Support). Given (i) a sub-DAG D' and (ii) a database DB, the support of D' is defined by the following equation:

$$sup(D') = \frac{\#\text{superDAGs of } D' \times \text{ avg weight of nodes in } D'}{\#\text{DAGs in } DB \times \text{ max weight of nodes in } DB}. \tag{1}$$

This definition of support allows us to discover sub-DAGs containing nodes that are not too frequent but are associated with heavy weights. See Example 3.

Example 3. Consider a sample DAG D, consisting of 10 directed edges (i.e., 10 triggering relations) on 20 nodes (i.e., 20 interactions): Professor A proposes three ideas (PRO), and each of them are rejected (REJ) by Professor B. Lecturer C makes 7 comments (COM), and each of them triggers Lecturer F's comments (COM). In other words, $D = (V, E)$, where (i) $V = \{v_1, ..., v_{20}\}$, (ii) $L(v_1) = L(v_3) = L(v_5) =$ PRO, (iii) $L(v_2) = L(v_4) = L(v_6) =$ REJ, (iv) $L(v_7) = ... = L(v_{20}) =$ COM, (v) $E = \{(v_1, v_2), (v_3, v_4), ..., (v_{19}, v_{20})\}$, (vi) $W(v_1) = ... = W(v_6) = 5$, and (vii) $W(v_7) = ... = W(v_{20}) = 1$. Here, the frequency of the pattern "A proposes an idea, which is rejected by B" is 3; the frequency of another pattern "C makes a comment, which is commented by F" is 7. Between them, the first pattern is more interesting than the second one because interactions between persons of higher rank (i.e., heavier weights) are usually more important and useful in analyzing decision-making meetings even when the frequency of these interactions is not too high. □

Definition 4 (Frequent Pattern or Fragment). Sub-DAGs, having support greater than the user-specific minimum support threshold *minsup* are considered **frequent sub-DAG pattern (or fragment)**. □

Definition 5 (Mining Frequent Interaction Patterns from Meeting DB). Given (i) a meeting database DB capturing human interactions in meetings, (ii) a user-specific minimum support threshold *minsup*, the problem of mining frequent interaction patterns is to discover from DB every frequent interaction pattern, i.e., every sub-DAG D' having $sup(D') \geq minsup$. □

Our proposed weighted DAG-based meeting mining algorithm (**WDAGmeet**) discovers frequent interaction patterns in the form of frequent sub-DAGs from weighted DAG database DB as follows. The algorithm first generates a set of all frequent nodes in DB. It then expands these nodes (i.e., singleton sub-DAGs) using the following four expansion rules:

1. **New Root:** A new root (with no incoming edge) is inserted.
2. **New Level:** A new topological level is introduced with the insertion of a new node into that level and the insertion of an edge from a node in the previous topological level.
3. **New Node:** A new node (with a label lexicographically greater than the last node in current topological level) is inserted into the current topological level.
4. **New Edge:** A new edge from a previously inserted node to the most recently inserted node is added.

See Example 4 for illustration of these rules. Note that, these rules are designed in such a way that no duplicate DAG is generated. In the process of expansion of already found frequent sub-DAGs, duplicate sub-DAGs may be generated. Although these duplicates do not affect the mining result, they certainly increase

Algorithm 1. WDAGmeet

```
Input  : (1) Meeting $DB$ of DAG, (2) $minsup$ threshold
Output: A set $F$ of frequent DAG interaction patterns
begin
    $F \leftarrow$ all the frequent nodes in all the DAGs in $DB$
    $tmpF \leftarrow F$
    while $tmpF \neq \emptyset$ do
        $tmp \leftarrow \emptyset$
        for $f \in tmpF$ do
            $tmp \leftarrow tmp \cup expandWithRoots(DB, f, minsup)$
            $tmp \leftarrow tmp \cup expandWithNewLevel(DB, f, minsup)$
            $tmp \leftarrow tmp \cup expandWithNewNode(DB, f, minsup)$
            $tmp \leftarrow tmp \cup expandWithNewEdge(DB, f, minsup)$
        end for
        $tmp \leftarrow pruneNonCannonical(tmp, minsup)$
        $tmp \leftarrow filterNotWantedFragments(tmp, minsup)$
        $tmp \leftarrow filterNotFrequentFragments(tmp, minsup)$
        $F \leftarrow F \cup findConnectedDAGs(tmp)$
        $tmpF \leftarrow tmp$
    end while
end
```

the runtime of the algorithm. To avoid generating duplicates, all newly expanded sub-DAGs are checked for duplicate canonical form [10] because the canonical form is unique for all duplicate isomorphic DAGs.

Example 4. Consider Fig. 3, which shows some examples of applications of the four expansion rules: WDAGmeet algorithm (a) inserts a *new root* PRO, (b) inserts another new root REQ, and (c) introduces a *new level* with the insertion of POS and of a triggering relation from PRO to POS. Afterwards, WDAGmeet (d) inserts into the current level a *new node* NEG triggered by REQ, (e) adds a *new edge* from PRO to the most recently inserted NEG. Similarly, WDAGmeet (f) introduces another new level (with the insertion of NEG, to which a triggering relation from another NEG is inserted) and (g) adds a new edge from PRO. □

Note that, some of the expanded DAGs are connected, but some are not. WDAGmeet algorithm only inserts frequent *connected* DAGs to the mining result.

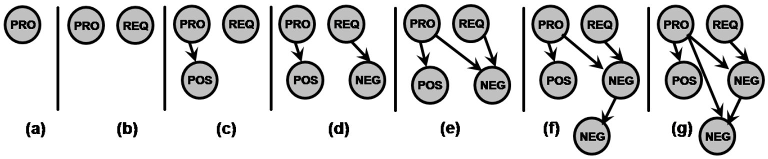


Fig. 3. Applications of expansion rules

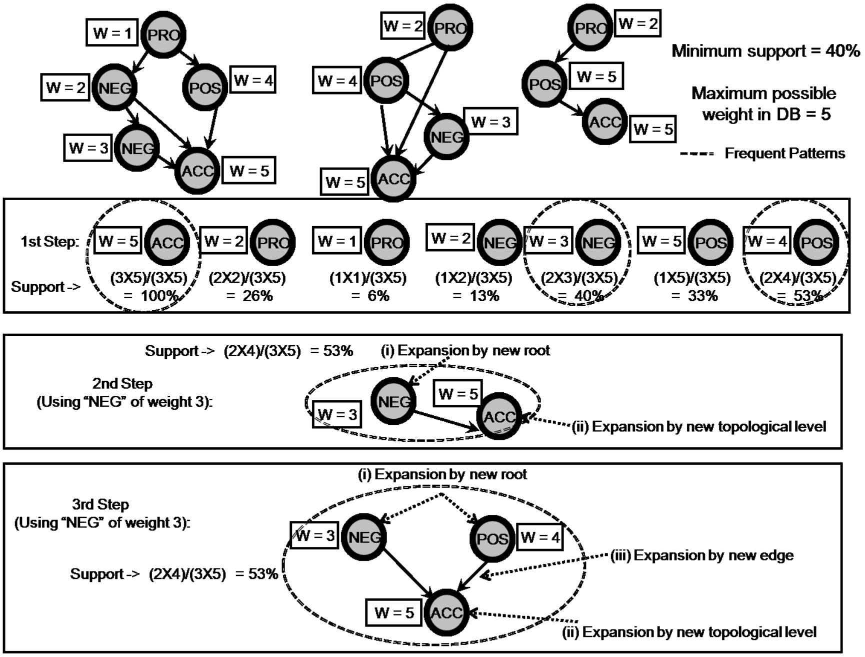

W = 1
PRO
W = 2
NEG
POS
W = 4
W = 3
NEG
ACC
W = 5
W = 2
PRO
W = 4
POS
W = 3
NEG
W = 5
ACC
PRO
W = 2
POS
W = 5
ACC
W = 5
Minimum support = 40%
Maximum possible weight in DB = 5
Frequent Patterns
1st Step:
Support ->
W = 5 ACC (3X5)/(3X5) = 100%
W = 2 PRO (2X2)/(3X5) = 26%
W = 1 PRO (1X1)/(3X5) = 6%
W = 2 NEG (1X2)/(3X5) = 13%
W = 3 NEG (2X3)/(3X5) = 40%
W = 5 POS (1X5)/(3X5) = 33%
W = 4 POS (2X4)/(3X5) = 53%
Support -> (2X4)/(3X5) = 53%
(i) Expansion by new root
2nd Step
(Using "NEG" of weight 3):
W = 3 NEG W = 5 ACC
(ii) Expansion by new topological level
(i) Expansion by new root
3rd Step
(Using "NEG" of weight 3):
Support -> (2X4)/(3X5) = 53%
W = 3 NEG
POS W = 4
(iii) Expansion by new edge
W = 5 ACC
(ii) Expansion by new topological level

Fig. 4. First few steps of WDAGmeet algorithm

The algorithm repeats the above expansion process until no new expansion is impossible. The pseudocode is given in Algorithm 1, and Fig. 4 illustrates the first few steps of the algorithm.

One important observation on the WDAGmeet algorithm is that, when the patterns are expanded, it adds not only frequent nodes but all possible nodes. The reason is that, the expansion rules do not satisfy the anti-monotone property: A pattern f may not be frequent because of low average-weight of the nodes contained in it, but connecting some nodes (of heavier weight or high support) can make f frequent.

4 Evaluation Results

First, we evaluated the functionality of our proposed WDAGmeet algorithm by comparing it with the existing tree-based mining method [12]. The tree-based method misses some important frequent patterns because it does not capture all triggering relations. As illustrated in Fig. 5, only one triggering relation is captured in the tree database for each triggered interaction. For instance, the tree captures the interaction ASK triggered by PRO but misses the one triggered by POS. Similarly, the tree captures the interaction NEG triggered by PRO but misses the one triggered by ASK. As such, the tree-based method does not generate the pattern POS-ASK-NEG as these three nodes are *not directly* connected in the tree. In fact, fragments containing siblings or ancestor's siblings of a node cannot be connected in the absence of their common ancestor in a tree. Hence, if the common ancestor is not frequent, the tree mining method fails to mine such fragments as a frequent pattern. In contrast, being internally connected with partial triggering relations, WDAGmeet discovers this kind of frequent interaction patterns, such as POS-ASK-NEG in the above example. This kind of frequent patterns reveals highly correlated interactions.

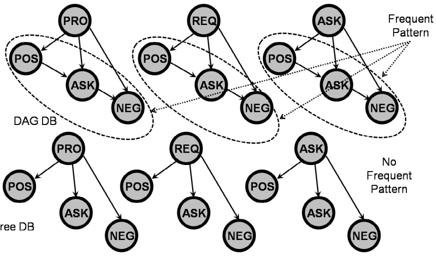


Fig. 5. DAG-based vs. tree-based representations of meetings

Next, we evaluated empirically the performance and effectiveness of WDAGmeet algorithm, which was implemented in C++. In our experiments, we used datasets based on sample meetings. We generated 10 synthetic datasets to simulate real meeting scenarios. Each dataset contains a description of (i) the meeting captured in a DAG, (ii) labeled interactions with their corresponding weights, and (iii) triggering relations (i.e. directed edges of the DAG). We used five distinct weights for ranking each nine interaction with one of the nine class labels. Experiments were run using an Intel Core i5 2.50 GHz machine with 2.94 GB of RAM and 32 bit OS (Windows 7).

Table 1 shows the number of discovered frequent patterns and the elapsed time to discover these frequent patterns when using different minimum support *minsup* threshold values performed on 10 different datasets having different sizes. Table 1 shows that, as the number of frequent patterns increased, the required time to discover these frequent patterns also increased.

Fig. 6 plots the number of frequent patterns and elapsed time vs. *DB* size in the right and left, respectively. One can observe that, on average, the number of discovered frequent patterns (or fragments) was loosely related to the size of DAGs. Dense DAGs usually generated more frequent patterns than sparse ones. When patterns were generated, it was more likely to locate those patterns in a dense DAG than a sparse one because the dense DAG contains most of the probable edges. In contrast, the probability of finding a frequent pattern was low in a sparse DAG. The increment of *DB* size can partially represent the sparseness of the DAG capturing interactions in meetings.

Then, we compared the performance of WDAGmeet algorithm with that of the existing tree-based mining method [12] empirically. Table 2 shows our experimental results, which can be explained as follows. During the mining

Table 1. #frequent patterns & elapsed time with various *minsup* for different *DB*

Size of *DB* (bytes)	*minsup* (%)	#frequent patterns	Elapsed time (seconds)
6917	40	415	440.589
	50	139	92.475
	60	26	17.672
	70	16	9.385
5804	30	481	401.539
	40	184	119.821
	50	25	8.917
	60	3	0.856
5465	50	135	86.668
	55	87	61.318
	60	73	53.787
	65	59	47.327

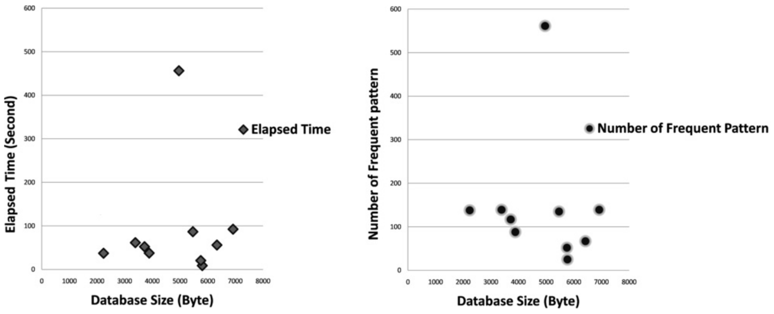


Fig. 6. Elapsed time and number of frequent patterns vs. *DB* size

process, any frequent pattern must be connected because neither WDAGmeet nor the tree-based method can search *DB* for a pattern or fragment that is not connected. As discussed earlier, the tree-based mining method missed some frequent patterns. In contrast, WDAGmeet algorithm did not miss these patterns. Moreover, WDAGmeet algorithm used weighted nodes for representing the importance/rank of persons triggering each interaction. This criterion decreased the number of frequent patterns discovered by WDAGmeet. Moreover, WDAGmeet distinguished multiple interactions initiated by different persons having different weights. In contrast, the tree-based method did not distinguish multiple interactions. Hence, as WDAGmeet captures all triggering and temporal relations, it generated fewer frequent patterns and did not miss any frequent patterns. In contrast, the tree-based method generated more frequent patterns but also missed some frequent patterns.

To summarize, WDAGmeet algorithm discovered frequent interaction patterns from weighted DAGs capturing human interactions in meetings in reasonable amounts of time. When *minsup* increased, the number of discovered frequent patterns decreased and the elapsed time also decreased. When compared with the

Table 2. #frequent patterns & elapsed time for tree-based vs. our DAG-based mining

Size of *DB* (bytes)	*minsup* (%)	#frequent patterns in tree-based method [12]	#frequent patterns in WDAGmeet
6917	40	635	415
	45	298	262
	50	153	139
5804	30	509	481
	35	482	442
	40	201	184
5465	50	146	135
	55	95	87
	60	79	73

existing tree-based method (which captures few triggering and all temporal relations), WDAGmeet algorithm captures all triggering relations as well as all temporal relations. Moreover, WDAGmeet algorithm does not miss any frequent interaction patterns. As an ongoing work, we plan to conduct more extensive experiments and compare the precision, recall and F-measure of our proposed WDAGmeet algorithm with those of the existing tree-based method.

5 Conclusions

In this paper, we modeled human interactions in meetings using a weighted directed acyclic graph (DAG). The weight indicates the rank or importance of the person who initiates one of the nine classes of interactions. Such a DAG-based representation of interaction flow captures both (i) temporal relations and (ii) triggering relations (which connect the triggering interaction to the triggered interaction) in meetings. Moreover, we also proposed DAG-based frequent pattern mining from interaction flow DAGs. Specifically, our proposed WDAGmeet algorithm mines weighted DAG-based meeting for frequent interaction patterns. The key idea is to model each session (especially decision-making sessions) of a meeting using DAGs. Moreover, DAGs also include patterns or fragments that are connected without any common ancestor, previously missed by the existing tree-based method. The integration of weight assignment to each interaction makes the meeting mining process more robust and worthwhile.

Evaluation results show that WDAGmeet algorithm was more effective in discovering frequent interaction patterns from weighted DAGs than the existing tree-based method. The mined frequent sub-DAGs can be served as foundations to further association rule mining. As ongoing work, we plan to integrate other types of human interactions. Moreover, the resulting mining algorithm can be customized to handle other classes of meetings such as medical interviews and business discussions. The property of assigning weights to the interactions and preserving all kinds of partially triggering relations add functionality of WDAGmeet in mining patterns from human interactions in meetings.

Acknowledgements. This project is partially supported by NSERC (Canada) and University of Manitoba.

References

1. Ahmed, C.F., Tanbeer, S.K., Jeong, B.-S., Lee, Y.-K.: An efficient candidate pruning technique for high utility pattern mining. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 749–756. Springer, Heidelberg (2009)
2. Ahmed, C.F., Tanbeer, S.K., Jeong, B.-S., Lee, Y.-K., Choi, H.-J.: Single-pass incremental and interactive mining for weighted frequent patterns. Expert Systems with Applications 39(9), 7976–7994 (2012)
3. Chen, Y.-L., Kao, H.-P., Ko, M.-T.: Mining DAG patterns from DAG databases. In: Li, Q., Wang, G., Feng, L. (eds.) WAIM 2004. LNCS, vol. 3129, pp. 579–588. Springer, Heidelberg (2004)
4. Leung, C.K.-S., Mateo, M.A.F., Brajczuk, D.A.: A tree-based approach for frequent pattern mining from uncertain data. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 653–661. Springer, Heidelberg (2008)
5. McCowan, L., Gatica-Perez, D., Bengio, S., Lathoud, G., Barnard, M., Zhang, D.: Automatic analysis of multimodal group actions in meetings. IEEE TPAMI 27(3), 305–317 (2005)
6. Otsuka, K., Sawada, H., Yamato, J.: Automatic inference of cross-modal nonverbal interactions in multiparty conversations: "who responds to whom, when, and how?" from gaze, head gestures, and utterances. In: ICMI 2007, pp. 255–262. ACM (2007)
7. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: Discovering periodic-frequent patterns in transactional databases. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 242–253. Springer, Heidelberg (2009)
8. Termier, A., Tamada, Y., Numata, K., Imoto, S., Washio, T., Higuchi, T.: DIGDAG, a first algorithm to mine closed frequent embedded sub-DAGs. In: MLG (2007)
9. Waibel, A., Bett, M., Finke, M., Stiefelhagen, R.: Meeting browser: tracking and summarizing meetings. In: DARPA Broadcast News Transcription and Understanding Workshop (1998)
10. Werth, T., Dreweke, A., Wörlein, M., Fischer, I., Philippsen, M.: DAGMA: mining directed acyclic graphs. In: IADIS ECDM 2008, pp. 11–18 (2008)
11. Yu, Z., Yu, Z., Ko, Y., Zhou, X., Nakamura, Y.: Inferring human interactions in meetings: A multimodal approach. In: Zhang, D., Portmann, M., Tan, A.-H., Indulska, J. (eds.) UIC 2009. LNCS, vol. 5585, pp. 14–24. Springer, Heidelberg (2009)
12. Yu, Z., Yu, Z., Zhou, X., Becker, C., Nakamura, Y.: Tree-based mining for discovering patterns of human interaction in meetings. IEEE TKDE 24(4), 759–768 (2012)

ClaSP: An Efficient Algorithm for Mining Frequent Closed Sequences

Antonio Gomariz[1,⋆], Manuel Campos[2], Roque Marin[1], and Bart Goethals[3]

[1] Information and Communication Engineering Dept., University of Murcia, Spain
[2] Languages and Systems Dept., University of Murcia, Spain
[3] Mathematics and Computer Science Dept., University of Antwerp, Belgium

Abstract. In this paper, we propose a new algorithm, called ClaSP for mining frequent closed sequential patterns in temporal transaction data. Our algorithm uses several efficient search space pruning methods together with a vertical database layout. Experiments on both synthetic and real datasets show that ClaSP outperforms currently well known state of the art methods, such as CloSpan.

1 Introduction

Sequence Data Mining (SDM) is a well-extended field of research in Temporal Data Mining that consist of looking for a set of patterns frequently enough occurring across time among a large number of objects in a given input database. The threshold to decide if a pattern is meaningful is called *minimum support*. SDM has been widely studied [6,9,3,5,2], with broad applications, such as the discovery of motifs in DNA sequences, analysis of customer purchase sequences, web click streams, and so forth.

The task of discovering the set of all frequent sequences in large databases is challenging as the search space is extremely large. Different strategies have been proposed so far, among which *SPADE*, exploiting a vertical database format [9], FreeSpan and PrefixSpan, based on projected pattern growth [3,5] are the most popular ones. These strategies show good performances in databases containing short frequent sequences or when the support threshold is not very low. Unfortunately, when long sequences are mined, or when a very low support threshold is used, the performance of such algorithms decreases dramatically and the number of frequent patterns increases sharply, resulting in too many meaningless and redundant patterns. Even worse, sometimes it is impossible to complete the algorithm execution due to a memory overflow.

One of the most interesting proposals to solve both problems are so called *closed sequences* [8], based on the same notion for regular frequent closed itemsets, as introduced by Pasquier et al. [4]. A frequent sequence is said to be closed,

[⋆] Corresponding author: agomariz@um.es. This work is partially financed by a PhD grant from the Seneca Foundation (Regional Agency for Science and Technology of the Region de Murcia), and by the Spanish Office for Science and Innovation through project TIN2009-14372-C03-01 and PlanE, and the European Union by means of the European Regional Development Fund (ERDF, FEDER).

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 50–61, 2013.

if there no exists a supersequence with the same support in the database. The final collection of closed sequences provides a much more simplified output, still keeping all the information about the frequency of each of the sequences. Some algorithms have been developed to find the complete set of closed sequences, where most of them are based on the Pattern Growh strategy [8,7].

In this paper, we propose a new algorithm, called ClaSP (Closed Sequential Patterns algorithm) which exploits several efficient search space pruning methods. Depending on the properties of the database, we argue about the desirability of using the vertical database format as compared to pattern growth techniques. We also show the suitability of the vertical database format in obtaining the frequent closed sequence set, and how, under some database configurations, a standard vertical database format algorithm can already be faster than Pattern Growth algorithms for closed sequences, by only adding a simple post-processing step. Experiments on both synthetic and real datasets show that ClaSP generates the same complete closed sequences as CloSpan [8] but has much better performance figures.

The remaining of the paper is organized as follows. Section 2 introduces the preliminary concepts of frequent closed sequential pattern mining and the notation used in the paper. In Section 3, we present the most relevant related works. In Section 4, the pruning methods and ClaSP algorithm are presented. The performance study is presented in section 5 and, finally, we state our conclusions in section 6.

2 Problem Setting

Let $\mathcal{I}$ be a set of items. A set $X = \{e_1, e_2, \ldots, e_k\} \subseteq \mathcal{I}$ is called an itemset or k-itemsets if it contains k items. For simplicity, from now on we denote an itemset I as a concatenation of items between brackets. So, $I_1 = (ab)$ and $I_2 = (bc)$ are both two 2-itemsets. Also, without loss of generality, we assume the items in every itemset are represented in a lexicographic order.

A sequence s is a tuple $s = \langle I_1 I_2 \ldots I_n \rangle$ with $I_i \in \mathcal{I}$, and $\forall i : 1 \leq i \leq n$. We denote the size of a sequence $|s|$ as the number of itemsets in that sequence. We denote the length of a sequence $(l = \sum_{i=1}^{n} |I_i|)$ as the number of items in it, and every sequence with k items is called a k-sequence. For instance, the sequence $\alpha = \langle (ab)(bc) \rangle$ is a 4-sequence with a size of 2 itemsets.

We say $\alpha = \langle I_{a_1} I_{a_2} \ldots I_{a_n} \rangle$ is a subsequence of another sequence $\beta = \langle I_{b_1} I_{b_2} \ldots I_{b_m} \rangle$ (or β is a supersequence of α), denoted as $\alpha \preceq \beta$, if there exist integers $1 \leq j_1 < j_2 < \ldots < j_n \leq m$ such that $I_{a_1} \subseteq I_{b_{j_1}}, I_{a_2} \subseteq I_{b_{j_2}}, \ldots, I_{a_n} \subseteq I_{b_{j_n}}$. For instance, $\langle (b)(c) \rangle$ is a subsequence of $\langle (ab)(bc) \rangle$, since $(b) \subseteq (ab)$ and $(c) \subseteq (bc)$ and the order in the itemsets is preserved. Furthermore, the sequence $\langle (b)(c) \rangle$ is not a subsequence of $\langle (abc) \rangle$.

In the rest of the work, we use the terms pattern and sequence interchangeably.

An input sequence is is a tuple $is = \langle id, s \rangle$ with $id \in \mathbb{N}$ and s is a sequence. We call id the identifier of the input sequence. We say that an input sequence is contains another sequence α, if $\alpha \preceq s$.

Table 1. A Sample Sequence Database

Sequence Id.	Sequence
1	$\langle(a)(ab)(bc)\rangle$
2	$\langle(a)(abc)\rangle$
3	$\langle(d)(a)(ab)(bc)\rangle$
4	$\langle(d)(ad)\rangle$

A sequence database D is collection of input sequences $D = \langle s_1 s_2 \ldots s_n\rangle$, incrementally ordered by the identifier of the contained sequences. In table 1 we show a sample input database D with four input sequences.

Definition 1. The *support* (or *frequency*) of a sequence, denoted as $\sigma(\alpha, D)$, is the total number of input sequences in the input database D that contain α. A pattern or sequence is called *frequent* if it occurs at least a given user specified threshold *min_sup*, called the minimum support. $\mathcal{FS}$ is the whole collection of frequent sequences. The problem of frequent sequence mining is now to find $\mathcal{FS}$ in a given input database, for a given minimum support threshold.

Given a sequence $\alpha = \langle I_1 I_2 \ldots I_n\rangle$ and an item e_i, we define the *s-extension* α' as the super-sequence of α, extending it with a new itemset containing a single item e_i, $\alpha' = \langle I_1 I_2 \ldots I_n I_{n+1}\rangle$, $I_{n+1} = (e_i)$. We define the *i-extension* of α if the last itemset I'_n of $\alpha' = \langle I_1 I_2 \ldots I'_n\rangle$ satisfies $(I'_n = I_n \cup e_i)$. That is, the item e_i is added to I_n. For instance, given the sequence $\alpha = \langle(a)(b)\rangle$ and an item $c \in \mathcal{I}$, the sequence $\beta = \langle(a)(b)(c)\rangle$ is an s-extension and $\gamma = \langle(a)(bc)\rangle$ is an i-extension.

Given two sequences β and γ such that both are s-extensions (or i-extensions) of a common prefix α, with items e_i and e_j respectively, we say β precedes γ, $\beta < \gamma$, if $e_i <_{lex} e_j$ in a lexicographic order. If, on the contrary, one of them is an s-extension and the other one is i-extension, the s-extension always precedes the i-extension.

Definition 2. If a frequent sequence α does not have another supersequence with the same support, we say that α is a *closed* sequence. Otherwise, if a frequent sequence β has a super-sequence γ with exactly the same support, we say that β is a non-closed sequence and γ *absorbs* β. The whole set of frequent closed sequences is denoted by $\mathcal{FCS}$. More formally, $\alpha \in \mathcal{FCS}$ if $\forall \beta \in \mathcal{FS}, \alpha \preceq \beta, \sigma(\alpha, D) \neq \sigma(\beta, D)$. The problem of closed sequence mining is now to find $\mathcal{FCS}$ in a given input database, for a given minimum support threshold.

Clearly, the collection of frequent closed sequences is smaller than the collection of all frequent sequences.

Example 1. *In our sample database, shown in table 1, for a support min_sup =* 2*, we find* $|\mathcal{FCS}| = 5$ *frequent closed sequences,* $\mathcal{FCS} = \{\langle(a)\rangle, \langle(d)(a)\rangle, \langle(a)(ab)\rangle, \langle(a)(bc)\rangle, \langle(a)(ab)(bc)\rangle\}$*, while the corresponding* $\mathcal{FS}$ *has* 27 *frequent sequences.*

3 Related works

Looking for frequent sequences in sequence databases was first proposed by Agrawal and Srikant [1,6]. Their algorithms (apriori-based) consist of executing a continuous loop of a candidate generation phase followed by a support checking phase. Two main drawbacks appear in those algorithms: 1) they need to do several scans of the database to check the support of the candidates; and 2) a breath-first search is needed for the candidate generation, leading to high memory consumption.

Later, two other strategies were proposed: 1) depth-first search based on a vertical database format [9] and 2) projected pattern growth [3,5]. The vertical database format strategy was created by Zaki in the Spade algorithm [9] which is capable of obtaining the frequent sequences without making several scans of the input database. His algorithm allows the decomposing of the original search tree in independent problems that can be solved in parallel in a depth-first search (DFS), thus enabling the processing of big databases.

The Pattern growth strategy was introduced by Han et al. [3] and it consists in algorithms that obtain the whole frequent sequence set by mean of techniques based on the so called projected pattern growth. The most representative algorithm in this strategy is PrefixSpan [5]. PrefixSpan defines a projected database as the set of suffixes with respect to a given prefix sequence. After projecting by a sequence, new frequent items are identified. This process is applied in a recursive manner by means of DFS, identifying the new frequent sequences as the concatenation of the prefix sequence with the frequent items that are found.

Prefixspan shows good performance and scales well in memory, especially with sparse databases or when databases mainly consist of small itemsets. However, when we deal with large dense databases that have large itemsets, the performance of Prefixpan is worse than that of Spade. In order to show this issue, we have conducted several tests. In a sequential database, several important properties have an influence on the algorithms execution, some of which are shown in Table 2.

Table 2. Parameters for IBM Quest Data Generator

Abbr.	Meaning
D	Number of sequences (in 000s)
C	Average itemset in a sequence
T	Average items in a itemset
N	Number of different items (in 000s)
S	Average itemsets in maximal sequences
I	Average items in maximal sequences

We define the database density as the quotient $\delta = \frac{T}{N}$. We have used the well-known data generator provided by IBM to run Spade and PrefixSpan under different configurations. In Figures 1 and 2 we can observe the behaviour of both Spade and Prefixspan when we vary the density and the number of itemsets.

In figure 1 we show the running time of the algorithms with a different number of items (100, 500 and 2500 items) with a constant $T = 20$ value. Since for a database, the density grows either if the numerator increases or the denominator decreases, the figures have been obtained just varying the denominator. Besides, figure 2 depicts the behaviour of the algorithms when the number of itemsets is changed between values of $C \in \{10, 40, 80\}$ while we keep the density constant ($\delta = \frac{20}{2500}$). The higher δ the more dense the database. We can see that PrefixSpan shows good results when both density and the number of itemsets are low, but when a database is denser and parameter C grows, we notice how Spade outperforms Prefixspan.

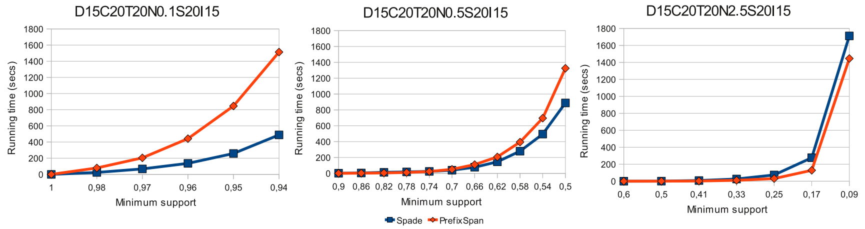


Fig. 1. Behaviour of Spade and PrefixSpan when density changes (in the number of items)

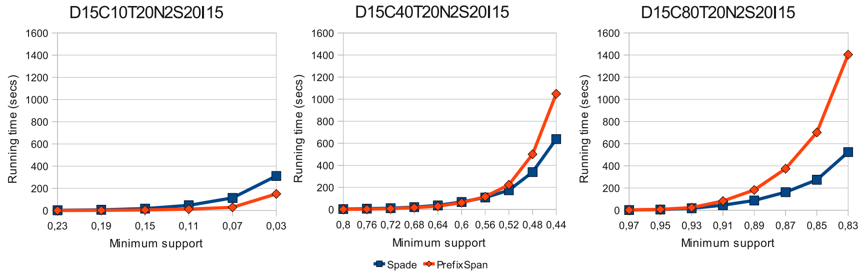


Fig. 2. Behaviour of Spade and PrefixSpan when the number of itemsets changes

For mining closed sequences, there exist two approaches: 1) run any algorithm for mining all frequent sequences and execute a post-processing step to filter out the set of closed sequences, or 2) obtain the set of closed sequences by gradually discarding the non-closed ones. Some algorithms have been developed to find the complete set of closed sequences. The most important algorithms developed so far, are CloSpan [8] and Bide [7], both derived from Prefixspan. While CloSpan uses a prefix tree to store the sequences and uses two methods to prune non-frequent sequences, Bide executes some checking steps in the original database that allows it to avoid maintaining the sequence tree in memory. However, to the best of our knowledge, there exist no algorithms for closed sequence mining based on the vertical database format as is presented here.

4 ClaSP: Algorithm and Implementation

In this section, we formulate and explain every step of our ClaSP algorithm. ClaSP has two main phases: The first one generates a subset of $\mathcal{FS}$ (and superset of $\mathcal{FCS}$) called Frequent Closed Candidates ($\mathcal{FCC}$), that is kept in main memory; and the second step executes a post-pruning phase to eliminate from $\mathcal{FCC}$ all non-closed sequences to finally obtain exactly $\mathcal{FCS}$.

Algorithm 1. ClaSP

```
1: $\mathcal{F}_1$ = {frequent 1-sequences}
2: $\mathcal{FCC} = \emptyset$, $\mathcal{FCS} = \emptyset$
3: for all  i $\in \mathcal{F}_1$  do
4:     $\mathcal{F}_{ie}$ = {frequent 1-sequences greater than i}
5:     $\mathcal{FCC}_i$=DFS-PRUNING(i,$\mathcal{F}_1$,$\mathcal{F}_{ie}$)
6:     $\mathcal{FCC} = \mathcal{FCC} \cup \mathcal{FCC}_i$
7: end for
8: $\mathcal{FCS}$ = N-ClosedStep($\mathcal{FCC}$)
Ensure: The final closed frequent pattern set $\mathcal{FCS}$
```

Algorithm 1, *ClaSP*, shows the pseudocode corresponding to the two main steps. It first finds every frequent 1-sequence, and after that, for all of frequent 1-sequences, the method *DFS-Pruning* is called recursively to explore the corresponding subtree (by doing a depth-first search). $\mathcal{FCC}$ is obtained when this process is done for all of the frequent 1-sequences and, finally, the algorithm ends removing the non-closed sequences that appear in $\mathcal{FCC}$.

Algorithm 2, *DFS-Pruning*, executes recursively both the candidate generation (by means of i-extensions and s-extensions) and the support checking, returning a part of $\mathcal{FCC}$ relative to the pattern p taken as parameter. The method takes as parameters two sets with the candidate items to do s-extensions and i-extensions respectively ($\mathcal{S}_n$ and $\mathcal{I}_n$ sets). The algorithm first checks if the current pattern p can be discarded, by using the method *checkAvoidable* (this algorithm is explained later in algorithm 5). Lines 4-9 perform all the s-extensions for the pattern p and keep in $\mathcal{S}_{temp}$ the items which make frequent extensions. In line 10, the method *ExpSiblings* (algorithm 4) is called, and there, *DFS-Pruning* is executed for each new frequent s-extensions. Lines 11-16 and 17 perform the same steps, with i-extensions. Finally, in line 19, the complete frequent patterns set (with a prefix p) is returned.

To store the patterns in memory, we use a lexicographic sequence tree. The elements in the tree are sorted by a lexicographic order according to extension comparisons (see section 2). In figure 3 we show the associated sequence tree for $\mathcal{FS}$ in our example and we denote an s-extension with a line, and an i-extension with a dotted line. This tree is traversed by algorithms 1, 2 and 4, using a depth-first traversal.

There are two main different changes added in ClaSP with respect to SPADE: (1) the step to check if the subtree of a pattern can be skipped (line 3 of algorithm 2), and (2) the step where the remaining non-closed patterns are eliminated (line 6 of algorithm 1).

Algorithm 2. DFS-Pruning(p, $\mathcal{S}_n$, $\mathcal{I}_n$)

```
Require: Current frequent pattern $p = (s_1, s_2, \dots, s_n)$, set of items for s-extension $\mathcal{S}_n$, set of items for i-extension $\mathcal{I}_n$
$\mathcal{S}_{temp} = \emptyset$, $\mathcal{I}_{temp} = \emptyset$
$\mathcal{F}_i = \emptyset$, $\mathcal{P}_s = \emptyset$, $\mathcal{P}_i = \emptyset$
if (not checkAvoidable( $p$, $\mathcal{I}(\mathcal{D}_p)$ )) then
    for all i $\in \mathcal{S}_n$ do
        if ($p' = (s_1, s_2, \dots, s_n, \{i\})$ is frequent) then
            $\mathcal{S}_{temp} = \mathcal{S}_{temp} \cup \{i\}$
            $\mathcal{P}_s = \mathcal{P}_s \cup \{p'\}$
        end if
    end for
    $\mathcal{F}_i = \mathcal{F}_i \cup \mathcal{P}_s \cup$ ExpSiblings($\mathcal{P}_s$,$\mathcal{S}_{temp}$,$\mathcal{S}_{temp}$)
    for all i $\in \mathcal{I}_n$ do
        if ($p' = (s_1, s_2, \dots, s_n \cup \{i\})$ is frequent) then
            $\mathcal{I}_{temp} = \mathcal{I}_{temp} \cup \{i\}$
            $\mathcal{P}_i = \mathcal{P}_i \cup \{p'\}$
        end if
    end for
    $\mathcal{F}_i = \mathcal{F}_i \cup \mathcal{P}_i \cup$ ExpSiblings($\mathcal{P}_s$,$\mathcal{S}_{temp}$,$\mathcal{I}_{temp}$)
end if
return $\mathcal{F}_i$
Ensure: Frequent pattern set $\mathcal{F}_i$ of this node and its siblings
```

Algorithm 3. N-ClosedStep($\mathcal{FCC}$)

```
Require: A frequent closed candidates set $\mathcal{FCC}$
$\mathcal{FCS} = \emptyset$
A hash table $H$ is created
for all $p \in \mathcal{FCC}$ do
    Add a new entry $\langle \mathcal{T}(\mathcal{D}_p), p \rangle$ in $H$
end for
for all entry $e \in H$ do
    for all $p_i \in e$ do
        for all $p_j \in e$, $j > i$ do
            if ($p_i.support() = p_j.support()$) then
                if ($p_i \preceq p_j$) then
                    Remove $p_i$ from $e$
                else
                    if ($p_j \preceq p_i$) then
                        Remove $p_j$ from $e$
                    end if
                end if
            end if
        end for
    end for
    $\mathcal{FC}_e$= all patterns $p \in e$
    $\mathcal{FCS} = \mathcal{FCS} \cup \mathcal{FC}_e$
end for
return $\mathcal{FCS}$
Ensure: The final closed frequent pattern set $\mathcal{FCS}$
```

Algorithm 4. ExpSiblings($\mathcal{P}$, $\mathcal{S}_s$, $\mathcal{S}_i$)

```
Require: The pattern set $\mathcal{P}$ which contains all the patterns whose children are going to be explore, the set of valid items $\mathcal{S}_s$ which generate the $\mathcal{P}$ set by means of s-extensions, the set of valid items $\mathcal{S}_i$ which generate the $\mathcal{P}$ set by means of i-extensions
$\mathcal{F}_s = \emptyset$
for all p $\in \mathcal{P}$ do
    $\mathcal{I}$ = Elements in $\mathcal{S}_i$ greater than the last item $e_i$ in $p$
    $\mathcal{F}_s = \mathcal{F}_s \cup$ DFS-Pruning($p$,$\mathcal{S}_s$,$\mathcal{I}$)
end for
return $\mathcal{F}_s$
Ensure: Frequent pattern set $\mathcal{F}_s$ for all of the patterns' siblings
```

Algorithm 5. CheckAvoidable(p, k)

```
Require: a frequent pattern $p$, its hash-key $k$, hash table $H$
$\mathcal{M}_k$ = Entries in the hash table with key $k$

if ($\mathcal{M}_k = \emptyset$) then
    Insert a new entry $\langle k,p \rangle$ in $H$
else
    for all pair $m \in \mathcal{M}_k$ do
        $p' = m.value()$
        if ($p$.support()=$p'.support()$) then
            //Backward sub-pattern
            if ($p \preceq p'$) then
                $p$ has the same descendants as $p'$, so $p$ points to $p'$ descendants

                return true
            else
                //Backward super-pattern
                if ($p' \preceq p$) then
                    $p$ has the same descendants as $p'$, so $p$ points to $p'$ descendants
                    Remove the current entry $\langle k, p' \rangle$ from $H$
                end if
            end if
        end if
    end for
    //Backward super-pattern executed?
    if (Pruning method 2 has been accomplished) then
        Add a new entry $\langle k, p \rangle$ in $H$
        return true
    end if
end if
//k does not exist in the hash table or it does exist but the present patterns are not related with p
Add a new entry $\langle k, p \rangle$ in $H$
return false
Ensure: It answers if the generation of $p$ can be avoided
```

To prune the space search, ClaSP used the method *CheckAvoidable* that is inspired on the pruning methods used in CloSpan. This method tries to find those patterns $p = \langle \alpha\, e_j \rangle$ and $p' = \langle \alpha\, e_i\, e_j \rangle$, such that, all of the appearances of p are in those of p', i.e., if every time we find a sequence α followed by an item e_j, there exists an item e_i between them, then we can safely avoid the exploration of the subtree associated to the pattern p. In order to find this kind of patterns, we define two numbers: 1) $l(s, p)$, is the size of all the suffixes with respect to p in sequence s, and 2) $\mathcal{I}(\mathcal{D}_p) = \sum_{i=1}^{n} l(s_i, p)$, the total number of remaining items with respect to p for the database $\mathcal{D}$, i.e. the addition of all of $l(s, p)$ for every sequence in the database. Using $\mathcal{I}(\mathcal{D})$ and the subsequence checking operation, in algorithm 5, ClaSP checks the equivalence between the $\mathcal{I}(\mathcal{D})$ values for two patterns: Given two sequences, s and s', such that $s \preceq s'$, if $\mathcal{I}(\mathcal{D}_s) = \mathcal{I}(\mathcal{D}_{s'})$, we can deduce that the support for all of their descendants is just the same.

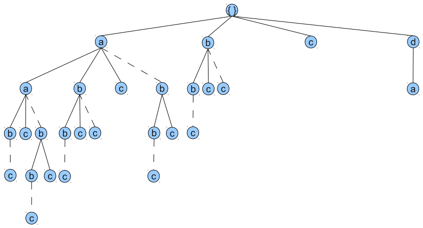


Fig. 3. Whole lexicographic sequence tree for our thorough example

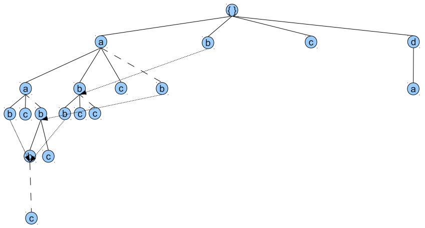


Fig. 4. Whole lexicographic sequence tree after processing ClaSP algorithm

In algorithm 5, the pruning phase is implemented by two methods: 1) Backward sub-pattern checking and 2) Backward super-pattern checking. The first one (lines 8-10) occurs when we find a pattern which is a subsequence of a pattern previously found with the same $\mathcal{I}(\mathcal{D})$ value. In that case, we can avoid exploring this new branch in the tree for this new pattern. The second method (lines 12-16 and 20-24) is the opposite situation and it occurs when we find a pattern that is a super-sequence of another pattern previously found with the same $\mathcal{I}(\mathcal{D})$ value. In this case we can transplant the descendants of the previous pattern to the node of this new pattern.

In figure 4 we show the ClaSP search tree w.r.t. our example without all pruned nodes. In our implementation, to store the relevant branches, we define a hash function with $\mathcal{I}(\mathcal{D})$ value as key and the pattern (i.e. the node in the tree for that pattern) as value ($\langle \mathcal{I}(\mathcal{D}_p), p \rangle$). We use a global hash table and, when we find a sequence p, if the backward sub-pattern condition is accomplished, we do not put the pair $\langle \mathcal{I}(\mathcal{D}_p), p \rangle$, whereas, if the backward super-pattern condition is true, we replace all the previous pairs $\langle \mathcal{I}(\mathcal{D}_{p'}), p' \rangle$ (s.t. $p' \preceq p$) with the new one $\langle \mathcal{I}(\mathcal{D}_p), p \rangle$. If instead we do not find any pattern with the same $\mathcal{I}(\mathcal{D})$ value of the pattern p, or those patterns with the same value are not related with p by means of the subsequence operation, we put the pair $\langle \mathcal{I}(\mathcal{D}_p), p \rangle$ to the global hash table (line 28). Note that when one of the two pruning conditions is true, we also need to check if the support for s and s' is the same since two $\mathcal{I}(\mathcal{D}_p)$ and $\mathcal{I}(\mathcal{D}_{p'})$ values can be equal but they do not necessarily have the same support.

We also need to consider all of the $\mathcal{I}(\mathcal{D})_s$ for every appearance in a sequence. For instance, in our example shown in table 1, regarding the three first sequences, if we consider just the first $\mathcal{I}(\mathcal{D}_s)$ for the first appearance, if we have the pattern $\langle(a)(b)\rangle$ in our example, we deduce that $\mathcal{I}(\mathcal{D}_{\langle(a)(ab)\rangle}) = \mathcal{I}(\mathcal{D}_{\langle(a)(b)\rangle})$ (both with value $\mathcal{I}(\mathcal{D}) = 5$), so we can avoid generating the descendants of $\langle(a)(b)\rangle$ because the are the same as in $\langle(a)(ab)\rangle$. However, we can check that $\langle(a)(bc)\rangle$ is frequent (with support 3), whereas $\langle(a)(abc)\rangle$ is not (support 1). This forces us to count in $\mathcal{I}(\mathcal{D}_s)$, all the number of items after every appearance.

Finally, regarding the non-closed pattern elimination (algorithm 3), the process consists of using a hash function with the support of a pattern as key and the pattern itself as value. If two patterns have the same support we check if one contains the other, and if this condition is satisfied, we remove the shorter pattern. Since the support value as key provoke a high number of collisions in the hash table (implemented with closed addressing), we use a $\mathcal{T}(\mathcal{D}_p) = \sum_{i=1}^{n} id(s_i)$ value, defined as the sum of all sequence ids where a pattern p appears. However, as the equivalence of $\mathcal{T}(\mathcal{D}_p)$ does not imply the equivalence of support, after checking that two patterns have the same $\mathcal{T}(\mathcal{D}_p)$ value, those patterns have to have the same support to remove one of them.

5 Performance Study

We exhaustively experimented on both synthetic and real world datasets. To generate the synthetic data, we have used the IBM data generator mentioned above (see section 3). In all our experiments we compare the performance of three algorithms: CloSpan, ClaSP and Spade. For the last algorithm we add the same non-closed candidate elimination phase which is used in ClaSP to obtain $\mathcal{FCS}$.

All experiments are done on a 4-cores of 2.4GHZ Intel Xeon, running Linux Ubuntu 10.04 Server edition. All the three algorithms are implemented in Java 6 SE with a Java Virtual Machine of 16GB of main memory.

Figure 5 shows the number of patterns and performance for the dataset D5C10T5N5S6I4 (-rept 1 -seq.npats 2000 -lit.npats 5000). Figure 5(a) shows the number of frequent patterns, the patterns processed by ClaSP, and the number of closed patterns. We can see how there is approximately an order of difference between these numbers, i.e. for every 100 frequent patterns we process around 10 patterns by ClaSP, and approximately only 1 of these 10 patterns is closed. Figure 5(b) shows the running time. ClaSP clearly outperforms both Spade and Clospan. For very low support (below 0.013), we have problems with the execution of Spade due to the space in memory taken for the algorithm.

Figure 6 shows a dataset with larger parameters of C, T and a lower N. This database is denser than the database above and in figure 6(a) we can observe that the difference between frequent patterns and processed patterns is not so big. Therefore, the pruning method checkAvoidable is not so effective and ClaSP

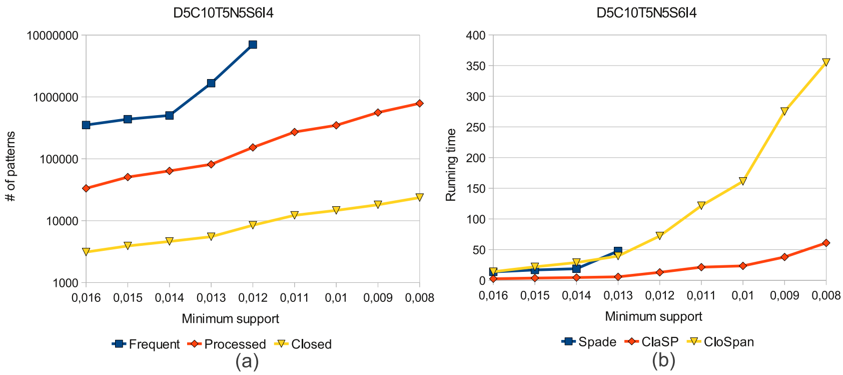


Fig. 5. Varying Support for Dataset D5C10T5N5S6I4(-seq.npats 2000 -lit.npats 5000)

and CloSpan are closer to the normal behavior of Spade and PrefixSpan, as is shown in figure 6(b). The results show that both ClaSP and Spade are much faster than Clospan.

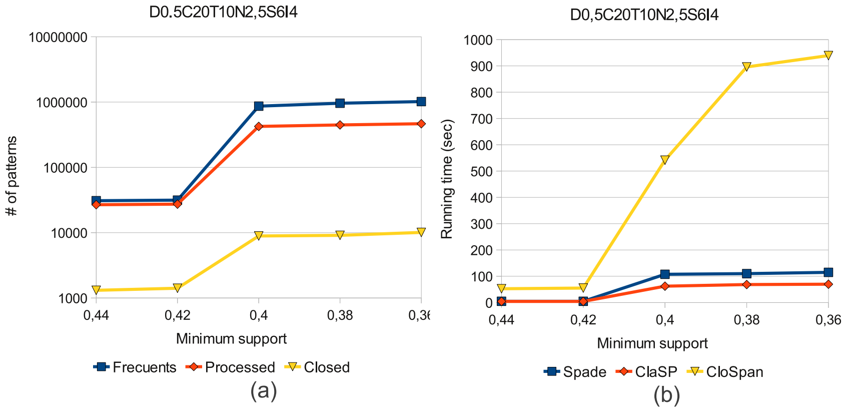


Fig. 6. Varying Support for Dataset D0.5C20T10N2.5S6I4(-seq.npats 2000 -lit.npats 5000)

Finally we test our algorithm with the gazelle dataset. This dataset comes from click-stream data from gazelle.com, which no longer exists. The dataset was once used in KDDCup-2000 competition and, basically, it includes a set of page views (each page contains a specific product information) in a legwear and legcare website. Each session contains page views done by a customer over a short period. Product pages viewed in one session are considered as an itemset, and different sessions for one user is considered as a sequence. The database contains 1423 different products and assortments which are viewed by 29369 different users. There are 29369 sequences, 35722 sessions (itemsets), and 87546 page views (items). The average number of sessions in a sequence is around 1. The average number of pageviews in a session is 2. The largest session contains 342 views, the longest sequence has 140 sessions, and the largest sequence contains 651 page views. Figure 7 shows the runtime with several support (from 0.03% to 0.015%). We compare the runtime behavior for both ClaSP and CloSpan and we can see how ClaSP outperforms CloSpan.

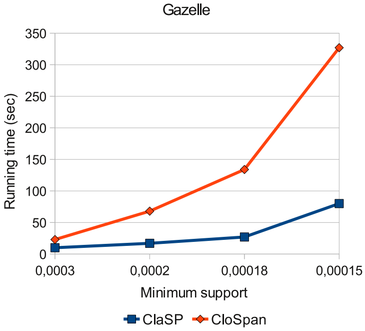


Fig. 7. Varying Support for Dataset Gazelle click stream

All the experiments show that ClaSP outperforms CloSpan, even if databases are sparse. This is because of, in very sparse databases, a high number of patterns are found only with extremely low support. Therefore, the lower the support is, the more patterns are found and the more items are chosen to create patterns. In this point, CloSpan, as PrefixSpan, is penalized since the algorithm has to projects several times the same item in the same sequence, having a worse time with respect to ClaSP. Besides, in those algorithms where Spade is better than Prefixspan, ClaSP is also faster than CloSpan.

6 Conclusions

In this paper, we study the principles for mining closed frequent sequential patterns and we compare the two main existing strategies to find frequent patterns. We show the benefits of using the vertical database format strategy against pattern growth algorithms, especially when facing dense datasets. Then, we introduced a new algorithm called ClaSP to mine frequent closed sequences. This algorithm is inspired on the Spade algorithm using a vertical database format strategy and uses a heuristic to prune non-closed sequences inspired by the CloSpan algorithm. To the best of our knowledge, this is the first work based on the vertical database strategy to solve the closed sequential pattern mining problem. In all our tests ClaSP outperforms CloSpan, and even, under certain datasets configuration, Spade also outperforms CloSpan when the non-closed elimination phase is executed after it.

References

1. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the Eleventh International Conference on Data Engineering, pp. 3–14. IEEE (1995)
2. Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 429–435. ACM (2002)
3. Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q.: FreeSpan: frequent pattern-projected sequential pattern mining. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 355–359 (2000)
4. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)
5. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.: Mining sequential patterns by pattern-growth: the PrefixSpan approach. IEEE Transactions on Knowledge and Data Engineering 16(11), 1424–1440 (2004)
6. Srikant, R., Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements. In: Apers, P.M.G., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 3–17. Springer, Heidelberg (1996)
7. Wang, J., Han, J., Li, C.: Frequent closed sequence mining without candidate maintenance. IEEE Transactions on Knowledge and Data Engineering 19(8), 1042–1056 (2007)
8. Yan, X., Han, J., Afshar, R.: CloSpan: Mining closed sequential patterns in large datasets. In: Proceedings of SIAM International Conference on Data Mining, pp. 166–177 (2003)
9. Zaki, M.J.: SPADE: An efficient algorithm for mining frequent sequences. Machine Learning 42(1), 31–60 (2001)

Efficient Mining of Contrast Patterns on Large Scale Imbalanced Real-Life Data

Jinjiu Li, Can Wang, Wei Wei, Mu Li, and Chunming Liu

Advanced Analytics Institute, University of Technology Sydney, Australia
{fJinjiu.Li,Can.Wang,Wei.Wei,Mu.Li,Chunming.Liug}@student.uts.edu.au

Abstract. Contrast pattern mining has been studied intensively for its strong discriminative capability. However, the state-of-the-art methods rarely consider the class imbalanced problem, which has been proved to be a big challenge in mining large scale data. This paper introduces a novel pattern, i.e. converging pattern, which refers to the itemsets whose supports contrast sharply from the minority class to the majority one. A novel algorithm, ConvergMiner, which adopts T*-tree and branch bound pruning strategies to mine converging patterns efficiently, is proposed. Substantial experiments in online banking fraud detection show that the ConvergMiner greatly outperforms the existing cost-sensitive classification methods in terms of predicative accuracy. In particular, the efficiency improves with the increase of data imbalance.

Keywords: Contrast Pattern, Class Imbalance, Fraud Detection.

1 Introduction

Contrast patterns [11] are the itemsets showing the discrimination between two classes in a data set, and they are very useful to detect anomalies. It has been widely studied in terms of emerging pattern [3], jumping emerging pattern [5], and contrast capturing method [11]. In addition, classifiers built by the contrast patterns, such as CAEP [4], are shown to outperform most of the existing methods (e.g. C4.5 [12], CMAR [8]) on accuracy. However, the existing contrast pattern mining methods do not consider the issue of class imbalance (i.e. the data distribution is extremely imbalanced among different classes). Such characteristics is commonly observed in the real-life applications.

Table 1 gives an example of four fraudulent patterns that appear in online banking with their False Positive Rates (FPR). There are 1,000,000 genuine transactions (Genuine for short) and 1,000 fraud transactions (Fraud for short) in the transaction set that involves 112 attributes. In order to catch frauds effectively, two criteria are proposed: $FPR \leq 0.8$ and the support in Genuine is no larger than 0.0001. We can see that only p_4 is qualified. In contrast, patterns p_1, p_2 and p_3 have rather high FPR and larger supports in the Genuine, so they can not be applied at an acceptable cost of the investigation fee spent on post-inspection.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 62–73, 2013.

Table 1. Patterns for Fraud Detection

Rules	$FPR = \lvert Genuine\rvert / \lvert Genuine + Fraud\rvert$	Frequency	
		Genuine (size=1M)	Fraud (size=1K)
$p_1 \rightarrow Fraud$	0.96	5000	200
$p_2 \rightarrow Fraud$	0.997	6000	20
$p_3 \rightarrow Fraud$	0.968	3000	100
$p_4 \rightarrow Fraud$	0.769	100	30

The fraudsters invariably try to disguise their behaviors maliciously and bypass the detection system, some typical features are then identified: (i) The itemsets that frequently occur in the Fraud are also usually frequent in the Genuine (such as p_1, p_2, p_3). (ii) The itemsets with a strong discriminative power generally are rarely seen ($support \leq 0.0001$) in Genuine (e.g. p_4). Therefore, there are three main challenges in mining contrast patterns to detect the Fraud. (1) A strict constraint on FPR demands an extremely small support in the Genuine (as shown in Table 1). Given a small support threshold, a huge number of itemsets will be generated, especially when plenty of attributes are involved. (2) Mining contrast patterns among the large scale data is computationally expensive. (3) Selecting the optimal pattern set for classification is also with a high computational complexity. It is common to see that the serious overlapping among patterns, on which the classifiers are built, negatively impact the overall performance in catching the Fraud.

However, the existing Emerging pattern miner MDB-LL_{border} [3] dose not consider the imbalance issue. By applying Max-Miner [1] to mine long patterns with a small support threshold in the Genuine, MDB-LL_{border} consumes a overwhelming memory and computational time due to challenges (1) and (2). Though jumping emerging patterns [5] can be quickly captured, they are rarely observed in the fraud detection scenario according to the feature (i). CAEP [4] suffers from the serious pattern overlapping confronting with challenge (3).

In this paper, we propose a novel approach to mine contrast patterns in the large scale imbalanced data by exploring the converging patterns, which effectively handle the above challenges. The main contributions are as follows.

- Define the converging patterns, a novel type of contrast patterns that significantly differentiate itemsets in the class imbalanced data; and propose an effective algorithm, called ConvergMiner, to mine the converging patterns.
- Introduce a series of branch bound pruning strategies to reduce the computational complexity; present a set of operators and theorems for border substraction to cut the computational time and memory cost; and design a new index structure T*-tree to support the fast checking of candidate patterns.
- Perform extensive experiments to evaluate the effectiveness of our approach and strategies against the state-of-art methods in terms of accuracy, convergence sensitivity and scalability.

The rest of the paper is organized as follows: Section 2 reviews the related work. Section 3 defines the problems and terminologies. Section 4 presents the approach of pattern border operation. Section 5 introduces two refining strategies, T*tree and border splitting. Section 6 proposes the algorithm of converging pattern mining. Section 7 proposes the pattern selection and scoring methods for prediction. Section 8 shows the evaluation. We conclude in Section 9.

2 Related Work

Several approaches have been proposed to mine the contrast patterns [11]. Emerging Patterns (EPs), firstly introduced in [3], uses growth rate to measure the support contrast of an itemset. Disjunctive emerging pattern [11] is a variant of EPs to discover the contrast patterns in a high dimensional data set. The work in [5] extends the concept of EPs, and introduces the Jumping Emerging Patterns (JEPs) whose supports increase abruptly from zero in one data set to non-zero in another one. According to [5,4], classifiers built by the EPs or JEPs outperform most methods (e.g. C4.5 [12], CMAR [8]) on accuracy.

All the above methods are proposed for the class balanced data. However, more researchers pay attention to the class imbalanced problems [13,10], which widely exit in the real world and greatly challenge the classic data mining algorithms. Tremendous research efforts have been made on the class-imbalanced data, for instance, Cost-Sensitive Neural Network [10], Ad-Cost [13], Cost-SVM [9], etc. None of them addresses the mining of contrast patterns in the class imbalanced data set, while we focus on solving this problem.

3 Problem Statement

Let $I = \{i_1, i_2, ..., i_m\}$ be a set of items, an itemset X is a subset of I. A transaction T is an itemset X whose number of elements is fixed according to the number of attributes in the data set. A data set D is a set of T. Suppose there are two classes denoted as C_t (the target class, e.g. Fraud) and C_b (the background class, e.g. Genuine), and two data sets D_t and D_b in D correspond to the respective classes C_t and C_b. Transactions in D_b are labeled as C_b, and

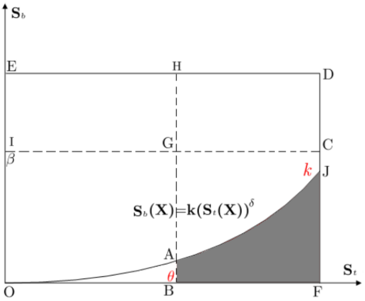


Fig. 1. An example of converging patterns

those in D_t are marked as C_t. $S_b(X)$ and $S_t(X)$ denote the supports of itemset X in D_b and D_t, respectively.

Definition 1. ***Converging Patterns*** *(CPs for short) are composed of the itemsets X that satisfy the following condition: $CPs = \{X|S_b(X) \leq k(S_t(X))^\delta$, $S_t(X) \geq \theta\}$, and the* ***Contrast Rate*** *of CPs is defined as $F(X) = (S_t(X))^\delta / S_b(X)$, where $k > 0$ is the contrast coefficient, $\delta > 0$ is the converging exponent, and $\theta > 0$ is the threshold of the minimal support in D_t.*

As the above definition indicates, CPs are controlled by k and δ. The larger the δ, the higher the contrast rate of the generated converging patterns. For example, in Fig. 1, itemsets are projected onto the support plane where the vertical axis stands for $S_b(X)$ and the horizontal axis denotes $S_t(X)$. According to Definition 1, the itemsets in the shadow region $\mathcal{I}_{ABFJ}$ compose CPs, denoted as $\{\mathcal{I}_{ABFJ}\}$. It is also derived by:

$$CPs = \{\mathcal{I}_{ABFJ}\} = \{\mathcal{I}_{BFDH}\} - \{\mathcal{I}_{GCDH}\} - \{\mathcal{I}_{AJCG}\}, \tag{1}$$

where $\mathcal{I}_\Box$ represents all the itemsets in region $\Box$.

The itemsets in $\mathcal{I}_{BFDH}$ can be obtained by the Max-Miner [1] in D_t, i.e. $\{\mathcal{I}_{BFDH}\} = \{X|S_t(X) \geq \theta\}$. For the second term in Equation (1), we have $\{\mathcal{I}_{GCDH}\} = \{\mathcal{I}_{BFDH}\} \cap \{\mathcal{I}_{ICDE}\}$. The mining of itemsets in $\mathcal{I}_{ICDE}$ is similar to that in $\mathcal{I}_{BFDH}$. Let β=max(k, π), where π is a proper support threshold for Max-Miner to output long patterns successfully in D_b. Unlike the algorithm in [3], which applies a strict growth rate β to mine EPs and does not work in imbalanced data as mentioned in Section 1. Then the itemsets in $\mathcal{I}_{ICDE}$ are obtained as $\{\mathcal{I}_{ICDE}\} = \{X|S_b(X) \geq \beta\}$. Thus, $\{\mathcal{I}_{GCDH}\} = \{X|S_t(X) \geq \theta\} \cap \{X|S_b(X) \geq \beta\}$. Therefore, the main task of finding the itemsets in $\mathcal{I}_{BFCG}$ becomes the effective filtering of all the itemsets in $\mathcal{I}_{AJCG}$, i.e., the last term in Equation (1).

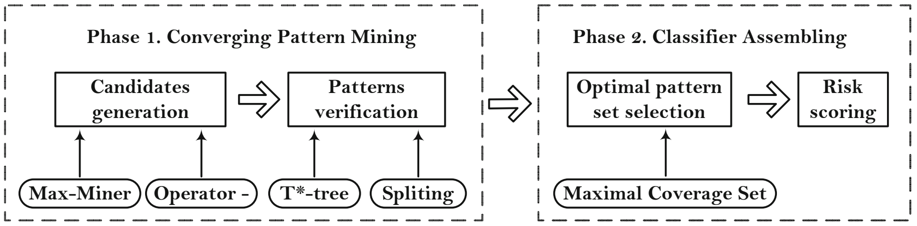


Fig. 2. Framework of building the anomaly detector powered by CPs, where oval boxes display the supporting techniques for each stage

Accordingly, the mining of CPs is fulfilled by two phases (as shown in Fig. 2): candidates generation and pattern verification (in phase 1), and predicative classifier building (in phase 2). Section 4 introduces the method to generate the candidate itemsets, section 5 provides the strategies to eliminate the redundancy, section 7 presents the techniques to select the optimal pattern set and calculate the prediction score.

4 Candidates Generation

As specified in Section 3, CPs is identified by Equation (1). In order to perform the substraction of itemsets quickly, we propose the Pattern Border to collect the itemsets and provide several theorems to support the retrieve of the candidate CPs by algebraic operations rather than traversing all elements in borders.

4.1 Pattern Border

Definition 2. *Given two itemsets $\mathcal{L}$ and $\mathcal{R}$ ($\mathcal{L} \subseteq \mathcal{R}$), the ordered interval [$\mathcal{L}, \mathcal{R}$] forms a* ***Pattern Border****, composed of a set of patterns. The collection of itemsets in* $[\mathcal{L}, \mathcal{R}]$ *is defined as:* $[\mathcal{L}, \mathcal{R}] = \{Y | \exists X \subseteq \mathcal{R}, \mathcal{L} \subseteq Y \subseteq X\}$.

Example 1. *Border* $[a, abcd]$ *contains 8 patterns:* $a, ab, ac, ad, abc, abd, acd, abcd$.

Pattern border is an important concept in CPs mining. A proper adjustment on the pattern border greatly avoids the full iteration of every candidate itemset in $\mathcal{I}_{AJCG}$, and dramatically speeds up the search of CPs. Since the smallest itemset in [$\mathcal{L}$, $\mathcal{R}$] is $\mathcal{L}$ and $\mathcal{R}$ is the largest one, an itemset $X \in [\mathcal{L}, \mathcal{R}]$ then satisfies the following conditions:

$$\max_{\mathcal{L} \subseteq X \subseteq \mathcal{R}} (S_t(X)) \leq S_t(\mathcal{L}), \min_{\mathcal{L} \subseteq X \subseteq \mathcal{R}} (S_t(X)) \leq S_t(\mathcal{R}) \tag{2}$$

Definition 3. *The* ***Upper Bound*** $F^u([\mathcal{L}, \mathcal{R}])$ *and* ***Lower Bound*** $F^l([\mathcal{L}, \mathcal{R}])$ *of the contrast rate* $F(X)$*, where* $X \in [\mathcal{L}, \mathcal{R}]$*, are defined as follows:*

$$F^l([\mathcal{L}, \mathcal{R}]) \leq F(X) \leq F^u([\mathcal{L}, \mathcal{R}]), \ \mathcal{L} \subseteq \mathrm{X} \subseteq \mathcal{R} \tag{3}$$

$$F^l([\mathcal{L}, \mathcal{R}]) = \frac{(S_t(\mathcal{R}))^\delta}{S_b(\mathcal{L})}, \ F^u([\mathcal{L}, \mathcal{R}]) = \frac{(S_t(\mathcal{L}))^\delta}{S_b(\mathcal{R})}. \tag{4}$$

In Section 5.2, $F^u(X)$ and $F^l(X)$ are applied to prune the searching space.

4.2 Subtraction of Pattern Borders

As shown in *Example* 1, pattern border is a simple and efficient way to collect patterns. Border substraction is frequently performed to generate the candidate itemsets in $\mathcal{I}_{BFCG}$, according to Equation (1). The most straightforward method is to enumerate each element in the border and eliminate the redundant elements. But, it is rather costly in both computational time and memory when the border is large. Thus, we implement the substraction of two borders only based on two operators × and −, rather than exploring the borders directly.

Let itemset $X \subseteq I$, $I_X = \{X' | X' \subseteq X\}$, and $I_X^+ = \{X' | X' \subseteq X, X' \neq \emptyset\}$, several definitions and theorems are proposed to support the border subtraction on the two sets of itemsets: S_1 and S_2.

Definition 4. *The* ***Operator*** $\times$ *for* S_1 *and* S_2 *is defined as:*

$$S_1 \times S_2 = \{X_1 \cup X_2 | X_1 \in S_1, X_2 \in S_2\} \tag{5}$$

The ***Operator*** $-$ *for* S_1 *and* S_2 *is defined as:*

$$S_1 - S_2 = \{X' | X' \in S_1, X' \notin S_2\} \tag{6}$$

In Particular, $I_{X_1} \times I_{X_2} = \{X'_1 \cup X'_2 | X'_1 \in I_{X_1}, X'_2 \in I_{X_2}\} = \{X'_1 \cup X'_2 | X'_1 \subseteq X_1, X'_2 \subseteq X_2\} = I_{X_1 \cup X_2}$, and $I_{X_1} - I_{X_2} = \{X' | X' \subseteq X_1, X' \nsubseteq X_2\}$. The operator $\times$ is used for splitting a big border, e.g. $[\emptyset, \{a, b, c, d\}] = I_{\{a,b\}} \times I_{\{c,d\}}$. The operator $-$ is adopted during the subtraction of two collections of itemsets, e.g. $[\emptyset, \{a, b, c, d\}] - [\emptyset, \{a, b\}] = I_{\{a,b\}} \times I^{+}_{\{c,d\}}$. It will be frequently executed when generating the candidate itemsets in $\mathcal{I}_{BFCG}$. We then easily get the following properties and theorems for the operators:

$$Distributive\ law : (I_{X_1} \cup I_{X_2}) - I_{X_3} = (I_{X_1} - I_{X_3}) \cup (I_{X_2} - I_{X_3}) \tag{7}$$

$$Commutative\ law : I_{X_1} \times I_{X_2} = I_{X_2} \times I_{X_1} \tag{8}$$

$$Associative\ law : (I_{X_1} \times I_{X_2}) \times I_{X_3} = I_{X_1} \times (I_{X_2} \times I_{X_3}) \tag{9}$$

$$Allocation\ law : I_{X_1} \times (I_{X_2} \cup I_{X_3}) = (I_{X_1} \times I_{X_2}) \cup (I_{X_1} \times I_{X_3}) \tag{10}$$

Theorem 1. $I_{X_1 \cup X_2} = (I^{+}_{X_1} \times I^{+}_{X_2}) \cup I^{+}_{X_1} \cup I^{+}_{X_2} \cup I_{\emptyset}$

Proof. $I_{X_1} \times I_{X_2} = (I^{+}_{X_1} \cup \emptyset) \times (I^{+}_{X_2} \cup \emptyset) = (I^{+}_{X_1} \times I^{+}_{X_2}) \cup I^{+}_{X_1} \cup I^{+}_{X_2} \cup I_{\emptyset}$

Theorem 1 decomposes $I_{X_1 \cup X_2}$ into smaller parts easily to be processed.

Theorem 2. *If* $X_1 \cap X_3 = \emptyset$*, then* $(I^{+}_{X_1} \times I_{X_2}) - I_{X_3} = I^{+}_{X_1} \times I_{X_2}$

Proof. For $\forall X \in (I^{+}_{X_1} \times I_{X_2})$, we have $X \cap X_1 \neq \emptyset$. Then $(I^{+}_{X_1} \times I_{X_2}) \cap I_{X_3} = \emptyset$ as $X_1 \cap X_3 = \emptyset$

Below, we illustrate the use of these two operators for the border substraction.The above techniques are applied to Algorithm 1 (line 4-7) in Section 6.

Example 2. *The border subtraction* $[\emptyset, abcdefgh] - [\emptyset, ab] - [\emptyset, bc]$ *is obtained by the following calculation:* $(I_{abcdefgh} - I_{ab}) - I_{bc} = (I_{ab} \times I^{+}_{cdefgh}) - I_{bc} = I_{ab} \times ((I_c \times I^{+}_{defgh}) \cup I^{+}_{c}) - I_{bc} = (I_{abc} \times I^{+}_{defgh}) \cup (I^{+}_{a} \times I^{+}_{c} \times I_b)$. *By iterating* $I^{+}_{X_2}$*, we get* $I_{X_1} \times I^{+}_{X_2} = I_{X_1} \times \sum_{i=1, x_i \in X_2}^{|X_2|} (I^{+}_{x_i} \times \prod_{j>i}^{|X_2|} I_{x_j})$. *The set* I^{+}_{defgh} *further splits into sub-borders:* $I^{+}_{defgh} = (I^{+}_{d} \times I_{efgh}) \cup (I^{+}_{e} \times I_{fgh}) \cup (I^{+}_{f} \times I_{gh}) \cup (I^{+}_{g} \times I_{h}) \cup I^{+}_{h}$. *Therefore, we get* $[\emptyset, abcdefgh] - [\emptyset, ab] - [\emptyset, bc] = [d, abcdefgh] \cup [e, abcefgh] \cup [f, abcfgh] \cup [g, abcgh] \cup [h, abch] \cup [ac, abc]$.

5 Pattern Verification

The qualification of itemsets in $\mathcal{I}_{AJCG}$ is verified in a Branch-and-Bound manner. Given a border $[\mathcal{L}, \mathcal{R}]$, $F^u(X)$ and $F^l(X)$ are estimated by Equation (4), and are used to check the qualification of $[\mathcal{L}, \mathcal{R}]$. Instead of scanning the database repeatedly, we propose the T*-tree to calculate $S_b(\mathcal{L})$, $S_t(\mathcal{L})$, $S_b(\mathcal{R})$ and $S_t(\mathcal{R})$, which are used to compute $F^u(X)$ and $F^l(X)$ by scanning database only once. Then, the strategies are presented to split $[\mathcal{L}, \mathcal{R}]$ for checking the sub borders.

5.1 T*-tree Index

Transaction Tree (T*-tree) extends the classic spatial index, i.e. R-tree [2], to obtain new properties which significantly accelerate the calculation of support. In a R-tree, an object is represented by its Minimal Bounding Box (MBB) [2], which is the minimum bounding rectangle surrounding the object. R-tree is built on all the MBBs recursively. In order to efficiently get the support of an itemset, we introduce a novel index T*-tree, in which all the transactions are treated as spatial objects wrapped by their MBBs. The query on T*-tree is to check how many transactions match a given pattern. Accordingly, the patterns to be queried are also mapped to MBBs.

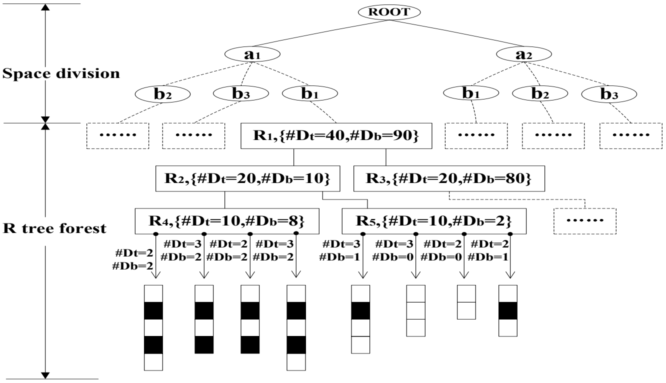


Fig. 3. An example of T*-tree, where R_i stands for MBB of an internal node

The main challenge of using T*-tree in a high dimensional data set is that the overlapping among nodes increases when more objects are inserted. Serious overlapping affects the query efficiency severely. Therefore, based on the data distribution, the data space is partitioned into multiple subspaces to reduce the overlapping of nodes as much as possible. The space partition follows a two-tier tree structure, as showed in Fig. 3. In this way, the computational time is dramatically cut by taking the following advantages:

1) A T*-tree consists of two layers: trunk and branch. All the attributes are sorted in a descending order on the gain ratio [6]. Top 10% (an empirical value, for instance) attributes are chosen as the trunk nodes, and the rest are the branch nodes. The top one attribute is the first level of trunk and so forth. With the growth of trunk levels, the data space is divided into smaller subspaces. After that, the branches are built in a similar process as R-tree.
2) Each node stores two values (i.e. $\sharp D_t$ and $\sharp D_b$, called *local supports*), which record the number of transactions in D_t and D_b, respectively.
3) The leaf nodes store the transaction chains that record the numbers of transactions covered by the same MBBs in D_t and D_b respectively. Multiple transactions can be stored in one transaction chain, especially when the transaction volume is huge in D_b.

5.2 Splitting Strategies

Given $[\mathcal{L}, \mathcal{R}]$, if $F^l([\mathcal{L}, \mathcal{R}]) \leq 1/k \leq F^u([\mathcal{L}, \mathcal{R}])$ (k is the contrast coefficient in Definition 1), then $[\mathcal{L}, \mathcal{R}]$ needs to be split into smaller borders for further validation. The splitting strategies below are used to speed up the search process, output CPs and remove unqualified borders quickly.

Strategy 1. If $F^u([\mathcal{L}, \mathcal{R}]) < 1/k$, then $F(X) \leq F^u([\mathcal{L}, \mathcal{R}]) < 1/k$. As a result, no itemset in $[\mathcal{L}, \mathcal{R}]$ is qualified to be chosen. So $[\mathcal{L}, \mathcal{R}]$ is safely removed.

Strategy 2. If $F^l([\mathcal{L}, \mathcal{R}]) \geq 1/k$, then $F(X) \geq F^l([\mathcal{L}, \mathcal{R}]) \geq 1/k$. Consequently, all the itemsets in $[\mathcal{L}, \mathcal{R}]$ must be selected. There are two directions to split $[\mathcal{L}, \mathcal{R}]$: for each single item $i \in \{\mathcal{R}\} - \{\mathcal{L}\}$, we extend $\mathcal{L}$ to the sub-border $[\mathcal{L} \cup \{i\}, \mathcal{R}]$, denoted as $\mathcal{L}^+$; or narrow down $\mathcal{R}$ to form the sub-border $[\mathcal{L}, \mathcal{R}/\{i\}]$, denoted as $\mathcal{R}^-$.

Strategy 3. If $F^u([\mathcal{L}, \mathcal{R}]) - 1/k \leq 1/k - F^l([\mathcal{L}, \mathcal{R}])$, it is quicker to locate the sub-borders that can be removed immediately with *Strategy 1* by narrowing their upper bound. The splitting must follow the direction in which F^u of sub-borders decreases as fast as possible. If $\max\limits_{i \in \{\mathcal{R}\} - \{\mathcal{L}\}} \{F^u([\mathcal{L}, \mathcal{R}]) - F^u([\mathcal{L} \cup \{i\}, \mathcal{R}])\} \geq \max\limits_{i \in \{\mathcal{R}\} - \{\mathcal{L}\}} \{F^u([\mathcal{L}, \mathcal{R}]) - F^u([\mathcal{L}, \mathcal{R}/\{i\}])\}$, the direction is $\mathcal{L}^+$; otherwise, $\mathcal{R}^-$.

Strategy 4. If $F^u([\mathcal{L}, \mathcal{R}]) - 1/k \geq 1/k - F^l([\mathcal{L}, \mathcal{R}])$, it is more efficient to qualify the sub-borders which can be delivered directly with *Strategy 2* by increasing their lower bound. The splitting must follow the direction in which the F^l of sub-borders increases as quickly as possible. If $\max\limits_{i \in \{\mathcal{R}\} - \{\mathcal{L}\}} \{F^l([\mathcal{L} \cup \{i\}, \mathcal{R}]) - F^l([\mathcal{L}, \mathcal{R}])\} \geq \max\limits_{i \in \{\mathcal{R}\} - \{\mathcal{L}\}} \{F^l([\mathcal{L}, \mathcal{R}/\{i\}]) - F^l([\mathcal{L}, \mathcal{R}])\}$, the direction is $\mathcal{L}^+$; otherwise, $\mathcal{R}^-$.

6 Algorithm of Converging Pattern Mining

Here, a novel algorithm ConvergMiner is proposed to mine CPs efficiently. Algorithm 1 presents the main process of mining CPs, and function *CheckContrast* is the key function to validate the qualification of candidate itemsets. There are four steps in the main process: **Step 1**. Build the T*-tree on D_t and D_b (Line 1). **Step 2**. Employ the Max-Miner to extract the pattern borders located at $\mathcal{I}_{ICDE}$ and $\mathcal{I}_{BFDH}$ (Line 2-3). **Step 3**. Perform the border substraction to obtain the pattern borders locate at $\mathcal{I}_{BFCG}$ (Line 4-7). **Step 4**. Search CPs in $\mathcal{I}_{BFCG}$ by a Divide-and-Conquer procedure (Line 8-9).

7 Scoring for Classification

There are two critical issues to be solved during the construction of an accurate CPs-based classifier: pattern selection and risk scoring. We adopt an effective measure, Maximal Coverage Gain (MCG) [7], to select the globally optimal pattern set. Once the optimal pattern set $\widehat{P}$ has been obtained, a base score for

Algorithm 1. ConvergMiner

```
Data: $D_t, D_b, k, \delta, \theta, \beta, len$
Result: Converging patterns
$H \leftarrow \emptyset$, Build $T^*$-tree on $D_t$ and $D_b$
$S^{'} = \{[\emptyset, X] | S_t(X) \geq \theta\}$          /* Get itemsets from $\mathcal{I}_{BFDH}$ */
$S^{''} = \{[\emptyset, X] | S_b(X) \geq \beta\}$          /* Get itemsets from $\mathcal{I}_{ICDE}$ */
for each $E$ in $S^{'}$ do
    for each $U$ in $S^{''}$ do
        $E \leftarrow E - U$  /* Obtain candidates by Opertaor — introduced in Section 4.2   */
    $H \leftarrow H \cup E$
for each $[\mathcal{L}, \mathcal{R}]$ in $H$ do
    $CheckContrast([\mathcal{L}, \mathcal{R}], T^*tree, k, \delta)$  /* Search CPs in $[\mathcal{L}, \mathcal{R}]$   */
Function CheckContrast($[\mathcal{L}, \mathcal{R}], T^*tree, k, \delta$)
Query T*-tree to collect $S_t(\mathcal{L}), S_t(\mathcal{R}), S_b(\mathcal{L}), S_b(\mathcal{R})$
if $S_b(\mathcal{L})$=0 or $F^l([\mathcal{L}, \mathcal{R}]) \geq \frac{1}{k}$ then // Based on strategy 2
    Output $[\mathcal{L}, \mathcal{R}]$ as CPs group
else if $S_b(\mathcal{R}) > 0$ and $F^u([\mathcal{L}, \mathcal{R}]) < \frac{1}{k}$ then // Based on strategy 1
    Remove $[\mathcal{L},\mathcal{R}]$
else if Direction($[\mathcal{L},\mathcal{R}],k,\delta$)=$\mathcal{R}^-$ then // Based on strategies 3 and 4
    for each $i \in \{\{\mathcal{R}\} - \{\mathcal{L}\}\}$ do
        CheckContrast($[\mathcal{L}$,R/{i}$]$,$k$,$\delta$ )
else // Based on strategies 3 and 4
    for each $i \in \{\{\mathcal{R}\} - \{\mathcal{L}\}\}$ do
        if $\mathcal{L}\cup\{i\} \subsetneq$H then
            CheckContrast($[\mathcal{L}\cup$\{i\}$,\mathcal{R}]$,$k$,$\delta$ )

```

each rule $r : p \to C_t$ in $\widehat{P}$ is calculated by Equation (11). The base score of p_i represents its impact in classifiers.

$$Base(p_i) = S_t(p_i) * F(p_i)/(1 + F(p_i)) \tag{11}$$

In the prediction phase, the score of u is calculated as follows:

$$\mathbb{S}(u) = \sum_{p_i \in \widehat{P}, u \prec p_i} Base(p_i) / \sum_{p_i \in P} Base(p_i) \tag{12}$$

Intuitively, given two transactions u_1 and u_2, if $\mathbb{S}(u_1) > \mathbb{S}(u_2)$, then the probability of $u_1 \in C_t$ is larger than that of $u_2 \in C_t$. Consequently, transaction u with a larger $\mathbb{S}(u)$ is more likely to be classified into C_t.

8 Experiment and Evaluation

Two real-life data sets are used: DS_1, the online banking transaction data from a major Australian bank. It contains 1,000,000 Genuine (D_b) and 1,000 Fraud (D_t), and the number of attributes involved is 112; DS_2, the social welfare payment claim data from Australia. It has 120,000 Genuine (D_b) and 2,120 Fraud (D_t) with 85 attributes. Below, ConvergMiner is compared with the state-of-the-art classification algorithms on these two class imbalanced data sets in terms of accuracy (i.e. Ada-Cost [13], Cost-NN [10], Cost-SVM [9] and CAEP [4]), efficiency (i.e. MDB-LL$_{border}$), and effectiveness of pruning strategies.

8.1 Accuracy

We compare the performance of the classifier powered by CPs with Ada-Cost [13], Cost-NN [10], Cost-SVM [9] and CAEP [4], from the perspective of False Positive Rate (FPR) and Detection Rate (DR, or True Positive Rate), which are defined as:

$$FPR = \frac{\text{False Alerts Number}}{\text{Genuine Instances Number}},\ DR = \frac{\text{Frauds Caught}}{\text{Total Frauds Number}} \tag{13}$$

In fact, the smaller the FPR and the larger the DR, the better the classifier. Fig. 4 shows that when 60% Fraud are caught, the FPRs of CPs, CAEP, Cost-NN, Ada-Cost and Cost-SVM are 6‰, 9‰, 12‰, 18‰ and 30‰, respectively. However, with the increase of FPR, all the classifiers achieve a higher DR. When $FPR = 11‰$, CPs catches 82% Fraud, but CAEP only catches 72%, Cost-NN catches 55%, Ada-cost catches 40%, and Cost-SVM catches 38%. The accuracy test on the data set 2 reveals the similar performance for the above methods (as shown in Fig. 5). Overall, we observe that at the same level of FPR, CPs achieves much higher DR than other methods; with the same DR, CPs obtains much lower FPR. In Fig. 4, when $FPR = 5\%$, CPs outperforms CAEP by 65%, Cost-NN by 80%, Ada-Cost by 90%, and Cost-SVM by 250%. In Fig. 5, for $DR = 60\%$, CPs outperforms CAEP by $FPR = 15‰$, which is only 60% of that of CAEP ($FPR = 24‰$). So, CPs outperforms benchmark methods in accuracy.

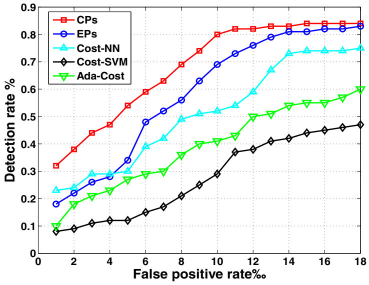


Fig. 4. ROC on DS_1

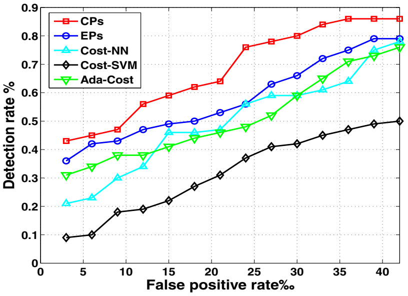


Fig. 5. ROC on DS_2

8.2 Efficiency

We compare the computation time consumed by ConvergMiner and MDB-LL$_{border}$ to evaluate their efficiency on DS_1. For both algorithms, we choose the same level of converging exponent δ=1. The contrast rate in ConvergMiner and MDB-LL$_{border}$ are $1/k$ and the growth rate, respectively. As shown in Fig. 6, with the increase of $1/k$, ConvergMiner gains more benefit from the splitting strategies, by which a huge number of sub-borders are eliminated more easily.

As a result, the number of borders to be split for the further checking decreases dramatically. In addition, MDB-LL$_{border}$ takes more time to process a huge number of the sub-border iterations due to the extremely low support in D_b. For instance, when the contrast rate is 1000, ConvergMiner takes only around 5% of the computation time consumed by MDB-LL$_{border}$. When $1/k$ is larger than 1000, MDB-LL$_{border}$ does not succeed in identifying the itemsets with the support less than 0.0005, but ConvergMiner still works stably. The reason is that ConvergMiner assigns a looser value rather than a fixed value to β (in D_b), and leaves the redundancy to the next step: bound checking and further validation. In summary, the results show that ConvergMiner significantly outperforms MDB-LL$_{border}$ on the imbalanced data set in terms of the efficiency.

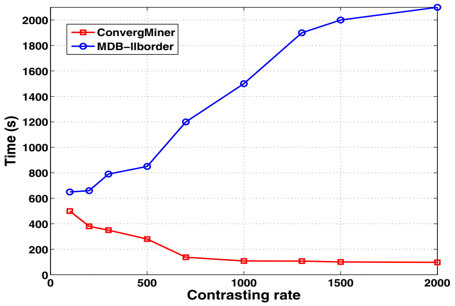


Fig. 6. Efficiency against contrast

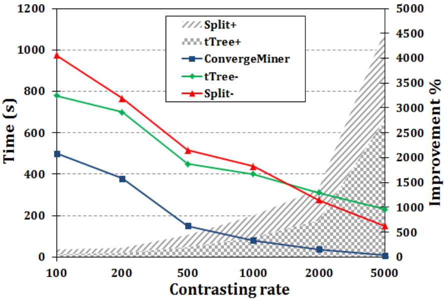


Fig. 7. Effectiveness on strategies

8.3 Effectiveness of the Pruning Strategies

Two benchmark algorithms, i.e. Split- and tTree-, are designed to evaluate the effectiveness of our pruning strategies on DS_1. Split- is a variant of ConvergMiner when the splitting strategies are replaced by a random one at line 17 in Function CheckContrast. tTree- is a variant of ConvergMiner by removing the local supports in T*-tree. Other components remain the same as ConvergMiner. Fig 7 displays the computational time of the three algorithms and the corresponding improvements gained from our pruning strategies under different contrast rates. ConvergMiner obtains the improvement of 50% (upon tTree-) and 90% (upon Split-), when the contrast rate is 100 due to the T*-tree and splitting strategies . With the increase of contrast rate, the improvement rate grows rapidly and reaches 2700% (upon tTree-) and 1700% (upon Split-) when the contrast rate is 5000. Thus, the proposed splitting strategies and T*-tree are effective in enhancing the computational efficiency.

9 Conclusion

In this paper, we introduce a novel type of patterns, i.e. converging patterns (CPs), on the extremely imbalanced data; and propose an efficient algorithm

ConvergMiner equipped with effective splitting strategies and a fast tree index to mine CPs. The experiments show that our algorithm greatly outperforms the state-of-the-art techniques on two real-life large scale imbalanced data sets in terms of accuracy and efficiency. In addition, ConvergMiner exhibits a strong capacity on the scalability and gains more advantages with a larger contrast rate. In future, we are going to mine CPs on the sequential data for online banking.

Acknowledgments. This work is sponsored by the Australian Research Council Discovery grant (DP1096218) and Linkage grant (LP100200774).

References

1. Bayardo, R.J.: Effciently mining long patterns from databases. In: SIGMOD 1998, pp. 85–93 (1998)
2. Beckmann, N., Kriegel, H., Schneider, R., Seeger, B.: The r*-tree: An efficient and robust access method for points and rectangles. In: SIGMOD 1990, pp. 322–331 (1990)
3. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: SIGKDD 1999, pp. 43–52 (1999)
4. Dong, G., Zhang, X., Wong, L., Li, J.: CAEP: Classification by aggregating emerging patterns. In: Arikawa, S., Nakata, I. (eds.) DS 1999. LNCS (LNAI), vol. 1721, pp. 30–42. Springer, Heidelberg (1999)
5. Fan, H., Ramamohanarao, K.: Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers. TKDE 18(6), 721–737 (2006)
6. Kent, J.: Information gain and a general measure of correlation. Biometrika 70(1), 163–173 (1983)
7. Li, J., Wang, C., Cao, L., Yu, P.S.: Large efficient selection of globally optimal rules on large imbalanced data based on rule coverage relationship analysis. In: SDM 2013 (accepted, 2013)
8. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In: ICDM 2001, pp. 369–376 (2001)
9. Li, Y., Kwok, J., Zhou, Z.: Cost-sensitive semi-supervised support vector machine. In: AI 2010, pp. 500–505 (2010)
10. Liu, X., Zhou, Z.: Training cost-sensitive neural networks with methods addressing the class imbalance problem. TKDE 18(1), 63–77 (2006)
11. Loekito, E., Bailey, J.: Fast mining of high dimensional expressive contrast patterns using binary decision diagrams. In: SIGKDD 2006, pp. 307–316 (2006)
12. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, Los Altos (1993)
13. Sun, Y., Kamel, M.: Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition 40(12), 3358–3378 (2007)

Online Cross-Lingual PLSI for Evolutionary Theme Patterns Analysis

Xin Xin, Kun Zhuang, Ying Fang, and Heyan Huang

School of Computer Science and Technology,
Beijing Institute of Technology, Beijing, China
{xxin,kzhuang,yfang,hhy63}@bit.edu.cn

Abstract. In this paper, we focus on the problem of evolutionary theme patterns (ETP) analysis in cross-lingual scenarios. Previously, cross-lingual topic models in batch mode have been explored. By directly applying such techniques in ETP analysis, however, two limitations would arise. (1) It is time-consuming to re-train all the latent themes for each time interval in the time sequence. (2) The latent themes between two adjacent time intervals might lose continuity. This motivates us to utilize online algorithms to solve these limitations. The research of online topic models is not novel, but previous work cannot be directly employed, because they mainly target at monolingual texts. Consequently, we propose an online cross-lingual topic model. By experimental verification in a real world dataset, we demonstrate that our algorithm performs well in the ETP analysis task. It can efficiently reduce the updating time complexity; and it is effective in solving the continuity limitation.

Keywords: Temporal Text Mining, Evolutionary Topic Patterns, Incremental/Online PLSI, Cross-lingual PLSI.

1 Introduction

Tracking the temporal evolutionary theme patterns (ETP) of documents is an important research issue nowadays. The original documents on the Web, such as news articles, research papers, etc., are generated in an unstructured stream, which would bring information overload problem to users. With the help of ETP analysis, it is more user-friendly to automatically group the documents in a temporal hierarchical structure. Figure 1(left) shows the ETP analysis result from the news articles (from NetEase, VOC, etc.) of the event Euro 2012. Before game starts, themes are mainly about the basic information of different teams, such as the stars, the strategies, etc.; after the game starts, the themes are evolved to the performances and reviews; before the final, the themes of the two participants in the final, Spain and Italy, merge into one theme, whose content includes the predictions from the fans, the comparison analysis, etc.; finally after the final, the theme is involved into the celebrations, the dominations, the reviews, and etc. We can see from this example that such a temporal hierarchical structure is very informative in tracking the development of an event. In addition,

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 74–85, 2013.

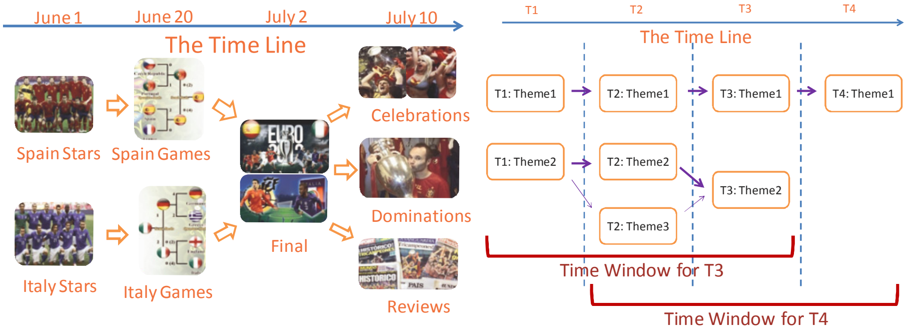


Fig. 1. An example of ETP analysis

it would be very helpful for information retrieval tasks from the semantic level. As a result, the ETP analysis has attracted more and more attention in both academia and industry [10–12].

In this paper, we focus on the issue of ETP analysis in the cross-lingual scenarios, which is different from previous ETP analysis [12] in monolingual texts. In this case, a topic is presented by documents of multiple languages rather than a monolingual one. Cross-lingual data would make the information more complete and the ETP analysis more accurate. For example, in Euro 2012 news, the discussion attitudes between a participant (e.g., Spain), and a third party member (e.g., China) might be quite different. The former would have bias to Spain; but the latter might be more neutral. Both of them should be considered in the ETP analysis.

Cross-lingual latent theme modeling is the key technique in cross-lingual ETP analysis. Competitive approaches include the cross-lingual probabilistic latent semantic indexing (PLSI) [15] model and the cross-lingual latent Dirichlet allocation (LDA) [9] model. The limitation of these competitive methods is that both of them are in batch mode. This would make the ETP analysis face the following two limitations.

Time-consuming Limitation. In the process of ETP analysis, the themes of documents in a new time interval should be integrated into the previous themes. In the batch mode, all the latent themes should be re-trained among the new documents and the old documents. In practice, the re-train process needs the enumeration of all the words in all the documents. Since the initial points are randomly chosen, it would also take many iterations of the enumeration before the model converges. Therefore, it is time-consuming to re-train the whole model for each time interval in the ETP analysis.

Continuity Limitation. If the re-train process is conducted for each time interval, the latent themes between adjacent time intervals might lose the continuity. The “continuity” has two meanings. 1) It is difficult to match the latent theme *ids* between the current time interval and the previous time interval. Because such *ids* are randomly assigned in the initial step of the algorithm. Thus the evolution process of themes might be not clear. 2) The latent themes between adjacent time intervals might have different category angles and might not match each other. For example, in Fig. 1 (left), the latent themes are divided by teams. However, the latent themes could also be divided by the activities from fans, the activities from the match, etc. In fact, these themes do exist in the data. But it is not proper to have one category in one time interval, and have another in the next. An ideal ETP tracking should keep the continuity in all the time intervals.

These two limitations motivate us to investigate online algorithms for ETP analysis. Intuitively, when new documents arrive, the online algorithms only make local update to the model instead of global one, thus the time complexity would be much less than re-training the whole model. In addition, while the online algorithm retains the main body of the previous model, the continuity would be kept. Thus both the above limitations could be solved.

Online algorithms for topic models are not novel research tasks either. Both online PLSI [4] and online LDA [6] have been explored. But these models are mainly for monolingual scenarios. Thus they cannot be directly employed. Therefore, in this paper, we combine the ideas of previous work and propose an online cross-lingual topic model for cross-lingual ETP analysis. Comparing between the PLSI model and the LDA model, the LDA model utilizes the regularization technique to smooth the data fitting, which indeed performs well in previous tasks, but it also increases the training cost and the model complexity. As a preliminary study, in this paper, we first choose to combine the cross-lingual PLSI [15] and the online PLSI [4] for simplicity. We leave the combination of the cross-lingual LDA[9] and the online LDA [6] as our future work.

The main contributions of this paper lie in that we combine the ideas of previous cross-lingual PLSI and online PLSI together and propose an online cross-lingual topic model. In addition we utilize it in ETP analysis and demonstrate its efficiency for reducing the time complexity and its effectiveness for keeping the continuity property.

In the following of this paper, to make it consistent with previous work and easy for comparisons, *many previous definitions and variables in [4, 12, 15] are retained if being not necessarily changed, and our contribution is to combine these ideas in the cross-lingual ETP analysis task.*

2 Problem Definition

2.1 Preliminary Definitions

Definition 1 (Cross-lingual Corpus). *Following [15], the cross-lingual corpus is a dataset with multiple languages. We utilize $C = \{C_1, C_2, ..., C_s\}$ to denote the set of data collections with s languages. Each C_i denotes the data collection of a single language, whose vocabulary is denoted by $W_i = \{w_1^i, w_2^i, ...w_{N_i}^i\}$. N_i is the total word number in the i_{th} language. Each data collection C_i contains a set of documents, $C_i = \{d_1^i, d_2^i, ..., d_{M_i}^i\}$. Each document is presented by a bag of words, and a function $c(w_k^i, d_j^i)$ is utilized to denote the occurrence count of word w_k^i in document d_j^i.*

Definition 2 (Cross-lingual Theme). *Following [15], a cross-lingual theme θ is presented as a multinomial distribution of words, denoted by $p(w|\theta)$, $w \in W_1 \bigcup W_2 \bigcup ... \bigcup W_s$. For each θ, we have $\sum_{i=1}^{s} \sum_{w \in Wi} p(w|\theta) = 1$. Under this definition, a cross-lingual theme would gather the words of all the languages with related semantic meanings together. This is a super set of monolingual theme. A monolingual theme could be extracted from the cross-lingual theme by normalization as $p_i(w^i|\theta) = \frac{p(w^i|\theta)}{\sum_{w \in W_i} p(w|\theta)}$.*

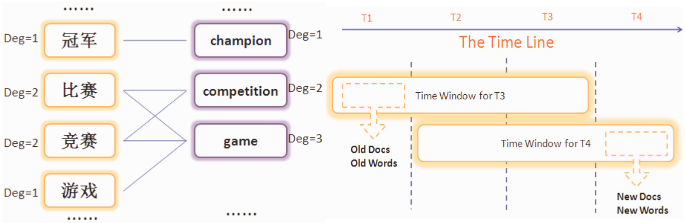
Definition 3 (Time Interval and Time Window). *As shown in Fig. 1 (right), the time line is divided into discrete time intervals. Each time interval is a fixed number of days, e.g., a week of seven days. For each time interval, the themes are automatically learned from the documents in both the current interval and the recent intervals to retain the overlap between new documents and old documents [12]. By doing this, the evolution analysis would be more smooth and robust. Thus a time window is defined for each time interval. It contains the current interval and a fixed number $l - 1$ of the recent intervals. The themes in a time interval is learned from the documents in its corresponding time window. Take $l = 3$ as an example, in Fig. 1 (right), the themes in interval T_3 is trained from the documents in the time window of T_1,T_2 and T_3; and the themes in T_4 is trained from the documents in a time window of T_2,T_3 and T_4.*

Definition 4 (Cross-lingual Evolutionary Transition). *Following [12], we utilize θ^{T_i} to denote a theme with a time stamp in the cross-lingual data. Suppose a theme $\theta_a^{T_{i-1}}$ and a theme $\theta_b^{T_i}$, if the similarity between them is larger than a threshold, it is defined that there is a cross-lingual evolutionary transition from $\theta_a^{T_{i-1}}$ to $\theta_b^{T_i}$.*

Definition 5 (Cross-lingual Theme Evolutionary Graph). *Following [12], the cross-lingual theme evolutionary graph $G = (N, E)$ is defined as a weighted graph as shown in Fig. 1 (right). In the graph, each node $v \in N$ denotes a theme with a time stamp; and each edge $e \in E$ denotes whether the two nodes in adjacent time intervals have an evolutionary transition. The weight of an edge (denoted by the thickness of the edge), stands for the similarity value between the current theme and the evolved theme. The larger the similarity is, the thicker the edge would be.*

2.2 The Task of Cross-Lingual ETP Analysis

Following the idea in [12], we define the task of cross-lingual ETP analysis as to draw a cross-lingual theme evolutionary graph from the unstructured data stream. The result of ETP analysis would give a clear temporal hierarchical overview of the themes and their evolutions for different events, which would serve for both users and other intelligent services.



Dictionary based Word Graph

Old Info. to be Deleted and New Info. to be Folded in

Fig. 2. Illustration for online cross-lingual PLSI

3 General Framework

The general framework for cross-lingual ETP analysis is a two-step process. The first step is to extract the themes in each time interval from their time windows; and the second step is to construct the evolutionary transitions among the themes in adjacent time intervals.

The second step is more intuitive by calculating the KL-divergence $D(\theta_2||\theta_1)$ between two themes [12]. In the cross-lingual case, it is defined as

$$D(\theta_2||\theta_1) = \sum_{w_i \in W_1 \bigcup W_2 \bigcup \ldots \bigcup W_s} p(w_i|\theta_2) \log \frac{p(w_i|\theta_2)}{p(w_i|\theta_1)}. \tag{1}$$

The main difference compared with [12] is that the cross-lingual theme contains words from different languages.

The first step is the key point that would be discussed in this paper. A naive way is to re-train all the documents in the time window in a batch mode. For example, the cross-lingual PLSI [15] model could be directly utilized here for theme extraction. However, as discussed in previous sections, it is time-consuming to re-train all the latent themes for each time interval; and the latent themes between two adjacent time intervals might lose continuity. Therefore, in this paper, we combine the idea in [15] and [4] to propose an online cross-lingual PLSI model for cross-lingual ETP analysis, which would be illustrated in detail in the following section.

4 Online Cross-Lingual PLSI

4.1 Cross-Lingual PLSI in Batch Mode

The cross-lingual PLSI [15] model is described by two kinds of parameters, $p(\theta|d)$ and $p(w|\theta)$. θ denotes the theme; d denotes the document; and w denotes the word.

The learning process of $p(\theta|d)$ and $p(w|\theta)$ is to optimize the linear combination of 1) the log-likelihood of the data and 2) the constraint function of cross-lingual connection, denoted as

$$F = (1-\lambda)L(C) + \lambda R(C), \tag{2}$$

where

$$L(C) = \sum_{i=1}^{s} \sum_{d \in C_i} \sum_{w} c(w,d) log \sum_{j=1}^{k} p(\theta_j|d)p(w|\theta_j), \tag{3}$$

$$R(C) = -\frac{1}{2} \sum_{<w_u,w_v>\in E} \sum_{j=1}^{k} (\frac{p(w_u|\theta_j)}{Deg(w_u)} - \frac{p(w_v|\theta_j)}{Deg(w_v)})^2. \tag{4}$$

$L(C)$ is the log-likelihood of the data in multiple languages. In constructing the constraint function $R(C)$, a multi-partite undirected graph $G =< W, E >$ is built from each bilingual dictionaries as shown in Fig. 2 (left). Each node in W denotes a word. If two words from different languages, e.g., w_u and w_v, could interpret each other, there would be an edge between them. $Dev(w_u)$ denotes the degree of the word w_u. From the definition, by optimizing the function $R(C)$, words from multiple languages that have similar meanings, would have similar probabilities within a theme.

The optimization is based on the EM algorithm as follows.

1. E-Step

$$P(\theta|w,d) = \frac{P(w|\theta)P(\theta|d)}{\sum_{\theta'} P(w|\theta')P(\theta'|d)} \tag{5}$$

2. M-Step

$$P(w|\theta) = \frac{\sum_{i=1}^{s} \sum_{d \in C_i} c(w,d)P(\theta|w,d)}{\sum_{i=1}^{s} \sum_{d \in C_i} \sum_{w' \in d} c(w',d)P(\theta|w',d)} \tag{6}$$

$$p(\theta|d) = \frac{\sum_{w \in d} c(w,d)P(\theta|w,d)}{\sum_{w \in d} c(w,d)} \tag{7}$$

$$p^{(t+1)}(w_u|\theta_j) = (1-\alpha)p^{(t)}(w_u|\theta_j) + \alpha \sum_{<w_u,w_v>\in E} \frac{1}{Deg(w_v)} p^{(t)}(w_v|\theta_j) \tag{8}$$

4.2 Proposed Cross-Lingual PLSI in Online Mode

Following the idea in [4], the process of the online cross-lingual PLSI is shown in Fig. 2 (right), including 1) discarding old documents and old terms; 2) folding in new documents and new terms and 3) updating the PLSI parameters. Compared with the online monolingual PLSI in [4], in this paper, step 2 and 3 are extended in order to fit the cross-lingual data. Therefore, in this section, we would focus these two steps. The first step could be referred in [4].

1. Fold in new documents and terms. The target of folding in new documents and terms is to estimate $P(\theta|d_{new})$ and $P(\theta|w_{new})$. The difference of our work from previous work [4] is that the process should be extended for incorporating the cross-lingual word relations. Because words in different languages that have similar meanings should have more probability to be assigned to the same theme. In the batch learning mode, this problem has been solved by a global update in Eq. 8; in the online learning mode, this would be solved by a local update. Consequently, we add an local update step as an extension of previous updating process. The whole process is conducted as follows by EM algorithms. By fixing the parameter of $P(w|\theta)$, $P(\theta|d_{new})$ could be estimated by iteration calculation of

$$P(\theta|w, d_{new}) = \frac{P(w|\theta)P(\theta|d_{new})}{\sum_{\theta' \in \Theta} P(w|\theta')P(\theta'|d_{new})}, . \tag{9}$$

$$P(\theta|d_{new}) = \frac{\sum_{w \in d_{new}} c(w, d_{new})P(\theta|w, d_{new})}{\sum_{\theta \in \Theta} \sum_{w \in d_{new}} c(w, d_{new})P(\theta'|w, d_{new})}. \tag{10}$$

After $P(\theta|d_{new})$ is estimated, $P(d_{new}|\theta)$ could be estimated by

$$P(d_{new}|\theta) = \frac{\sum_{w \in d_{new}} c(w, d_{new})P(\theta|w, d_{new})}{\sum_{d \in D_{new}} \sum_{w \in d} c(w, d)P(\theta|w, d)}, \tag{11}$$

where D_{new} is all the documents in the new time window, including documents in the remaining intervals and the new interval. By fixing the parameter of $P(d_{new}|\theta)$, $P(\theta|w_{new})$ could be estimated by iteration calculation of

$$P(\theta|w_{new, d_{new}}) = \frac{P(d_{new}|\theta)P(\theta|w_{new})}{\sum_{\theta' \in \Theta} P(d_{new}|\theta')P(\theta'|w_{new})}, \tag{12}$$

$$P(\theta|w_{new}) = \frac{\sum_{d \in D_{new}} c(w_{new}, d)P(\theta|w_{new}, d)}{\sum_{d' \in D_{new}} c(w_{new}, d')}. \tag{13}$$

$$P'(\theta|w_{new}) = (1-\alpha)P(\theta|w_{new}) + \alpha \sum_{<w_{new}, w_v> \in E} \frac{1}{Deg(w_{new})} P(\theta|w_v) \tag{14}$$

In the last equation, the cross-lingual smoothing is incorporated following the idea of [15]. Consequently, the similar words in different languages would have similar latent theme distributions.

2. Update the PLSI parameters. In the folding in process, $p(w_{old}|\theta)$ has not been changed and $p(w_{new}|\theta)$ has not been incorporated. Thus a normalization is utilized following [4] as

$$P(w|\theta) = \frac{\sum_{d \in D_{new}} c(w, d)P(\theta|w, d)}{\sum_{d' \in D_{new}} \sum_{w' \in d'} c(w', d')P(\theta|w', d')}. \tag{15}$$

If calculation complexity is limited in a small range, the updating has been finished. But if more calculation is allowed, Eq. 5, Eq. 6, Eq. 7 and Eq. 8 could be executed iteratively to optimize the parameters further.

5 Experiments

The experiments are targeted at justifying the following issues for the proposed online cross-lingual PLSI model.

1. What is its overall performance in the ETP analysis?
2. Whether it is efficient in reducing the time complexity?
3. Whether it is effective in solving the continuity limitation?

To issue 1, we show an intuitive example of the ETP analysis result; and we also compare it with a version of monolingual topic model by translating multiple languages into a single language, in order to show the advantages of cross-lingual. To issue 2, we compare the speed of convergence of the proposed model with the original cross-lingual model in batch mode. To issue 3, we give both qualitative and quantitative justification compared with the original cross-lingual model in batch mode.

5.1 Dataset

Our dataset is a set of news articles, which is collected between June 1st 2012 to June 23th 2012. The articles are either in English or in Chinese. Totally, 269,144 Chinese articles and 64,897 English articles are collected. A pre-processing is conducted before the experiments, including splitting words, removing the stop words and stemming. We choose mandarintools[1] to build the bilingual word graph as shown in Fig. 2 (left).

The dataset is divided into three time intervals evenly, denoted as T_1, T_2 and T_3. The time window length is $l = 2$. Thus the time window for T_2 include T_1 and T_2; and the time window for T_3 is T_2 and T_3. When documents of T_3 arrive, the topic model should discard the old words and documents in T_1 and fold in the new words and documents in T_3. In the proposed online PLSI, this process is natural. In the original PLSI in batch mode, the model should be re-trained using the documents in T_2 and T_3.

5.2 Overall Performance

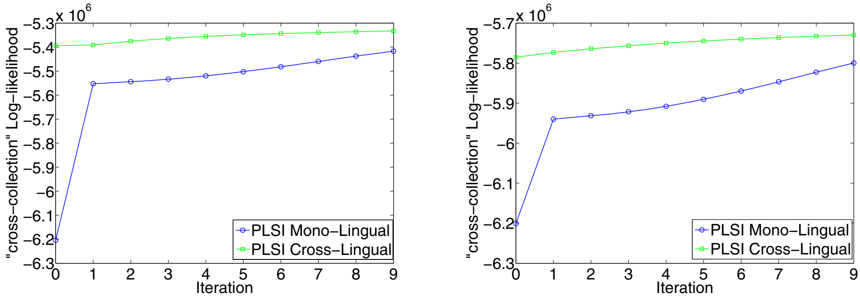
To evaluate the overall performance, we utilize the proposed model to extract the themes in T_2 and T_3. Then the cross-lingual evolutionary transitions are generated by calculating the KL-divergence using Eq. 1. A cross-lingual theme evolutionary graph is drawn like Fig. 1 (right). Due to the space limitation, we do not demonstrate the whole graph, but we show part of it instead for analysis.

[1] `mandarintools.com`

Table 1. Overall performance example

Theme extraction in T_2		Theme extraction in T_3 by PLSI in batch mode		Theme extraction in T_3 by the proposed model	
Theme 2.1	Theme 2.2	Theme 3.1’	Theme 3.2’	Theme 3.1	Theme 3.2

England
卡希尔(Cahill)
Terry
France
保罗(Paul)
揭幕(opening)
热身赛
(warm-up match)
Hodgson
Curtis
球衣(jersey)
瑞典(Sweden)
	Italy
	Spain
	克罗地亚(Croatia)
	进球数(goals scored)
	球迷(fan)
	Cassano
	皮尔洛(Pirlo)
	托雷斯(Torres)
	player
	集训(train)
	预测(predict)
		警方(police)
		酒店(hotel)
		worker
		sex
		话题(topic)
		中国(China)
		种族歧视
		(racial discrimination)
		country
		watch
		乌克兰(Ukraine)
			coach
			乌克兰(Ukraine)
			卡希尔(Cahill)
			德国(Germany)
			Spain
			意大利(Italian)
			英格兰(England)
			大战(contest)
			出线(advance)
			knock
				England
				球迷(fan)
				乌克兰(Ukraine)
				hotel
				Spain
				杰拉德(Gerrard)
				鲁尼(Rooney)
				sex
				底线(end line)
				team
				出线(advance)
					Italy
					Ireland
					克罗地亚(Croatia)
					巴洛特利(Balotelli)
					默契(tacit)
					barbecue
					赌球(gambling)
					积分(score)
					球衣(jersey)
					卡萨诺(Cassano)
					final



(a) $\alpha = 0.2$

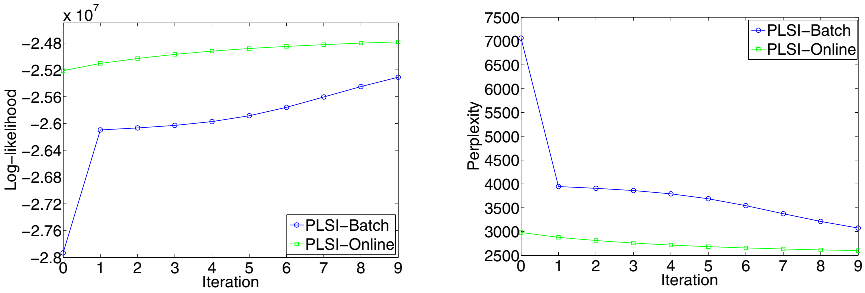
(b) $\alpha = 0.8$

Fig. 3. Comparison with monolingual topic model

We choose the event of Euro 2012 as an example. Table. 1 shows the ETP analysis result. Column 1 shows the theme extraction result for T_2; Column 2 shows the theme extraction result for T_3 from cross-lingual PLSI in batch mode, which would be utilized to show the improvement of continuity later; and Column 3 shows the theme extraction result for T_3 from the proposed model. *Theme* 3.1 is evolved from *theme* 2.1; and *theme* 3.2 is evolved from *theme* 2.2.

It can be observed that in *theme* 2.1, stories are related to the England team, such as the stars, the opponent team, etc. The stories are about warm-up matches, openings, etc. After being evolved to *theme* 3.1, stories are also related to England team. But as time goes by, the content of the stories has been changed to whether the England team can advance, the behavior of its star Rooney, etc. *Theme* 2.2 and *theme* 3.2 have similar theme evolution, but they are related to the Italy team. We can also see that the English media is more open in sensitive topics such as “sex” than the Chinese media, which shows the complementarity of information among the cross-lingual data. This example demonstrates that proposed model is effective in cross-lingual ETP tasks.

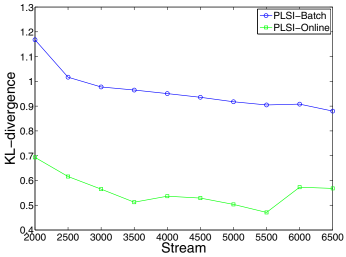
To show the advantage of cross-lingual topic model, we compare it with the monolingual PLSI by translating the Chinese documents into English documents. The experimental setup for translation can be referred to [15]. For evaluation,



(a) log-likelihood (b) perplexity

Fig. 4. Comparison with cross-lingual PLSI in batch mode (1)

#Theme	PLSI-Batch Convergence time (ms)	PLSI-Online Convergence time (ms)
8	190300	42375 (22.2%)
16	504269	91857 (18.2%)
24	796977	145785 (18.3%)



(a) execution time (b) continuity

Fig. 5. Comparison with cross-lingual PLSI in batch mode (2)

we utilize the “cross-collection” log-likelihood defined in [15]. Fig. 3 shows the experimental result for different parameters. α is the parameter used in Eq. 8 and Eq. 14. It could be observed that the cross-lingual topic model constantly outperform the monolingual topic model in each iteration. This is because the cross-lingual topic model could well smooth the connection between different languages. The result is consistent with the result in [15].

5.3 Justification for Reducing the Time Complexity

In order to justify the performance of reducing the time complexity, we compare our proposed model with the original cross-lingual PLSI [15] in batch mode. Following the evaluation metrics in [1, 4], we utilize the log-likelihood and perplexity to evaluate the performance of the model in each iteration. For perplexity, the smaller the value, the better the performance. The definition of the metrics could be found in [1, 4]. For the proposed online model, the parameters are updated from the parameters of the previous model; for the original model in batch mode, the parameters are randomly allocated at first and are re-trained thoroughly.

Fig. 4 and Fig. 5(a) shows the experimental results. It could be observed that the proposed online model converges much faster than the model in batch mode, in both log-likelihood and perplexity metrics. From Fig. 5(a), the convergence time of the proposed model is around 20% of the original model's, which is consistent with the result in [4] (15% to 20% of the batch mode in monolingual case). This demonstrates that the proposed model is efficient for reducing the time complexity in folding in the new documents and words.

5.4 Justification for Solving the Continuity Limitation

In qualitative analysis, we show an example of the ETP analysis from the original PLSI [15] in batch mode in Table. 1. We can see the difference between this model and our proposed model. The themes in T_2 are divided by different teams. One for England and one for Italy. After the evolution in T_3, the themes extracted by our proposed model are also divided by different teams. However, the themes extracted by the original PLSI in batch mode are divided by other dimensions. *theme* $3.1'$ is about the activities of fans; and *theme* $3.2'$ is about the activities of the game. This example demonstrates directly that the proposed model performs better in keeping the continuity.

In quantitative analysis, following the idea in [4], we utilize the average KL divergence rate of the two closest latent variables in two adjacent time windows to quantify the continuity performance. The detailed definition of this metric could be found in [4]. Fig. 5(b) shows the performance of our proposed model and the original model in batch mode. It could be observed that the performance of the proposed model outperforms the original model constantly. The average distance of our proposed model is 56% of the original model's.

From above justifications, it can be concluded that the proposed model is effective in solving the continuity limitation.

6 Related Work

The primary work related to ETP analysis in history is the topic detection and tracking (TDT) task [14]. The main difference of the TDT tasks from the ETP analysis is that TDT is in the document level, while ETP is in the word level. Recent work of ETP analysis include [5, 8, 10–12]. The main difference from our work is that our model can model cross-lingual themes. Topic model is the key technique in this paper. Two representatives of topic model include PLSI [7] and LDA [2]. Many variances of topic models are presented over these years, including the online learning [1, 4, 6, 8]; the cross-lingual modeling [3, 9, 13, 15], and etc.

7 Conclusion and Future Work

In this paper, we propose an online cross-lingual PLSI model and utilize this model in the ETP analysis tasks. Experimental verification from a real world dataset demonstrates that the proposed model performs well. It is efficient to reduce the training time in folding in new documents; and the proposed model is effective in keeping the continuity property of themes in different time intervals.

In the future, we will try to combine previous online LDA and cross-lingual LDA model in ETP analysis. We believe through the regularization techniques in LDA, the performance would obtain another improvement.

Acknowledgments. The work described in this paper was fully supported by the National Basic Research Program of China (973 Program, Grant No. 2013CB329605). The authors also would like to thank the reviewers for their helpful comments.

References

1. AlSumait, L., Barbará, D., Domeniconi, C.: On-line lda: adaptive topic models for mining text streams with applications to topic detection and tracking. In: Proceedings of ICDM 2008, pp. 3–12. IEEE (2008)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research 3, 993–1022 (2003)
3. Boyd-Graber, J., Blei, D.M.: Multilingual topic models for unaligned text. In: Proceedings of UAI 2009, pp. 75–82. AUAI Press (2009)
4. Chou, T.C., Chen, M.C.: Using incremental plsi for threshold-resilient online event analysis. IEEE Transactions on Knowledge and Data Engineering 20(3), 289–299 (2008)
5. He, Q., Chen, B., Pei, J., Qiu, B., Mitra, P., Giles, L.: Detecting topic evolution in scientific literature: how can citations help? In: Proceeding of CIKM 2009, pp. 957–966. ACM (2009)
6. Hoffman, M.D., Blei, D.M., Bach, F.: Online learning for latent dirichlet allocation. In: Proceedings of NIPS 2010, vol. 23, pp. 856–864 (2010)
7. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of SIGIR 1999, pp. 50–57. ACM (1999)
8. Iwata, T., Yamada, T., Sakurai, Y., Ueda, N.: Online multiscale dynamic topic models. In: Proceedings of KDD 2010, pp. 663–672. ACM (2010)
9. Jagarlamudi, J., Daumé III, H.: Extracting multilingual topics from unaligned comparable corpora. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 444–456. Springer, Heidelberg (2010)
10. Kleinberg, J.: Bursty and hierarchical structure in streams. In: Proceedings of the KDD 2003, vol. 7, pp. 373–397 (2003)
11. Lin, C.X., Mei, Q., Han, J., Jiang, Y., Danilevsky, M.: The joint inference of topic diffusion and evolution in social communities. In: Proceedings of ICDM 2011, pp. 378–387. IEEE (2011)
12. Mei, Q., Zhai, C.X.: Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In: Proceedings of the KDD 2005, pp. 198–207. ACM (2005)
13. Ni, X., Sun, J.T., Hu, J., Chen, Z.: Cross lingual text classification by mining multilingual topics from wikipedia. In: Proceedings of WSDM 2011, pp. 375–384. ACM (2011)
14. Wang, C., Zhang, M., Ma, S., Ru, L.: Automatic online news issue construction in web environment. In: Proceedings of WWW 2008, pp. 457–466. ACM (2008)
15. Zhang, D., Mei, Q., Zhai, C.X.: Cross-lingual latent topic extraction. In: Proceedings of ACL 2010. Association for Computational Linguistics, pp. 1128–1137 (2010)

F-Trail: Finding Patterns in Taxi Trajectories

Yasuko Matsubara[1], Lei Li[2], Evangelos Papalexakis[3], David Lo[4],
Yasushi Sakurai[1], and Christos Faloutsos[3]

[1] NTT Communication Science Labs
[2] University of California, Berkeley
[3] Carnegie Mellon University
[4] Singapore Management University

Abstract. Given a large number of taxi trajectories, we would like to find interesting and unexpected patterns from the data. How can we summarize the major trends, and how can we spot anomalies? The analysis of trajectories has been an issue of considerable interest with many applications such as tracking trails of migrating animals and predicting the path of hurricanes. Several recent works propose methods on clustering and indexing trajectories data. However, these approaches are not especially well suited to pattern discovery with respect to the dynamics of social and economic behavior. To further analyze a huge collection of taxi trajectories, we develop a novel method, called F-Trail, which allows us to find meaningful patterns and anomalies. Our approach has the following advantages: (a) it is fast, and scales linearly on the input size, (b) it is effective, leading to novel discoveries, and surprising outliers. We demonstrate the effectiveness of our approach, by performing experiments on real taxi trajectories. In fact, F-Trail does produce concise, informative and interesting patterns.

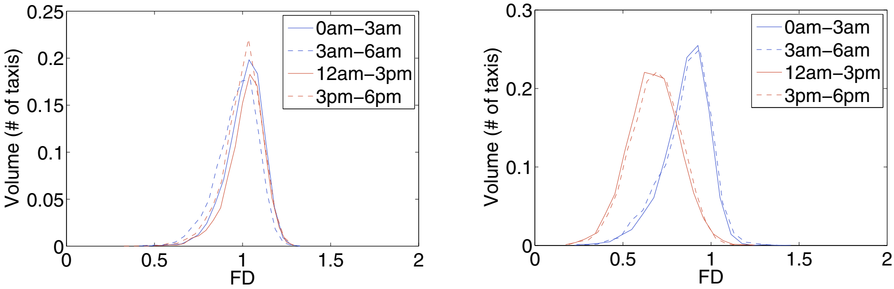
1 Introduction

What patterns can we derive, using trajectory data from a whole fleet of taxis? What is the normal pattern of activity, and which (if any) outliers exist? We seek to discover such patterns, so that we can spot anomalies, and help the taxi operating company understand general trends, with the ultimate goals of maximizing fuel efficiency, profits, and passenger satisfaction. Trajectory analysis has attracted a lot of interest, including trajectories of (migrating) animals [21,8,7,10], of hurricane paths [4,2], as well as from an indexing point of view [6,16]. The latter studied indexing, but not pattern discovery; among the former, the emphasis was on clustering and distance functions on trajectories.

Motivation and Challenges. Uncovering rules governing collective taxi behavior is a challenging task because of the myriad factors that influence an individual's decision to take a particular action. In this work, we study 10,000 trajectories generated by anonymous taxi drivers, with the aim of measuring social and economic activity. Intuitively, the task of this paper is as follows:

Given a set of GPS coordinates for every taxi, every few minutes, and its status (i.e., 'full' or 'empty'), find the general trends of the fleet of taxis, and spot outliers.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 86–98, 2013.



(a) Full taxis

(b) Empty taxis

Fig. 1. Different behavior of empty taxis, at night: PDF of fractal dimension (FD) of all trips, for each segment of a day (0-3am, 3-6am etc). (a) Full taxis (b) empty ones. Notice that trajectories of full taxis are linear-like (fractal dimension FD≈1), for any time slot; empty taxis have a wandering behavior (FD < 1) during the day, but linear behavior during the night.

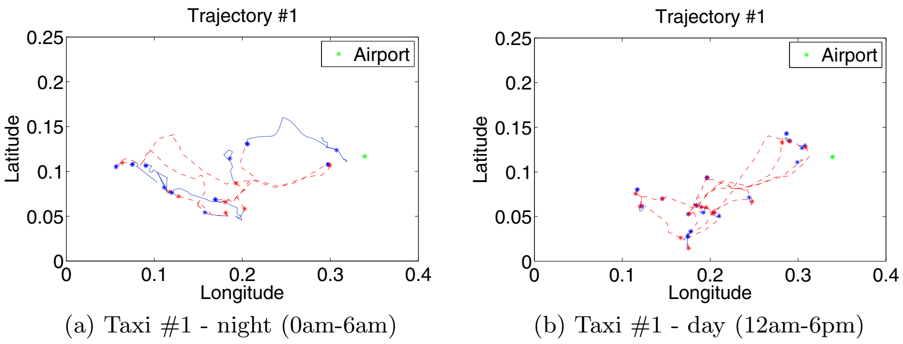
Here, we present a novel method, F-Trail, for finding meaningful patterns and anomalies in a huge number of trajectories. Our approach can scalably and automatically identify typical patterns of taxi behavior, and actually "see" such patterns from the point of view of topology. More specifically, we propose using the *fractal dimension* as a characteristic for trend analysis and extreme detection.

Figure 1 shows a snapshot of our discoveries: Namely, taxi drivers follow linear-like paths when 'full'; and convoluted, 'wandering' paths when empty during the day, but clearly different behavior when 'empty' at night. More specifically Figure 1(a) is the probability density function (PDF) of the 'fractal dimension' (FD) of all trajectories with a passenger (say, "full taxi trails"). Since $FD \approx 1$, for any time of the day, we conclude that, when 'full', taxi drivers follow linear, piece-wise linear, and in general, smooth paths. Conversely, Figure 1(b) shows that, when 'empty', they follow short, wandering paths, creating a bursty-like pattern (see blue dots in Figure 2(b)), with a much lower FD.

The only exception is during the night (solid blue and dotted blue curves in Figure 1(b)): the drivers abandon their 'wandering' behavior, since they don't expect to find nearby customers, and instead go almost straight to taxi plazas or the airport (green dot in Figures 2(a-b)). Notice the linear-like blue paths ('empty trails') in Figure 2(a). We present several more observations, later.

Contributions. In this paper we propose a new approach, namely F-Trail, which includes fast and effective techniques that can learn the key trends of a large collection of taxi trails. The contributions of this paper are as follows:

1. *Effective*: We apply F-Trail to a real trajectory dataset, which allows us to identify major trends in taxi behavior, and spot outliers.
2. *Adaptive:* F-Trail describes the common behavior and anomalies of taxi trajectories from the point of view of an individual taxi.
3. *Scalable*: The careful design of F-Trail makes it linear on the number of input size in terms of wall clock time.



(a) Taxi #1 - night (0am-6am) (b) Taxi #1 - day (12am-6pm)

Fig. 2. Night vs day trajectories of the same taxi: (a) during the night, the blue (='empty') trail is linear-like, probably going to the airport (green dot), or to taxi plazas; (b) during the daytime, the blue (empty) trail is point-like, due to wandering behavior, to find nearby passengers. In all cases, the red (='full') trails are linear-like. Note that we anonymize latitude and longitude information due to privacy concerns.

Outline. The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 introduces our approach and describes how to analyze individual taxi behavior. We conclude in Section 4.

2 Related Work

The previous work on mining trajectory data can be grouped into the following categories: (1) design of distance and similarity scores and (2) indexing methods for spatial-temporal databases. For trajectory similarity functions, Vlachos et al. [21] use the longest common sub-sequence, while [8,7] use minimal description language. Given a similarity score, [21] study the trajectories of marine mammals, while [8,7] use it to find clusters and outlier trajectories. Gaffney et al. [4,2] use generative models to group the trajectories of moving objects such as hurricanes. Giannotti et al. [5] study the trajectory pattern mining problem. Very recently, Yuan et al. [24,23,11] study a large number of taxi trajectories in Beijing, and present new and sophisticated models to find the different functional regions, and optimal driving route.

Remotely related is the work on indexing and searching moving objects: The work in [16] builds index on moving both spatial and temporal dimensions with pre-aggregation to support OLAP operations. In related work [6,3] also propose various solutions for answering region retrieval queries, predicting the past and future positions. Similarity search and pattern discovery in time sequences have also attracted huge interest [13,18,15].

One of the contributions of F-Trail is that it uses fractal concepts to spot patterns in trajectories. Fractals and self-similarity have been used in numerous settings, like financial time series analysis [12], modeling ethernet traffic [22], social network analysis [14,9], and numerous other settings (see, e.g., [19,17]).

3 F-Trail

Given several thousands of trajectories, we need to find commonalities and extremes. What do the trips have in common? Can we extract features from such trajectories, to help us understand the dataset? We propose extracting the *fractal dimension* of every such trip. The fractal dimension has several desirable properties: (a) it is invariant to affine transformations, (b) it is fast to compute, and (c) it captures the complexity of the trajectory. Next we give brief background and the intuition behind our proposed solution.

3.1 Preliminaries

(Hausdorff) Fractal Dimension. There exist many fractal dimensions (Hausdorff, Minkowski, correlation, information, etc). Among them, we use only the first one, which is formally defined as follows:

The *(Hausdorff) fractal dimension*, or simply the *fractal dimension* for a range of scales (r_1, r_2) and for a given, self-similar, point-set in an n-dimensional address space, is defined as the exponent of the law [19]:

$$N(r) \;=\; C\, r^{-FD} \quad (r_1 < r < r_2) \tag{1}$$

where (r_1, r_2) is a suitable range of scales, and $N(r)$ is the number of non-empty cells, when we impose a grid of side r, on our dataset. Intuitively, if we plot $N(r)$ vs r in log-log scales, the plot will be linear for the range of scales (r_1, r_2). We refer to such plots as *Hausdorff plots*, and we report the slope and constant (= intercept), for each taxi trajectory.

Computational Complexity. Linear. More specifically, we have:

Lemma 1. *The computation time for the fractal dimension is $O(N)$, that is, linear on the number of points N.*

Proof: Using the so-called *box-counting* method [19], we only need to go over the data points a few times. **QED**

Basic Properties for Trajectory Data. Our dataset consists of 10,000 trajectories taken from anonymized taxi drivers in the large city, where each trajectory corresponds to the trail of each taxi for an entire day. The dataset has the following attributes: GPS coordinates (i.e., longitude, latitude), a timestamp, and the passenger status (i.e., 'full' or 'empty').

Figure 3 shows an example of a taxi trajectory of an entire day. The horizontal and vertical axes show the longitude and latitude of the GPS points respectively, where red and blue lines show the trails of the taxi for each status (full and empty). Note that we anonymize latitude and longitude information due to privacy concerns. The red and blue dots indicate the locations of 'pick-up' and 'drop-off' points. The green dot at (0.3391, 0.1168) is the location of the international airport, and (0.15 - 0.25, 0.06) is a downtown area. In this figure, we can see that most trajectory lines are between the airport and downtown via highways.

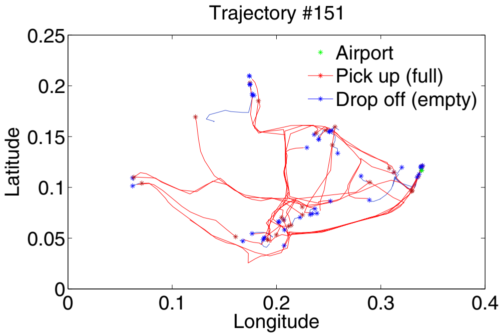


Fig. 3. Plotting conventions for a taxi trajectory: the red/blue lines indicate trails of the taxi with/without passengers, respectively. The red and blue dots indicate the locations of pick-up and drop-off points, respectively.

Here we define some terminology. We will refer to each set of trail points with passengers (shown as red lines in Figure 3), as a *'full' trail*, and refer to a set of points without passengers (i.e., empty) as an *"empty" trail*. That is, the entire trajectory consists of a combination of 'full' and 'empty' trails.

3.2 Intuition - Fractal Dimension as a Feature

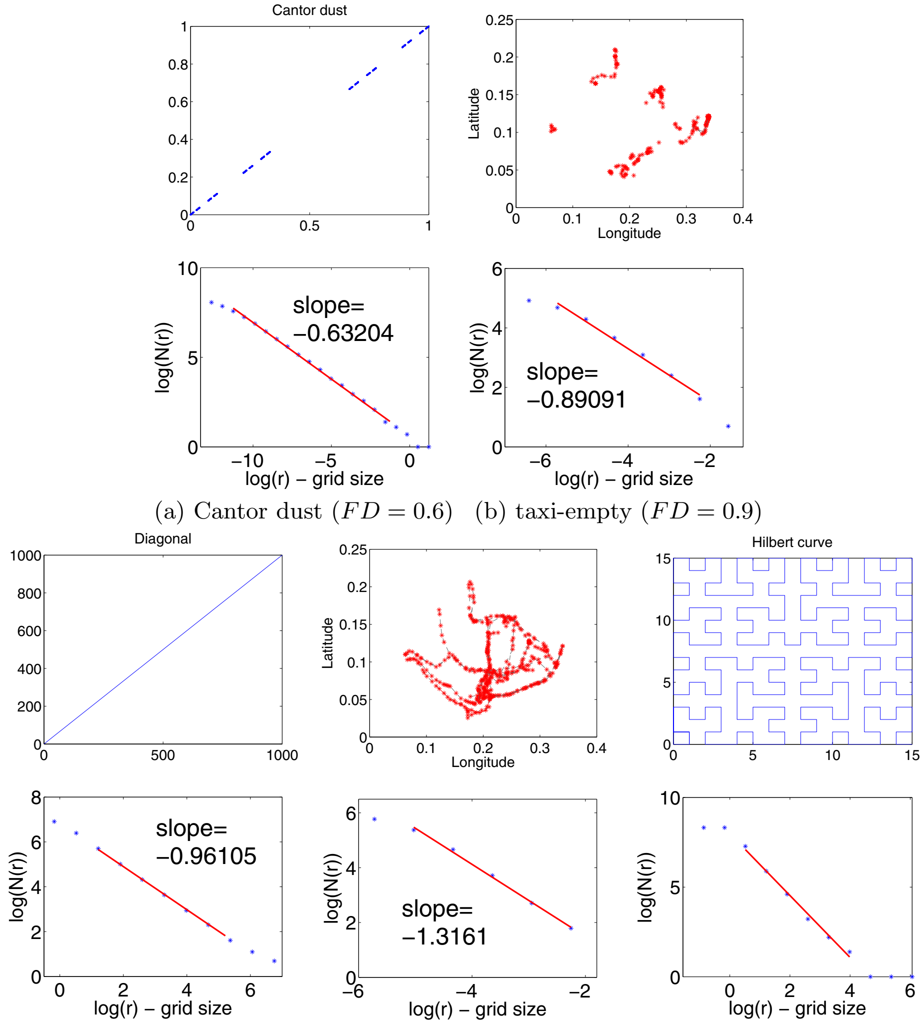
Our goal is to analyze the trajectories, and specifically to characterize the underlying behavior of taxi drivers, and gain insight into how and why the observed characteristics arise.

The taxi drivers have social and economic priorities and follow their own strategies for success. We want to extract detailed information on their behavior, especially regarding their mobility patterns. There exist numerous time-series analysis methods, including FFT and wavelets, but they depend strongly on the locations of trajectories, which makes it hard to find the economic strategies and social behavior. How can we characterize the shape of trajectories, while ignoring their location?

Approach 1. *We propose to use the fractal dimension of each taxi trajectory, as a feature for finding patterns and groups.*

Several real datasets are self-similar, and thus have an intrinsic, or *fractal* dimension: the peripheries of clouds and rain patches ($D = 1.3$), coast-lines ($D = 1.1$ to 1.58 for Norway), and many more [1]. Are our taxi trajectories self-similar? The answer is 'yes', for the vast majority of them. The ones that don't, are either too short, or deserve further examination, being the exceptions. Next we give the intuition and necessary definitions.

Figure 4 shows some trajectories as well as the tools to measure their fractal dimension: the odd columns (i.e., Figure (a,c,e)) are synthetic, and the even ones (i.e., Figure (b,d)) are real trajectories. Intuitively, a set of points (like our taxis' (x,y) coordinates per unit time) is a fractal if it exhibits self-similarity over all scales. The way to interpret the value of the fractal dimension is as follows:



(a) Cantor dust ($FD = 0.6$) (b) taxi-empty ($FD = 0.9$)

(c) line ($FD = 1$) (d) taxi-full ($FD = 1.3$) (e) Hilbert curve ($FD = 2.0$)

Fig. 4. Intuition behind trajectories and their fractal dimension(s). Top row: trajectories (sets of (x,y) points). Bottom row: the 'Hausdorff' plots, in log-log scales - the slope is the fractal dimension. Columns have trajectories of increasing fractal dimension (≈ complexity), from $FD = 0.6$ up to 2.

- $FD = 1$: This happens when the trajectory has iso-spaced points, along a line or a smooth curve, (see synthetic dataset Figure 4(c)).
- $FD > 1$: This happens when the taxi does twists and turns, like the real trajectory of Figure 4(d).
- $FD < 1$, This happens when the taxi does many stops, like the real trajectory of Figure 4(b).

– $FD = 0$, if the taxi is completely static, in which case the trajectory reduces to a (multi)point.

The two extremes in Figure 4(a,e) correspond to the so-called 'Cantor dust', and the 'Hilbert curve'. The former is derived from a line segment, by recursively deleting the middle third, and has fractal dimension $\log(2)/\log(3) \approx 0.63$. The latter is a *space filling curve*, with fractal dimension = 2, covering the whole 2-d space, in the limit.

Thus far we have introduced the fractal dimension for individual trajectory analysis in order to understand the taxi behavior. However, the behavior of each taxi could vary in a month, a week, or even in a single day since the dynamic strategy typically beats the static strategy. Actually, each empty taxi exhibits distinct behavior in different time ranges according to the distribution of passengers (see Figure 1).

Approach 2. *We propose to apply a short-window approach to the fractal dimension, which is more flexible for trajectory analysis.*

Instead of handling the entire trajectory, we locally analyze the fractal dimension of each snapshots to obtain a better understanding of time-varying social behavior.

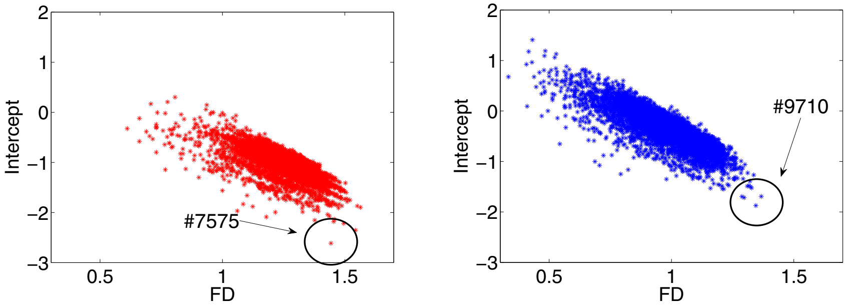
3.3 F-Trail Analysis

We now introduce our approach and describe how to analyze individual taxi behavior.

Fractal Trajectories - Are There Clusters? For a few trajectories, a human could eye-ball them, and derive the above patterns. But, how can we accomplish this automatically for thousands of trajectories? Our first idea is to compute the fractal dimensions for individual trajectory analysis. We begin by investigating the sociability of taxi movement by measuring the fractal dimensions of trajectories of two statuses (i.e., 'full' and 'empty' trails). We compute the fractal dimension of each trail, which help us to understand how the taxi drivers find the passengers and how they pick them up.

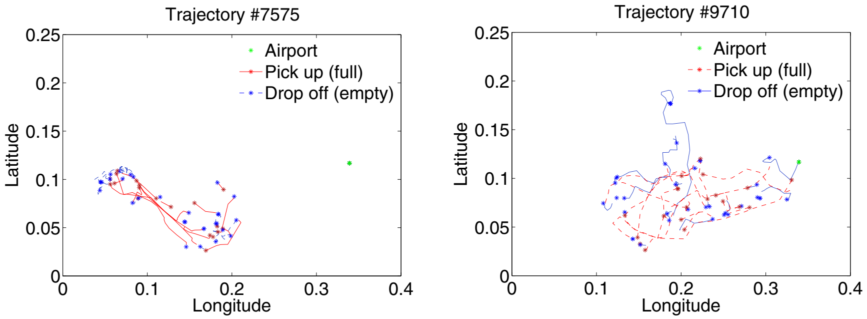
Do taxi drivers take their passengers over direct paths? Are their trajectories different, when they are empty, looking for passengers? It turns out that F-Trail can answer both questions, and the answers are 'yes', with a few twists. Let's see the details.

Figure 5 shows the fractal dimension (FD) versus the intercept of full and empty trails. For most 'full' trails, the fractal dimension is between $1.1 - 1.3$, while for 'empty' trails, it is between $0.8 - 1.1$. Here, $FD = 0$ is the burstiest (i.e., static taxi), $FD = 1$ corresponds to a taxi moving uniformly on a line or smooth curve, and $FD = 2$ (maximum) would be for a taxi that is uniformly distributed over the whole address space. This figure shows remarkable differences in the behaviors of each status. Thus, we have:



(a) FD (full trails) (b) FD (empty trails)

Fig. 5. Pattern and extremes: FD vs intercept plots of the trajectories: Each dot corresponds to a trail of (a) full taxi and (b) empty taxi. Full taxis tend to have higher fractal dimension. Also note the extremes, like 'full' taxi #7575 and 'free' taxi #9710, in circles.



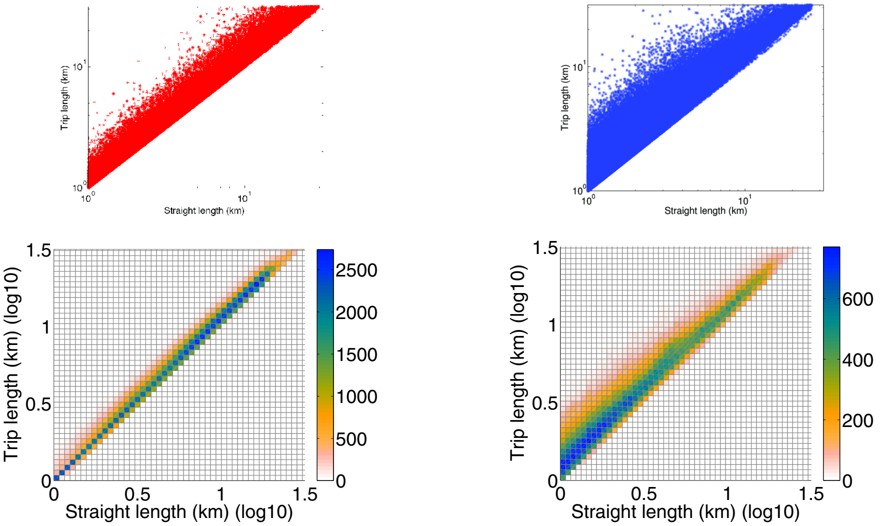
(a) Extreme (in the "full" status) (b) Extreme (in the "empty" status)

Fig. 6. Extremes of full and empty trails: (a) full trail of taxi # 7575 (red line) has high FD, but the intercept (≈ total length) is the lowest: a lot of short, straight-like rides. (b) Empty trail of taxi # 9710, having the highest FD: it's blue rides are long, and mainly straight like, which means no local 'wandering' for its next passenger.

Observation 1 (Typical behavior). *Typical taxi behavior over an entire day is to have fractal dimensions between* $FD = 1.1 - 1.3$ *for full trails and* $FD = 0.8 - 1.1$ *for empty trails.*

Notice that the above observation is invariant to affine transformations. Moreover, it helps us spot a clear distinction between 'full' and 'empty' trail sets (see Figure 5, FD vs. intercept plot). This is because, unlike many full taxis that head straight for their destinations, empty taxis frequently do many stops and turns, to find a new passenger, which leads to low fractal dimension.

A further observation that we can derive from the fractal dimensions of trajectories shown in the same figure is that there are several extremes for each of the full and empty trail sets. For instance, the trajectory #7575 is the one



(a) Full taxis (slope=1.04, intercept=0.03) (b) Free taxis (slope=0.97, intercept=0.16)

Fig. 7. Straight distance vs. trip length of each ride (log-log scales): scatter plots (top) and density plots (bottom). Full taxis pick the shortest path (intercept ≈0); empty taxis drive about 40% more length than necessary (intercept=0.16, $10^{0.16} \approx 1.4$).

extreme, due to its low intercept, which implies that the particular trajectory covers a very small area. In fact, the taxi driver adopted a different strategy and focused only on a highly-populated area (see Figure 6(a)). The trajectory #9710 is an extreme example of the empty status and shows a high fractal dimension. Actually the trajectory includes many line-like trips (long blue segments), for his/her long 'passenger search' (see Figure 6(b)), in contrast to the majority of taxi drivers, who wander locally, looking for their next passenger.

Thus we have almost answered the first question: taxis seem take their passengers to smooth, line-like curves. The next natural question is: are these the shortest paths? We address this question next.

Trip Length vs. Crow's Flight - Any Waste? So far we have examined the dynamics of the taxi behaviors at an individual level, and proposed a simple model capturing the fractal dimensions. To check whether 'full' taxis actually use the shortest path, we do another scatter-plot: For every trip, we plot the "crow's flight" length (= Euclidean distance between pick-up point and drop-off point) and the reported length (= sum of lengths between successive location snapshots). Figure 7 shows these scatter plots, where every dot is a trip. Of course, nothing is below the diagonal, and there is heavy over-plotting. The bottom row contains the density plots.

Notice that the 'full' trips are almost on the diagonal (slope=1, intercept≈0), which means that the taxi drivers are efficient, in the shortest path sense.

Observation 2 (Taxi driver efficiency). *Most taxi drivers take their passengers via the shortest path (or very close to it).*

By contrast, when 'empty', the slope is still 1, but the intercept is higher (0.16, in log-log scales), which means that the drivers 'wander', with many turns and returns, until they find a new passenger. Mathematically: $l = 10^{0.16} * s^1$ or

$$l \approx 1.4 * s \tag{2}$$

where l is the reported length of the trip, and s is the straight ("crow's flight") length.

Observation 3 (40 percent wandering). *Drivers on 'empty' status have more convoluted trajectories than on 'full' status, driving about* 40% *longer than necessary.*

This observation agrees with the intuition: Drop-offs are typically in residential areas, with many, narrow roads, and the taxi drivers have to turn and loop, until they find another passenger. Thus, the trajectory is more convoluted. In contrast, 'full' taxis typically go on highways, which are straight or with a few smooth turns and thus the trajectories are simpler and more efficient.

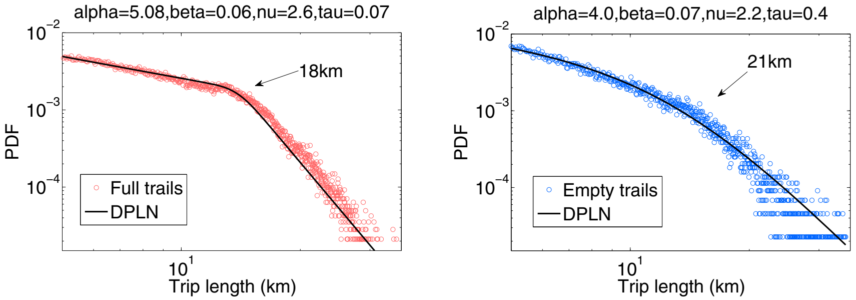
Fractal Dimension - Any Changes with the Hour of the Day? So far we have seen that most empty taxis are likely to pick up new passengers geographically close to the last drop-off point, and thus we would expect that this strategy to minimize their effort (e.g., maximize fuel efficiency, profits). However, as described in the introduction, this is not always the case. We thus introduce a *short-window* approach to the fractal dimension (FD), which enable us to characterize the taxi behavior for each time interval (i.e., snapshot). Specifically, instead of analyzing all the (x,y) snapshots of, say, taxi i, we study separately the snapshots at 0-3am, 3-6am, etc.

As we have seen in Figure 1, the behavior of 'full' taxis is the same for all hours of the day, but the 'empty' ones vary in a very interesting way:

Observation 4 ('empty' at night: differ). *There is a clearly distinct pattern of empty taxis at night time: instead of convoluted, 'wandering' paths, drivers choose line-like paths.*

In retrospect, the observation above makes sense: During the night (0-6am), the drop-off place is probably at a residential area, and there is slim chance to find another passenger nearby. Thus, taxi drivers choose to drive straight to places with higher chances of demand (airport, down-town, etc).

Power Law in the Trip-Length Distribution? Whenever there is self-similarity and fractals, we often have power laws and scale-free distributions. Does this hold for the trip-lengths, in our case? Surprisingly, it turns out that the double Pareto lognormal (DPLN) distribution yields good fits to our data.



(a) Full taxis (b) Empty taxis

Fig. 8. PDF of trip length (length vs count, in log-log scales): knees at characteristic scales of ≈20 km, which is roughly the radius of the city

The DPLN distribution generalizes the power-law and lognormal distributions, and is expressed by the following equation,

$$f(x|\alpha,\beta,\nu,\tau) = \frac{\alpha\beta}{\alpha+\beta}[\exp^{\alpha\nu+\alpha^2\tau^2/2} x^{-\alpha-1}\Phi(\frac{\log x-\nu-\alpha\tau^2}{\tau}) + x^{\beta-1}\exp^{-\beta\nu+\beta^2\tau^2/2}\Phi^c(\frac{\log x-\nu+\beta\tau^2}{\tau})] \tag{3}$$

where Φ and Φ^c are the cdf and complementary cdf of $N(0,1)$, and for further details (e.g., parameters α, β, ν, τ), see [20].

Figure 8 shows the DPLN fitting results of trip-length vs count distribution. The figure shows the PDF for the (a) full (red circles) and (b) empty (blue circles) rides, in log-log scales. There is a power-law behavior up to the 'knee' (at about 20 km), and then a sharp (power-law) drop after that. Notice that the knee is at roughly the radius of the city of study. Consequently, we have:

Observation 5 (Trip length: DPLN behavior). *The trip length distribution is skewed, for both 'full' and 'empty' rides, with a power-law that has a 'knee' at* ≈ 20 *km. This is exactly the so-called "doubly Pareto lognormal" (DPLN).*

From Figure 8 (a), we observe two types of customers: the 'below-city-radius' ones, that take short trips, or airport-to-downtown; and the (much more rare) 'above-city-radius' ones, that maybe hire a taxi for sightseeing, or for several days. Such results (i.e., trip-length DPLN fitting) could be used to analyze the tradeoff between the cost of finding passengers (Figure 8 (b)) and the fares received from them (Figure 8 (a)) to design pricing structure, which would maximize the total revenue.

4 Conclusions

We investigated patterns of human mobility on a large collection of taxi trajectories, and proposed a new method, F-Trail, to find meaningful patterns and anomalies. Our approach is (a) linear on the input size, and (b) able to spot meaningful general trends, as well as outliers. In more detail, our main discoveries are as follows:

- Typical for taxi trajectories is to have fractal dimension between 1.1 to 1.3 for full trails and 0.8 to 1.1 for empty trails, which means that most taxis take their passengers to smooth, line-like curves, while empty taxis have many stops or bursty-like patterns.
- Most surprisingly, we found that most taxi drivers change their strategies in different time ranges according to passenger demand, i.e., there was a very interesting deviation of FD for empty taxi during the night.

Acknowledgement. We would like to thank the taxi company that provides us the data. This work was sponsored by the Defense Threat Reduction Agency and was accomplished under contract No. HDTRA1-10-1-0120 and by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053.

References

1. Mei, Q., Zhai, C.X.: Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In: Proceedings of the KDD 2005, pp. 198–207. ACM (2005)
2. Chudova, D., Gaffney, S., Mjolsness, E., Smyth, P.: Translation-invariant mixture models for curve clustering. In: KDD, pp. 79–88 (2003)
3. Cudre-Mauroux, P., Wu, E., Madden, S.: Trajstore: An adaptive storage system for very large trajectory data sets. In: ICDE, pp. 109–120 (March 2010)
4. Gaffney, S., Smyth, P.: Trajectory clustering with mixtures of regression models. In: KDD, pp. 63–72 (1999)
5. Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D.: Trajectory pattern mining. In: KDD, pp. 330–339 (2007)
6. Hadjieleftheriou, M., Kollios, G., Tsotras, V.J., Gunopulos, D.: Indexing spatiotemporal archives. VLDB J. 15(2), 143–164 (2006)
7. Lee, J.-G., Han, J., Li, X.: Trajectory outlier detection: A partition-and-detect framework. In: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, pp. 140–149. IEEE Computer Society, Washington, DC (2008)
8. Lee, J.-G., Han, J., Whang, K.-Y.: Trajectory clustering: a partition-and-group framework. In: SIGMOD, pp. 593–604 (2007)
9. Leskovec, J., Chakrabarti, D., Kleinberg, J.M., Faloutsos, C.: Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 133–145. Springer, Heidelberg (2005)
10. Li, Z., Ding, B., Han, J., Kays, R., Nye, P.: Mining periodic behaviors for moving objects. In: KDD, pp. 1099–1108 (2010)
11. Liu, W., Zheng, Y., Chawla, S., Yuan, J., Xing, X.: Discovering spatio-temporal causal interactions in traffic data streams. In: KDD, pp. 1010–1018 (2011)
12. Mandelbrot, B.: Fractal Geometry of Nature. W.H. Freeman, New York (1977)
13. Matsubara, Y., Sakurai, Y., Faloutsos, C., Iwata, T., Yoshikawa, M.: Fast mining and forecasting of complex time-stamped events. In: KDD, pp. 271–279 (2012)

14. Matsubara, Y., Sakurai, Y., Prakash, B.A., Li, L., Faloutsos, C.: Rise and fall patterns of information diffusion: model and implications. In: KDD, pp. 6–14 (2012)
15. Matsubara, Y., Sakurai, Y., Yoshikawa, M.: Scalable algorithms for distribution search. In: ICDM, pp. 347–356 (2009)
16. Papadias, D., Tao, Y., Zhang, J., Mamoulis, N., Shen, Q., Sun, J.: Indexing and retrieval of historical aggregate information about moving objects. In: IEEE Data Engineering Bulletin (2002)
17. Peitgen, H.-O., Juergens, H., Saupe, D.: Chaos and Fractals: New Frontiers of Science. Springer-Verlag New York Inc. (1992)
18. Sakurai, Y., Faloutsos, C., Yamamuro, M.: Stream monitoring under the time warping distance. In: Proceedings of ICDE, Istanbul, Turkey, pp. 1046–1055 (April 2007)
19. Schroeder, M.: Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise. W.H. Freeman and Company, New York (1991)
20. Seshadri, M., Machiraju, S., Sridharan, A., Bolot, J., Faloutsos, C., Leskovec, J.: Mobile call graphs: beyond power-law and lognormal distributions. In: KDD, Las Vegas, Nevada, USA, pp. 596–604 (2008)
21. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. In: Proceedings of the 18th International Conference on Data Engineering, pp. 673–684 (2002)
22. Willinger, W., Taqqu, M., Sherman, R., Wilson, D.V.: Self-similarity through high variability: statistical analysis of ethernet LAN traffic at the source level. ACM SIGCOMM 1995. Computer Communication Review 25, 100–113 (1995)
23. Yuan, J., Zheng, Y., Xie, X.: Discovering regions of different functions in a city using human mobility and pois. In: KDD, pp. 186–194 (2012)
24. Yuan, J., Zheng, Y., Xie, X., Sun, G.: Driving with knowledge from the physical world. In: KDD, pp. 316–324 (2011)

Mining Appliance Usage Patterns in Smart Home Environment

Yi-Cheng Chen[1], Yu-Lun Ko[1], Wen-Chih Peng[1], and Wang-Chien Lee[2]

[1] Department of Computer Science, National Chiao Tung University, Taiwan
[2] Department of Computer Science and Engineering, The Pennsylvania State University, USA
{ejen.cs95g,jimmy.cs99g}@nctu.edu.tw, wcpeng@cs.nctu.edu.tw,
wlee@cse.psu.edu

Abstract. Nowadays, due to the great advent of sensor technology, the data of all appliances in a house can be collected easily. However, with a huge amount of appliance usage log data, it is not an easy task for residents to visualize how the appliances are used. Mining algorithms is necessary to discover appliance usage patterns that capture representative usage behavior of appliances. If some of our representative patterns of appliance electricity usages are available, we may be able to adapt our usage behaviors to conserve the energy easily. In this paper, we introduce (i) two types of usage patterns which capture the representative usage behaviors of appliances in a smart home environment and (ii) the corresponding algorithms for discovering usage patterns efficiently. Finally, we apply our algorithms on a real-world dataset to show the practicability of usage pattern mining.

Keywords: appliance usage pattern, interval-based sequential pattern, smart environment.

1 Introduction

Recently, concerns over global climate changes have motivated significant efforts in reducing the emissions of CO_2 and other GHGs (greenhouse gases). Many researchers focus on the reduction of electricity usage in the residential sector because it is a significant contributor of greenhouse gas emissions. Nevertheless, electricity conservation is an arduous task for the residential users due to the lack of detailed electricity usage. If an electricity bill is high, we usually can tell that it is expensive rather than "why" it is expensive. We argue that if representative patterns of appliance electricity usages are available, residents can adapt their appliance usage behaviors to conserve the energy effectively.

Due to the great advent of sensor technology, the electricity usage data of all appliances in a house can be collected easily. In particular, power meters are deployed to collect appliance usage log data. With the appliance usage log data, one could visualize how the appliances are used. With a huge amount of appliance usage log data collected, it is necessary to propose mining algorithms to discover appliance usage patterns that capture representative usage behavior of appliances. Appliance usage

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 99–110, 2013.

patterns are able to help users understand how they use their appliances and furthermore, appliance usage patterns could be used to detect abnormal usages of appliances or utilize to design an intelligent control for appliances.

However, extracting meaningful usage information is a complex issue. Our appliance usage behaviors usually vary at different time and season, e.g., many behaviors of the same appliances in summer and in winter are totally different. For example, the air-conditioner may be used everyday in summers, but hardly turned on in other seasons. Many appliances may also have unique usage patterns, e.g., some appliances are seasonal-type and some ones belong to daily-type. For example, while the air-conditioner (a seasonal appliance) usually is turned on frequently only in the summer, the light usually is turned on and off frequently everyday and thus belongs to daily-type. Obviously, how to mine representative usage patterns which can describe diversified appliance usage effectively and efficiently is a challenging problem.

Previous researches of usage patterns mainly focused on energy disaggregation [2, 3, 4, 8, 9, 12] and appliance recognition [1, 5, 6, 7, 10, 11]. To the best of our knowledge, very few studies facilitate the energy saving with the discovery and utilization of appliance usage patterns. The contributions of this paper are: (1) from analyzing appliance usage data, two types of usage patterns, daily behavior-based usage pattern (DBUP) and clustered-based statistical usage pattern (CSUP), are proposed to represent and describe the complex behaviors of appliance usage effectively; (2) two mining algorithms are developed to discover appliance usage patterns efficiently by employing some effective pruning strategies to reduce the search space; (3) the proposed methods have been applied to a real dataset to validate its practicability and show the support for residents to adapt their appliance usage behaviors for conserve the energy.

The rest of the paper is organized as follows. Section 2 and Section 3 provide the related work and preliminary, respectively. Section 4 introduces the two types of appliance usage patterns. Section 5 presents the result of experimental evaluation. Finally, Section 6 concludes the paper.

2 Related Work

In this section, we discuss some previous works utilized usage patterns for energy disaggregation and appliance recognition. Farinaccio et al. [3] used some patterns, such as number of ON-OFF switches, to disaggregate the whole-house electricity consumption into a number of major end-uses. Suzuki et al. [12] used a new NIALM technique based on integer programming to disaggregate residential power use. Lin et al. [9] used a dynamic Bayesian network and filter to disaggregate the data online. Kim et al. [8] investigated the effectiveness of several unsupervised disaggregation methods on low frequency power measurements collected in real homes. They also proposed a usage pattern which consists of on-duration distribution and dependency between appliances. Goncalves et al. [4] explored an unsupervised approach to determine the number of appliances in the household, including their power consumption and state, at any given moment. Chen et al. [2] disaggregated utility consumption from smart meters into specific usage that associated with human activities. They proposed a novel statistical framework for disaggregation on coarse granular smart meter readings by modeling fixture characteristic, household behavior, and activity correlations.

Prudenzi [11] utilized an artificial neural network based procedure for identifying the electrical signatures of residential appliances. Ito et al. [5] extracted features from the current (e.g., amplitude, form, timing) to develop appliance signatures. For appliance recognition, Kato et al. [6] used Principal Component Analysis to extract features from electric signals and classified them by Support Vector Machine. Aritoni et al. [1] proposed a software prototype which can be used to understand the household appliances behavior. Some of these works and the characteristics of workable solutions were discussed by Matthews et al. [10].

3 Preliminaries

Definition 1 (usage-point and usage-point log). Let $\mathcal{S}= \{ON, OFF\}$ be the set of states. Without loss of generality, we define a set of uniformly spaced time points based on the natural number N. We say the pair $(s_i, t_i) \in \mathcal{S} \times N$ is an usage-point, where $s_i \in \mathcal{S}$, $t_i \in N$. The usage-point log of an appliance is $U_P = \langle D_1, D_2, \ldots, D_n \rangle$, where D_i is the daily usage-point sequence of the appliance, $1 \le i \le n$. D_i is a sequence of usage-point, i.e., $D_i = \langle (s_{i1}, t_{i1}), (s_{i2}, t_{i2}), \ldots, (s_{im}, t_{im}) \rangle$.

Definition 2 (*ON*-switch and *OFF*-switch). Given a daily usage-point sequence $D_i = \langle (s_{i1}, t_{i1}), (s_{i2}, t_{i2}), \ldots, (s_{im}, t_{im}) \rangle$ of an appliance, the *ON*-switch is the usage point (s_{ij}, t_{ij}), where $1 \le j \le m$, $s_{ij} = ON$ and $s_{i(j-1)} = OFF$; the *OFF*-switch is the usage point (s_{ik}, t_{ik}), where $1 \le k \le m$, $s_{ik} = OFF$ and $s_{i(k-1)} = ON$.

Definition 3 (usage-interval and usage-interval log). Given a daily usage-point sequence $D_i = \langle (s_{i1}, t_{i1}), (s_{i2}, t_{i2}), \ldots, (s_{im}, t_{im}) \rangle$ of an appliance, let t_{ip} and t_{iq} be the time of *j*th *ON*-switch (ON, t_{ip}) and *j*th *OFF*-switch (OFF, t_{iq}) in D_i, respectively[1]. The *j*th usage-interval of D_i is thus the duration time between the *j*th *ON*-switch and *j*th *OFF*-switch, represented as a triplet $(j, t_{ij}, d_{ij}) \in N \times N \times N$, where $t_{ij} = t_{ip}$ and $d_{ij} = t_{iq} - t_{ip}$. The usage-interval log of an appliance is $U_I = \langle I_1, I_2, \ldots, I_n \rangle$, where I_i is the daily usage-interval sequence of the appliance, $1 \le i \le n$. I_i is a sequence of usage-interval, i.e., $I_{\mathrm{i}} = \langle (1, t_{i1}, d_{i1}), (2, t_{i2}, d_{i2}), \ldots, (\ell, t_{i\ell}, d_{i\ell}) \rangle$.

We use Fig. 1 as an example to explain Definition 2 and Definition 3. Given a daily usage-point sequence of light as shown in Fig. 1, obviously, the first *ON*-switch and *OFF*-switch are (*ON*, 6:00) and (*OFF*, 8:00), respectively; the second *ON*-switch and *OFF*-switch are (*ON*, 14:00) and (*OFF*, 20:00), respectively. Hence, the usage interval sequence of 14 April, 2013 is ⟨(1, 6:00, 2 hrs), (2, 14:00, 6 hrs)⟩.

[1] Without loss of generality, in Definition 3, we assume that an appliance is turned *ON* before it is turned *OFF* in a daily usage log.

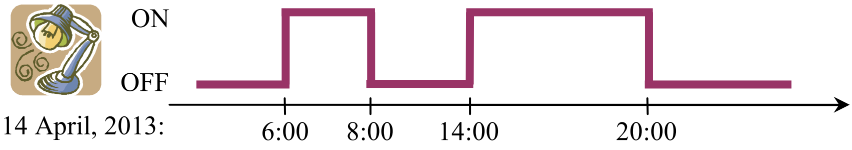


Fig. 1. An example of daily usage-point sequence

4 Mining Appliance Usage Patterns

In this section, we will define two types of appliance usage patterns, *daily behavior-based usage pattern* (*DBUP*) and *clustered-based statistical usage pattern* (*CSUP*). For each type of appliance usage patterns, we will first define its representation and then propose our algorithms to mine appliance usage patterns.

4.1 Algorithm of Mining Daily Behavior-Based Usage Pattern

DBUP mainly focuses on extracting representative behaviors of daily appliance usage. The approach is shown in Fig. 2. For ex tracting DBUP from an usage point log, first, we treat a dail y usage-point sequence as a daily behavior of an appliance. Then, similar daily usage behaviors are clu stered in the same group. A hierarchical cluster method is proposed to group similar behaviors together. Finally, we evaluate the centroid behaviors of those clusters and output them as the daily behavior-based usage patterns.

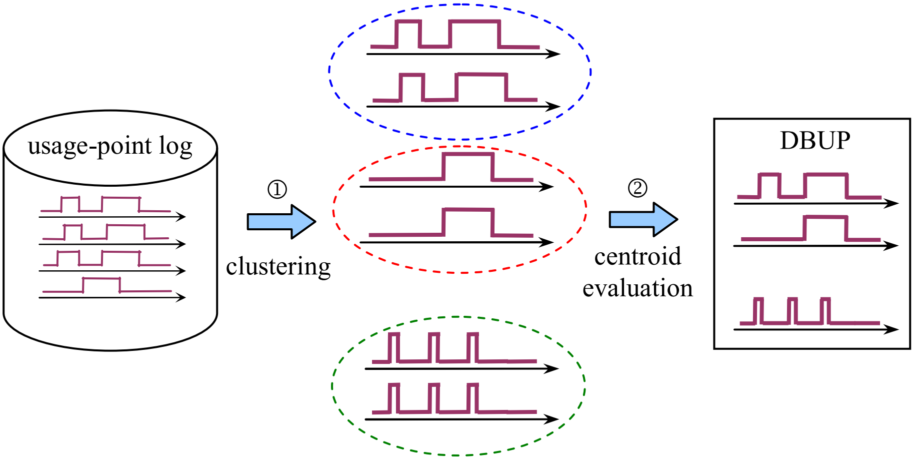


Fig. 2. Mining daily behavior-based usage patterns

An efficient method, DBUP-Miner, is developed for mining daily behavior-based usage patterns. The pseudo code i s shown in Algorithm 1. DBUP-Miner first calls sub-procedure Hierarchical_Clustering to enumerate the clusters of all daily usage-point sequences (line 1, algorithm 1). Hierarchical_Clustering is designed to cluster the similar daily usage behavior. At first, each daily usage-point sequence is considered as a cluster (line 4, algorithm 1). To cluster usage-point sequences (i.e., 0 and 1 time series sequences), we need to measure the similarity among all input time series sequences. In the clustering process, each piece of time series data can be viewed as a point located in an abstract space, and the distances between these points are usually figured out by a similarity function. A time series similarity qualifies the distance between the sequences of time series data as points in the clustering space. One important point for similarity function is that the time-shifting constrain needs to be considered, i.e., the range of local time shift should be limited. In this paper, we adopt EDR as the similarity function that can deal with local time shifting under a time shifting constrain, and can deal with noises without being compromised with too much amplitude shifting.

```
Algorithm 1: DBUP-Miner ($U_p$)

Input: An usage-point log $U_p = \langle D_1, D_2, \ldots, D_n \rangle$, where $D_i$ is the daily usage-
        point sequence $\langle (s_{i1}, t_{i1}), (s_{i2}, t_{i2}), \ldots, (s_{im}, t_{im}) \rangle$
Output: A set of usage-point sequences $\{S1, S2, \ldots\}$

01: $CU_p \leftarrow$ Hierarchical_Clustering ($U_p$);
02: $\{S1, S2, \ldots\} \leftarrow$ evaluate the centroid of each cluster in $CU_p$;
03: output $\{S1, S2, \ldots\}$;

Procedure Hierarchical_Clustering ($U_p$)
04: Let each usage-point sequence in $U_p$ be a cluster;
05: for each cluster $C_j \in U_p$ do
06:     for each cluster $C_k \in U_p - C_j$ do
07:         if (distance ($C_j$, $C_k$) $\leq \sigma$) and (distance ($C_j$, $C_k$) is minimum in $U_p$) then
08:             merge $C_j$ and $C_k$ to a new cluster $C_r$;
09:             update distance between $C_r$ and other clusters;
10:             $U_p \leftarrow U_p - \{C_j, C_k\}$; $U_p \leftarrow U_p \cup C_r$;
11: output $U_p$;
```

For each two clusters, if their distance is smaller than or equal to the threshold σ and is minimum, DBUP-Miner merges two clusters to a new cluster and updates the similarity between new cluster and other clusters (Lines 5-9, algorithm 1). The setting of threshold σ is usually heuristic. We will discuss in detail in experimental results and show the effect upon the mining results. Finally, after clustering, we evaluate the centroid of the each cluster and output these representative daily usage behaviors (lines 2-3, algorithm 1). We compute mean of each cluster as the representative centroid. With DBUP-Miner, we can obtain the daily behavior-based usage patterns efficiently.

4.2 Algorithm of Mining Cluster-Based Statistical Usage Pattern

For an appliance, the starting time and the duration of the *j*th *ON-OFF* switch in each day are also informative. We propose to employ CSUP, to extract the representative usage-interval from the daily log. CSUP clusters each starting time and duration of the *j*th usage-interval on an appliance in a given usage-interval log. Notice that, different to DBUP, the input data for extracting CSUP is a set of usage-interval sequences instead of usage-point sequence, since we want to discover some general behavior to express the starting time and duration of each *j*th usage-interval. CSUP adopts a hierarchical method to cluster similar usage-intervals in the same group. Finally, we evaluate the centroid usage-intervals of the clusters and output them as a clustered-based statistical usage pattern. The approach of mining CSUP is as shown in Fig. 3.

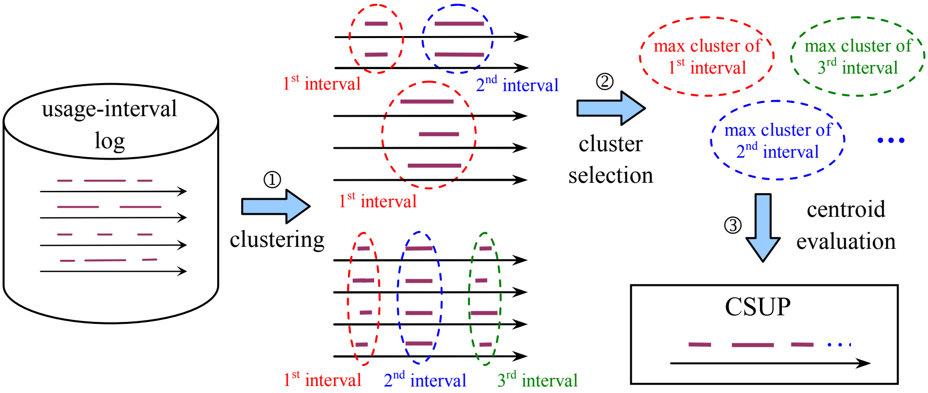


Fig. 3. Mining clustered-based statistical usage pattern

Algorithm 2 illustrates the main framework which includes the necessary processing steps of CSUP-Miner. Given usage-interval log, minimum support μ, CSUP Miner first accumulates the occurrences of the *j*th usage-interval in each usage-interval sequence and store it into a matrix number[*j*] (j=1,2,…) (lines 1-3, algorithm 2). Then, we check whether the count in number[*j*] is larger than or equal to minimum count $\mu \times n$. If number[*j*] is smaller than $\mu \times n$, the *j*th usage-interval will be pruned, since it is nonrepresentative to describe a usage behavior (line 4, algorithm 2). For the frequent *j*th usage-interval, CSUP-Miner calls the sub-procedure Hierarchical_Clustering to cluster the similar usage-intervals and stores the result of the clustering into CG_j (lines 5-6, algorithm 2).

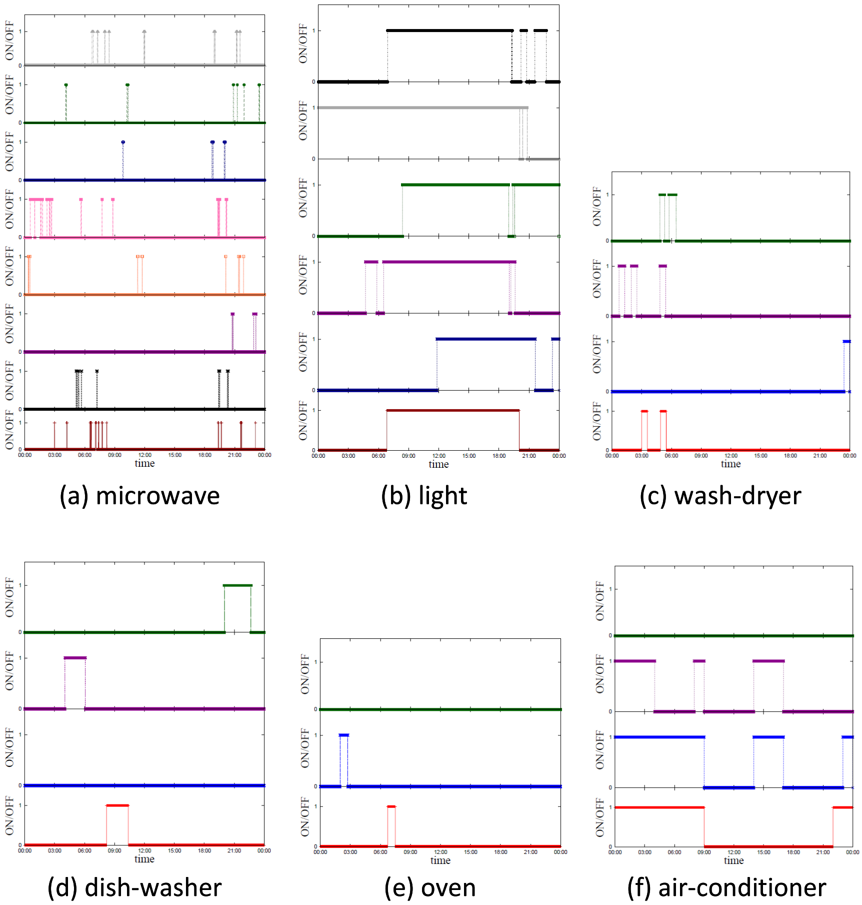
The clustering method is similar to the one used in DBUP-Miner. The hierarchical clustering method utilizes two attributes, starting time and duration, for clustering all *j*th usage-intervals. At first, each usage-interval (i.e., (j, t_{ij}, d_{ij})) is a cluster (line 9, algorithm 2) and we group two closest usage-intervals which have the closest distance and the distance is smaller than or equal to threshold σ (lines 12-13, algorithm 2). The effect of threshold σ will be discussed in experimental results. Then, we update the similarity between new cluster and other clusters (line 14, algorithm 2) and recursively run above steps until the distances among all clusters are not smaller than or equal to σ(lines 10-15, algorithm 2).

To find the representative usage-interval to describe the *j*th time for turning on the appliance, we select the maximum cluster from CG_j and evaluate the centroid (mean starting time and mean duration) of all usage-intervals in this cluster to represent the *j*th usage-interval (line 7, algorithm 2). Finally, we output all representative usage-intervals as the clustered-based statistical usage patterns (line 8, algorithm 2).

```
Algorithm 2: CSUP-Miner ($U_I$, $\mu$)

Input: An usage-interval log $U_I = \langle I_1, I_2, \ldots, I_n \rangle$, where $I_i$ is the daily usage-interval sequence
        $\langle (1, t_{i1}, d_{i1}),\ (2, t_{i2}, d_{i2}), \ldots, (\ell, t_{i\ell}, d_{i\ell}) \rangle$ ; minimum support $\mu$
Output: A representative usage-interval sequence $\{(1, \tau_1, \delta_1), (2, \tau_2, \delta_2), \ldots\}$

01: for each $I_i \in U_I$ do
02:     for $j = 1$ to $\ell$ do
03:         $number[j] \leftarrow number[j] + 1$;    // accumulate the occurrences of usage-intervals;
04: for each $j$ that $number[j] \geq (\mu * n)$ do    // remove the infrequent usage-interval;
05:     $G_j \leftarrow \{(j, t_{1j}, d_{1j}), (j, t_{2j}, d_{2j}), \ldots, (j, t_{nj}, d_{nj})\}$;
06:     $CG_j \leftarrow$ Hierarchical_Clustering ($G_j$);
07:     $(j, \tau_j, \delta_j) \leftarrow$ evaluate the centroid of maximum cluster in $CG_j$;
08: output all $(j, \tau_j, \delta_j)$;

Procedure Hierarchical_Clustering (G)
09: Let each usage-nterval in $G$ be a cluster;
10: for each cluster $C_p \in G$ do
11:     for each cluster $C_q \in G - C_p$ do
12:         if (distance ($C_p$, $C_q$) $\leq \sigma$) and (distance ($C_p$, $C_q$) is minimum in $G$) then
13:             merge $C_p$ and $C_q$ to a new cluster $C_r$;
14:             update distance between $C_r$ and other clusters;
15:             $G \leftarrow G - \{C_p, C_q\}$ ; $G \leftarrow G \cup C_r$ ;
16: output $G$;
```

5 Experimental Results

All algorithms were implemented in C++ language and tested on a Pentium D 3.0 GHz with 2 GB of main memory running Windows XP system. This performance study has been conducted on real world dataset. The real dataset is collected in a smart home environment, which consists six appliances (microwave, dish-washer, wash-dryer, light, oven and air-conditioner) for 45 days. We transfer the power consumption to the state of *ON* or *OFF* for each appliance. Hence, a day has about 22,550 usage points. To show the practicability of proposed two types of usage patterns, we perform two kinds of experiments. First, we extract DBUP and CSUP from a real dataset. From discovered patterns, the characteristic of each appliance is elaborated. Then, the relationship between threshold setting and the pattern length are discussed in detail. We also indicate the influence of the minimum threshold.



(a) microwave (b) light (c) wash-dryer

(d) dish-washer (e) oven (f) air-conditioner

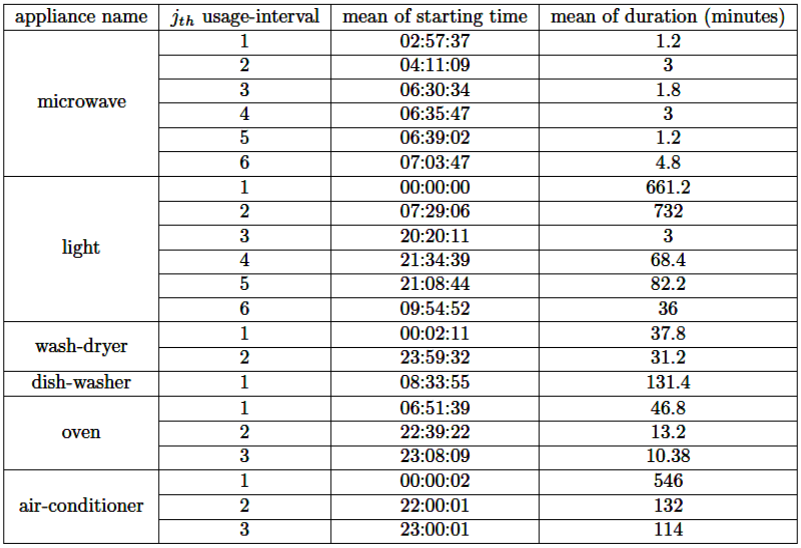
Fig. 4. Daily behavior-based usage pattern mined from collected real dataset

5.1 DBUP and CSUP from Real Dataset

Fig. 4 shows the discovered daily behavior-based usage pattern of six appliances with similarity threshold σ = 800. The setting of threshold σ is set empirically which will be discussed in details in next section. The x axis is daily time (from 00:00:00 to 23:59:59) and the y axis is the states of *ON* and *OFF*. The curves with different colors represent different daily behaviors. DBUP reveals the usage information of an appliance, such as how many times the appliance is turned on, the starting time of usage, and duration of usage. For example, Fig. 4(a) shows the DBUP of a microwave which exhibits eight representative behaviors (different colors of curve). The third behavior (black curve) describes three times for turning on the microwave. The first turn-on time is about 08:45 and the duration is about 50 seconds; the second turn-on time is about 19:00 and the duration is also about 2 minutes; the last turn-on time is about 19:30 and the duration is about 45 seconds.

We also can observe that different appliance has different usage characteristic, i.e., different distribution of using time and duration of turn-on time. For example, in Fig. 4(a), the microwave, in general, is used evenly in a day and has short usage duration; however, oven usually is turned on at a fixed time and has longer usage duration as shown in Fig. 4(b). Hence, with DBUP, residents can know their representative behaviors of daily appliance usage.

Table 1. Clustered-based statistical usage pattern mined from collected real dataset



appliance name	j_{th} usage-interval	mean of starting time	mean of duration (minutes)
microwave	1	02:57:37	1.2
	2	04:11:09	3
	3	06:30:34	1.8
	4	06:35:47	3
	5	06:39:02	1.2
	6	07:03:47	4.8
light	1	00:00:00	661.2
	2	07:29:06	732
	3	20:20:11	3
	4	21:34:39	68.4
	5	21:08:44	82.2
	6	09:54:52	36
wash-dryer	1	00:02:11	37.8
	2	23:59:32	31.2
dish-washer	1	08:33:55	131.4
oven	1	06:51:39	46.8
	2	22:39:22	13.2
	3	23:08:09	10.38
air-conditioner	1	00:00:02	546
	2	22:00:01	132
	3	23:00:01	114

Table 1 lists the discovered clustered-based statistical usage pattern of six appliances (with minimum support μ = 10%). The setting of threshold μ will be discussed in next section. For an appliance, CSUP can provide residents three information, how many times the appliance is turned on, mean of starting time, and mean of the usage duration. We use the microwave in Table 1 for discussion. There are six usage-intervals for the microwave. The mean starting time and mean duration of first usage interval are 02:57:37 and 1.2 minutes, respectively. In other words, it expresses that residents usually use the microwave six times in a day and turn on the microwave at about 02:57 and use it about 1.2 minutes in average at first usage.

Different appliances have different usage frequencies and turn-on times. For example, as shown in Table 1, the microwave usually is turned on/off six times a day, but the dish washer usually is used only once a day. The usage durations of two appliances are also different, e.g., oven usually is used for a long time due to the nature of its usage. From the discovered CSUP, residents can know, in general, how many times they use an appliance in one day, and each usage exists for how long. In summary, DBUP and CSUP provide residents their representative patterns of appliance usage. With this useful information, resident may make appropriate adaption for electricity conservation more easily.

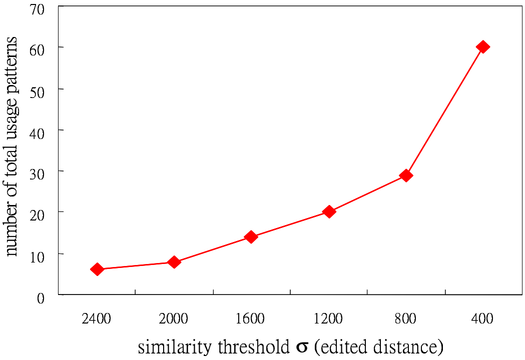


Fig. 5. The number of generated DBUP patterns with varying similarity threshold

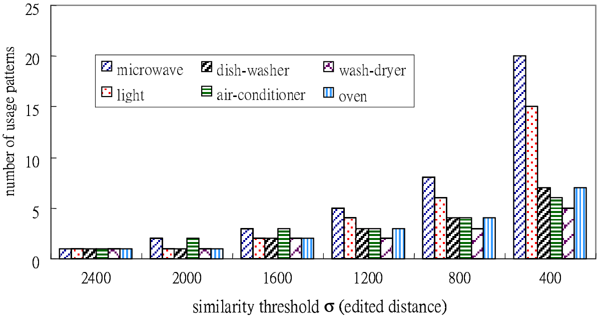


Fig. 6. The distribution of generated DBUP patterns with varying minimum support

5.2 The Influence of Minimum Thresholds on Mining Results

In the following experiments, we examine the effects of the threshold setting. Appropriate similarity threshold σ and minimum support μ are very critical for mining DBUP and CSUP. Fig. 5 and Fig. 6 show the number of generated daily behavior-based usage patterns and the distribution with varying similarity threshold σ. The setting of σ will affect the number of generated DBUP directly. As mentioned in Section 4.1, DBUP Miner uses EDR as the similarity function for hierarchical clustering. EDR computes the total edit distances between two input usage-point sequences (i.e., 0 and 1 time series sequences); and the similarity threshold decide how closed the sequences are grouped into the same cluster. Obviously, the smaller the similarity threshold, the larger the number of DBUP patterns. As shown in Fig 5, when similarity threshold is 400, 62 DBUPs are extracted from the collected real dataset. As can be seen from Fig. 6, we can obtain an amount of usage patterns for all appliances with σ = 400. When the similarity threshold is greater than 1,600, most of the generated usage patterns are with length one or two.

Finally, we discuss the relation between the length of generated CSUP and the minimum support μ. In Fig. 7, we show the length of usage-interval sequence (CSUP) on our real dataset with varying minimum support thresholds which is from 10% to 90%. Obviously, when the minimum support value increases, the length of generated patterns decreases. As shown in the figure, we can generate longer CSUP, especially for microwave and light, if the minimum support μ is larger than 30%. However, when the minimum support reaches to 50%, some appliances do not have any CSUP patterns (dish-washer, air-conditioner, wash-dryer and oven). Consequently, from the experiments, we can observe that the threshold setting is very critical for the process of mining DBUP and CSUP.

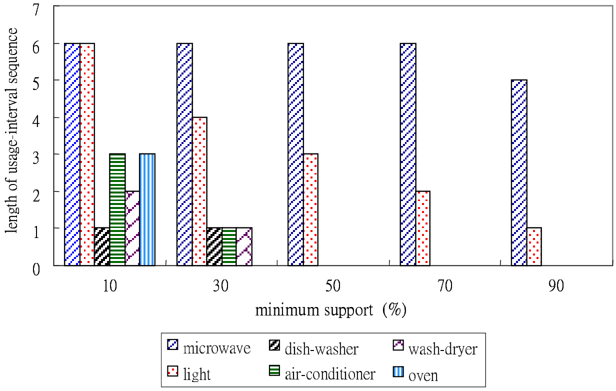


Fig. 7. Experiments of influence on minimum support setting

6 Conclusion

Recently, considerable concern has arisen over the electricity conservation due to the issue of greenhouse gas emissions. If representative behaviors of appliance usages are available, residents may adapt their appliance usage behaviors to conserve energy effectively. However, how to extract representative usage patterns which can describe diversified appliance usage effectively and efficiently is a challenging issue. In this paper, from analyzing appliance usage data, two types of usage patterns, daily behavior-based usage pattern (DBUP) and clustered-based statistical usage pattern (CSUP), are proposed to represent and describe the complex behaviors of appliance usage. The mining algorithms also have been developed to discover DBUP and CSUP efficiently. Finally, the proposed methods have been applied to real dataset to validate the practicability and showed the support for residents to adapt their appliance usage behaviors for electricity conservation.

Acknowledgement. Wen-Chih Peng was supported in part by the National Science Council, Project No. 100-2218-E-009-016-MY3 and 100-2218-E-009-013-MY3, by Taiwan MoE ATU Program, by D-Link and by Microsoft.

References

1. Aritoni, O., Negru, V.: A Methodology for Household Appliances Behavior Rec-ognition in AmI Systems Integration. In: 7th International Conference on Auto-matic and Autonomous Systems (ICAS 2011), pp. 175–178 (2011)
2. Chen, F., Dai, J., Wang, B., Sahu, S., Naphade, M., Lu, C.T.: Activity Analysis Based on Low Sample Rate Smart Meters. In: 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2011), pp. 240–248 (2011)
3. Farinaccio, L., Zmeureanu, R.: Using a pattern recognition approach to disaggregate the total electricity consumption in a house into the major end-uses. Energy and Buildings 30(3), 245–259 (1999)
4. Goncalves, H., Ocneanu, A., Bergés, M.: Unsupervised disaggregation of appliances using aggregated consumption data. In: KDD Workshop on Data Mining Applications in Sustainability (SustKDD 2011) (2011)
5. Ito, M., Uda, R., Ichimura, S., Tago, K., Hoshi, T., Matsushita, Y.: A method of appliance detection based on features of power waveform. In: 4th IEEE Symposium on Applications and the Internet (SAINT 2004), pp. 291–294 (2004)
6. Kato, T., Cho, H.S., Lee, D., Toyomura, T., Yamazaki, T.: Appliance recognition from electric current signals for information-energy integrated network in home environments. In: Mokhtari, M., Khalil, I., Bauchet, J., Zhang, D., Nugent, C. (eds.) ICOST 2009. LNCS, vol. 5597, pp. 150–157. Springer, Heidelberg (2009)
7. Kolter, J.Z., Johnson, M.J.: REDD: A public data set for energy disaggregation research. In: KDD Workshop on Data Mining Applications in Sustainability (SustKDD 2011) (2011)
8. Kim, H., Marwah, M., Arlitt, M., Lyon, G., Han, J.: Unsupervised disaggregation of low frequency power measurements. In: 11th SIAM International Conference on Data Mining (SDM 2011), pp. 747–758 (2011)
9. Lin, G., Lee, S., Hsu, J., Jih, W.: Applying power meters for appliance recognition on the electric panel. In: 5th IEEE Conference on Industrial Electronics and Applications (ISIEA 2010), pp. 2254–2259 (2010)
10. Matthews, H., Soibelman, L., Berges, M., Goldman, E.: Automatically disaggre-gating the total electrical load in residential buildings: a profile of the required solution. In: Intelligent Computing in Engineering, pp. 381–389 (2008)
11. Prudenzi, A.: A neuron nets based procedure for identifying domestic appliances pattern-of-use from energy recordings at meter panel. In: IEEE Power Engineering Society Winter Meeting, vol. 2, pp. 491–496 (2002)
12. Suzuki, K., Inagaki, S., Suzuki, T., Nakamura, H., Ito, K.: Nonintrusive appliance load monitoring based on integer programming. In: International Conference on Instrumentation, Control and Information Technology, pp. 2742–2747 (2008)

Computational Models of Stress in Reading Using Physiological and Physical Sensor Data

Nandita Sharma and Tom Gedeon

Research School of Computer Science
Australian National University, Canberra, Australia
{Nandita.Sharma,Tom.Gedeon}@anu.edu.au

Abstract. Stress is a major problem facing our world today and it is important to develop an objective understanding of how average individuals respond to stress in a typical activity like reading. The aim for this paper is to determine whether stress patterns can be recognized using individual-independent computational models from sensor based stress response signals induced by reading text with stressful content. The response signals were obtained by sensors that sourced various physiological and physical signals, from which hundreds of features were derived. The paper proposes feature selection methods to deal with redundant and irrelevant features and improve the performance of classifications obtained from models based on artificial neural networks (ANNs) and support vector machines (SVMs). A genetic algorithm (GA) and a novel method based on *pseudo-independence* of features are proposed as feature selection methods for the classifiers. Classification performances for the proposed classifiers are compared. The performance of the individual-independent classifiers improved when the feature selection methods were used. The GA-SVM hybrid produced the best results with a stress recognition rate of 98%.

Keywords: stress classification, artificial neural networks, genetic algorithms, support vector machines, physiological signals, physical signals, reading.

1 Introduction

Stress is part of everyday life and it has been widely accepted that stress which leads to less favorable states (such as anxiety, fear or anger) is a growing concern for people and society. The term, *stress*, was coined by Hans Selye. He defined it as “the non-specific response of the body to any demand for change” [1]. Stress is the body’s reaction or response to the imbalance caused between demands and resources available to a person. Stress is seen as a natural alarm, resistance and exhaustion [2] system for the body to prepare for a fight or flight response to protect the body from threats and changes. When experienced for longer periods of time without being managed, stress has been widely recognized as a major growing concern. It has the potential to cause chronic illnesses (e.g. cardiovascular diseases, diabetes and some forms of cancer) and increase economic costs in societies, especially in developed countries [3, 4]. Benefits of stress research range from improving day-to-day activities, through

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 111–122, 2013.

increasing work productivity to benefitting the wider society - motivating interest, making it a beneficial area of research and posing technical challenges in Computer Science. Various computational methods have been used to objectively define and classify stress to differentiate conditions causing stress from other conditions. The methods developed have used simplistic models formed from techniques like Bayesian networks [5], decision trees [6] and support vector machines [7]. These models have been built from a relatively smaller set of stress features than the sets used in the models in this paper.

The human body's response signals obtained from non-invasive methods that reflect reactions of individuals and their bodies to stressful situations have been used to interpret stress levels. These measures have provided a basis for defining stress objectively. Stress response signals used in this paper fall into two categories – physiological and physical signals. Physiological signals include the galvanic skin response (GSR), electrocardiogram (ECG) and blood pressure (BP). Unlike these signals, we define physical signals as signals where changes by the human body can be seen by humans without the need for equipment and tools that need to be attached to individuals to detect general fluctuations. However, sophisticated equipment and sensors using vision technologies are still needed to obtain physical signals at sampling rates sufficient for data analysis and modeling like the ones used in this paper. Physical signals include eye gaze and pupil dilation signals. GSR, ECG, BP, eye gaze and pupil dilation signals have been used to detect stress in literature [5, 8, 9] but this combination has not been reported in literature so far. We use this combination of sensor signals in this paper and refer to them as primary signals for stress.

Hundreds of stress features can be derived from primary signals for stress to classify stress classes. However, this set of features may include redundant and irrelevant features which may outweigh the more effective features showing stress patterns. This could cause a stress classification model to produce lower quality classifications. Since this paper is dealing with sensor data, some features may suffer from corruption as well. In order to achieve a good classification model that is robust to such potential features that may reduce the performance of classifications, appropriate feature selection methods must be developed and adopted by classifiers. A feature selection method that selects features in order to reduce redundancy using correlation based analysis could be used [10]. In addition, a genetic algorithm (GA) could also be used to select subsets of features for optimizing stress classifications. A GA is a global search algorithm and has been commonly used to solve optimization problems [11]. The search algorithm is based on the concept of natural evolution. It evolves a population of candidate solutions using crossover, mutation and selection methods in search for a population of a better quality. GAs have been successfully used to select features derived from physiological signals [12, 13].

This paper describes the method for collecting and developing computational models for recognizing stress patterns in response signals observed from individuals while reading *stressed* and *non-stressed* labeled text validated by participants. It details an experiment conducted to collect sensor and participant-reported data where experiment participants read text with stressful and non-stressful content during which multiple response signals were recorded. Several approaches for stress recognition in reading are developed, compared and discussed including methods for selecting features from hundreds of features derived from the response signals. The paper concludes with a summary of the findings and suggests directions for future work.

2 Data Collection from Reading Experiment

Thirty-five undergraduate students were recruited as experiment participants. The participant cohort was made up of 25 males and 10 females over the age of 18 years old. Each participant had to understand the requirements of the experiment from a written set of experiment instructions with the guidance of the experiment instructor before they provided their consent to take part in the experiment. Afterwards, physiological stress sensors were attached to the participant and physical stress sensors were calibrated. The instructor notified the participant to start reading, which triggered a sequence of text paragraphs. After finishing the reading, participants had to do an assessment based on the reading. An outline of the process of the experiment for an experiment participant is shown in Fig. 1.

Each participant had physiological and physical measurements taken over the 12 minutes reading time period. During the reading period, a participant read *stressed* and *non-stressed* types of text validated by participants. Stressed text had stressful content in the direction towards distress, fear and tension. Each participant read three stressed and three non-stressed text. Each text had approximately 360 words and was displayed on a computer monitor for participants to read. For consistency, each text was displayed on a 1050 x 1680 pixel Dell monitor, displayed for 60 seconds and positioned at the same location of the computer screen for each participant. Each line of the paragraph had 70 characters including spaces.

Results from the experiment survey validated the text classes. This is a common method used in literature to validate stress classes for tasks [14]. Participants found the paragraphs that were labeled stressed stressful and text labeled non-stressed as not stressful with a statistical significance of $p < 0.001$ according to the Wilcoxon test.

The physiological and physical sensor signals (which we refer to as primary stress signals) captured during the experiment were GSR, ECG, BP, eye gaze and pupil diameter signals. Biopac ECG100C, Biopac GSR100C and Finapres Finger Cuff systems were used to take ECG, GSR and blood pressure recordings at a sampling rate of 1000 Hz. Eye gaze and pupil dilation signals were obtained using Seeing Machines FaceLAB system with a pair of infrared cameras at 60 Hz. There were other signals that were derived from the primary stress signals to form other stress response signals. These signals included the heart rate variability signal, which was calculated from consecutive ECG peaks and another popular signal used for stress detection [15, 16].

Features were derived from the primary stress signals. Statistics (e.g. mean and standard deviation) were calculated for the signal measurements for each 5 second interval during the stressed and non-stressed reading. Measures such as the number of peaks for periodic signals, the distance an eye covered, the number of forward and backward tracking fixations, and the proportion of the time the eye fixated on different regions of the computer screen over 5 second intervals were also obtained. The statistic and measure values formed the stress feature set. There were 215 features altogether.

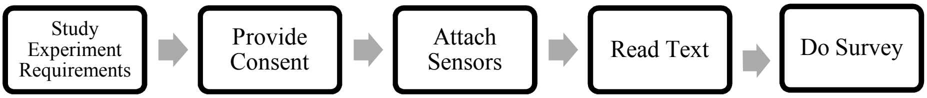


Fig. 1. Process followed by participants during the reading experiment

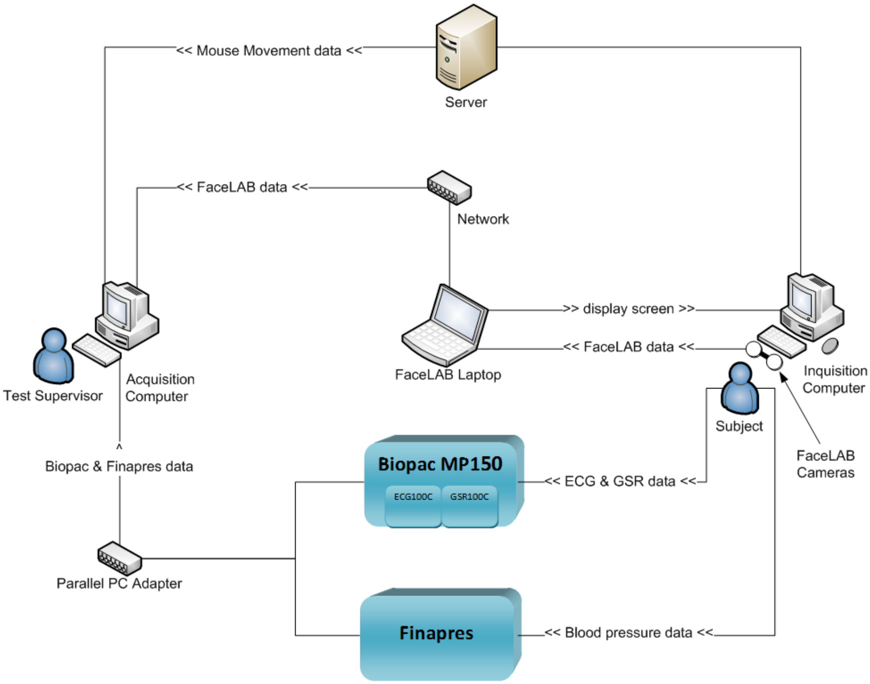


Fig. 2. Equipment setup for the reading experiment

3 Feature Selection

Features used for developing models for classification had an effect on the performance of classifiers. Selecting effective features by reducing redundant and irrelevant features have been known to improve the quality of pattern recognition [17] because it generalizes the patterns in the data better and helps develop a generalized model that captures necessary data patterns. In turn, this improves the quality for classifications. In this paper, a feature selection method based on correlation of features and a genetic algorithm (GA) approach were developed and used as feature selection methods for stress classification.

The features derived from the stress primary signals may have had redundant data so a feature selection approach based on *correlation coefficients* for features was developed. Correlation analysis using correlation coefficients has been reported to detect some redundancies in data [10]. It also took into account the time-varying nature of features and enabled comparison of features on this basis.

A correlation coefficient is a measure for the strength of the linear relationship between features. Consider two features, X and Y, with x_t and y_t values at time-step t in X and Y respectively, $\overline{X}$ and $\overline{Y}$ the means and σ_X and σ_Y are the standard deviations for X and Y, then the correlation coefficient r_{XY} is defined by

$$r_{XY} = \frac{\sum_{t=0}^{T}(x_t-\bar{X})(y_t-\bar{Y})}{(T+1)\sigma_X\sigma_Y} \tag{1}$$

The values for r_{XY} fulfill the following equation:

$$|r_{XY}| \leq 1 \tag{2}$$

If the value for $r_{XY} = 0$, then features X and Y are independent otherwise the features are correlated. However, stress features may have noise originating from data collection and the human body so as a result. In addition, the definition for independence of features for stress classification may be too strict. This motivated the use of different degrees for feature independence. In order to distinguish from independence defined by the strict criteria, we coin the term *pseudo-independence* to mean independence at a certain degree. Suppose the degree of independence is set at $r_{XY} \leq 0.1$ and r_{XY} is found to be 0.05, then features X and Y are pseudo-independent. On the other hand if r_{XY} is found to be 0.25, then features X and Y are not pseudo-independent.

The pseudo-independent based feature selection method, *pseudo-independent feature selection algorithm* (PISA), was used to select stress features that a classifier was provided to detect stress patterns. Given a set of features, each feature was compared with each of the other features to deter mine whether they were not pseudo-independent features. Features and their not pseudo-independent features were found and used to generate a set of pseudo-independent features for classification. The process by which a set of pseudo-independent features were obtained was by PISA. PISA is defined in Fig. 3.

Given a set of features and a set of features that are not pseudo-independent to each feature, PISA firstly finds features that are pseudo-independent to every other feature. Then it selects one feature from a cluster of not pseudo-independent features. The feature selected from a cluster is based on its not pseudo-independent features – the number of features and whether the features have been already selected. Features with a higher number of not pseudo-independent features have a greater chance of being selected. This characteristic of PISA minimizes the impact of highly correlated noisy features on classification.

Algorithm: Pseudo-Independent Feature Selection Algorithm (PISA). Find features that are pseudo-independent to other features given a set of features and their pseudo-independent features

Inputs:

- A set of features, *feature_set*
- Collection of features that are correlated to some degree to the other features in *feature_set* with a mapping to the features that they are not pseudo-independent with in *feature_set, corr_feat_mapping*

Output:

- Collection of pseudo-independent features

Method:

```
notcorr_feat_collection ← get features that are in feature_set and not in
    corr_feat_mapping
for each feature feat selected in descending order of the number of features
    that they are correlated with in corr_feat_mapping {
    corr_feats ← get the features correlated with feat from
        corr_feat_mapping
    if none of the features in corr_feats are in notcorr_feat_collection {
        then add feat in notcorr_feat_collection }}
```

Fig. 3. An algorithm to generate a set of pseudo-independent features (PISA) for stress classification

To illustrate PISA, consider an example input set comprising features and the features that are not pseudo-independent to the features presented in Fig. 4. Firstly, PISA determines features that do not have any not pseudo-independent features. A set {f4} is the result at this point. Now, the rest of the pseudo-independent features need to be appended to the set. There are multiple possibilities and they are {f1, f3, f4} and {f2, f4}. After applying the algorithm the result will be {f2, f4}, which is the smaller feature set of the multiple feature sets with pseudo-independent features. This approach reduces the risk of selecting features that were derived from corrupted sensor data and negatively affect the performance for stress classification. Suppose the feature set {f1, f2, f3} were derived from corrupted sensor data and if {f1, f3, f4} was selected as a set of features for classification, then intuitively the possibility of a classification model to capture better stress patterns would be lower than if {f2, f4} was chosen instead.

Inputs:
feature_set: {f1, f2, f3, f4}
corr_feat_mapping:

Feature		*Feature(s) not pseudo-independent to associated feature*
f1	→	{f2}
f2	→	{f1, f3}
f3	→	{f2}

Output: {f2, f4}

Fig. 4. An example to illustrate the inputs and corresponding output for the PI algorithm presented in Fig. 3 using a simple set of features

Another feature selection method used for stress classification in reading was based on a GA. A GA is a global search algorithm that was used to select features to improve the quality of stress classifications. The GA search evolved a population of subsets of features using crossover, mutation and selection methods in search for a population of subset of features that produced a better quality stress classification. A subset of features is referred to as an *individual* or *chromosome*. The quality for each chromosome in the population was defined by the quality of classifications produced when a classifier was provided with the features in the chromosome.

The initial population for the GAs was set up to have all the features. The number of features in the chromosomes varied but the chromosome length was fixed. The length of a chromosome was equal to the number of features in the feature space. A chromosome was a binary string where the index for a bit represented a feature and the bit value indicated whether the feature was used in the classification.

The parameters for the GAs implemented were set as provided in Table 1.

Table 1. Parameter settings for GAs used for feature selection

GA Parameter	Value/Setting
population size	100
number of generations	2000
crossover rate	0.8
mutation rate	0.01
crossover type	MATLAB's Scattered Crossover
mutation type	MATLAB's Uniform Mutation
selection type	MATLAB's Stochastic Uniform Selection

4 Computational Stress Classifiers

Classification models developed for stress pattern recognition from primary stress signals were based on an artificial neural network (ANN) and a support vector machine (SVM). The models were extended to incorporate feature selection phases either using the PISA or GA approaches. Each classification model was defined to capture individual-independent stress patterns. The accuracy and F-Score were calculated for each approach to determine the quality of the classification.

The stress reading data set was divided up into 3 subsets – training, validation and test sets – where 50% of the data samples were used for training a classification model and the rest of the data set was divided up equally for validating and testing the model. MATLAB was used to implement and test the models.

4.1 Artificial Neural Network Based Stress Classifiers

ANNs, inspired by biological neural networks, have capabilities for learning patterns to recognize characteristics in input tuples by classes. An ANN is made up of interconnected processors, known as *artificial neurons*, which are connected by weighted links that pass signals between neurons. In this paper, feed-forward ANNs trained using backpropagation were used. Three topologies were used to classify stress in reading. Each of the ANNs was provided stress features as inputs based on a selection method. Therefore, the ANNs differed only on the number of inputs. The ANN based stress classification models were:

- **ANN:** an artificial neural network classification model that was provided with all the features in the stress feature set as input to recognize stress patterns
- **PISA+ANN:** ANN with inputs as features produced by PISA
- **GA+ANN:** ANN with inputs as features produced by a GA

The MATLAB adapt function was used for training the ANNs on an incremental basis. Each network was trained for 1000 epochs using the Levenberg-Marquardt algorithm. The network had 7 hidden neurons and one neuron in the output layer. Future work could investigate optimizing the topology of the ANN for stress classification on the reading data set.

4.2 Support Vector Machine Based Stress Classifiers

SVMs have been widely used in literature for classification problems including classifications based on physiological data [18]. Provided a set of training samples, a SVM transforms the data samples using a nonlinear mapping to a higher dimension with the aim to determine a *hyperplane* that partitions data by class or labels. A hyperplane is chosen based on *support vectors*, which are training data samples that define maximum *margins* from the support vectors to the hyperplane to form the best decision boundary. This contributes to the resistance to data overfitting and helps to generalize classifications well.

Despite the useful characteristics, SVMs are still not robust to feature sets with redundant and irrelevant features. As a consequence, hybrids of SVM with PISA or GA were used to deal with ineffective features to investigate whether the hybrids with feature selection methods improved the quality of the classification.

The SVM based stress classification models developed were:

- **SVM:** a support vector machine classification model that was provided with all the features in the stress feature set as input similar to the ANN
- **PISA+SVM:** SVM with inputs as features produced by PISA
- **GA+SVM:** SVM with inputs as features produced by a GA

5 Results and Discussion

The six ANN and SVM based techniques were implemented and tested on the reading data set for stress recognition. Classification results were obtained using 10-fold cross-validation. The results are presented in Table 2. Classifiers with feature selection methods performed better than classifiers that used all stress features to model stress patterns. The hybrid classifiers had stress recognition rates and F-score values that were at least 8% and 12% better respectively. Classifiers with a GA as the feature selection method produced the highest stress recognition rates with GA+SVM as the best performing technique.

Performance measures for the classification techniques show that it was beneficial to use the feature selection methods to model stress patterns in reading. The stress features would have had redundant features and PISA would have reduced it. PISA compared every feature to every other feature in a pair-wise fashion and took a greedy approach in selecting features. Unlike PISA, the GA took a global view of the features and would have managed to reduce more redundant, irrelevant and corrupted features.

In terms of execution time, the GA based approaches, GA+ANN and GA+SVM, took longer times than the other techniques to produce solutions. It took PISA less than one second to select the features for the classification models whereas the execution times for the GA based approaches were in the order of hours. Classification without a feature selection method or with PISA took relatively a similar amount of time. Empirical execution times for the different approaches are shown in Table 3.

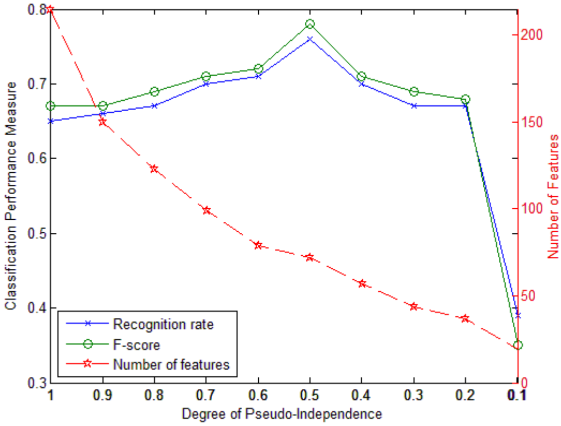
The execution times for GA based approaches were recorded after the search reached convergence except for GA+ANN, which took a lot longer to execute. GA+ANN took at least 3 days and it took the other techniques not more than a few hours to produce a solution. Therefore, it was not practical to let the GA+ANN search execute for longer. Table 2 and Table 3 have a * with the values to show results at the point when the GA+ANN search was terminated.

Table 2. Performance measures for stress recognition using the different approaches based on 10-fold cross-validation

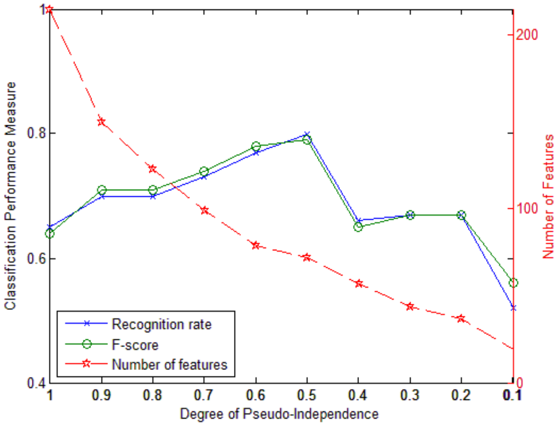
Classification Performance Measure	ANN	PISA+ANN	GA+ANN	SVM	PISA+SVM	GA+SVM
Accuracy	0.68	0.76	0.82*	0.67	0.80	**0.98**
F-score	0.67	0.79	0.82*	0.67	0.79	**0.98**

Table 3. Relative execution times for the different classification techniques

	ANN	PISA+ANN	GA+ANN	SVM	PISA+SVM	GA+SVM
Execution time (minutes)	5.1	5.1	4582*	0.5	0.5	275



(a)



(b)

Fig. 5. Performances for stress recognition using different degrees of pseudo-independence and PISA as the feature selection method for the classification (a) ANN based classification (b) SVM based classification

Due to the relative long execution times for the GA based approaches to search for a better stress classification result, other feature selection approaches can be investigated in the future. With shorter execution times and classification performance results for classifiers using PISA, PISA has the potential to increase the performance for classifications. In future, PISA based classifiers could be extended to have a more complex definition for pseudo-independence.

ANN and SVM classification results using PISA over a range of degrees of pseudo-independence were also obtained to determine the effect of the different degrees of pseudo-independence on the classification results. The results are displayed in graphs shown in Fig. 5. The graphs show the stress recognition rates and F-score along with the number of features for the different degrees of pseudo-independence. For both, ANN and SVM, the plot for the classification results show a positive overall rate of change and then it becomes negative after 0.5 degree of pseudo-independence. The best classification results are produced when the degree of pseudo-independence is 0.5. At this degree, every feature is pseudo-independent at the degree of 0.5 to every other feature in the set of features used as input to develop th e ANN and SVM classification models.

6 Conclusion and Future Work

Classification models were developed to recognize individual-independent stress patterns in physiological and physical data for reading. The use of a feature selection method that dealt with redundant features improved the quality of the classification. However, classification models based on a genetic algorithm provided better recognition rates for stress than a correlation based feature selection method. On the other hand, genetic algorithm based approaches required much longer execution times but the correlation based feature selection method hardly had any impact on the execution time. In future, the correlation based feature selection method could be extended to have a wider definition for feature independence. Further, a hybrid of the two feature selection methods presented in this work could be developed to reduce the execution time for the genetic algorithm based search. Moreover in this work, features were selected from an individual-independent viewpoint. Future work could investigate feature selection based on relationships of features for each individual and analyze their effect of the approach on classifications.

References

[1] Selye, H.: The stress syndrome. The American Journal of Nursing 65, 97–99 (1965)

[2] Hoffman-Goetz, L., Pedersen, B.K.: Exercise and the immune system: a model of the stress response? Immunology Today 15, 382–387 (1994)

[3] The-American-Institute-of-Stress. America's No. 1 Health Problem - Why is there more stress today? (August 05, 2010), http://www.stress.org/americas.htm

[4] Lifeline-Australia, Stress Costs Taxpayer $300K Every Day (2009), http://www.lifeline.org.au

[5] Liao, W., Zhang, W., Zhu, Z., Ji, Q.: A real-time human stress monitoring system using dynamic bayesian network. In: Computer Vision and Pattern Recognition - Workshops, CVPR Workshops (2005)

[6] Zhai, J., Barreto, A.: Stress recognition using non-invasive technology. In: Proceedings of the 19th International Florida Artificial Intelligence Research Society Conference, FLAIRS, pp. 395–400 (2006)

[7] Dou, Q.: An SVM ranking approach to stress assignment. University of Alberta (2009)

[8] Labbé, E., Schmidt, N., Babin, J., Pharr, M.: Coping with stress: the effectiveness of different types of music. Applied Psychophysiology and Biofeedback 32, 163–168 (2007)

[9] Healey, J.A., Picard, R.W.: Detecting stress during real-world driving tasks using physiological sensors. IEEE T ransactions on Intelligent Transportation Systems 6, 156–166 (2005)

[10] Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: 12th International Conference on Machine Learning, pp. 856–863 (2003)

[11] Goldberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addison-wesley (1989)

[12] Park, B.J., Jang, E.H., Kim, S.H., Huh, C., Sohn, J.H.: Feature selection on multi-physiological signals for emotion recognition. In: 2011 International Conference on Engineering and Industries (ICEI), Korea, pp. 1–6 (2011)

[13] Niu, X., Chen, L., Chen, Q.: Research on genetic algorithm based on emotion recognition using physiological signals. In: 2011 International Conference on Computational Problem-Solving (ICCP), pp. 614–618 (2011)

[14] Hill, J.D., Boy le, L.N.: Driver stress as influenced by driving maneuvers and roadway conditions. Transportation Research Part F: T raffic Psychology and Behaviour 10, 177–186 (2007)

[15] Ferreira, P., Sanches, P., Höök, K., Jaensson, T.: License to chill!: how to empower users to cope with stress. In: Proceedings of the 5th Nordic Conference on Human-Computer Interaction: Building Bridges, pp. 123–132 (2008)

[16] Dishman, R.K., Nakamura, Y., Garcia, M.E., Thompson, R.W., Dunn, A.L., Blair, S.N.: Heart rate variability, trait anxiety, and perceived stress among physically fit men and women. International Journal of Psychophysiology 37, 121–133 (2000)

[17] Siedlecki, W., Sklansky, J.: A note on genetic algorithms for large-scale feature selection. Pattern Recognition Letters 10, 335–347 (1989)

[18] Cheng, B.: Emotion recognition from physiological signals using support vector machine. In: Wu, Y. (ed.) Software Engineering and Knowledge Engineering. AISC, vol. 114, pp. 49–52. Springer, Heidelberg (2012)

Latent Patient Profile Modelling and Applications with Mixed-Variate Restricted Boltzmann Machine

Tu Dinh Nguyen, Truyen Tran, Dinh Phung, and Svetha Venkatesh

Center for Pattern Recognition and Data Analytics
School of Information Technology, Deakin University, Geelong, Australia
{ngtu,truyen.tran,dinh.phung,svetha.venkatesh}@deakin.edu.au

Abstract. Efficient management of chronic diseases is critical in modern health care. We consider *diabetes mellitus*, and our ongoing goal is to examine how machine learning can deliver information for clinical efficiency. The challenge is to aggregate highly heterogeneous sources including demographics, diagnoses, pathologies and treatments, and extract similar groups so that care plans can be designed. To this end, we extend our recent model, the mixed-variate restricted Boltzmann machine (MV.RBM), as it seamlessly integrates multiple data types for each patient aggregated over time and outputs a homogeneous representation called "latent profile" that can be used for patient clustering, visualisation, disease correlation analysis and prediction. We demonstrate that the method outperforms all baselines on these tasks - the primary characteristics of patients in the same groups are able to be identified and the good result can be achieved for the diagnosis codes prediction.

1 Introduction

Chronic diseases are rampant. Health care costs are increasingly related to such diseases. *Diabetes mellitus* is one such chronic disease from which 346 million people worldwide suffer, as estimated by The World Health Organization (WHO) [1]. Only about $5 - 10\%$ of them have Type I diabetes mellitus, whilst Type II comprises the rest. The people who suffer Type I are not able to produce insulin. In contrast, Type II diabetes means that there is an inability to absorb insulin. In 2004, 3.4 million people died from complications of high blood sugar. The incidence of diabetes mellitus is increasing, and being diagnosed in younger people. This leads to serious complications - deterioration in blood vessels, eyes, kidneys and nerves. It is a chronic, lifelong disease.

Escalating health costs are associated with such chronic diseases. To provide high quality healthcare, care plans are issued to patients to manage them within the community, taking steps in advance so that these people are not hospitalised. Thus, it is imperative to identify groups of patients with similar characteristics so that they can be covered by a coherent care plan. Additionally, if the hospital can predict the disease codes arising from escalating complication of chronic disease, it can adjust financial and manpower resources. Thus useful prediction of codes for chronic disease can lead to service efficiency.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 123–135, 2013.

Clustering is a natural selection for this task. However, medical data is complex – it is mixed-type containing Boolean data (e.g., male/female), continuous quantities (e.g., age), single categories (e.g., regions), and repeated categories (e.g., disease codes). Traditional clustering methods cannot naturally integrate such data and we choose to extend the our recent model, the mixed-variate restricted Boltzmann machine (MV.RBM) [2]. The mixed-variate RBM uncovers *latent profile* factors, enabling subsequent clustering. Using a cohort of $6,931$ chronic diabetes patients with data from 2007 to 2011, we collect $3,178$ diagnosis codes (treated as repeated categories) and combine it with region-of-birth (as categories) and age (as Gaussian variables) to form our dataset. We show clustering results obtained from running affinity propagation (AP) [3], containing 10 clusters and qualitatively evaluate the disease codes of groups. We demonstrate that the mixed-variate RBM followed by AP outperforms all baseline methods – Bayesian mixture model and affinity propagation on the original diagnosis codes, and k-means and AP on latent profiles, discovered by just the plain RBM [4].

Predicting disease codes for future years enables hospitals to prepare finance, equipment and logistics for individual requirements of patients. Thus prediction of disease codes forms the next part of our study. Using the mixed-variate RBM and the dataset described above, we demonstrate that our method outperforms other methods, establishing the versatility of the latent profile discovery with mixed-variate RBM.

In short, our main contributions are: (i) Novel extension and application of a powerful data mining tool, mixed-variate RBM, to a complex hospital chronic disease dataset, for clustering and understanding of disease codes within subgroups; (ii) Disease code prediction, using the model; and (iii) Demonstration of the method and showing that it outperforms baseline models, in both clustering and prediction on this complex data.

The significance of our work is to build a framework that is able to support healthcare centres and clinicians delivering outcomes that can integrate with their operations to enhance clinical efficiencies. Using such systems, the management and supervision on diabetes patients in particular as well as other kinds of diseases patients in general would have the potential to improve.

The rest of paper is organized as follows. The next section presents our patient profile modelling framework. Next, we describe our implementation on the diabetes cohort and demonstrate efficiencies of our methods. Section 4 discusses related work and modelling choices as well as the other potentials of the proposed framework, followed by conclusions in Section 5.

2 Latent Patient Profiling

A patient profile in modern hospitals typically consists of multiple records including demographics, admissions, diagnoses, surgeries, pathologies and medication. Each record contains several fields, each of which is type-specific. For example, age can be considered as a continuous quantity but a diagnosis code is a discrete element. At the first approximation, each patient can be represented by using a long vector of mixed types[1]. However, joint modelling of mixed types is known to be highly challenging even for a small set of variables [5,6]. Complete patient profiling, on the other hand, requires handling of thousands of variables. Of equal importance is that the profiling should readily support a variety of clinical analysis tasks such as patient clustering, visualisation and disease prediction. In what follows, we develop a representational and computational scheme to capture such heterogeneity in an efficient way. In particular, we extend the our recently introduced machinery known as mixed-variate restricted Boltzmann machine (MV.RBM) [2] for the task.

2.1 Mixed-Variate Restricted Boltzmann Machines

A restricted Boltzmann machine (RBM) is a bipartite undirected graphical model with two layers, where the input layer consists of visible units and the other layer the binary hidden units [7]. See, for example, Fig. 1 for an illustration. A mixed-variate RBM is a RBM with inhomogeneous input units, each of which has the own type. More formally, let $\boldsymbol{v}$ denote the joint set of visible variables: $\boldsymbol{v} = (v_1, v_2, ..., v_N)$, $\boldsymbol{h}$ the joint set of binary hidden units: $\boldsymbol{h} = (h_1, h_2, ..., h_K)$, where $h_k \in \{0, 1\}$ for all k. Each visible unit encodes type-specific information, and the hidden units capture the *latent factors* not presented in the observations. Thus the MV.RBM can be seen as a way to transform inhomogeneous observational record into a *homogeneous representation* of the patient profile. Another way to view this as a mixture model where there are 2^K mixture components. This capacity is arguably important to capture all factors of variation in the patient cohort.

The MV.RBM defines a Boltzmann distribution over all variables: $P(\boldsymbol{v}, \boldsymbol{h}; \psi) = e^{-E(\boldsymbol{v}, \boldsymbol{h})}/Z(\psi)$, where $E(\boldsymbol{v}, \boldsymbol{h})$ is model energy, $Z(\psi)$ is the normalising constant and ψ is model parameter. In particular, the energy is defined as

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\left(\sum_i F_i(v_i) + \sum_i a_i v_i + \sum_k b_k h_k + \sum_{ik} h_k W_{ik} v_i\right) \tag{1}$$

where $\boldsymbol{a} = (a_1, a_2, .., a_N), \boldsymbol{b} = (b_1, b_2, ..., b_K)$ are biases of visible and hidden units, and $\boldsymbol{W} = [W_{ik}]$ represents the weights connecting hidden and visible units, and $F_i(v_i)$ are type-specific function. The bipartite structure allows conditional independence among intra-layer variables which lead to the following factorisations:

$$P(\boldsymbol{v} \mid \boldsymbol{h}) = \prod_{i=1}^{N} P(v_i \mid \boldsymbol{h}) \tag{2}$$

$$P(\boldsymbol{h} \mid \boldsymbol{v}) = \prod_{k=1}^{K} P(h_k \mid \boldsymbol{v}) \tag{3}$$

The conditional separation of types in Eq. (2) is critical: This allows independent specification of type-specific data generative models and at the same

[1] Since each field may be repeated over time (e.g., diagnosis codes), we need an aggregation scheme to summarize the field. Here we use the simple counting for diagnosis codes.

time achieves higher-order dependencies through the "pooling" layer $\boldsymbol{h}$. For example, let $f_i(\boldsymbol{h}) = a_i + \sum_k W_{ik} h_k$, the *binary* units would be specified as: $P(v_i \mid \boldsymbol{h}) = 1/\left(1 + e^{-f_i(\boldsymbol{h})}\right)$ (i.e., $F_i(v_i) = 0$); the *Gaussian* units: $P(v_i \mid \boldsymbol{h}) = \mathcal{N}\left(\sigma_i^2 f_i(\boldsymbol{h}); \sigma_i\right)$ (i.e., $F_i(v_i) = -v_i^2/2\sigma_i^2$), and the *categorical* units: $P(v_i \mid \boldsymbol{h}) = e^{f_i(\boldsymbol{h})}/\sum_j e^{f_j(\boldsymbol{h})}$.

The model is typically estimated by maximising the data log-likelihood $\mathcal{L} = \log P(\boldsymbol{v}; \psi) = \log \sum_{\boldsymbol{h}} P(\boldsymbol{v}, \boldsymbol{h}; \psi)$. The parameters are updated in a gradient ascent fashion as follows:

$$\psi \leftarrow \psi + \nu \left(\mathbb{E}_{\boldsymbol{v},\boldsymbol{h}} \left[\frac{\partial E(\boldsymbol{v}, \boldsymbol{h})}{\partial \psi} \right] - \mathbb{E}_{\boldsymbol{h}|\boldsymbol{v}} \left[\frac{\partial E(\boldsymbol{v}, \boldsymbol{h})}{\partial \psi} \right] \right) \tag{4}$$

for some learning rate $\nu > 0$. Here $\mathbb{E}_{\boldsymbol{v},\boldsymbol{h}}$ denotes the expectation with respect to the full model distribution $P(\boldsymbol{v}, \boldsymbol{h}; \psi)$, $\mathbb{E}_{\boldsymbol{h}|\boldsymbol{v}}$ the conditional distribution given the known $\boldsymbol{v}$. Whilst the conditional expectation can be compute efficiently, the full expectation is generally intractable. Thus we must resort to approximate methods, and in this paper, we choose a truncated MCMC-based method known as contrastive divergence (CD) [8] as it proves to be fast and accurate. A MCMC chain is obtained by alternating between $\hat{\boldsymbol{v}} \sim P\left(\boldsymbol{v} \mid \hat{\boldsymbol{h}}\right)$ and $\hat{\boldsymbol{h}} \sim P(\boldsymbol{h} \mid \hat{\boldsymbol{v}})$.

2.2 MV.RBM for Patient Profiling

The goal of patient profiling is to construct an effective *personal representation* from multiple hospital records. Here we focus mainly on patient demographics (e.g., *age*, *gender* and *region-of-birth*) and their existing health conditions (e.g., existing *diagnoses*). For simplicity, we consider a binary gender (male/female). Further, age can be considered as a continuous quantity and thus a Gaussian unit can be used[2]; and region-of-birth and diagnosis as categorical variables. However, since the same diagnosis can be repeated during the course of readmissions, it is better to include them all. In particular, we adopt the idea from the "replicated softmax" [4] where repeated diagnoses share the same parameters. In the end, we build one MV.RBM per patient due to the difference in the diagnosis sets. Further, to balance the contribution of the hidden units against the variation in input length, it is important to make a change to the energy model in Eq. (1) as follows: $D\boldsymbol{b} \leftarrow \boldsymbol{b}$ where D is the total number of input variables for each patient. We note that these parameter sharing and balancing are not readily present in the current MV.RBM of Truyen et al [2].

Once the model has been estimated, the *latent profiles* are generated by computing the posterior vector $\hat{\boldsymbol{h}} = \left(P\left(h_1^1 \mid \boldsymbol{v}\right), P\left(h_2^1 \mid \boldsymbol{v}\right), ..., P\left(h_K^1 \mid \boldsymbol{v}\right)\right)$, where $P\left(h_k^1 \mid \boldsymbol{v}\right)$is a shorthand for $P(h_k = 1 \mid \boldsymbol{v})$ – the probability that the k-th latent factor is activated given the demographic and clinical input $\boldsymbol{v}$:

[2] Although the distribution of ages for a particular disease is generally not Gaussian, our model is a mixture of exponentially many components (2^K, see Sec. 2.1 for detail), and thus can capture any distribution with high accuracy.

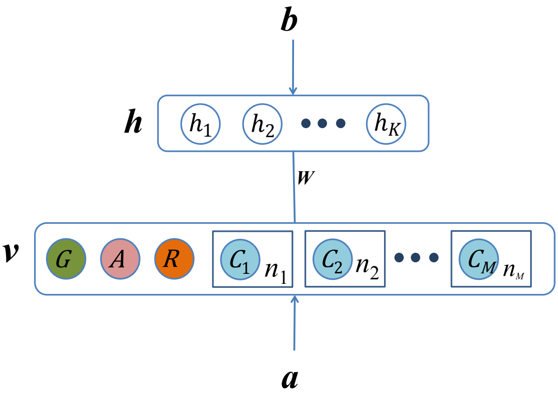


Fig. 1. Patient profiling using mixed-variate RBMs. The top layer represents stochastic binary units. The bottom layer encodes multiple type-specific inputs: A for age (continuous), G for gender (binary), R for region-of-birth, C_k for diagnosis codes. The circles within squares denote the replicated diagnosis codes (categorical) where the integers $\{n_k\}$ denotes the number of replications.

$$P\left(h_k^1 \mid \boldsymbol{v}\right) = \frac{1}{1 + \exp\left\{-Db_k - \sum_i W_{ik} v_i\right\}}.$$

As we will then demonstrate in Section 3, the latent profile can be used as input for a variety of analysis tasks such as patient clustering and visualisation.

Interestingly, the model also enables a certain degree of *disease prediction*, i.e., we want to guess which diagnoses will be positive for the patient in the future[3]. Although this may appear to be an impossible task, it is plausible statistically because some diseases are highly correlated or even causative, and there are certain pathways that a disease may progress. More specifically, subset of diagnoses at time $t+1$ can be predicted by searching for the mode of the following conditional distribution:

$$P\left(\boldsymbol{v}^{(t+1)} \mid \boldsymbol{v}^{(1:t)}\right) = \sum_{\boldsymbol{h}} P\left(\boldsymbol{v}^{(t+1)}, \boldsymbol{h} \mid \boldsymbol{v}^{(1:t)}\right).$$

Unfortunately the search is intractable as we need to traverse through the space of all possible disease combinations, which has the size of 2^M where M is the set of diagnosis codes. To simplify the task and to reuse of the readily discovered latent profile $\hat{\boldsymbol{h}}^{(1:t)}$, we assume that (i) the model distribution is not changed due to the "unseen" future, (ii) the latent profile at this point captures everything we can say about the state of the patient, and (iii) future diseases are conditionally

[3] Although this appears to resemble the traditional collaborative filtering, it is more complicated since diseases may be recurrent, and the strict temporal orders must be observed to make the model clinically plausible.

independent given the current latent profile. This leads to the following *mean-field* approximation[4]:

$$P\left(v_j^{(t+1)} \mid \boldsymbol{v}^{(1:t)}\right) \approx \frac{\exp\left\{a_j + \sum_k W_{jk} P\left(h_k^1 \mid \boldsymbol{v}^{(1:t)}\right)\right\}}{\sum_i \exp\left\{a_i + \sum_k W_{ik} P\left(h_k^1 \mid \boldsymbol{v}^{(1:t)}\right)\right\}}. \quad (5)$$

3 Implementation and Results

In this section we present the analysis of patient profiles using the data obtained from Barwon Health, Victoria, Australia[5], during the period of 2007 − 2011 using the extended MV.RBM described in Section 2. In particular, we evaluate the capacity of the MV.RBM for patient clustering and for predicting future diseases. For the former task, the MV.RBM is can be seen as a way to transform complex input data into a homogeneous vector from which post-processing steps (e.g., clustering and visualisation) can take place. For the prediction task, the MV.RBM acts as a classifier that map inputs into outputs.

Our main interest is in the *diabetes* cohort of $7,746$ patients. There are two types of diabetes: Type I is typically present in younger patients who are not able to produce insulin; and Type II is more popular in the older group who, on the other hand, cannot adsorb insulin. One of the most important indicators of diabetes is the high blood sugar level compared to the general population. Diabetes are typically associated with multiple diseases and complications: The cohort contains $5,083$ distinct diagnosis codes, many of which are related to other conditions and diseases such as obesity, tobacco use and heart problems. For robustness, we remove those rare diagnosis codes with less than 4 occurrences in the data. This results in a dataset of $6,931$ patients who originally came from 102 regions and were diagnosed with totally $3,178$ unique codes. The inclusion of age and gender into the model is obvious: they are not only related to and contributing to the diabetes types, they are also largely associated with other complications. Information about the regions-of-origin is also important for diabetes because it is strongly related to the social conditions and lifestyles, which are of critical importance to the proactive control of the blood sugar level, which is by far the most cost-effective method to mitigate diabetes-related consequences.

3.1 Implementation

Continuous variables are first normalised across the cohort so that the Gaussian inputs have zero-means and unit variances. We employ 1-step contrastive divergence (CD) [8] for learning. Learning rates vary from type to type and they are chosen so that reconstruction errors at each data sweep are gradually reduced.

[4] This result is obtained by first disconnecting the future diagnosis-codes from the latent units and then find the *suboptimal* factorised distribution $Q\left(\boldsymbol{v}^{(t+1)}, \boldsymbol{h} \mid \boldsymbol{v}^{(1:t)}\right) = \prod_j Q_j\left(v_j^{(t+1)} \mid \boldsymbol{v}^{(1:t)}\right) \prod_k P\left(h_k \mid \boldsymbol{v}^{(1:t)}\right)$ that minimises the Kullback-Leibner divergence from the original distribution $P\left(\boldsymbol{v}^{(t+1)}, \boldsymbol{h} \mid \boldsymbol{v}^{(1:t)}\right)$.

[5] Ethics approval 12/83.

Parameters are updated after each mini-batch of 100 patients, and learning is terminated after 100 data sweeps. The number of hidden units is determined empirically to be 200 since large size does not necessarily improve the clustering/prediction performance.

For *patient clustering*, once the model has been learned, the hidden posteriors that are computed using Eq. (3) can be used as the new representation of the data. To enable fast bitwise implementation (e.g., see [9]), we then convert the continuous posteriors into binary activation as follows: $\hat{h}_k = 1$ if $P(h_k^1 \mid \boldsymbol{v}) \geq \rho_1$ and $\hat{P}_k = 0$ otherwise for all $k = 1, 2.., K$ and some threshold $\rho_1 \in (0, 1)$. We then apply well-known clustering methods including affinity propagation (AP) [3], k-means and Bayesian mixture models (BMM). The AP is of particular interest for our exploratory analysis because it is capable of automatically determining the number of clusters. It requires the similarity measure between two patients, and in our binary profiles, a natural measure is the Jaccard coefficient:

$$J(p, q) = \frac{\mid S\{p\} \cap S\{q\} \mid}{\mid S\{p\} \cup S\{q\} \mid} \tag{6}$$

where $S\{p\}$ is the set of activated hidden units for patient p. Another hyperparameter is the so-called 'preference' which we empirically set to the average of all pairwise similarities multiplied by -20. This setting gives a reasonable clustering.

The other two clustering methods require a prior number of clusters, and here we use the output from the AP. For the k-means, we use the the Hamming distance between activation vectors of the two patients[6]. The BMM is a Bayesian model with multinomial emission probability.

The task of *disease prediction* is translated into predicting diagnoses in the future for each patient. We split data into 2 subsets: The earlier subset, which contains those diagnoses in the period of $2007 - 2010$, is used to train the MV.RBM; and the later subset is used to evaluate the prediction performance. In the MV.RBM, we order the future diseases according to the probability that the disease occurs as in Eq. (5).

3.2 Patient Clustering

First we wish to validate that the latent profiles discovered by the MV.RBM are informative enough so that *clinically meaningful* clusters can be formed. Fig. 2 shows the 10 clusters returned by the AP and the similarity between every patient pair (depicted in colour, where the similarity increases with from blue to red). It is interesting that, out of 10 groups, we are able to discover a group whose conditions are mostly related to Type I diabetes (Figs. 3a and 3b), and another group associated with Type II diabetes (Figs. 4a and 4b). The grouping properties can be examined visually using a visualisation tool known as t-SNE [10] to project the latent profiles onto 2D. Fig. 5a depicts the distribution of patients, where the colours are based on the group indices assigned earlier by the AP.

[6] The centroid of each cluster is chosen according to the median elementwise.

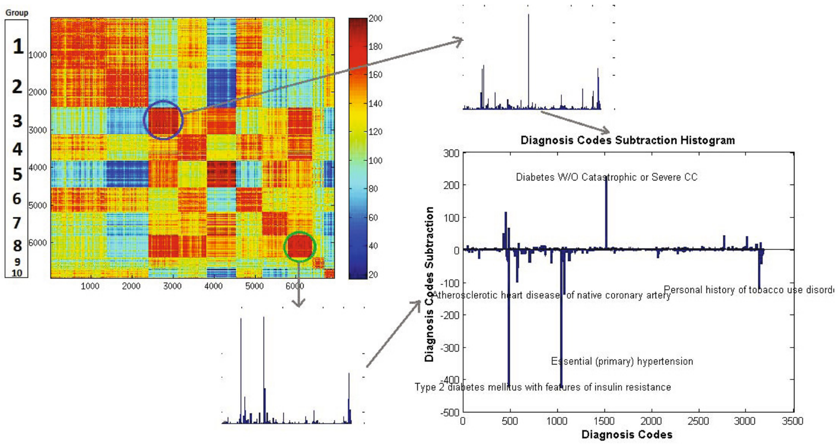
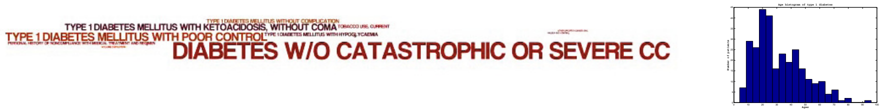
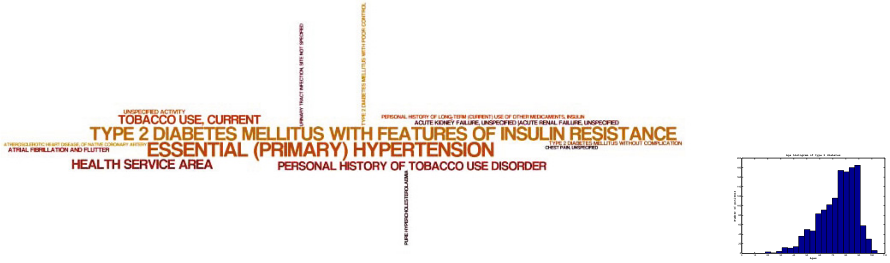


Fig. 2. Similarity matrix and diagnosis codes histograms. The matrix represents resemblances of pairwise patients while histograms show quantity of diagnoses. Group 3 and Group 8 look highly overlapping at the diagnosis level (top-left figure), but in fact, their clinical conditions are significantly different when we subtract the two histograms (lower-right figure). (Best viewed in colors).



(a) Tag cloud of diagnosis descriptions. (b) Age histogram.

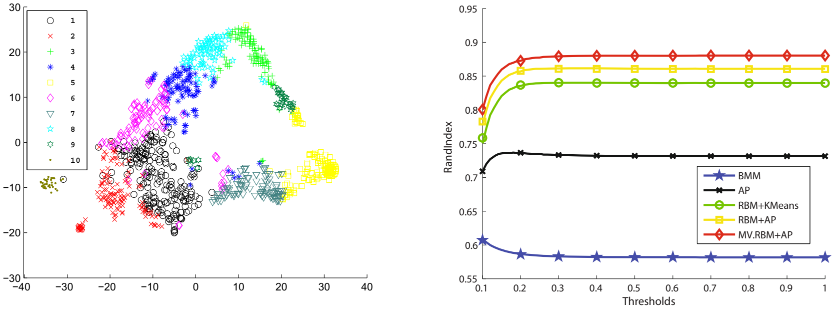
Fig. 3. Type I diabetes mellitus: Primary diagnoses and age distribution. Two figures confirms the existing knowledge that Type I diabetes mellitus often occurs in the younger population.



(a) Tag cloud of diagnosis descriptions. (b) Age histogram.

Fig. 4. Type II Diabetes mellitus: Primary diagnoses and age distribution. We can see that the age distribution is distinct from the Type I group.

For quantitative evaluation, we calculate the *Rand-index* [11] to assess the quality of resulting clusters, given that we do not have cluster labels. The Rand-index is the pairwise accuracy between any two patients. To judge whether two patients share the same cluster, we consult the diagnosis code hierarchy of the ICD-10 [12]. We use hierarchical assessment since a diagnosis code may have multiple levels. E11.12, for example, has two levels: E11 and E11.12. The lower level code specifies disease more clearly whilst the higher is more abstract. Therefore we have two ways for pairwise judgement: the Jaccard coefficient (Section 3.1) and code 'cluster' which is the grouping of codes that belong to the same disease class, as specified by the latest WHO standard ICD-10. At the lowest level, two patients are considered similar if the two corresponding code sets are sufficiently overlapping, i.e., their Jaccard coefficient is greater than a threshold $\rho_2 \in (0, 1)$. At higher level, on the other hand, we consider two patients to be clinically similar if they share higher level diabetes code of the same code 'cluster'. For instance, two patients with two codes E11.12 and E11.20 are similar at the level E11[7], but they are dissimilar at the lower level. Note that this hierarchical division is for evaluation only. We use codes at the lowest level as replicated softmax units in our model (Section 2.2).



(a) (b)

Fig. 5. Visualisation and quantitative assessment of clusters. (a) t-SNE [10] projection on 2, 000 latent profiles. Groups are labelled by the outputs of the AP. (Best viewed in colors). (b) Rand-index curves in patient clustering. *AP*: affinity propagation, *BMM*: Bayesian mixture model, *RBM*: MV.RBM with diagnosis codes only.

Fig. 5b reports the Rank-indices with respect to the assessment at the lowest level in the ICD-10 hierarchy for clustering methods with and without MV.RBM pre-processing. At the next ICD-10 level, the MV.RBM/AP achieves a Rand-index of 0.6040, which is, again, the highest among all methods, e.g., using the RBM/AP yields the score of 0.5870, and using AP on diagnosis codes yields 0.5529. This clearly demonstrates that (i) MV.RBM latent profiles would lead to better clustering that those using diagnosis codes directly, and (ii) modelling mixed-types would be better than using just one input type (e.g., the diagnosis codes).

[7] This code group is for non-insulin-dependent *diabetes mellitus*.

3.3 Disease Prediction

The prediction results are summarised in Fig. 6, where the ROC curve of the MV.RBM is compared against that of the baseline using k-nearest neighbours (k-NN). The k-NN approach to disease ranking is as follows: For each patient, a neighbourhood of the 50 most similar patients is collected based on the Jaccard coefficient over sets of unique diagnoses. The diagnoses are then ranked according to their occurrence frequency within the neighbourhood. As can be seen from the figure, the latent profile approaches outperform the k-NN method. The MV.RBM with contextual information such as age, gender and region-of-birth proves to be useful. In particular the the areas under the ROC curve (AUC) of the MV.RBMs are 0.84 (with contextual information) and 0.82 (without contextual information). These are clearly better than the score obtained by k-NN, which is 0.77.

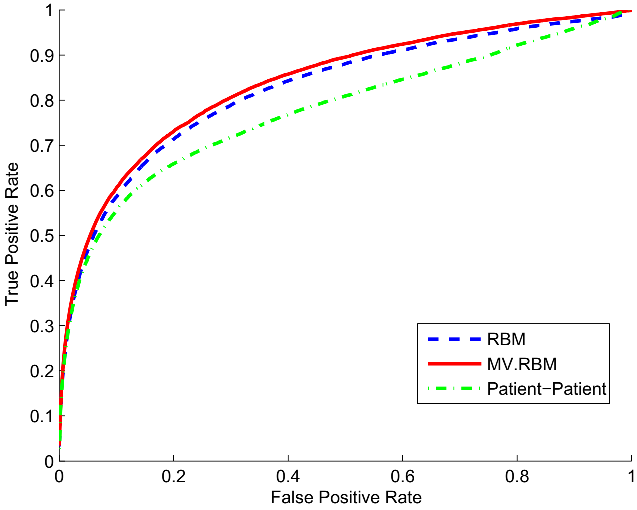


Fig. 6. ROC curves in disease prediction. *RBM* is MV.RBM with diagnosis codes only; *Patient-Patient* is the k-nearest neighbours method. Best viewed in colors.

4 Discussion and Related Work

This work is part of our ongoing effort to apply data mining and statistical techniques to understand the complex health databases in order to improve the services efficiency within health organisations and across coordinated networks. This line of research has recently attracted considerable interest in the data mining community (e.g., see [13,14]). Our focus on diabetes is motivated by the pressing demands to deliver personalised cares for the large population on an ongoing basis [15,16]. However, we wish to emphasize that the approach is quite general and could be applicable to other cohorts.

In terms of modelling, our work adds a few more flavours to the current mixed-variate analysis in biomedical domains [5,6,17]. The existing literature offers three approaches: The first is to specify the direct type-specific conditional relationship between two variables (e.g., see [5]), the second is to assume that each observable is generated from a latent variable (latent variables then encode the dependencies) (e.g., see [6]), and the third is to construct joint cumulative distributions using copula [18,17]. The drawback of the first approach is that it requires far more domain knowledge and statistical expertise to design a correct model even for a small number of variables. The second approach lifts the direct dependencies to the latent variables level. All approaches are, however, not very scalable to realistic setting of the hospital records. Our treatment using MV.RBM [2], along with the line of work using RBMs for representing complex data [19,20,21], offers the *fourth alternative*: Direct pairwise dependencies are substituted by indirect long-range dependencies. Not only this simplifies the model design, the inference is much more scalable: each MCMC sweep through all variables takes only linear time. Our most recent work in [21], while enjoying the similar computational efficiency, offers a better interpretation through the use of latent variables to capture the generative mechanism of data types.

Latent profiling could be important for other applications such as patient retrieval, i.e., we want to retrieve patients with clinically similar conditions to the patient under study. In this setting, using raw diagnosis codes may miss those whose codes are different from the present patient even if they share the same clinical conditions. The use of MV.RBM, on the other hand, would project these patients onto similar latent profiles. It is also of interest to ask whether it is justifiable for the choice of parameter sharing for repeated diagnoses. To get answer, we experimented with the “counting” treatments in which each code is considered as a Poisson variable, and our clustering/prediction results indicate that it is much better to employ the parameter sharing trick. This may due to the fact that under the Poisson treatment, diagnoses are assumed to “arrive” independently, while in reality diagnoses are generally correlated.

5 Conclusion

We have presented a latent profiling framework using our recently introduced architecture known as mixed-variate restricted Boltzmann machines (MV.RBM). The goal was to develop a representational and computational scheme that can handle complex, inhomogeneous data from real hospital settings. The MV.RBM was adapted to handle recurrent diagnoses by parameter sharing and variable balancing. We evaluated this scheme on a cohort of complex diseases such as diabetes, where there are many influential factors, and diagnoses are often repeated, correlated and causative. It is demonstrated that the chosen scheme is highly effective for exploratory tasks such as patient clustering and visualisation and predictive tasks such as 1-year diagnosis prognosis.

References

1. World Health Organization: Diabetes (2012), `http://www.who.int/mediacentre/factsheets/fs312/en/index.html` (accessed September 2012)
2. Tran, T., Phung, D.Q., Venkatesh, S.: Mixed-variate restricted Boltzmann machines. Journal of Machine Learning Research - Proceedings Track 20, 213–229 (2011)
3. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science 315, 972–976 (2007)
4. Salakhutdinov, R., Hinton, G.: Replicated softmax: an undirected topic model. Advances in Neural Information Processing Systems 22, 1607–1614 (2009)
5. McCulloch, C.: Joint modelling of mixed outcome types using latent variables. Statistical Methods in Medical Research 17(1), 53 (2008)
6. Dunson, D., Herring, A.: Bayesian latent variable models for mixed discrete outcomes. Biostatistics 6(1), 11 (2005)
7. Freund, Y., Haussler, D.: Unsupervised learning of distributions on binary vectors using two layer networks. Santa Cruz, CA, USA. Tech. Rep. (1994)
8. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
9. Salakhutdinov, R., Hinton, G.: Semantic hashing. In: SIGIR Workshop on Information Retrieval and Applications of Graphical Models, vol. 500(3). ACM Special Interest Group on Information Retrieva (2007)
10. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research 9, 2579–2605 (2008)
11. Rand, W.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association, 846–850 (1971)
12. World Health Organization: ICD-10th (2010), `http://apps.who.int/classifications/icd10/browse/2010/en` (accessed September 2012)
13. Khosla, A., Cao, Y., Lin, C.C.-Y., Chiu, H.-K., Hu, J., Lee, H.: An integrated machine learning approach to stroke prediction. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 183–192. ACM (2010)
14. Luo, D., Wang, F., Sun, J., Markatou, M., Hu, J., Ebadollahi, S.: Sor: Scalable orthogonal regression for non-redundant feature selection and its healthcare applications. In: SIAM Data Mining Conference (2012)
15. Ben-Hur, A., Iverson, T., Iyer, H.: Predicting the risk of type 2 diabetes using insurance claims data. In: Neural Information Processing System Foundation (2010)
16. Neuvirth, H., Ozery-Flato, M., Hu, J., Laserson, J., Kohn, M.S., Ebadollahi, S., Rosen-Zvi, M.: Toward personalized care management of patients at risk: the diabetes case study. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 395–403. ACM (2011)
17. de Leon, A.R., Wu, B.: Copula-based regression models for a bivariate mixed discrete and continuous outcome. Statistics in Medicine 30(2), 175–185 (2011)
18. Song, P.X.-K., Li, M., Yuan, Y.: Joint regression analysis of correlated data using gaussian copulas. Biometrics 65(1), 60–68 (2009)
19. Truyen, T., Phung, D., Venkatesh, S.: Ordinal Boltzmann machines for collaborative filtering. In: Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI), Montreal, Canada (June 2009)

20. Tran, T., Phung, D., Venkatesh, S.: Cumulative restricted Boltzmann machines for ordinal matrix data analysis. In: Proc. of 4th Asian Conference on Machine Learning (ACML), Singapore (2012)
21. Tran, T., Phung, D., Venkatesh, S.: Embedded Restricted Boltzmann Machines for fusion of mixed data type and applications in social measurements analysis. In: Proc. of the 15th International Conference on Information Fusion (FUSION), Singapore (2012)

MassBayes: A New Generative Classifier with Multi-dimensional Likelihood Estimation

Sunil Aryal and Kai Ming Ting

Gippsland School of Information Technology
Monash University, Australia
{sunil.aryal,kaiming.ting}@monash.edu

Abstract. Existing generative classifiers (e.g., BayesNet and A*n*DE) make independence assumptions and estimate one-dimensional likelihood. This paper presents a new generative classifier called *MassBayes* that estimates multi-dimensional likelihood without making any explicit assumptions. It aggregates the multi-dimensional likelihoods estimated from random subsets of the training data using varying size random feature subsets. Our empirical evaluations show that MassBayes yields better classification accuracy than the existing generative classifiers in large data sets. As it works with fixed-size subsets of training data, it has constant training time complexity and constant space complexity, and it can easily scale up to very large data sets.

Keywords: Generative classifier, Likelihood estimation, MassBayes.

1 Introduction

The learning task in classification is to learn a model from a labelled training set that maps each instance to one of the predefined classes. The model learned is then used to predict a class label for each unseen test instance. Each instance $\mathbf{x}$ is represented by a d-dimensional vector $\langle x_1, x_2, \cdots, x_d \rangle$ and given a class label $y \in \{y_1, y_2, \cdots, y_c\}$, where c is the total number of classes. The training set D is a collection of labelled instances $\{(\mathbf{x}^{(i)}, y^{(i)})\}$ $(i = 1, 2, \cdots, N)$.

The generative approach of classifier learning models the joint distribution $p(\mathbf{x}, y)$ and predicts the most probable class as:

$$\hat{y} = \arg\max_{y} \ p(\mathbf{x}, y) \tag{1}$$

Using the product rule, the joint probability can be factorised as:

$$p(\mathbf{x}, y) = p(y) \times p(\mathbf{x}|y) \tag{2}$$

Generative classifiers learn either the joint distribution $p(\mathbf{x}, y)$ or the likelihood $p(\mathbf{x}|y)$. However, estimating $p(\mathbf{x}, y)$ or $p(\mathbf{x}|y)$ directly from data using existing data modelling techniques is difficult. Density estimators such as Kernel Density Estimation [1], k-Nearest Neighbour [1] and Density Estimation Trees [2] are impractical in large data sets due to their high time and space complexities. The research has thus focused on learning one-dimensional likelihood to approximate $p(\mathbf{x}, y)$ in different ways.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 136–148, 2013.

Existing generative classifiers allow limited probabilistic dependencies among attributes and assume some kind of conditional independence. Different generative classifiers make different assumptions and allow different level of dependencies. They learn a network (or its simplification) of probabilistic relationship between the attributes and estimate the likelihood at each node given its parents from D (i.e., one-dimensional likelihood estimation). The joint distribution $p(\mathbf{x}, y)$ is estimated as the product of likelihood of each attribute given their parents in the network:

$$\hat{p}(\mathbf{x}, y) = p(x_1|\pi_1) \times p(x_2|\pi_2) \times \cdots \times p(x_d|\pi_d) \times p(y|\pi_y) \tag{3}$$

where π_i is $parent(x_i)$ and π_y is $parent(y)$.

Though these one-dimensional likelihood generative classifiers have been shown to perform well [3,4,5,6,7], we hypothesize that a multi-dimensional likelihood generative classifier will produce even better results.

In this paper, we propose an ensemble approach to estimate multi-dimensional likelihood without making any explicit assumption about attribute independence. The idea is to construct an ensemble of t multi-dimensional likelihood estimators using random sub-samples $\mathcal{D}_i \subset D$ ($i = 1, 2, \cdots, t$). Each estimator estimates the multi-dimensional likelihood using a random subset of d attributes from $\mathcal{D}_i$. The average estimation from t estimators provides a good approximation of $p(\mathbf{x}|y)$. We call the resulting generative classifier *MassBayes*. It has constant space complexity and constant training time complexity because it employs a fixed-size training subset to build each of the t estimators.

The rest of the paper is structured as follows. Section 2 provides a brief overview of well-known generative classifiers. The proposed method is described in Section 3 followed by the implementation details in Section 4. The empirical evaluation results are presented in Section 5. Finally, we provide conclusions and directions for future research in Section 6.

2 Existing Generative Classifiers

Naive Bayes (NB) [3] is the simplest generative approach that estimates $p(\mathbf{x}, y)$ by assuming that the attributes are statistically independent given y:

$$\hat{p}(\mathbf{x}, y)_{NB} = p(y) \prod_{i=1}^{d} p(x_i|y) \tag{4}$$

Despite the strong independence assumption, it has been shown that NB produces impressive results in many application domains [3,4]. Its simplicity and clear probabilistic semantics have motivated researchers to explore different extensions of NB to improve its performance by relaxing the unrealistic assumption.

BayesNet [5] learns a network of probabilistic relationship among the attributes including the class attribute from the training data. Each node in the network is independent of its non-descendants given the state of its parents. At each node, the conditional probabilities with respect to its parents are learned from D. The joint probability $p(\mathbf{x}, y)$ is estimated as:

$$\hat{p}(\mathbf{x}, y)_{BayesNet} = p(y|\pi_y) \prod_{i=1}^{d} p(x_i|\pi_i) \tag{5}$$

Learning an optimal network requires searching over a set of every possible network, which is exponential in d. It is intractable in high-dimensional problems [8]. NB is the simplest form of a Bayesian network, where each attribute is dependent on y only.

In another simplification of BayesNet, AnDE [7] relaxes the independence assumption by allowing dependency between y and a fixed number of privileged attributes or super-parents. The other attributes are assumed to be independent given the n super-parents and y. AnDE with $n = 0$, A0DE, is NB. AnDE avoids the expensive searching in learning probabilistic dependencies by constructing an ensemble of n-dependence estimators. The joint probability $p(\mathbf{x}, y)$ is estimated as:

$$\hat{p}(\mathbf{x}, y)_{AnDE} = \sum_{s \in S^n} p(\mathbf{x}_s, y) \prod_{j \in \{1,2,\cdots,d\} \setminus s} p(x_j|\mathbf{x}_s, y) \tag{6}$$

where S^n is the collection of all subsets of size n of the set of d attributes $\{1, 2, \cdots, d\}$; and $\mathbf{x}_s$ is a n-dimensional vector of values of $\mathbf{x}$ defined by s.

It has been shown that A1DE and A2DE produce better predictive accuracy than the other state-of-the-art generative classifiers [6,7]. However, it only allows dependencies on a fixed number of attributes and y. Because of the high time complexity of $O\left(N\binom{d}{n+1}\right)$ [1] and space complexity of $O\left(c\binom{d}{n+1}v^{n+1}\right)$, where v is the average number of values for an attribute [7], only A2DE or A3DE is feasible even for a moderate number of dimensions. Furthermore, selecting an appropriate value of n for a particular data set requires a search.

AnDE and many other implementations of BayesNet require all the attributes to be discrete. The continuous-valued attributes must be discretised using a discretisation method before building a classifier.

3 MassBayes: A New Generative Classifier

Rather than aggregating an ensemble of n-dependence single-dimensional likelihood estimators, we propose to aggregate an ensemble of t multi-dimensional likelihood estimators where each likelihood is estimated using different random subsets of d attributes from data. The likelihood $p(\mathbf{x}|y)$ is estimated as:

$$\hat{p}(\mathbf{x}|y) = \frac{1}{t} \sum_{g \in G_t} p(\mathbf{x}_g|y) \tag{7}$$

where G_t is a collection of t subsets of varying sizes of d attributes; and $\mathbf{x}_g$ is a $|g|$-dimensional vector of values of $\mathbf{x}$ defined by g; and $1 \leq |g| \leq d$.

Each $p(\mathbf{x}_g|y)$ is estimated using a random subset of training instances $\mathcal{D} \subset D$, where $|\mathcal{D}| = \psi < N$.

$$\hat{p}(\mathbf{x}_g|y) = \frac{|\mathcal{D}_{y,\mathbf{x}_g}|}{|\mathcal{D}_y|} \tag{8}$$

[1] $\binom{d}{n}$ is a binomial coefficient of n out of d.

where $|\mathcal{D}_{y,\mathbf{x}_g}|$ is the number of instances having attribute values $\mathbf{x}_g$ belonging to class y in $\mathcal{D}$ and $|\mathcal{D}_y|$ is the number of instances belonging to class y in $\mathcal{D}$.

Rather than relying on a specific discretisation method in the preprocessing step, we propose to build a model directly from data, akin to an adaptive multi-dimensional histogram, to determine $\mathbf{x}_g$ which adapts to the local data distribution. The feature space partitioning we employed (to be discussed in Section 4) produces large regions in sparse area and small regions in the dense area of the data distribution.

Let $T(\cdot)$ be a function that divides the feature space into non-overlapping regions and $T(\mathbf{x})$ be the region where $\mathbf{x}$ falls. In a multi-dimensional space, each instance in $\mathcal{D}$ can be isolated by splitting only on few dimensions i.e., only a subset of d attributes ($g \subset \{1, 2, \cdots, d\}$) is used to define $T(\mathbf{x})$. Hence, $|\mathcal{D}_{y,\mathbf{x}_g}|$ is the number of instances belonging to class y in the region $T(\mathbf{x})$. Let $p(T(\mathbf{x})|y)$ be the probability of region $T(\mathbf{x})$ when only class y instances in $\mathcal{D}$ are considered.

$$p(T(\mathbf{x})|y) = \hat{p}(\mathbf{x}_g|y) = \frac{|\mathcal{D}_{y,\mathbf{x}_g}|}{|\mathcal{D}_y|} \tag{9}$$

The new generative classifier, called *MassBayes*, estimates the joint distribution as:

$$\hat{p}(\mathbf{x}, y)_{MassBayes} = p(y)\ \frac{1}{t} \sum_{g \in G_t} p(\mathbf{x}_g|y) = p(y)\ \frac{1}{t} \sum_{i=1}^{t} p(T_i(\mathbf{x})|y) \tag{10}$$

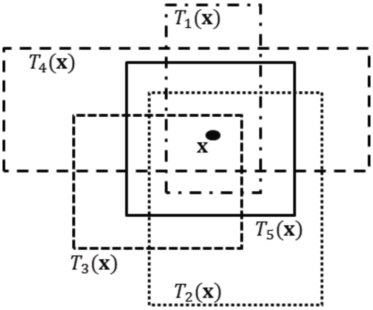


Fig. 1. Different regions from different $T_i(\cdot)$ $(i = 1, 2, \cdots, 5)$ that cover $\mathbf{x}$

The average probability of t different regions $T_i(\mathbf{x})$ $(i = 1, 2, \cdots, t)$, constructed using $\mathcal{D}_i \subset D$, provides a good estimate for $p(\mathbf{x}|y)$ as it estimates the multi-dimensional likelihood by considering the distribution in different local neighbourhood of $\mathbf{x}$ in the data space. An illustrative example is provided in Figure 1. Note that, the estimator employed in MassBayes is not a true density estimator as it does not integrate to 1.

MassBayes has the following characteristics in comparison with A*nDE*:

1. In each estimator, A*n*DE estimates one-dimensional likelihood given a fixed number of super-parents and y, whereas MassBayes estimates multi-dimensional likelihood using varying number of dimensions.
2. In A*n*DE, the ensemble size is fixed to $\binom{d}{n}$. But, MassBayes allows the flexibility for users to set the ensemble size.

3. A*n*DE requires continuous-valued attributes to be discretised before building the model. The performance of A*n*DE is affected by the discretisation technique used. In contrast, MassBayes builds models directly from data. It can be viewed as a dynamic multi-dimensional discretisation where the information loss is minimised by averaging over multiple models.
4. Each model in MassBayes is built with training subset of size $\psi < N$ which gives rise to the constant training time. In contrast, each model in A*n*DE is trained using the entire training set.
5. A*n*DE is a deterministic algorithm whereas MassBayes is a randomised algorithm.
6. Like A*n*DE, MassBayes is a generative classifier without search.

4 Implementation

In order to partition the feature space to define the regions $T_i(\cdot)$, we use the implementation described by Ting and Wells (2010) using a binary tree called *h:d-tree* [9]. A parameter h defines the maximum level of sub-division. The maximum height of a tree is $h \times d$.

Let the data space that covers the instances in $\mathcal{D}$ be $\varDelta$. The data space $\varDelta$ is adjusted to become δ using a random perturbation conducted as follows. For each dimension j, a split point v_j is chosen randomly within the range $max_j(\varDelta) - min_j(\varDelta)$. Then, the new range δ_j along dimension j is defined as $[v_j - r, v_j + r]$, where $r = max(v_j - min_j(\varDelta), max_j(\varDelta) - v_j)$. The new range on all dimensions defines the adjusted work space for the tree building process.

A subset $\mathcal{D}$ is constructed from D by sampling ψ instances without replacement. The sampling process is restarted with D when all the instances are used. The random adjustment of the work space and random sub-sampling, as described earlier, ensure that no two trees are identical.

The dimension to split is selected from a randomised set of d dimensions in a round-robin manner at each level of a tree. A tree is constructed by splitting the work space into two equal-volume half spaces at each level. The process is then repeated recursively on each non-empty half-space. The tree building process stops when there is only one instance in a node or the maximum height is reached.

At the leaf node, the number of instances in the node belonging to each class is stored. Figure 2 shows a typical example of an implementation of $T(\cdot)$ as an *h:d-tree* for $h = 2$ and $d = 2$. The dotted lines enclosed the instances in $\mathcal{D}$ and the solid lines enclosed the adjusted work space which has ranges δ_1 and δ_2 on x_1 and x_2 dimensions. $R1, R2, R3, R4$ and $R5$ represent different regions in $T(\cdot)$ depending on the data distribution in $\mathcal{D}$. Region $R1$ is defined by splitting the work space in x_1 dimension only, $g = \{1\}$, whereas the other four regions use dimensions x_1 and x_2, i.e., $g = \{1, 2\}$.

In the original implementation by Ting and Wells (2010) for mass estimation, each tree is built to the maximum height of $h \times d$ resulting in equal-size regions regardless of the data distribution [9]. In our implementation, in order to adapt

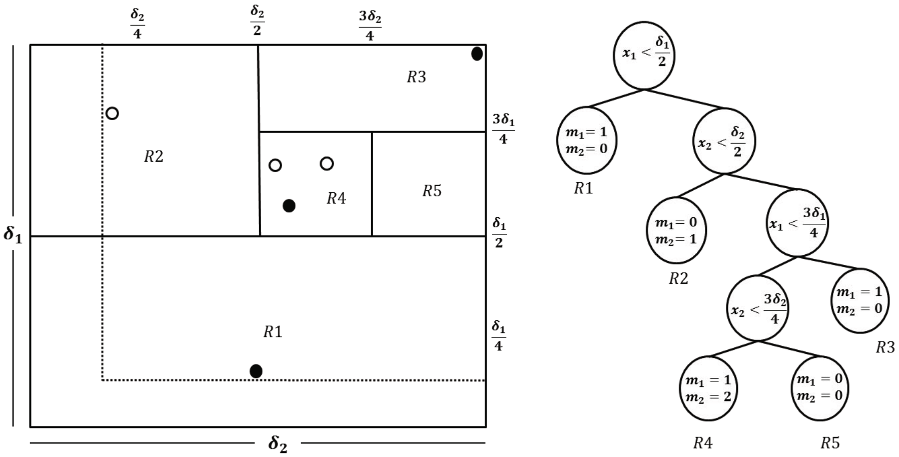


Fig. 2. An example of an h:d-*tree* for $h = 2$ and $d = 2$

to the data distribution, the tree building stops early once the instances are separated. We use the same algorithm as used by Ting and Wells (2010) to generate h:d-*trees* to represent $T_i(\cdot)$ in [9] with the required modification.

The procedures to generate t trees from a given data set D are provided in Algorithms 1 and 2.

The maximum height of each tree is hd, and ψ instances have to be assigned to either of the two child nodes at each level of a tree. Hence, the total training time complexity to construct t trees is $O(thd\psi)$. There are a maximum of ψ (as $\psi < 2^{hd}$ in general) leaf nodes in each tree. The total space complexity is $O(t(d + c)\psi)$.

The time and space complexities of two variants of NB (NB-KDE that estimates $p(x_i|y)$ through kernel density estimation [4]; and NB-Disc that estimates $p(x_i|y)$ through discretisation [10]), AnDE and MassBayes are presented in Table 1. Both training time complexity and space complexity of MassBayes are

Table 1. Time and space complexities of different generative classifiers

Classifiers	Training time	Testing time	Space
NB-KDE [4]	$O(Nd)$	$O(cmd)$	$O(cmd)$
NB-Disc [6]	$O(Nd)$	$O(cd)$	$O(cdv)$
AnDE [7]	$O\left(N\binom{d}{n+1}\right)$	$O\left(cd\binom{d}{n}\right)$	$O\left(c\binom{d}{n+1}v^{n+1}\right)$
MassBayes	$O(thd\psi)$	$O(thd)$	$O\left(t(d+c)\psi\right)$

N: total number of training instances, m: average number of training instances in a class, d: number of dimensions, c: number of classes, v: average number of discrete values of an attribute, n: number of super-parents, t: number of trees, h: level of divisions, and ψ: sample size.

independent of N. Note that the complexities for NB-Disc and AnDE do not include the additional discretisation needed in the preprocessing.

Algorithm 1. BuildTrees(D, t, ψ, h)

Inputs: D - input data, t - number of trees, ψ - sub-sampling size, h - number of times an attribute is employed in a path.
Output: F - a set of t *h:d–trees*

```
$H \leftarrow h \times d$ {Maximum height of a tree}
**Initialize** $F$
**for** $i = 1$ to $t$ **do**
    $\mathcal{D} \leftarrow sample(D, \psi)$ {strictly without replacement}
    $(min,max) \leftarrow$ InitialiseWorkSpace($\mathcal{D}$)
    $A \leftarrow$ {Randomised list of $d$ attributes.}
    $F \leftarrow F \cup$ SingleTree($\mathcal{D}, min, max, 0, A$)
**end for**
**return** $F$
```

Algorithm 2. SingleTree($\mathcal{D}, min, max, \ell, A$)

Inputs: $\mathcal{D}$ - input data, min & max - arrays of minimum and maximum values for each attribute that define a work space, A - a randomised list of d attributes, ℓ - current height level.
Output: an *h:d–tree*

```
**Initialize** $Node(\cdot)$
**while** ($\ell < H$ and $|\mathcal{D}| > 1$) **do**
    $q \leftarrow nextAttribute(A, \ell)$ {Retrieve an attribute from $A$ based on height level.}
    $mid_q \leftarrow (max_q + min_q)/2$
    $\mathcal{D}_l \leftarrow filter(\mathcal{D}, q < mid_p)$
    $\mathcal{D}_r \leftarrow filter(\mathcal{D}, q \geq mid_q)$
    **if** ($|\mathcal{D}_l| = 0$ ) or ($|\mathcal{D}_r| = 0$) **then** {Reduce range for single-branch node.}
        **if** ($|\mathcal{D}_l| > 0$ ) **then** $max_q \leftarrow mid_q$
        **else** $min_q \leftarrow mid_q$
        **end if**
        $\ell \leftarrow \ell + 1$
        continue at the start of while loop
    **end if**
    {Build two nodes: $Left$ and $Right$ as a result of a split into two half-spaces.}
    $temp \leftarrow max_q$; $max_q \leftarrow mid_q$
    $Left \leftarrow$ SingleTree($\mathcal{D}_l, min, max, \ell + 1, A$)
    $max_q \leftarrow temp$; $min_q \leftarrow mid_q$
    $Right \leftarrow$ SingleTree($\mathcal{D}_r, min, max, \ell + 1, A$)
    terminate while loop
**end while**
$classCount \leftarrow updateClassCount(\mathcal{D})$
**return** $Node(Left, Right, SplitAtt \leftarrow q, SplitValue \leftarrow mid_q, classCount)$
```

5 Empirical Evaluation

This section presents the results of the experiments conducted to evaluate the performance of MassBayes against seven well known contenders: two variants of NB (NB-KDE and NB-Disc), BayesNet, three variants of AnDE (A1DE, A2DE, A3DE) and decision tree J48 (i.e., the WEKA [11] version of C4.5 [12]).

MassBayes was implemented in Java using the WEKA platform [11] which also has implementations of NB, BayesNet, A1DE and J48. For A2DE and A3DE, we used the WEKA implementations provided by the authors of AnDE.

All the experiments were conducted using a 10-fold cross validation in a Linux machine with 2.27 GHz processor and 100 GB memory. The average accuracy (%) and the average runtime (seconds) over a 10-fold cross validation were reported. A two-standard-error significance test was conducted to check whether the difference in accuracies of two classifiers was significant. A win or loss was counted if the difference was significant; otherwise, it was a draw.

Ten data sets with $N > 10000$ were used. All the attributes in the data sets are numeric. The properties of the data sets are provided in Table 2. The RingCurve, Wave and OneBig data sets were three synthetic data sets and the rest were real-world data sets from UCI Machine Learning Repository [13]. RingCurve and Wave are subsets of the RingCurve-Wave-TriGaussian data set used in [9] and OneBig is the data set used in [14].

Table 2. Properties of the data sets used

Data sets	#N	#d	#c	Data sets	#N	#d	#c
CoverType	581012	10	7	RingCurve	20000	2	2
MiniBooNE	129596	50	2	Letters	20000	16	26
OneBig	68000	20	10	Magic04	19020	10	2
Shuttle	58000	8	7	Mamograph	11183	6	2
Wave	20000	2	2	Pendigits	10992	16	10

For AnDE, BayesNet and NB-Disc, data sets were discretised by a supervised discretisation technique based on minimum entropy [15] as suggested by the authors of AnDE before building the classification models.

Two variants of MassBayes were used: MassBayes with ($\psi = 5000$) and MassBayes$'$ ($\psi = N$). The other two parameters t and h were set as default to 100 and 10, respectively.

For BayesNet, the parameter *'maximum number of parents'* was set to 100 to examine whether a large number of parents produces better results; and the parameter *'initialise as Naive Bayes'* was set to *'false'* to initialise an empty network structure. The default values were used for the rest of the parameters. All the other classifiers were executed with the default parameter settings.

Table 3. Average classification accuracies (%) over a 10-fold cross validation

Data	Mass	Mass	A3	A2	A1	Bayes	NB-	NB-	
sets	Bayes′	Bayes	DE	DE	DE	Net	KDE	Disc	J48
CoverType	94.00	78.21	88.16	80.81	72.89	87.79	66.72	66.61	92.39
MiniBooNE	92.68	91.11	N/A*	91.48	89.58	90.25	86.07	86.29	90.47
OneBig	100.00	100.00	N/A*	99.81	99.69	99.99	99.98	99.97	99.84
Shuttle	99.89	99.89	99.94	99.94	99.85	99.93	92.68	94.36	99.97
Letters	96.63	95.63	95.11	94.31	88.81	86.97	74.21	73.94	87.92
RingCurve	100.00	100.00	99.99	99.99	99.99	99.99	99.27	99.48	99.91
Wave	100.00	100.00	78.51	78.51	78.51	78.51	77.91	78.51	99.79
Magic04	85.72	85.53	85.08	84.57	83.00	83.46	76.13	78.27	85.01
Mamograph	98.69	98.71	98.51	98.37	98.42	98.54	97.86	97.62	98.57
Pendigits	99.45	99.28	98.80	98.82	97.84	96.81	88.64	87.9	96.56

* Did not complete because of integer overflow error.

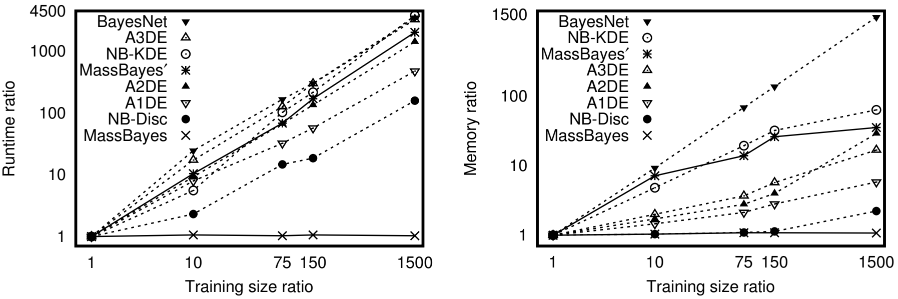
Table 4. Win:Loss:Draw counts of MassBayes over the other contenders in terms of classification accuracy based on the two-standard-error significance test

	A3DE	A2DE	A1DE	BayesNet	NB-KDE	NB-Disc	J48
MassBayes′	4:1:3	7:1:2	7:0:3	7:1:2	10:0:0	10:0:0	7:1:2
MassBayes	3:2:3	6:3:1	7:0:3	6:2:2	10:0:0	10:0:0	6:2:2

Table 5. Average runtime (seconds) over a 10-fold cross validation

	Mass	Mass	A3	A2	A1	Bayes	NB-	NB-	
Data sets	Bayes′	Bayes	DE	DE	DE	Net	KDE	Disc	J48
CoverType	1075.8	45.7	45.6	13.9	4.9	387.9	96.3	3.2	3690.7
MiniBooNE	431.1	33.7	N/A	231.3	5.9	308.9	831.6	2.1	323.8
OneBig	113.9	10.5	N/A	11.6	3.9	432.5	253.0	0.8	15.1
Shuttle	48.5	8.0	1.8	0.7	0.5	6.8	1.5	0.4	4.2
Letters	18.9	5.5	11.5	2.6	0.8	4.9	2.5	0.4	7.3
RingCurve	4.4	2.3	0.2	0.2	0.2	0.3	2.4	0.2	0.4
Wave	4.9	2.1	0.2	0.2	0.2	0.2	2.5	0.1	0.6
Magic04	10.9	3.9	0.7	0.5	0.2	0.7	8.8	0.2	3.4
Mamograph	4.7	3.1	0.3	0.2	0.2	0.3	0.5	0.2	0.4
Pendigits	7.3	3.6	5.7	0.9	0.4	1.8	1.6	0.2	1.2

Table 3 shows the average classification accuracies of MassBayes′ and MassBayes in comparison to the other contenders. The results of the two-standard-error significance test in Table 4 show that both MassBayes′ and MassBayes produced better classification accuracy than the other contenders in most data sets.



(a) Runtime

(b) Space

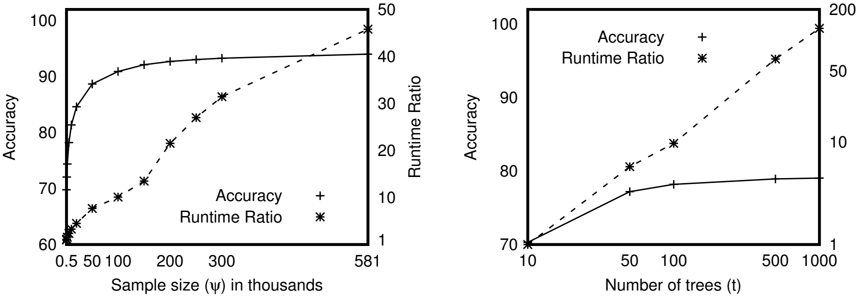
Fig. 3. Scale-up test: MassBayes versus existing generative classifiers. The base for training size ratio is 7000 instances and the bases for runtime ratio and memory ratio are the training time and memory required to save a classification model for 7000 instances. Axes are on logarithmic scales of base 10.

MassBayes produced slightly poorer results than A2DE, A3DE, BayesNet and J48 in CoverType. This was because the default sample size was not enough to yield a good estimate. The accuracy was increased up to 84.62% with $\psi = 20000$ and 88.66% with $\psi = 50000$. More samples are required to grow the trees further to model the distributions well if the class distributions in the feature space are complex. Figure 4(a) shows the improvement in accuracy of MassBayes in CoverType when the sample size was increased.

Table 5 presents the average runtime. In terms of runtime, MassBayes was an order of magnitude faster than A2DE in MiniBooNE; BayesNet in CoverType, MiniBooNE and OneBig; NB-KDE in MiniBooNE and OneBig; and J48 in CoverType and MiniBooNE. It was of the same order of magnitude as A3DE, A2DE, BayesNet, NB-KDE and J48 in many cases and an order of magnitude slower than NB-Disc and A1DE. MassBayes′ was an order of magnitude slower than the other contenders in many data sets. However, it was of the same order of magnitude as A3DE in Letters; A2DE in MiniBooNE; BayesNet and NB-KDE in MiniBooNE and OneBig; and J48 in CoverType and MiniBooNE.

Note that the reported runtime results for AnDE, BayesNet and NB-Disc did not include the discretisation time that must be done as a preprocessing step, which give the existing generative classifiers (except NB-KDE) an unfair advantage over MassBayes. The discretisation time can be substantially large in large data sets. For example, the discretisation took 52 seconds in the largest data set, CoverType. This discretisation time alone was more than the total runtime of MassBayes. Thus, MassBayes in effect runs faster than all existing generative classifiers on equal footing.

In order to examine the scalability of the classifiers in terms of training time and space requirements with the increase in training size N, we used the 48-dimensional (42 irrelevant attributes with constant values) RingCurve-Wave-Tri-Gaussian data set previously employed by Ting and Wells (2010) in [9]. The training data size was increased from 7000 to 70000, half-a-million, 1 million and



(a) Sample size (ψ)

(b) Trees (t)

Fig. 4. Effect of parameters ψ and t on the classification accuracy and runtime of MassBayes in the CoverType data set. The base for the runtime ratio while varying ψ and t is the total runtime (training and testing over a 10-fold cross validation) for $\psi = 500$ and $t = 10$, respectively. The horizontal axis of t and the vertical axis of runtime ratio in (b) are on logarithmic scales of base 10.

10 million by a factor of 1, 10, 75, 150 and 1500, respectively. Figure 3 shows the increase in classification model building time and memory space required to store the classification model for different generative classifiers. Note that the discretisation time was not included in the presented results. The discretisation time increases linearly with the increase in training data size. This additional time for discretisation will increase the training time of AnDE, BayesNet and NB-Disc. MassBayes had constant training time and constant space requirements.

In order to examine the sensitivity of the parameters ψ, t and h in classification accuracy and runtime of MassBayes, we conducted a set of experiments by varying one parameter and fixing the other two to the default values. The result of the experiment varying ψ and t in the largest data set (CoverType) is shown in Figure 4. The increase in runtime was plotted as a ratio to show the factor of runtime increased when the parameters were increased.

In general, accuracy increased up to a certain point and remained flat when each of the three parameters was increased. This indicates that the parameters of MassBayes are not too sensitive in terms of classification accuracy if they are set to sufficiently high values. The runtime increased linearly with t and sublinearly with ψ. With fixed sample size ($\psi = 5000$), increase in h after a certain point did not affect the runtime because the tree building process stopped before reaching the maximum level h once the instances are separated.

6 Conclusions and Future Work

In this paper, we presented a new generative classifier called *MassBayes* that approximates $p(\mathbf{x}|y)$ by aggregating multi-dimensional likelihoods estimated using varying size subsets of features from random subsets of training data. In contrast, existing generative classifiers make assumptions about attribute independence and estimate single-dimensional likelihood only. Our empirical results show that MassBayes produced better classification accuracy than the existing generative classifiers in large data sets.

In terms of runtime, it scales better than the existing generative classifiers in large data sets as it builds models in an ensemble using fixed-size data subsets. The constant training time and space complexities make it an ideal classifier for large data sets and data streams.

Future work includes applying the proposed method in data sets with discrete and mixed attributes and investigating the effectiveness of MassBayes in the data stream context. In this paper, we have rigorously assessed MassBayes with the state-of-the-art Bayesian classifiers. In the near future, we will assess its performance against some well-known discriminative classifiers and their ensembles. The feature space partitioning can be implemented in various ways. It would be interesting to investigate a more intelligent way of feature space partitioning rather than dividing at mid-point of a randomly selected dimension.

Acknowledgement. This work is supported by the Air Force Research Laboratory, under agreement# FA2386-11-1-4112. Sunil Aryal is partially supported by Monash University Postgraduate Publications Award to write this paper. We would like to thank Geoff Webb and the anonymous reviewers for their helpful comments.

References

1. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapmal & Hall/CRC (1986)
2. Ram, P., Gray, A.G.: Density Estimation Trees. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 627–635. ACM, New York (2011)
3. Langley, P., Iba, W., Thompson, K.: An Analysis of Bayesian Classifiers. In: Proceedings of the Tenth National Conference on Artificial Intelligence, pp. 399–406 (1992)
4. Langley, P., John, G.H.: Estimating continuous distribution in Bayesian classifiers. In: Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence (1995)
5. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian Network Classifiers. Machine Learning 29, 131–163 (1997)
6. Webb, G.I., Boughton, J.R., Wang, Z.: Not So Naive Bayes: Aggregating one-dependence estimators. Machine Learning 58, 5–24 (2005)
7. Webb, G., Boughton, J., Zheng, F., Ting, K., Salem, H.: Learning by extrapolation from marginal to full-multivariate probability distributions: decreasingly naive Bayesian classification. Machine Learning 86, 233–272 (2012)
8. Chickering, D.M.: Learning Bayesian Networks is NP-Complete. In: Fisher, D., Lenz, H.J. (eds.) Learning from Data: Artificial Intelligence and Statistics V, pp. 121–130. Springer, Heidelberg (1996)
9. Ting, K.M., Wells, J.R.: Multi-Dimensional Mass Estimation and Mass-Based Clustering. In: Proceedings of IEEE ICDM, pp. 511–520 (2010)

10. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and Unsupervised Discretization of Continuous Features. In: Proceedings of the 12th International Conference on Machine Learning, pp. 194–202. Morgan Kaufmann (1995)
11. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1) (2009)
12. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
13. Frank, A., Asuncion, A.: UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences (2010), `http://archive.ics.uci.edu/ml`
14. Nanopoulos, A., Theodoridis, Y., Manolopoulos, Y.: Indexed-based density biased sampling for clustering applications. IEEE Transaction on Data and Knowledge Engineering 57(1), 37–63 (2006)
15. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous valued attributes for classification learning. In: Proceedings of 14th International Joint Conference on Artificial Intelligence, pp. 1034–1040 (1995)

Fast and Effective Single Pass Bayesian Learning

Nayyar A. Zaidi and Geoffrey I. Webb

Faculty of Information Technology, Monash University, VIC 3800, Australia
{nayyar.zaidi,geoff.webb}@monash.edu

Abstract. The rapid growth in data makes ever more urgent the quest for highly scalable learning algorithms that can maximize the benefit that can be derived from the information implicit in big data. Where data are too big to reside in core, efficient learning requires minimal data access. Single pass learning accesses each data point once only, providing the most efficient data access possible without resorting to sampling. The AnDE family of classifiers are effective single pass learners. We investigate two extensions to A2DE, subsumption resolution and MI-weighting. Neither of these techniques require additional data access. Both reduce A2DE's learning bias, improving its effectiveness for big data. Furthermore, we demonstrate that the techniques are complementary. The resulting combined technique delivers computationally efficient low-bias learning well suited to learning from big data.

Keywords: Averaged n-Dependence Estimators, Subsumption Resolution, Big Data, Naive Bayes, Bias-Variance Trade-off.

1 Introduction

When data are too big to reside in RAM, machine learning has two options. The first is learn from a sample, thereby potentially losing information implicit in the data as a whole. The second is to process the data out-of-core. In the latter case, data access is very expensive, and single-pass learning becomes very desirable. The Averaged n-Dependence Estimators (AnDE) family of Bayesian learning algorithms provide efficient single pass learning with accuracy competitive with the state-of-the-art in-core learning [1]. In addition, AnDE classifiers

- have time complexity linear with respect to the number of training examples,
- directly handle multiple class problems,
- directly handle missing values, and
- do not require parameter tuning.

These features make them strong contenders for application with big data.

Previous research has shown that as n is increased, the bias of the AnDE algorithms decreases, at the cost of an increase in variance [1]. Variance tends to decrease as data quantity increases, so for big data low bias algorithms tend to have an advantage [2]. Hence, for large data, larger n is desirable. Unfortunately, however, large n has high time and space complexity, especially as the

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 149–160, 2013.

dimensionality of the data increases. In practice, A2DE has proven effective for moderate dimensional data.

A number of techniques have demonstrated a capacity to lower the bias of A1DE with negligible computational cost. Subsumption Resolution (SR) [3] achieves this with a form of lazy (classification time) feature elimination. Weightily Averaged One-Dependence Estimators (WAODE) [4] achieves it by weighting the sub-models. While previous studies have demonstrated the independent effectiveness of each of these algorithms, their interoperability has not previously been investigated. In this paper we investigate whether they are compatible and the extent to which applying both together reduces bias relative to applying each alone. Further, neither of these techniques has been studied in the context of AnDE with n greater than 1. We herein investigate their effectiveness when applied to A2DE, both severally and jointly. We reveal that they are indeed effective at further reducing A2DE's bias with minimal additional computation.

The rest of this paper is organized as follows. We discuss related work and our proposed improvements to A2DE in section 2. We will discuss experimental results in section 3. We conclude in section 4.

2 Semi-naive Bayes Method - AnDE

We seek to estimate $\mathrm{P}(y \mid \mathbf{x})$, where y is a class label and $\mathbf{x}$ is a vector of attribute values $\mathbf{x} = \langle x_1, \ldots x_m \rangle$. For notational convenience we define

$$x_{\{i,j,\ldots q\}} = \langle x_i, x_j, \ldots, x_q \rangle.$$

For example, $x_{\{2,3,5\}} = \langle x_2, x_3, x_5 \rangle$. We use $\hat{\mathrm{P}}(\cdot)$ to denote an estimate of $\mathrm{P}(\cdot)$.

AnDE aims to estimate $\mathrm{P}(y \mid \mathbf{x})$ using $\mathrm{P}(y \mid \mathbf{x}) \propto \mathrm{P}(y, \mathbf{x})$ and hence normalizing each $\hat{\mathrm{P}}(y, \mathbf{x})$ to derive the respective $\hat{\mathrm{P}}(y \mid \mathbf{x})$. The required joint probability is estimated using

$$\hat{\mathrm{P}}_{\mathrm{AnDE}}(y, \mathbf{x}) = \begin{cases} \sum_{s \in \binom{\mathcal{A}}{n}} \delta(x_s) \hat{\mathrm{P}}(y, x_s) \prod_{i=1}^{a} \hat{\mathrm{P}}(x_i \mid y, x_s) / \sum_{s \in \binom{\mathcal{A}}{n}} \delta(x_s) & : \sum_{s \in \binom{\mathcal{A}}{n}} \delta(x_s) > 0 \\ \hat{\mathrm{P}}_{\mathrm{A(n\text{-}1)DE}}(y, \mathbf{x}) & : \text{otherwise} \end{cases} \tag{1}$$

where $\binom{\mathcal{A}}{n}$ indicates the set of all size-n subsets of $\{1, \ldots a\}$ and $\delta(x_\alpha)$ is a function that is 1 if the training data contains an object with the value x_α, otherwise 0.

Note that $\mathrm{P}(x_i \mid y, x_s) = 1$ when $i \in s$. Whereas other probability estimates should be smoothed or regularized, smoothed estimates should not be used in this case, and in practice these values are not included in the calculation.

Subsumption resolution [3] is an effective technique for rectifying a specific class of extreme violations of the attribute independence assumption, those where $\mathrm{P}(x_i \mid x_j) = 1.0$. In this case $\mathrm{P}(y \mid \mathbf{x}) = \mathrm{P}(y \mid x_{\{1 \ldots i-1, i+1 \ldots m\}})$ and hence all inaccuracies introduced into $\hat{\mathrm{P}}(y \mid \mathbf{x})$ by this violation of the attribute independence assumption can be avoided by dropping x_i from (1). For example, when the attribute values include *female* and *pregnant* only the latter should be used, when they include *male* and *not-pregnant* only the former should be used, and when they include *female* and *not-pregnant* both should be used. This requires, however, that one infer whether $\mathrm{P}(x_i \mid y, x_s) = 1$ for each pair of attribute values. In the current research we infer that $\mathrm{P}(x_i \mid x_j) = 1.0$ if $\#(x_j) = \#(x_i, x_j) > 100$, where $\#(x_j)$ is the count of the number of times attribute value x_j occurs in the data and $\#(x_i, x_j)$ is the count of the number of times both x_i and x_j occur together in the data. To prevent both attribute values being deleted if they cover exactly the same data, we delete the one with the higher index if $\#(x_i) = \#(x_j)$.

$$\hat{\mathrm{P}}_{\mathrm{AnDE}^{\mathrm{SR}}}(y, \mathbf{x}) = \hat{\mathrm{P}}_{\mathrm{AnDE}}(y, x_{\{i \in \mathbf{x} : \neg \exists j \in \mathbf{x} \#(x_i) = \#(x_i, x_j) > 100 \wedge [\#(x_j) > \#(x_i) \vee j < i]\}})$$

Subsumption resolution has been shown to be effective at reducing the bias of A1DE [5,3].

Another approach to reducing bias in AnDE that has been shown to be effective for A1DE [6,4,7] is to weight the sub-models, modifying (1) to

$$\hat{\mathrm{P}}_{\mathrm{WAnDE}}(y, \mathbf{x}) = \begin{cases} \sum_{s \in \binom{\mathcal{A}}{n}} \delta(x_s) w_s \hat{\mathrm{P}}(y, x_s) \prod_{i=1}^{a} \hat{\mathrm{P}}(x_i \mid y, x_s) / \sum_{s \in \binom{\mathcal{A}}{n}} \delta(x_s) & : \sum_{s \in \binom{\mathcal{A}}{n}} \delta(x_s) > 0 \\ \hat{\mathrm{P}}_{\mathrm{WA(n\text{-}1)DE}}(y, \mathbf{x}) & : \text{otherwise} \end{cases}$$

WAODE [4] weights A1DE, where s is a single attribute value. It sets w_s to the mutual information of the attribute with the class. WAODE is effective at reducing the bias of A1DE with minimal computational overhead. We here generalize that strategy to MI-weighted AnDE, using $w_s = \mathrm{MI}(S, Y)$,

$$\mathrm{MI}(s, Y) = \sum_{y \in Y} \sum_{x_s \in X_s} \mathrm{P}(x_s, y) \log \frac{\mathrm{P}(x_s, y)}{\mathrm{P}(x_s)\mathrm{P}(y)} \tag{2}$$

where Y is the set of class labels and X_s is the cross product of values for attributes with indices in s.

While subsumption resolution and weighting have each been shown to reduce the bias of AnDE in isolation, they have not previously been used in conjunction. To assess the effect of doing so we also evaluate MI-weighted AnDESR,

$$\hat{\mathrm{P}}_{\mathrm{WAnDE}^{\mathrm{SR}}}(y, \mathbf{x}) = \hat{\mathrm{P}}_{\mathrm{WAnDE}}(y, x_{\{i \in \mathbf{x} : \neg \exists j \in \mathbf{x} \#(x_i) = \#(x_i, x_j) > 100 \wedge [\#(x_j) > \#(x_i) \vee j < i]\}})$$

2.1 Computational overheads

AnDE has training time complexity of $\mathrm{O}(t\binom{m}{n+1})$ and classification time complexity of $\mathrm{O}(km\binom{m}{n})$ for classifying a single example, where t is the number of training examples.

Subsumption resolution requires no additional training time and at classification time requires $\binom{m}{2}$ comparisons to identify any subsumed attribute values,

Table 1. Data sets

Domain	Case	Att	Class	Domain	Case	Att	Class
Abalone	4177	9	3	Liver Disorders (Bupa)	345	7	2
Adult	48842	15	2	Lung Cancer	32	57	3
Annealing	898	39	6	Lymphography	148	19	4
Audiology	226	70	24	MAGIC Gamma Telescope	19020	11	2
Auto Imports	205	26	7	Mushrooms	8124	23	2
Balance Scale	625	5	3	Nettalk(Phoneme)	5438	8	52
Breast Cancer (Wisconsin)	699	10	2	New-Thyroid	215	6	3
Car Evaluation	1728	8	4	Nursery	12960	9	5
Census-Income (KDD)	299285	40	2	Optical Digits	5620	49	10
Connect-4 Opening	67557	43	3	Page Blocks Classification	5473	11	5
Contact-lenses	24	5	3	Pen Digits	10992	17	10
Contraceptive Method Choice	1473	10	3	Pima Indians Diabetes	768	9	2
Covertype	581012	55	7	Postoperative Patient	90	9	3
Credit Screening	690	16	2	Primary Tumor	339	18	22
Echocardiogram	131	7	2	Promoter Gene Sequences	106	58	2
German	1000	21	2	Segment	2310	20	7
Glass Identification	214	10	3	Sick-euthyroid	3772	30	2
Haberman's Survival	306	4	2	Sign	12546	9	3
Heart Disease (Cleveland)	303	14	2	Sonar Classification	208	61	2
Hepatitis	155	20	2	Splice-junction Gene Sequences	3190	62	3
Horse Colic	368	22	2	Statlog (Shuttle)	58000	10	7
House Votes 84	435	17	2	Syncon	600	61	6
Hungarian	294	14	2	Teaching Assistant Evaluation	151	6	3
Hypothyroid(Garavan)	3772	30	4	Tic-Tac-Toe Endgame	958	10	2
Ionosphere	351	35	2	Vehicle	846	19	4
Iris Classification	150	5	3	Volcanoes	1520	4	4
King-rook-vs-king-pawn	3196	37	2	Vowel	990	14	11
Labor Negotiations	57	17	2	Waveform-5000	5000	41	3
LED	1000	8	10	Wine Recognition	178	14	3
Dermatology	366	35	6	Zoo	101	17	7
Cylinder	540	40	2	Letter Recognition	20000	17	26
Spambase	4601	58	2	Localization	164860	7	3
Wall-following	5456	25	4	Poker-hand	1025010	11	10
yeast	1484	9	10	Thyroid	9169	30	20
Satellite	6435	37	6	Musk1	476	167	2
Chess	551	40	2				

and hence does not increase the classification time complexity so long as $n > 0$. In practice subsumption resolution can substantially reduce classification time by reducing the number combinations of attribute values that must be processed.

MI weighted AnDE requires the calculation of the weights at training time, $\mathrm{O}(k\binom{m}{n})$. In practice this is dominated by the training time complexity of regular AnDE and hence does not increase the effective complexity and the additional training time impost is modest. The classification time impact is negligible.

3 Experimental Results

The experiments are conducted in the Weka work-bench (version 3.5.7) on data sets described in table 1. Each algorithm is tested on each data set using 20 rounds of 2-fold cross validation. Probability estimates were smoothed using m-estimation [8] with $m = 1$.

The bias-variance decomposition provides valuable insights into the components of the error of learned classifiers. *Bias* denotes the systematic component of error, which describes how closely the learner is able to describe the decision surfaces for a domain. *Variance* describes the component of error that stems from sampling, which reflects the sensitivity of the learner to variations in the training sample [9,10]. There are a number of different bias-variance decomposition definitions. In this research, we use the bias and variance definitions of [9], together with the repeated cross-validation bias-variance estimation method [10]. When two algorithms are compared, we count the number of data sets for which one algorithm performs better, equally well or worse than the other on a given measure. A standard binomial sign test, assuming that wins and losses are equiprobable, is applied to these records. We assess a difference as significant if the outcome of a two-tailed binomial sign test is less than 0.05. The base probabilities of each algorithm are estimated using m-estimation, since in our initial experiments it leads to more accurate probabilities than Laplace estimation for naive Bayes, A1DE and A2DE. The data sets are divided into four categories. First, consisting of all 71 data sets. Second, large data sets with number of instances $> 10,000$. Third, medium data sets with number of instances > 1000 and $< 10,000$. Fourth, small data sets with number of instances < 1000. The following techniques are compared:

- NB, Standard naive Bayes with m-estimates of probabilities.
- A1DE, $\hat{\mathrm{P}}_{\mathrm{AnDE}}(y, \mathbf{x})$ with $n = 1$.
- A1DE-S, $\hat{\mathrm{P}}_{\mathrm{AnDE}^{\mathrm{SR}}}(y, \mathbf{x})$ with $n = 1$.
- A1DE-W, $\hat{\mathrm{P}}_{\mathrm{WAnDE}}(y, \mathbf{x})$ with $n = 1$.
- A1DE-SW, $\hat{\mathrm{P}}_{\mathrm{WAnDE}^{\mathrm{SR}}}(y, \mathbf{x})$ with $n = 1$.
- A2DE, $\hat{\mathrm{P}}_{\mathrm{AnDE}}(y, \mathbf{x})$ with $n = 2$.
- A2DE-S, $\hat{\mathrm{P}}_{\mathrm{AnDE}^{\mathrm{SR}}}(y, \mathbf{x})$ with $n = 2$.
- A2DE-W, $\hat{\mathrm{P}}_{\mathrm{WAnDE}}(y, \mathbf{x})$ with $n = 2$.
- A2DE-SW, $\hat{\mathrm{P}}_{\mathrm{WAnDE}^{\mathrm{SR}}}(y, \mathbf{x})$ with $n = 2$.
- RF10, Random Forest with 10 decision trees.

Numeric attributes are discretized using MDL discretization [11] for all compared techniques except Random Forest. Bias, variance, 0-1 Loss and RMSE results are reported in the following sections.

3.1 Comparison of Bias and Variance

The WDL bias and variance results are shown in Tables 2 and 3 respectively with significant ($\alpha = 0.05$) results shown in bold. We summarize the results as:

- Both weighting and subsumption resolution reduce the bias of both A1DE and A2DE significantly more often than they increase it.
- Jointly applying both weighting and subsumption resolution to either A1DE or A2DE reduces bias significantly more often than it increases it relative to applying either alone.
- Both weighting and subsumption resolution increase the variance of both A1DE and A2DE more often than they decrease it, although these results are not always statistically significant.

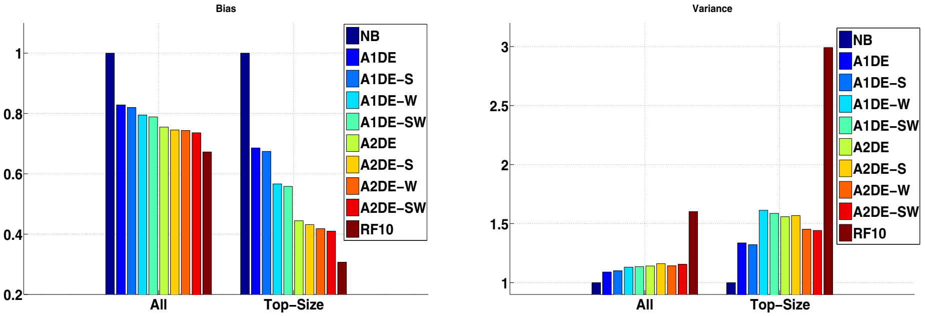


Fig. 1. Averaged Bias (left) and Variance (Right) results normalized with respect to NB. The error-bars are ordered in the same sequence as in the legend.

Table 2. Win/Draw/Loss of Bias Comparison, all data sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**56/3/12**								
A1DE-S	**56/2/13**	**34/33/4**							
A1DE-W	**56/3/12**	**51/4/16**	**41/4/26**						
A1DE-SW	**59/2/10**	**51/6/14**	**44/5/22**	**25/41/5**					
A2DE	**57/2/12**	**53/3/15**	**47/5/19**	**44/3/24**	**41/4/26**				
A2DE-S	**57/2/12**	**51/3/17**	**50/4/17**	**48/3/20**	**48/3/20**	**31/35/5**			
A2DE-W	**57/2/12**	**54/4/13**	**52/4/15**	**52/5/14**	**49/5/17**	**48/7/16**	36/8/27		
A2DE-SW	**58/2/11**	**54/4/13**	**54/4/13**	**53/4/14**	**52/4/15**	**50/6/15**	**45/7/19**	**32/32/7**	
RF10	**57/1/13**	**54/2/15**	**53/2/16**	**51/3/17**	**51/3/17**	**49/4/18**	**49/3/19**	**49/4/18**	**49/4/18**

– Jointly applying both weighting and subsumption resolution to either A1DE or A2DE increases variance more often than it decrease it relative to applying either alone, but these differences are also not always statistically significant.
– Random Forest has lower bias and higher variance significantly more often than the reverse relative to all AnDE variants.

The average bias and variance results are shown in figure 1. One can see that RF10 has better bias than any member of the AnDE family but worse variance.

3.2 Comparison of the Accuracy - 0-1 Loss and RMSE

The above results show that subsumption resolution and weighting both reduce bias at the cost of an increase in variance. These two techniques have synergistic effect. Used together they further reduce bias at cost of increased variance. If we accept that as data quantity increases, the bias term will increasingly dominate error, we should expect these strategies to be most effective at decreasing error for larger data sets.

Table 3. Win/Draw/Loss of Variance Comparison, all data sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**23/3/45**								
A1DE-S	**22/2/47**	13/33/25							
A1DE-W	**21/2/48**	**18/6/47**	**17/7/47**						
A1DE-SW	**20/2/49**	**18/6/47**	**17/7/47**	11/43/17					
A2DE	**22/3/46**	28/3/40	**25/3/43**	38/4/29	37/3/31				
A2DE-S	**20/2/49**	**20/2/49**	**22/5/44**	29/3/39	29/2/40	**10/35/26**			
A2DE-W	**20/3/48**	**26/3/42**	26/2/43	30/4/37	30/4/37	22/11/38	36/9/26		
A2DE-SW	**19/3/49**	**23/2/46**	**24/2/45**	26/5/40	28/4/39	**21/7/43**	29/9/33	**9/33/29**	
RF10	**8/1/62**	**8/2/61**	**9/2/60**	**8/4/59**	**9/3/59**	**6/2/63**	**7/2/62**	**6/4/61**	**6/4/61**

The WDL 0-1 Loss and RMSE results are shown in Table 4 and 5 respectively. The significant ($\alpha = 0.05$) results are shown in bold. We summarize the results as:

- Subsumption resolution decreases error more often than not relative to both A1DE and A2DE for both measures of error and for almost all of the different data collections. The exceptions are A1DE, 0-1 loss, medium data and A2DE, 0-1 loss, small data for which there are draws. However, not all these results are statistically significant.
- Subsumption resolution with weighting can decrease error for both measures of error for the first two collections (all and large data sets). As predicted, the effectiveness reduces as data set sizes reduce and for medium data sets, subsumption resolution with weighting can have slightly worst performance relative to weighting in terms of 0-1 loss but better in terms of RMSE. The results, however, are non-significant. The same pattern can be observed in smaller data sets with subsumption resolution and weighting not very effective.
- Subsumption resolution in tandem with weighting can project AnDE to be competitive to RF10, winning significantly on all data sets in terms of the two error measures on all and small data sets. On medium data sets, it results in winning significantly often for A2DE and non-significant often for A1DE over RF10. On large data sets, both A1DE and A2DE lose to RF10. The results are, however, not significant. With five wins and seven losses over RF10, we conjecture, that AnDE with subsumption resolution and weighting, with all desirable properties of learning from big data, is a strong contender for big data learning.

To give an indication of the magnitude of the differences in performance, the average 0-1 Loss results and RMSE results are shown in figures 2 and 3 respectively. It is apparent that A2DE-SW achieves lower average 0-1 loss and RMSE on all of small, medium and large datasets, although this advantage does diminish to being very slight for the largest datasets.

Table 4. Win/Draw/Loss of 0-1 Loss Comparison

All Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**53/4/14**								
A1DE-S	**51/4/16**	**27/31/13**							
A1DE-W	**50/2/19**	35/8/28	29/8/34						
A1DE-SW	**48/3/20**	38/6/27	32/10/29	20/42/9					
A2DE	**54/3/14**	**50/4/17**	**48/4/19**	**45/8/18**	**41/10/20**				
A2DE-S	**49/3/19**	**46/3/22**	**45/4/22**	**44/5/22**	**43/5/23**	23/34/14			
A2DE-W	**48/2/21**	**46/3/22**	**45/4/22**	**47/6/18**	**46/6/19**	36/8/27	35/9/27		
A2DE-SW	**47/2/22**	**45/2/24**	**42/3/26**	**45/7/19**	**44/6/21**	37/9/25	36/11/24	21/34/16	
RF10	40/1/30	28/2/41	26/5/40	**24/2/45**	**24/2/45**	**22/3/46**	**20/4/47**	**17/3/51**	**17/3/51**

Large Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**12/0/0**								
A1DE-S	**12/0/0**	7/4/1							
A1DE-W	**12/0/0**	**9/2/1**	7/1/4						
A1DE-SW	**12/0/0**	**10/1/1**	8/2/2	5/6/1					
A2DE	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**11/0/1**				
A2DE-S	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**7/5/0**			
A2DE-W	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	9/1/2	5/1/6		
A2DE-SW	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	9/1/2	8/1/3	**6/6/0**	
RF10	**12/0/0**	9/0/3	9/0/3	9/0/3	9/0/3	7/1/4	6/1/5	5/1/6	5/1/6

Medium Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**18/1/0**								
A1DE-S	**19/0/0**	7/5/7							
A1DE-W	**19/0/0**	13/1/5	10/3/6						
A1DE-SW	**18/1/0**	12/1/6	10/4/5	5/8/6					
A2DE	**19/0/0**	**17/0/2**	**15/1/3**	11/1/7	11/1/7				
A2DE-S	**19/0/0**	**16/0/3**	**14/1/4**	12/1/6	12/1/6	6/9/4			
A2DE-W	**19/0/0**	**17/0/2**	**16/2/1**	**15/2/2**	**14/2/3**	**13/3/3**	**13/3/3**		
A2DE-SW	**19/0/0**	**16/0/3**	**14/1/4**	**14/2/3**	**14/2/3**	11/4/4	11/5/3	5/7/7	
RF10	**15/0/4**	10/0/9	8/3/8	6/1/12	6/1/12	6/1/12	5/2/12	**4/1/14**	**4/1/14**

Small Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	23/3/14								
A1DE-S	20/4/16	13/22/5							
A1DE-W	19/2/19	13/5/22	12/4/24						
A1DE-SW	18/2/20	16/4/20	14/4/22	**10/28/2**					
A2DE	23/3/14	21/4/15	21/3/16	22/7/11	19/9/12				
A2DE-S	18/3/19	18/3/19	19/3/18	20/4/16	19/4/17	10/20/10			
A2DE-W	17/2/21	17/3/20	17/2/21	20/4/16	20/4/16	14/4/22	17/5/18		
A2DE-SW	16/2/22	17/2/21	16/2/22	19/5/16	18/4/18	17/4/19	17/5/18	10/21/9	
RF10	13/1/26	**9/2/29**	**9/2/29**	**9/1/30**	**9/1/30**	**9/1/30**	**9/1/30**	**8/1/31**	**8/1/31**

Table 5. Win/Draw/Loss of RMSE Comparison

All Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**59/2/10**								
A1DE-S	**59/2/10**	**32/32/7**							
A1DE-W	**58/1/12**	39/5/27	29/5/37						
A1DE-SW	**59/1/11**	**44/4/23**	35/4/32	**24/42/5**					
A2DE	**59/2/10**	**49/3/19**	**41/4/26**	**50/1/20**	**47/1/23**				
A2DE-S	**57/2/12**	**47/1/23**	**45/2/24**	**48/3/20**	**47/3/21**	**28/30/13**			
A2DE-W	**53/2/16**	**44/1/26**	**44/1/26**	**46/4/21**	**45/3/23**	**41/8/22**	28/10/33		
A2DE-SW	**54/1/16**	**44/1/26**	**44/1/26**	**46/3/22**	**46/3/22**	**41/6/24**	35/11/25	**25/34/12**	
RF10	42/0/29	32/0/39	30/0/41	28/2/41	28/1/42	**23/0/48**	**22/1/48**	**19/1/51**	**16/1/54**

Large Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**12/0/0**								
A1DE-S	**12/0/0**	7/4/1							
A1DE-W	**12/0/0**	8/2/2	6/1/5						
A1DE-SW	**12/0/0**	9/1/2	6/1/5	5/6/1					
A2DE	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**11/0/1**				
A2DE-S	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	7/4/1			
A2DE-W	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	9/1/2	4/0/8		
A2DE-SW	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**12/0/0**	**10/0/2**	8/1/3	7/4/1	
RF10	**12/0/0**	9/0/3	9/0/3	9/0/3	9/0/3	6/0/6	6/0/6	6/0/6	5/0/7

Medium Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**18/1/0**								
A1DE-S	**18/1/0**	8/7/4							
A1DE-W	**18/1/0**	**15/2/2**	**13/3/3**						
A1DE-SW	**18/1/0**	**14/2/3**	**15/2/2**	7/10/2					
A2DE	**18/1/0**	**15/1/3**	**13/2/4**	10/1/8	10/1/8				
A2DE-S	**17/2/0**	**15/1/3**	**14/2/3**	11/1/7	10/1/8	8/7/4			
A2DE-W	**17/2/0**	**15/1/3**	**16/1/2**	**14/1/4**	13/1/5	**14/4/1**	**12/4/3**		
A2DE-SW	**17/1/1**	**15/1/3**	**15/1/3**	**14/1/4**	**14/1/4**	**12/4/3**	**12/4/3**	6/9/4	
RF10	14/0/5	10/0/9	10/0/9	6/1/12	7/0/12	7/0/12	7/0/12	**3/0/16**	**3/0/16**

Small Data Sets

	NB	A1DE	A1DE-S	A1DE-W	A1DE-SW	A2DE	A2DE-S	A2DE-W	A2DE-SW
A1DE	**29/1/10**								
A1DE-S	**29/1/10**	**17/21/2**							
A1DE-W	**28/0/12**	16/1/23	**10/1/29**						
A1DE-SW	**29/0/11**	21/1/18	14/1/25	**12/26/2**					
A2DE	**29/1/10**	22/2/16	16/2/22	**28/0/12**	26/0/14				
A2DE-S	**28/0/12**	20/0/20	19/0/21	25/2/13	25/2/13	13/19/8			
A2DE-W	24/0/16	17/0/23	16/0/24	20/3/17	20/2/18	18/3/19	12/6/22		
A2DE-SW	25/0/15	17/0/23	17/0/23	20/2/18	20/2/18	19/2/19	15/6/19	12/21/7	
RF10	16/0/24	**13/0/27**	**11/0/29**	13/1/26	**12/1/27**	**10/0/30**	**9/1/30**	**10/1/29**	**8/1/31**

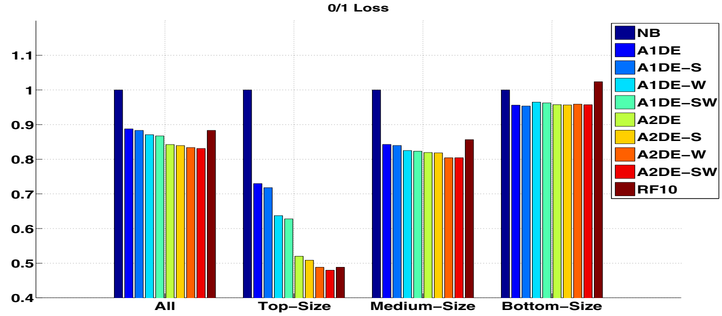


Fig. 2. Average 0-1 Loss results on 4 different collections of data sets normalized with respect to NB. The error-bars are ordered in the same sequence as in the legend.

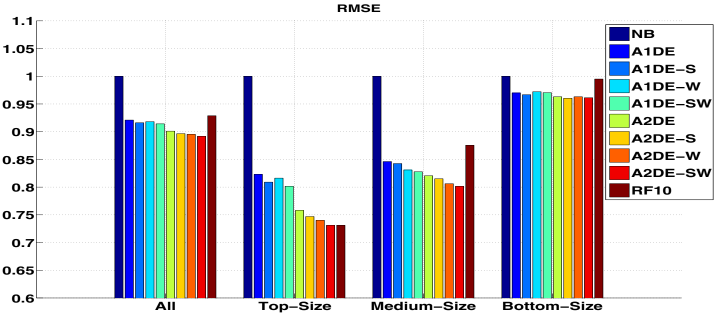


Fig. 3. Average RMSE results on 4 different collections of data sets normalized with respect to NB. The error-bars are ordered in the same sequence as in the legend.

3.3 Analysis of Classification and Learning Time

The average results of classification and learning time for all the compared techniques are shown in figure 4. One can see that subsumption resolution can greatly reduce A2DE's classification time. While A2DE-S and A2DE-SW require only slightly less training time on average than RF10, the training time complexity of AnDE and its variants is linear with respect to data quantity while RF10's is super-linear, as shown by the difference between training times for all data and for large data. The training time advantage would substantially increase if RF10 were applied to data that were too large to maintain in RAM. A2DE and its variants require substantially more classification time than RF10, even with the decreases introduced by subsumption resolution. However, it can be seen that the classification time of RF10 is also super-linear with respect to training set size, whereas AnDE's is not. This is due to the size of the trees increasing as the data quantity increases.

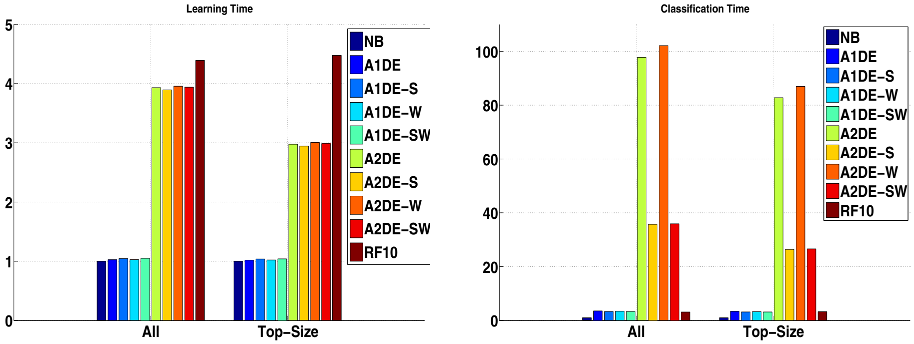


Fig. 4. Averaged Learning and Classification timing results normalized with respect to NB. The error-bars are ordered in the same sequence as in the legend.

3.4 Code

The code of the methods proposed in this work can be obtained from the website, `https://sourceforge.net/projects/averagedndepend/`.

4 Conclusion

AnDE is a strong contender for learning from big data due to its capacity to learn in a single pass through the training data, and consequent training time complexity that is linear with respect to the number of training examples. Weighting using mutual information and subsumption resolution have both previously been demonstrated to be computationally efficient approaches to further reducing the bias of A1DE. As low bias is desirable when learning from large data, it is important to assess the extent to which each of these approaches can reduce the bias of A1DE's lower bias sibling, A2DE. Further, it is important to assess the extent to which these two approaches can augment one another.

The experimental evidence is conclusive. We confirm previous findings that each technique reduces A1DE's bias. We demonstrate that each technique is just as effective at reducing A2DE's bias as it is at reducing A1DE's. We find further that there is strong synergy between the two techniques and that they operate in tandem to reduce the bias of both A1DE and A2DE more effectively than does either in isolation. As is inevitable, these gains in bias come at a cost in increased variance. This bias/variance trade-off can be expected to play out in different error outcomes for different types of data. In particular, for big data, where variance can be expected to be low, low bias can be expected to result in low error [2]. Our experiments demonstrate that this expectation is born out in practice, with both weighting and subsumption resolution reducing error on the largest datasets significantly more often than not relative to standard A2DE and with the two in tandem significantly often further reducing the error relative to MI-weighting alone, and often, but not significantly so, further reducing the error of subsumption resolution alone.

We compared A2DE with MI-weighting and subsumption resolution against the state-of-the-art in-core learning algorithm Random Forest. Random Forest is a lower bias algorithm. However, that bias advantage comes with a considerable variance disadvantage. Even for datasets with 10,000+ training examples Random Forest achieved lower error slightly less often than higher relative to A2DE-SW.

Using only single-pass learning, A2DE with MI-weighting and subsumption resolution achieves accuracy that is very competitive with the state-of-the-art in-core learning, making it a desirable algorithm for learning from very large data.

Acknowledgment. This research has been supported by the Australian Research Council under grant DP110101427 and Asian Office of Aerospace Research and Development, Air Force Office of Scientific Research under contract FA23861214030.

References

1. Webb, G.I., Boughton, J., Zheng, F., Ting, K.M., Salem, H.: Learning by extrapolation from marginal to full-multivariate probability distributions: Decreasingly naive Bayesian classification. Machine Learning 86(2), 233–272 (2012)
2. Brain, D., Webb, G.I.: The need for low bias algorithms in classification learning from large data sets. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, pp. 62–73. Springer, Heidelberg (2002)
3. Zheng, F., Webb, G.I., Suraweera, P., Zhu, L.: Subsumption resolution: An efficient and effective technique for semi-naive Bayesian learning. Machine Learning 87(1), 93–125 (2012)
4. Jiang, L., Zhang, H.: Weightily averaged one-dependence estimators. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 970–974. Springer, Heidelberg (2006)
5. Zheng, F., Webb, G.I.: Efficient lazy elimination for averaged one-dependence estimators. In: Proceedings of the Twenty-Third International Conference on Machine Learning (ICML 2006), pp. 1113–1120 (2006)
6. Cerquides, J., de Mántaras, R.L.: Robust Bayesian linear classifier ensembles. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 72–83. Springer, Heidelberg (2005)
7. Yang, Y., Webb, G., Cerquides, J., Korb, K., Boughton, J., Ting, K.: To select or to weigh: A comparative study of linear combination schemes for superparent-one-dependence estimators. IEEE Transactions on Knowledge and Data Engineering 19(12), 1652–1665 (2007)
8. Cestnik, B.: Estimating probabilities: A crucial task in machine learning. In: Proceedings of the Ninth European Conference on Artificial Intelligence (ECAI 1990). Pitman, London (1990)
9. Kohavi, R., Wolpert, D.: Bias plus variance decomposition for zero-one loss functions. In: Proceedings of the Thirteenth International Conference on Machine Learning, pp. 275–283. Morgan Kaufmann, San Francisco (1996)
10. Webb, G.I.: Multiboosting: A technique for combining boosting and wagging. Machine Learning 40(2), 159–196 (2000)
11. Fayyad, U.M., Irani, K.B.: On the handling of continuous-valued attributes in decision tree generation. Machine Learning 8(1), 87–102 (1992)

Sparse Reductions for Fixed-Size Least Squares Support Vector Machines on Large Scale Data

Raghvendra Mall and Johan A.K. Suykens

Department of Electral Engineering, ESAT-SCD, Katholieke Universiteit Leuven,
Kasteelpark Arenberg,10 B-3001 Leuven, Belgium

Abstract. Fixed-Size Least Squares Support Vector Machines (FS-LSSVM) is a powerful tool for solving large scale classification and regression problems. FS-LSSVM solves an over-determined system of M linear equations by using Nyström approximations on a set of prototype vectors (PVs) in the primal. This introduces sparsity in the model along with ability to scale for large datasets. But there exists no formal method for selection of the right value of M. In this paper, we investigate the sparsity-error trade-off by introducing a second level of sparsity after performing one iteration of FS-LSSVM. This helps to overcome the problem of selecting a right number of initial PVs as the final model is highly sparse and dependent on only a few appropriately selected prototype vectors (SV) is a subset of the PVs. The first proposed method performs an iterative approximation of L_0-norm which acts as a regularizer. The second method belongs to the category of threshold methods, where we set a window and select the SV set from correctly classified PVs closer and farther from the decision boundaries in the case of classification. For regression, we obtain the SV set by selecting the PVs with least minimum squared error (mse). Experiments on real world datasets from the UCI repository illustrate that highly sparse models are obtained without significant trade-off in error estimations scalable to large scale datasets.

1 Introduction

LSSVM [3] and SVM [4] are state of the art learning algorithms in classification and regression. The SVM model has inherent sparsity whereas the LSSVM model lacks sparsity. However, previous works like [1],[5],[6] address the problem of sparsity for LSSVM. One such approach was introduced in [7] and uses a *fixed-size least squares support vector machines*. The major benefit which we obtain from FS-LSSVM is its applicability to large scale datasets. It provides a solution to the LSSVM problem in the primal space resulting in a parametric model and sparse representation. The method uses an explicit expression for the feature map using the Nyström method [8],[9] as proposed in [16]. In [7], the authors obtain the initial set of prototype vectors (PVs) i.e. M vectors while maximizing the quadratic Rènyi entropy criterion leading to a sparse representation in the primal space. The error of FS-LSSVM model approximates to that of LSSVM for $M \ll N$. But this is not the sparsest solution and selecting an initial value of M

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 161–173, 2013.

is an existent problem. In [11], they try to overcome this problem by iteratively building up a set of conjugate vectors of increasing cardinality to approximately solve the over-determined FS-LSSVM linear system. But if few iterations don't suffice to result in a good approximation then the cardinality will be M.

The L_0-norm counts the number of non-zero elements of a vector. It results in very sparse models by returning models of low complexity and acts as a regularizer. However, obtaining this L_0-norm is an NP-hard problem. Several approximations to it are discussed in [12]. In this paper, we modify the iterative sparsifying procedure introduced in [13] and used for LSSVM the technique as shown in [14] and reformulate it for the FS-LSSVM. We apply this formulation on FS-LSSVM because for large scale datasets like Magic Gamma, Adult and Slice Localization we are overwhelmed with memory $O(N^2)$ and computational time $O(N^3)$ constraints when applying the L_0-norm scheme directly on LSSVM [14] or SVM [13]. The second proposed method performs an iteration of FS-LSSVM and then based on a user-defined window selects a subset of PVs as SV. For classification, the selected vectors satisfy the property of being correctly classified and are either closer or farther from the decision boundary since they well determine the extent of the classes. But for regression, the SV set comprises those PVs which have the least *mse* and are best-fitted by the regressor. Once the SV set is determined we re-perform FS-LSSVM resulting in highly sparse models without significant trade-off in accuracy and scalable to large scale datasets.

The contribution of this work involves providing smaller solutions which use $M' < M$ PVs for FS-LSSVM, obtaining highly sparse models with guarantees of low complexity (L_0-norm of $\tilde{w}$) and overcoming the problem of memory and computational constraints faced by L_0-norm based approaches for LSSVM and SVM on large scale datasets. Sparseness enables exploiting memory and computationally efficiency, e.g. matrix multiplications and inversions. The solutions that we propose utilize the best of both FS-LSSVM and sparsity inducing measures on LSSVM and SVM resulting in highly sparse and scalable models. Table 1 provides a conceptual overview of LSSVM, FS-LSSVM and proposes Reduced FS-LSSVM along with the notations used in the rest of the paper. Figures 1 and 2 illustrate our proposed approaches on the Ripley and Motorcycle dataset.

Table 1. For Reduced FS-LSSVM, we first perform FS-LSSVM in the Primal. Then a sparsifying procedure is performed in the Dual of FS-LSSVM as highlighted in the box in the middle resulting in a reduced SV set. Then FS-LSSVM is re-performed in the Primal as highlighted by the box on the right. We propose two sparsifying procedures namely L_0 reduced FS-LSSVM and Window reduced FS-LSSVM.

	LSSVM		**FS-LSSVM**		**Reduced FS-LSSVM**
SV/Train	N/N		M/N		M'/M
Primal	w		$\tilde{w}$		$\boxed{w'}$
	$\phi(\cdot) \in \mathbb{R}^{N_h}$	Step 1	$\tilde{\phi}(\cdot) \in \mathbb{R}^{M}$	Step 2	$\boxed{\phi'(\cdot) \in \mathbb{R}^{M'}}$
Dual	α	→	$\boxed{\tilde{\alpha}}$	→	α'
	$K \in \mathbb{R}^{N \times N}$		$\boxed{\tilde{K} \in \mathbb{R}^{M \times M}}$		$K' \in \mathbb{R}^{M' \times M'}$

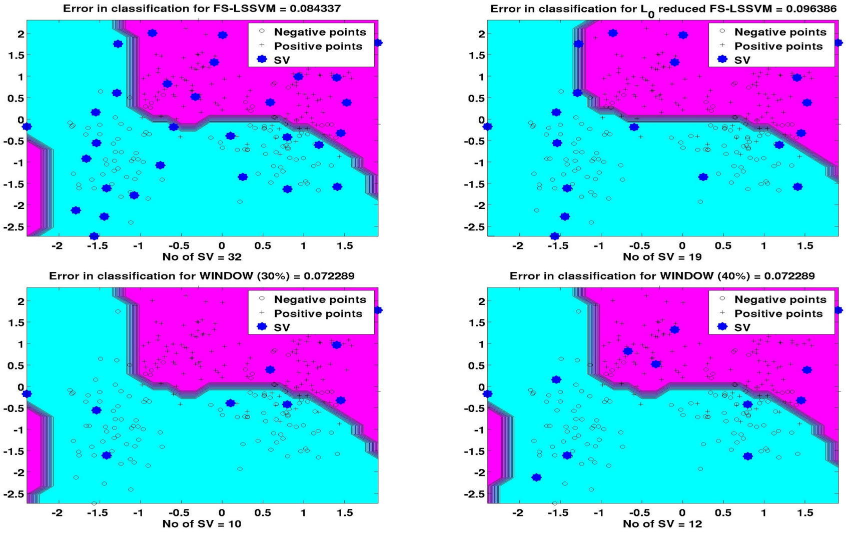


Fig. 1. Comparison of the best randomization result out of 10 randomizations for the proposed methods with FS-LSSVM for Ripley Classification data

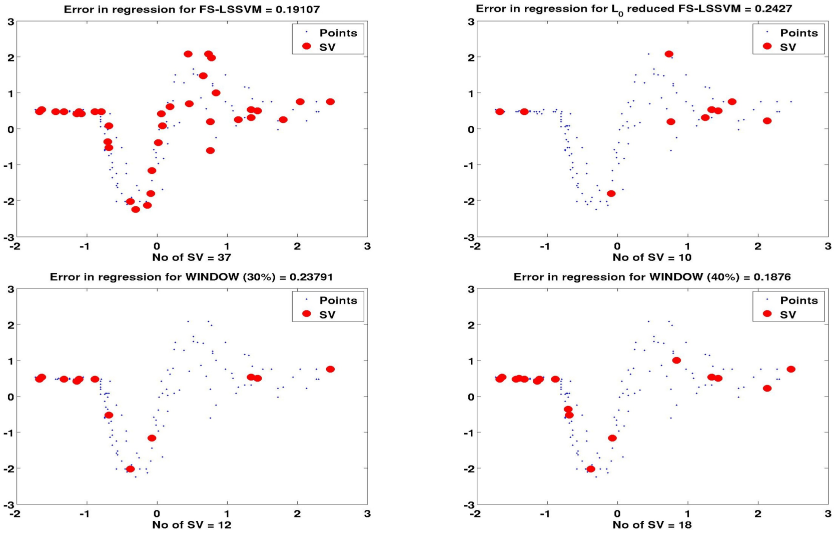


Fig. 2. Comparison of the best randomization result out of 10 randomizations for the proposed methods with FS-LSSVM for Motorcycle Regression data

2 L_0 reduced FS-LSSVM

Algorithm 1 gives a brief summary of the FS-LSSVM method. We first propose an approach using the L_0-norm to reduce $\tilde{w}$ and acting as a regularizer in the objective function. It tries to estimate the optimal subset of PVs leading to sparse solutions. For our formulation, the objective function is to minimize the error estimations of these prototype vectors regulated by L_0-norm of $\tilde{w}$. We modify the procedure described in [13], [14] and consider the following generalized primal problem:

$$\begin{aligned} \min_{\tilde{\alpha},\tilde{b},\tilde{e}} \quad & J(\tilde{\alpha},\tilde{e}) = \frac{1}{2}\sum_j \lambda_j \tilde{\alpha}_j^2 + \frac{\gamma}{2}\sum_i \tilde{e}_i^2 \\ \text{s.t.} \quad & \sum_j \tilde{\alpha}_j \tilde{K}_{ij} + \tilde{b} = y_i - \tilde{e}_i,\ i = 1,\ldots,M \end{aligned} \tag{1}$$

where $\tilde{w} \in \mathbb{R}^M$ and can be written as $\tilde{w} = \sum\limits_j \tilde{\alpha}_j \tilde{\phi}(x_j)$. The regularization term is now not on $\| \tilde{w} \|^2$ but on $\| \tilde{\alpha} \|^2$. The regularization weights are given by the prefix λ_j coefficients. This formulation is similar to [14] with the difference that it is made applicable here to large scale datasets. These $\tilde{\alpha}_j$ are coefficients of linear combination of the features which result in $\tilde{w}$ vector. The set of PVs is represented as $\mathcal{S}_{PV}$. The $\tilde{e_i}$ are error estimates and are determined only for the vectors belonging to the set $\mathcal{S}_{PV}$. Thus, the training set comprises of the vectors belonging to the set $\mathcal{S}_{PV}$.

Introducing the coefficient β for Lagrangian $\mathcal{L}$ one obtains: $\partial\mathcal{L}/\partial\tilde{\alpha}_j = 0 \Rightarrow \tilde{\alpha}_j = \sum\limits_j \beta_i K_{ij}/\lambda_j$, $\partial\mathcal{L}/\partial\tilde{b} = 0 \Rightarrow \sum\limits_i \beta_i = 0$, $\partial\mathcal{L}/\partial\tilde{e}_i = 0 \Rightarrow \beta_i = \gamma\tilde{e}_i$, $\partial\mathcal{L}/\partial\beta_i = 0 \Rightarrow \sum\limits_j \tilde{\alpha}_j K_{ij} + \tilde{b} = y_i - \tilde{e}_i$, $\forall i$.Combining the conditions $\partial\mathcal{L}/\partial\tilde{\alpha}_i = 0$, $\partial\mathcal{L}/\partial\tilde{e}_i = 0$ and $\partial\mathcal{L}/\partial\beta_i = 0$ and after little algebraic manipulation yields $\sum\limits_k \beta_k H_{ik} + \tilde{b} = y_i$, with $H = \tilde{K} diag(\lambda)^{-1}\tilde{K} + I_M/\gamma$ and $\tilde{K}$ is a kernel matrix. The kernel matrix $\tilde{K}$ is defined as $\tilde{K}_{ij} = \tilde{\phi}(x_i)^\intercal\tilde{\phi}(x_j)$ where $x_i \in \mathcal{S}_{PV}, x_j \in \mathcal{S}_{PV}$, $\tilde{\phi}(x_i) \in \mathbb{R}^M$ and $H \in \mathbb{R}^{M\times M}$.

Algorithm 1. Fixed-Size LSSVM method

```
Data: $\mathbb{D}_n = \{(x_i, y_i) : x_i \in \mathbb{R}^d, y_i \in \{+1,-1\}$ for classification & $y_i \in \mathbb{R}$ for
        regression, $i = 1,\ldots,N\}$.
Determine the kernel bandwidth using the multivariate rule-of-thumb [17].
Given the number of PVs, perform prototype vector selection using quadratic
Rènyi entropy criterion.
Determine the tuning parameters $\sigma$ and $\gamma$ performing fast $v$-fold cross
validation as described in [10].
Given the optimal tuning parameters, get FS-LSSVM parameters $\tilde{w}$ & $\tilde{b}$.
```

This, together with $\partial\mathcal{L}/\partial\tilde{b} = 0$, results in the linear system

$$\left[\begin{array}{c|c} 0 & 1_k^\intercal \\ \hline 1_k & H \end{array}\right] \left[\begin{array}{c} \tilde{b} \\ \hline \beta \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right] \tag{2}$$

The procedure to obtain sparseness involves iteratively solving the system (2) for different values of λ and is described in Algorithm 2. Considering the t^{th} iteration, we can build the matrix $H^t = \tilde{K}diag(\lambda^t)^{-1}\tilde{K} + I_M/\gamma$ and solve the system of linear equations to obtain the value of β^t and $\tilde{b}^t$. From this solution we get $\tilde{\alpha}^{t+1}$ and most of its element tend to zero, the $diag(\lambda^{t+1})^{-1}$ will end up having many zeros along the diagonal due to the values allocated to λ^{t+1}. It was shown in [13] that as $t \to \infty$, $\tilde{\alpha}^t$ converges to a stationary point $\tilde{\alpha}^*$ and this model is guaranteed to be sparse and result in set SV. This iterative sparsifying procedure converges to a local minimum as the L_0-norm problem is NP-hard. Since this $\tilde{\alpha}^*$ depends on the initial choice of weights, we set them to the FS-LSSVM solution $\tilde{w}$, so as to avoid ending up in different local minimal solutions.

Algorithm 2. L_0 reduced FS-LSSVM method

```
Data: Solve FS-LSSVM to obtain initial $\tilde{w}$ & $\tilde{b}$
$\tilde{\alpha} = \tilde{w}(1 : M)$
$\lambda_i \leftarrow \tilde{\alpha}_i, i = 1, \dots, M$
while convergence do
    $H \leftarrow \tilde{K}diag(\lambda)^{-1}\tilde{K} + I_M/\gamma$
    Solve system (2) to give $\beta$ and $\tilde{b}$
    $\tilde{\alpha} \leftarrow diag(\lambda)^{-1}\tilde{K}\beta$
    $\lambda_i \leftarrow 1/\tilde{\alpha}_i^2, i = 1, \dots, M'$
Result: indices = find($|\tilde{\alpha}_i| > 0$)
```

The *convergence* of Algorithm 2 is assumed when the $\parallel \tilde{\alpha}^t - \tilde{\alpha}^{t+1} \parallel /M'$ is lower than 10^{-4} or when the number of iterations t exceeds 50. The result of the approach is the *indices* of those PVs for which $|\tilde{\alpha}_i| > 10^{-6}$. These *indices* provide the set of most appropriate prototype vectors (SV). The FS-LSSVM method (Algorithm 1) is re-perfomed using only this set SV. We are training only on the set $\mathcal{C}_{PV}$ and not on the entire training data because the H matrix becomes $N \times N$ matrix which cannot in memory for large scale datasets.

The time complexity of the proposed methods is bounded by solving the linear system of equations (2). An interesting observation is that the H matrix becomes sparser after each iteration. This is due to the fact that $diag(\lambda)^{-1} = diag(\tilde{\alpha}_1^2, \dots, \tilde{\alpha}_M^2)$ and most of these $\tilde{\alpha}_i \to 0$. Thus the H matrix becomes sparser in each iteration such that after some iterations inverting H matrix is equivalent to inverting each element of the H matrix. The computation time is dominated by matrix multiplication to construct the H matrix. The H matrix construction can be formulated as multiplications of two matrices i.e. $P = \tilde{K}diag(\lambda)^{-1}$ and $H = P\tilde{K}$. The P matrix will become sparser as it multiplies the $\tilde{K}$ matrix with $diag(\lambda)^{-1}$. Let $\tilde{M}$ be the number of columns in P matrix with elements $\neq 0$.

This number i.e. $\tilde{M}$ can be much less than M. Thus, for the L_0 reduced FS-LSSVM the time required for the sparsifying procedure is given by $\mathcal{O}(M^2\tilde{M})$ and the average memory requirement is $\mathcal{O}(M^2)$.

3 Window Reduced FS-LSSVM

In [5], it was proposed to remove the support vectors with smaller $|\alpha_i|$ for the LSSVM method. But this approach doesn't greatly reduce the number of support vectors. In [6], the authors proposed to remove support vectors with the larger $y_i f(x_i)$ as they are farther from decision boundary and easiest to classify. But these support vectors are important as they determine the true extent of a class.

We propose window based SV selection method for both classification and regression. For classification, we select the vectors which are correctly classified and closer and farther from the decision boundary. An initial FS-LSSVM method determines the $\tilde{y}$ for the PVs. To find the reduced set SV, we first remove the prototype vectors which were misclassified ($\tilde{y} \neq y$) as shown in [2]. Then, we sort the estimated $f(x_j)$, $\forall j \in \mathit{CorrectPV}$ where $\mathit{CorrectPV}$ is the set of correctly classified PVs to obtain a sorted vector S. This sorted vector is divided into two halves one containing the sorted positive estimations ($\tilde{y}$) corresponding to positive class and the other containing sorted negative values ($\tilde{y}$) corresponding to negative class. The points closer to the decision boundary have smaller positive and smaller negative estimations ($|\tilde{y}|$) and the points farther from the decision boundary have the larger positive and larger negative estimations ($|\tilde{y}|$) as depicted in Figure 1. So these vectors corresponding to these estimations are selected. Selecting correctly classified vectors closer to decision boundary prevents over-fitting and selecting vectors farther from the decision boundary helps to identify the extent of the classes.

For regression, we select the prototype vectors which have least mse after one iteration of FS-LSSVM. We estimate the squared errors for the PVs and out of these prototype vectors select those vectors which have the least mse to form the set SV. They are the most appropriately estimated vectors as they have the least error and so are most helpful in estimating a generalization of the regression function. By selection of prototype vectors which have least mse we prevent selection of outliers as depicted in Figure 2. Finally, a FS-LSSVM regression is re-performed on this set SV.

The percentage of vectors selected from the initial set of prototype vectors is determined by the window. We experimented with various window size i.e. $(30, 40, 50)$ percent of the initial prototype vectors (PVs). For classification, we selected half of the window from the positive class and the other half from the negative class. In case the classes are not balanced and number of PVs in one class is less than half the window size then all the correctly classified vectors from those PVs are selected. In this case, we observe that the number of selected prototype vectors (SV) can be less than window size. The methodology to perform this Window reduced FS-LSSVM is presented in Algorithm 3.

This method result in better generalization error with smaller variance achieving sparsity. The trade-off with estimated error is not significant and in several cases it leads to better results as will be shown in the experimental results. As we increase the window size the variation in estimated error decreases and estimated error also decreases until the median of the estimated error becomes nearly constant as is depicted in Figure 3.

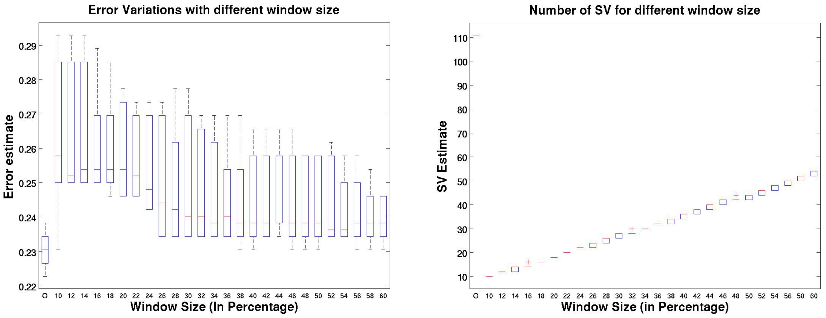


Fig. 3. Trends in error & SV with increasing window size for Diabetes dataset compared with the FS-LSSVM method represented here as ‘O’

Algorithm 3. Window reduced FS-LSSVM

```
Data: $\mathbb{D}_n = \{(x_i, y_i) : x_i \in \mathbb{R}^d, y_i \in \{+1, -1\}$ for $Classification$ & $y_i \in \mathbb{R}$ for
        $Regression$, $i = 1, \ldots, N\}$.
Perform FS-LSSVM using the initial set of PVs of size $M$ on training data.
if Classification then
    $CorrectPV$ = Remove misclassified prototype vectors
    $S = sort(f(x_i)) \; \forall i \in CorrectPV$;
    $A = S(:) > 0$; $B = S(:) < 0$;
    $begin = windowsize/4$;
    $endA = size(A) - windowsize/4$;
    $endB = size(B) - windowsize/4$;
    SV = $[A[begin \;\; endA]; B[begin \;\; endB]]$;
if Regression then
    Estimate the squared error for the initially selected PVs
    SV = Select the PVs with least mean squared error
Re-perform the FS-LSSVM method using the reduced set SV of size $M'$ on
training data
```

4 Computational Complexity and Experimental Results

4.1 Computational Complexity

The computation time of FS-LSSVM method involves:

- Solving a linear system of size $M+1$ where M is the number of prototype vectors selected initially (PV).
- Calculating the Nyström approximation and eigenvalue decomposition of the kernel matrix of size M once.
- Forming the matrix product $[\tilde{\phi}(x_1), \tilde{\phi}(x_2), \ldots, \tilde{\phi}(x_n)]^\intercal [\tilde{\phi}(x_1), \tilde{\phi}(x_2), \ldots, \tilde{\phi}(x_n)]$.

The computation time is $\mathcal{O}(NM^2)$ where N is dataset size as shown in [10]. We already presented the computation time for the iterative sparsifying procedure for L_0 reduced FS-LSSVM. For this approach, the computation time $\mathcal{O}(M^3)$. So, it doesn't have an impact on the overall computational complexity as we will observe from the experimental results. In our experiments, we selected $M = \lceil k \times \sqrt{N} \rceil$ where $k \in \mathbb{N}$, the complexity of L_0 reduced FS-LSSVM can be rewritten as $\mathcal{O}(k^2N^2)$. We experimented with various values of k and observed that after certain values of k, the change in estimated error becomes nearly irrelevant. In our experiments, we choose the value of k corresponding to the first instance after which the change in error estimations becomes negligible.

For the window based method, we have to run the FS-LSSVM once and based on window size obtain the set SV which is always less than PVs i.e. $M' \leq M$. The time-complexity for re-performing the FS-LSSVM on the set SV is $\mathcal{O}(M'^2N)$ where N is the size of the dataset. The overall time complexity of the approach is $\mathcal{O}(M^2N)$ required for Nyström approximation and the average memory requirement is $\mathcal{O}(NM)$.

4.2 Dataset Description

All the datasets on which the experiments were conducted are from UCI benchmark repository [15]. For classification, we experimented with Ripley (RIP), Breast-Cancer (BC), Diabetes (DIB), Spambase (SPAM), Magic Gamma (MGT) and Adult (ADU). The corresponding dataset size are 250,682,768,4061,19020, 48842 respectively. The corresponding k values for determining the initial number of prototype vector are 2,6,4,3,3 and 3 respectively. The datasets Motorcycle, Boston Housing, Concrete and Slice Localization are used for regression whose size is $111, 506, 1030, 53500$ and their k values are 6,5,6 and 3 respectively.

4.3 Experiments

All the experiments are performed on a PC machine with Intel Core i7 CPU and 8 GB RAM under Matlab 2008a. We use the RBF-kernel for kernel matrix construction in all cases. As a pre-processing step, all records containing unknown values are removed from consideration. Input values have been normalized. We compare the performance of our proposed approaches with the normal FS-LSSVM classifier/regressor, L_0 LSSVM [14], SVM and ν-SVM. The last two methods are implemented in the LIBSVM software with default parameters. All methods use a cache size of 8 GB. Shrinking is applied in the SVM case. All comparisons are made on 10 randomizations of the methods.

The comparison is performed on an out-of-sample test set consisting of 1/3 of the data. The first 2/3 of the data is reserved for training and cross-validation. The tuning parameters σ and γ for the proposed FS-LSSVM methods and SVM methods are obtained by first determining good initial starting values using the method of coupled simulated annealing (CSA) in [18]. After that a derivative-free simplex search is performed. This extra step is a fine tuning procedure resulting in more optimal tuning parameters and better performance.

Table 2 provides a comparison of the mean estimated error ± its deviation, mean number of selected prototype vectors SV and a comparison of the mean computation time ± its deviation for 10 randomizations of the proposed approaches with FS-LSSVM and SVM methods for various classification and regression data sets. Figure 4 represents the estimated error, run time and variations in number of selected prototype vectors for Adult (ADU) and Slice Localization (SL) datasets respectively.

4.4 Performance Analysis

The proposed approaches i.e L_0 reduced FS-LSSVM and Window reduced FS-LSSVM method introduce more sparsity in comparison to FS-LSSVM and SVM methods without significant trade-off for classification. For smaller datasets L_0 LSSVM produces extremely few support vectors but for datasets like SPAM, Boston Housing and Concrete it produces more support vectors. For some datasets like breast-cancer and diabetes, it can be seen from Table 2 that proposed approaches results in better error estimations than other methods with much smaller set SV. For datasets like SPAM and MGT, the trade-off in error is not significant considering the reduction in number of PVs (only 78 prototype vectors required by L_0 reduced FS-LSSVM for classifying nearly 20,000 points). From Figure 4, we observe the performance for Adult dataset. The window based methods result in lower error estimate using fewer but more appropriate SV. Thus the idea of selecting correctly classified points closer and farther from the decision boundary results in better determining the extent of the classes. These sparse solutions typically lead to better generalization of the classes. The mean time complexity for different randomizations is nearly the same.

For regression, as we are trying to estimate a continuous function, if we greatly reduce the PVs to estimate that function, then the estimated error would be higher. For datasets like boston housing and concrete, the estimated error by the proposed methods is more than FS-LSSVM method but the amount of sparsity introduced is quite significant. These methods result in reduced but more generalized regressor functions and the variation in the estimated error of window based approach is lesser as in comparison to L_0-norm based method. This is because in each randomization, the L_0-norm reduces to different number of prototype vectors whereas the reduced number of prototype vectors (SV) for window based method is fixed and is uninfluenced by variations caused by outliers as the SV have least *mse*. This can be observed for the Slice Localization dataset in Figure 4. For this dataset, L_0 reduced FS-LSSVM estimates lower error than window approach. This is because for this dense dataset, the L_0-norm based

Table 2. Comparison of performance of different methods for UCI repository datasets. The '*' represents no cross-validation and performance on fixed tuning parameters due to computational burden and '-' represents cannot run due to memory constraints.

Test Classification(Error and Mean SV)												
	RIP		BC		DIB		SPAM		MGT		ADU	
Algorithm	Error	SV	Error	SV	Error	SV	Error	SV	Error	SV	Error	SV
FS-LSSVM	**0.0924 ± 0.01**	32	0.047 ± 0.004	155	0.233 ± 0.002	111	0.077 ± 0.002	204	**0.1361 ± 0.001**	414	0.146 ± 0.0003	664
L_0 LSSVM	0.183 ± 0.1	7	0.04 ± 0.011	17	0.248 ± 0.034	10	**0.0726 ± 0.005**	340	-	-	-	-
C-SVC	0.153 ± 0.04	81	0.054 ± 0.07	318	0.249 ± 0.0343	290	0.074 ± 0.0007	800	0.144 ± 0.0146	7000	0.1519(*)	11,085
ν-SVC	0.1422 ± 0.03	86	0.061 ± 0.05	330	0.242 ± 0.0078	331	0.113 ± 0.0007	1525	0.156 ± 0.0142	7252	0.161(*)	12,205
L_0 reduced FS-LSSVM	0.1044 ± 0.0085	17	**0.0304 ± 0.009**	19	0.239 ± 0.03	36	0.1144 ± 0.053	144	0.1469 ± 0.004	115	0.1519 ± 0.002	78
Window (30%)	0.095 ± 0.009	10	**0.0441 ± 0.0036**	46	**0.2234 ± 0.006**	26	0.1020 ± 0.002	60	0.1522 ± 0.002	90	**0.1449 ± 0.003**	154
Window (40%)	**0.091 ± 0.015**	12	**0.0432 ± 0.0028**	59	**0.2258 ± 0.008**	36	0.0967 ± 0.002	78	0.1481 ± 0.0013	110	**0.1447 ± 0.003**	183
Window (50%)	**0.091 ± 0.009**	16	**0.0432 ± 0.0019**	66	**0.2242 ± 0.005**	45	0.0955 ± 0.002	98	0.1468 ± 0.0007	130	**0.1446 ± 0.003**	213

Test Regression(Error and Mean SV)								
	Motorcycle		Boston Housing		Concrete		Slice Localization	
Algorithm	Error	SV	Error	SV	Error	SV	Error	SV
FS-LSSVM	**0.2027 ± 0.007**	37	**0.1182 ± 0.001**	112	**0.1187 ± 0.006**	192	**0.053 ± 0.0**	694
L_0 LSSVM	0.2537 ± 0.0723	9	0.16 ± 0.05	65	0.165 ± 0.07	215	-	-
ϵ-SVR	0.2571 ± 0.0743	69	0.158 ± 0.037	226	0.23 ± 0.02	670	0.1017(*)	13,012
ν-SVR	0.2427 ± 0.04	51	0.16 ± 0.04	226	0.22 ± 0.02	330	0.0921(*)	12,524
L_0 reduced FS-LSSVM	0.62 ± 0.29	5	0.1775 ± 0.057	48	0.2286 ± 0.03	43	0.1042 ± 0.06	495
Window (30%)	0.32 ± 0.03	12	0.1465 ± 0.02	34	0.2 ± 0.005	58	0.1313 ± 0.0008	208
Window (40%)	0.23 ± 0.03	15	0.1406 ± 0.006	46	0.1731 ± 0.006	78	0.116 ± 0.0005	278
Window (50%)	0.23 ± 0.02	21	**0.1297 ± 0.009**	56	0.1681 ± 0.0039	96	0.1044 ± 0.0006	347

Train & Test Classification(Computation Time)						
Algorithm	RIP	BC	DIB	SPAM	MGT	ADU
FS-LSSVM	4.346 ± 0.143	28.2 ± 0.691	15.02 ± 0.688	181.1 ± 7.75	3105.6 ± 68.6	23,565 ± 1748
L_0 LSSVM	5.12 ± 0.9	28.2 ± 5.4	39.1 ± 5.9	950 ± 78.5	-	-
C-SVC	13.7 ± 4	43.36 ± 3.37	24.8 ± 3.1	1010 ± 785	20,603 ± 396	139,730(*)
ν-SVC	13.6 ± 4.5	41.67 ± 2.436	23.23 ± 2.3	785 ± 22	35,299 ± 357	139,927(*)
L_0 reduced FS-LSSVM	4.349 ± 0.28	28.472 ± 1.07	15.381 ± 0.5	193.374 ± 8.35	3185.7 ± 100.2	22,485 ± 650
Window (30%)	4.28 ± 0.22	28.645 ± 1.45	14.5 ± 0.5	171.78 ± 4.82	3082.9 ± 48	22,734 ± 1520
Window (40%)	4.33 ± 0.27	27.958 ± 1.21	14.6 ± 0.75	172.62 ± 5.3	3052 ± 42	22,466 ± 1705
Window (50%)	4.36 ± 0.27	28.461 ± 1.3	14.255 ± 0.5	167.39 ± 2.31	3129 ± 84	22,621 ± 2040

Train & Test Regression(Computational Time)				
Algorithm	Motorcycle	Boston Housing	Concrete	Slice Localization
FS-LSSVM	3.42 ± 0.12	14.6 ± 0.27	55.43 ± 0.83	37,152 ± 1047
L_0 LSSVM	10.6 ± 1.75	158.86 ± 3.2	753 ± 35.5	-
ϵ-SVR	24.6 ± 4.75	63 ± 1	55.43 ± 0.83	252,438(*)
ν-SVR	25.5 ± 3.9	61 ± 1	168 ± 3	242,724(*)
L_0 reduced FS-LSSVM	3.426 ± 0.18	15.4 ± 0.35	131 ± 2	36,547 ± 1992
Window (30%)	3.52 ± 0.244	15.04 ± 0.17	56.75 ± 1.35	36,087 ± 1066
Window (40%)	3.425 ± 0.153	15.04 ± 0.27	55.67 ± 0.81	35,906 ± 1082
Window (50%)	3.5 ± 0.226	15.04 ± 0.15	55.06 ± 1.088	36,183 ± 1825

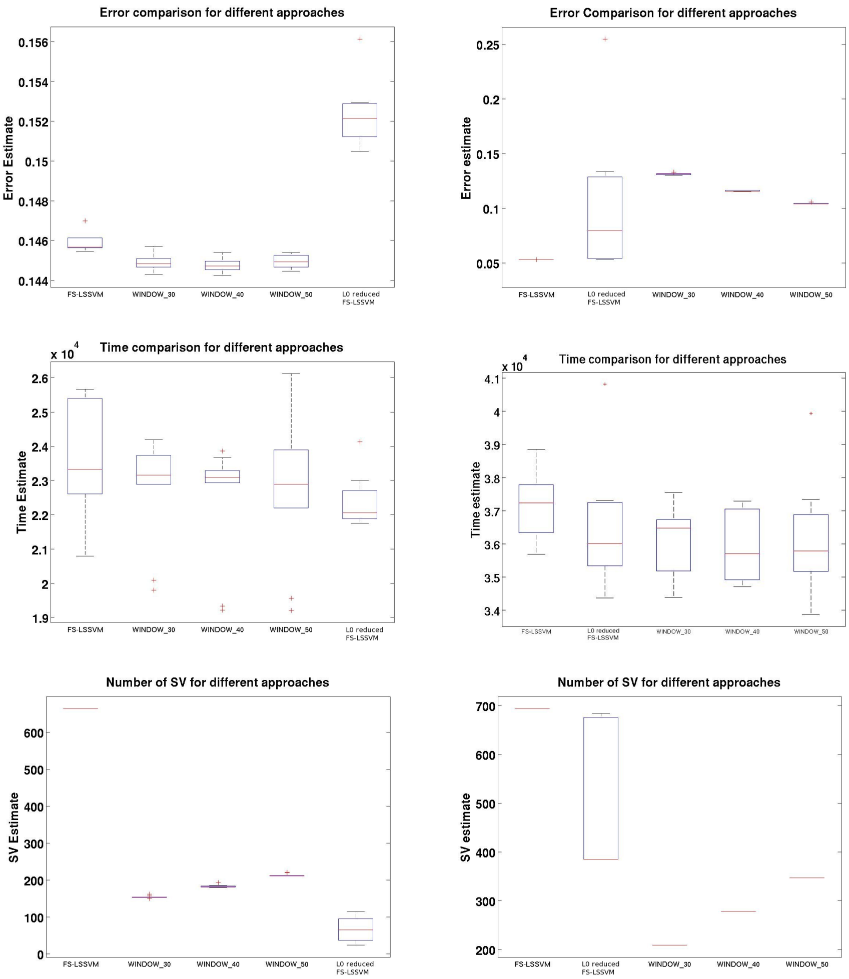


Fig. 4. Comparison of performance of proposed approaches with FS-LSSVM method for Adult & Slice Localization datasets

FS-LSSVM requires more SV (495) than window based method $(208, 278, 347)$ which signifies more vectors are required for better error estimation. The proposed models are of magnitude $(2-10)$x sparser than the FS-LSSVM method.

5 Conclusion

In this paper, we proposed two sparse reductions to FS-LSSVM namely L_0 reduced and Window reduced FS-LSSVM. These methods are highly suitable for mining large scale datasets overcoming the problems faced by L_0 LSSVM [14] and FS-LSSVM. We developed the L_0 reduced FS-LSSVM based on iteratively sparsifying L_0-norm training on the initial set of PVs. We also introduced a Window reduced FS-LSSVM trying to better determine the underlying structure of model by selection of more appropriate prototype vectors (SV). The resulting approaches are compared with normal FS-LSSVM, L_0 LSSVM and two kinds of SVM (C-SVM and ν-SVM from LIBSVM software) with promising performances and timing results using smaller and sparser models.

Acknowledgements. This work was supported by Research Council KUL, ERC AdG A-DATADRIVE-B, GOA/10/09MaNet, CoE EF/05/006, FWO G.0588.09, G.0377.12, SBO POM, IUAP P6/04 DYSCO, COST intelliCIS.

References

[1] Hoegaerts, L., Suykens, J.A.K., Vandewalle, J., De Moor, B.: A comparison of pruning algorithms for sparse least squares support vector machines. In: Pal, N.R., Kasabov, N., Mudi, R.K., Pal, S., Parui, S.K. (eds.) ICONIP 2004. LNCS, vol. 3316, pp. 1247–1253. Springer, Heidelberg (2004)

[2] Geebelen, D., Suykens, J.A.K., Vandewalle, J.: Reducing the Number of Support Vectors of SVM classifiers using the Smoothed Seperable Case Approximation. IEEE Transactions on Neural Networks and Learning Systems 23(4), 682–688 (2012)

[3] Suykens, J.A.K., Vandewalle, J.: Least Squares Support Vector Machine Classifiers. Neural Processing Letters 9(3), 293–300 (1999)

[4] Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer (1995)

[5] Suykens, J.A.K., Lukas, L., Vandewalle, J.: Sparse approximation using Least Squares Support Vector Machines. In: Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS 2000), pp. 757–760 (2000)

[6] Li, Y., Lin, C., Zhang, W.: Improved Sparse Least-Squares Support Vector Machine Classifiers. Neurocomputing 69(13), 1655–1658 (2006)

[7] Suykens, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J.: Least Squares Support Vector Machines. World Scientific Publishing Co., Pte., Ltd., Singapore (2002)

[8] Nyström, E.J.: Über die praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. Acta Mathematica 54, 185–204 (1930)

[9] Baker, C.T.H.: The Numerical Treatment of Integral Equations. Oxford Claredon Press (1983)

[10] De Brabanter, K., De Brabanter, J., Suykens, J.A.K., De Moor, B.: Optimized Fixed-Size Kernel Models for Large Data Sets. Computational Statistics & Data Analysis 54(6), 1484–1504 (2010)

[11] Karsmakers, P., Pelckmans, K., De Brabanter, K., Hamme, H.V., Suykens, J.A.K.: Sparse conjugate directions pursuit with application to fixed-size kernel methods. Machine Learning, Special Issue on Model Selection and Optimization in Machine Learning 85(1), 109–148 (2011)

[12] Weston, J., Elisseeff, A., Schölkopf, B., Tipping, M.: Use of the Zero Norm with Linear Models and Kernel Methods. Journal of Machine Learning Research 3, 1439–1461 (2003)

[13] Huang, K., Zheng, D., Sun, J., et al.: Sparse Learning for Support Vector Classification. Pattern Recognition Letters 31(13), 1944–1951 (2010)

[14] Lopez, J., De Brabanter, K., Dorronsoro, J.R., Suykens, J.A.K.: Sparse LSSVMs with L_0-norm minimization. In: ESANN 2011, pp. 189–194 (2011)

[15] Blake, C.L., Merz, C.J.: UCI repository of machine learning databases, Irvine, CA (1998), `http://archive.ics.uci.edu/ml/datasets.html`

[16] Williams, C.K.I., Seeger, M.: Using the Nyström method to speed up kernel machines. In: Advances in Neural Information Processing Systems, vol. 13, pp. 682–688 (2001)

[17] Scott, D.W., Sain, S.R.: Multi-dimensional Density Estimation. Data Mining and Computational Statistics 23, 229–263 (2004)

[18] Xavier de Souza, S., Suykens, J.A.K., Vandewalle, J., Bolle, D.: Coupled Simulated Annealing for Continuous Global Optimization. IEEE Transactions on Systems, Man, and Cybertics - Part B 40(2), 320–335 (2010)

Discovery of Regional Co-location Patterns with k-Nearest Neighbor Graph

Feng Qian[1], Kevin Chiew[2], Qinming He[1], Hao Huang[1], and Lianhang Ma[1]

[1] College of Computer Science and Technology, Zhejiang University, P.R. China
[2] School of Engineering, Tan Tao University, Vietnam

Abstract. The spatial co-location pattern mining discovers the subsets of features of which the events are frequently located together in a geographic space. The current research on this topic adopts a distance threshold that has limitations in spatial data sets with various magnitudes of neighborhood distances, especially for mining of regional co-location patterns. In this paper, we propose a hierarchical co-location mining framework by considering both varieties of neighborhood distances and spatial heterogeneity. By adopting k-nearest neighbor graph (kNNG) instead of distance threshold, we propose "distance variation coefficient" as a new measure to drive the mining process and determine an individual neighborhood relationship graph for each region. The experimental results on a real world data set verify the effectiveness of our framework.

Keywords: co-location pattern, kNNG, variation coefficient.

1 Introduction

The spatial co-location pattern mining [1] discovers the subsets of features (co-locations) of which the events are frequently located together in a geographic space. It has been applied to many areas like mobile commerce, earth science, biology, public health, and transportation [2]. Figure 1(a) shows a sample data set containing instances of six spatial features represented by distinct shapes. The instances of features describe the presence of their instances at different locations in a 2D or 3D space. A careful review reveals four co-location patterns as illustrated in Figure 1(c). To discover these patterns, the current research (see e.g., [2,3]) adopts an approach with two phases, namely (1) converting the spatial data set into a neighborhood relationship graph (NRG for short) using a distance threshold as illustrated in Figure 1(b) in which the distance threshold is defined as the maximal distance allowed for two events to be neighbors; and (2) finding prevalent co-locations based on their clique instances in the derived graph.

The first phase implicitly assumes the normal distances of neighbors being smaller than the predefined distance threshold. This requires an approximately uniform distribution of spatial events across the space, as well as the joint distributions of features. In real life however, the data density often varies across different areas, leading to more complex joint distributions. For such data sets with various densities, an improper distance threshold may severely affect the mining results due to two reasons. (1) First, a small distance threshold may ignore many clique instances of prevalent co-locations in sparse areas, resulting in these co-locations being under-estimated. (2) Second, a large

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 174–186, 2013.

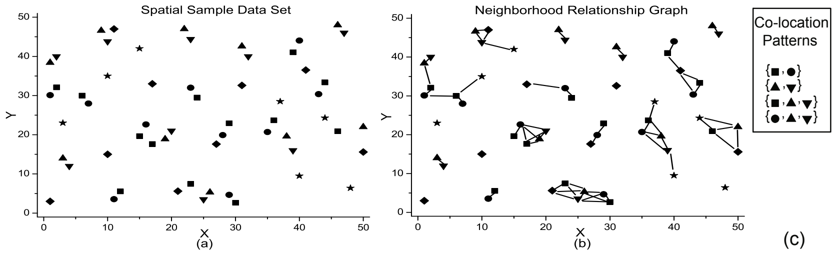


Fig. 1. An illustrative example of spatial co-location pattern discovery

distance threshold may introduce irrelevant clique instances of candidate co-locations in density areas, resulting in these co-locations being over-estimated.

Given the above concerns, we propose to find the regional co-location patterns as an extension to the conventional two-phase approaches. Our motivation comes from (1) the inconsistency of neighborhood distances in a data space; that is, the distance between any two neighboring events varies across the space since the data densities vary from region to region in the space. This inconsistency motivates us to investigate a new measure "distance variation coefficient" (see Section 3) rather than distance threshold to drive the mining process; and (2) the existence of spatial heterogeneity which demonstrates that most geographic processes vary by locations [4], and indicates that the inconsistent co-location sets may be found from different regions [5]. For instance, back to our example of mobile commerce, the requesting patterns in business regions are usually different from those in tourist regions.

As a mining strategy, we propose to hierarchically merge spatial events into local regions in the form of NRGs followed by passing them to the second phase of the conventional approaches. To enable the regional co-location pattern discovery meaningful, we partially inherit from the conventional approaches the assumption that a reasonable region is supposed to have relatively consistent neighborhood distances. This means the distances among the data points and their neighbors inside the region vary within a small range. Besides, we assume that different regions have inconsistent co-locating information in terms of spatial heterogeneity. Given these, this paper addresses two problems: (1) identifying regions with consistent neighborhood distances and co-locating information; and (2) specifying an individual NRG for each identified region.

We adopt k-nearest neighbor graphs (kNNG) that capture more natural neighborhoods [6] to describe the consistency of neighborhood distances within a region. Similar to distance thresholds, predefining k value for each region may lead to under-estimation or over-estimation of co-locations. Instead, we introduce a new measure "distance variation co-efficient" to control the range of distance varying and automatically determine k value for each region. With kNNG, the regions' NRGs are naturally prepared.

We then are able to define the similarity of co-location information between adjacent regions by passing these graphs to the second phase of the conventional approaches. The definition of similarity is a measure about whether the corresponding regions share consistent co-location information and are qualified to be merged. Analogous to kNNG, the mutual k-nearest neighbors (MkNN) naturally capture the inter-connectivity of adjacent regions [7,8]. We adopt MkNN to exclude the real region boundaries when hierarchically merging the regions.

In summary, our contributions are as follows. (1) We propose a hierarchical mining framework to discover regional co-locations by considering both varieties of neighborhood distances and spatial heterogeneity. (2) We propose a novel “distance variation coefficient to drive the mining process and determine an individual NRG for each region. (3) We evaluate our mining algorithm with experiments on a real world data set by comparing against the conventional approaches using distance thresholds.

The remaining sections are organized as follows. We first review the current research results on spatial co-location pattern mining in Section 2, followed by giving a formal description for the research problem and some basic definitions in Section 3. We then present our hierarchical mining framework to discover regional co-location patterns in Section 4, together with the experimental evaluation in Section 5 before concluding our work in Section 6.

2 Literature Review

Various algorithms have been proposed for spatial co-location pattern mining. They can be classified into four types as reviewed in the following.

One main type of these techniques are the aforementioned two-phase approaches, by which the NRG is determined by a predefined distance threshold. This general framework was first proposed by Shekhar *et al.* [9]. Within the framework, different algorithms such as joinless algorithm [2], synchronic sweep algorithm [10], and density-based algorithm [3], were proposed to improve the performance of mining process, especially the efficiency of collecting clique instances. For example, Xiao *et al.* [3] proposed to search the dense areas with high priorities so as to speed up the decision making of the algorithm. The approach was also used in spatio-temporal data sets by introducing a time factor as the time interval threshold [11].

As a distortion of the first type of approaches, the second type diversifies the objective of spatial co-location pattern mining. For example, it was extended to mine complex spatial co-location patterns (e.g., one-to-many, self-colocating, self-exclusive, and multi-feature exclusive) [12] and maximal co-location patterns [13]. Huang *et al.* [14] also adjusted the interest measure to treat the case with rare events. Yoo *et al.* [15] proposed to find the N-most prevalent co-location patterns.

The third type replaces the usage of distance threshold in the fir t phase. Huang *et al.* [16] proposed to use density ratio of different features to describe the neighborhood relationship together with a clustering algorithm. A buffer-based model [17] was also proposed to describe the neighborhood relationship for dealing with extended spatial objects such as lines and polygons. Sheng *et al.* [18] used the influence functions to model the spatial features. Among these work, a similar neighborhood related threshold or function has to be predefined by users.

These three types of techniques focus on the global co-location patterns. That is, the first and second types adopt a predefined distance threshold to determine the NRG, while the third type replaces it with a similar neighborhood related threshold. The fourth type assumes that the neighborhood distances are consistent. The data sets with various densities are not sophisticatedly treated in these work.

The fourth type of techniques discovers the regional co-location patterns. Celik *et al.* [19] straightforwardly applied the conventional approaches to a set of zones, where a zonal space has to be specified by users. Eick *et al.* [5] adopted the prototype-based clustering [20] to find regional co-location patterns. The interestingness of co-locations was scored in its fitness function. As an input of this approach, every event has a vector value of all the features, in which each item needs to be a continuous type. However, this work did not explore the monotonic property of the interesting measure proposed by Shekhar *et al.* [9] which introduces the pruning techniques to the mining process. Moreover, this approach may not be applicable to the discrete type of inputs.

3 Problem Formulation

In this section, we will present the problem statement after some basic definition related to regional co-location mining.

3.1 Basic Definitions

A spatial data set is an input of the spatial co-location pattern mining algorithm.

Definition 1 (Spatial data set). *A spatial data set has a set of non-spatial features* $F = \{f_1, f_2, \ldots, f_n\}$*, and consists of a set of spatial events* $E = \{E_1, E_2, \ldots, E_n\}$*, where* E_i *(*$1 \leqslant i \leqslant n$*) is a set of events of feature* f_i*. Every event* $e_j \in E_i$ *(*$1 \leqslant j \leqslant |E_i|$*) has a vector information of ⟨feature type* f_i*, event ID* j*, spatial location (x, y)⟩.* ∎

Given the neighborhood constraint, a spatial data set or part of it (one of its regions) can be converted into an NRG as the foundation for co-location discovery. The conventional approaches adopt the distance threshold to describe this neighborhood constraint which may lead to limitations as discussed in Section I. To get rid of those limitations, we adopt the kNNG to define the NRG in the following.

Definition 2 (Neighborhood relationship graph (NRG)). *The neighborhood relationship graph* $\mathcal{G}$ *is implemented by kNNG, in which each vertex represents a spatial event, and there is an edge connecting two vertices if they have different features and either of them is among the kNNs of the other one. The graph's vertices and edges are denoted as* $V(\mathcal{G})$ *and* $E(\mathcal{G})$ = *kNNG(*$V(\mathcal{G})$*), and each edge is assigned with a weight that is the Euclidean distance between two spatial events connected by an edge.* ∎

In our approach, each NRG corresponds to an individual region. In practice, kNNG can be calculated by firstly finding the kNN of spatial events and then filtering out the edges whose end points share the same feature type. In the context of NRG represented by kNNG, the neighborhood distance is the weight of an edge in the graph.

Given the above neighborhood constraint, the interestingness of prevalent co-locations within a region is defined by an interesting measure known as participation index which is in turn defined from participation ratio. We borrow the definitions [1,2] as follows.

Definition 3 (Participation ratio). *Given the co-location* C *(*$C \subset F$*), its participation ratio of* f_i *(*$f_i \in C$*) is defined as* $Pr(C, f_i) = \frac{\pi_{f_i}(table\_instance(C))}{table\_instance(f_i)}$*, where* π *is the relational projection operation with duplication elimination and* $table\_instance$ *is the collection of clique instances of co-locations or features, in each instance of co-locations the spatial events are neighbors to each other.* ∎

Definition 4 (Participation index). *The participation index of co-location C is $Pi(C) = \min_{f_i \in C}\{Pr(C, f_i)\}$, which measures the prevalence of C.* ∎

With a predefine prevalence threshold θ, the second phase of the conventional algorithms such as join [1] or joinless algorithm [2] can wisely discover for each region a set of co-locations of which the value of participation index is not less than threshold θ (i.e., $Pi(C) \geqslant \theta$). Moreover, due to the monotonic property of participation index, i.e., $Pi(C') \geqslant Pi(C) \,\forall\, C' \subset C$, pruning techniques such as *apriori* [21] may be introduced into the mining process.

As foregoing, we assume that a reasonable region has relatively consistent neighborhood distances, meaning that the edge weights in an NRG have small variation. In a usual application domain of co-location discovery, geographers and biologists care about the spatial patterns under specific spatial frameworks following clumped, random or uniform distribution [22]. In clustering applications, classic algorithms (e.g., k-means algorithm [20]) often assume that the clumped clusters are compact, implying that they have consistent neighborhood distances inside. As for the data sets with random or uniform distribution, the local regions are also reasonable to hold the consistency. Based on the above applications, we define the distance variation coefficient to investigate the neighborhood distances of regions as follows.

Definition 5 (Distance variation coefficient). *The distance variation coefficient of the NRG $\mathcal{G}$ is defined as $\Omega(\mathcal{G}) = \frac{\sigma(E(\mathcal{G}))}{\mu(E(\mathcal{G}))}$, where $\mu(\cdot)$ and $\sigma(\cdot)$ are statistical operations for calculating the mean value and standard deviation of the weights of all edges in $\mathcal{G}$.* ∎

Our mining framework allows us to hierarchically merge regions followed by finding their prevalent co-location sets. During the merging process, the range of distance varying is controlled by a corresponding distance variation threshold ϵ. The algorithm does not stop until the distance variation of every newly merged region is greater than ϵ, i.e., $\Omega(\mathcal{G}) > \epsilon$.

As another assumption w.r.t. spatial heterogeneity, we define the similarity of regions in the following. By gradually combining the most similar regions under the distance variation constraint, our algorithm guarantees the maximum consistency of co-locating information inside the regions.

Definition 6 (Similarity of NRGs). *Given the prevalence threshold θ and two NRGs $\mathcal{G}_1$ and $\mathcal{G}_2$, their similarity is defined as $\mathcal{R}(\mathcal{G}_1, \mathcal{G}_2) = \frac{1}{\mathcal{L}-1}\sum_{i=2}^{\mathcal{L}} J(\mathcal{C}_{1i}, \mathcal{C}_{2i})$, where $\mathcal{C}_{1i}$ is the set of size i co-locations of $\mathcal{G}_1$ each of which is prevalent (i.e., $Pi(\mathcal{C}_{1i}) \geqslant \theta$), $\mathcal{C}_{2i}$ the set of size i co-locations of $\mathcal{G}_2$ each of which is prevalent (i.e., $Pi(\mathcal{C}_{2i}) \geqslant \theta$), $J(\cdot)$ the Jaccard index defined as $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$, and $\mathcal{L}$ the maximum size of the prevalent co-locations of either $\mathcal{G}_1$ or $\mathcal{G}_2$.* ∎

The size of a co-location C is the number of distinct features it contains, namely $|\{f_i | f_i \in C\}|$ [1]. We indicate $Pi(\mathcal{C}_{1i}) \geqslant \theta$ if $Pi(C) \geqslant \theta \,\forall\, C \subset \mathcal{C}_{1i}$. The Jaccard index is a statistic commonly used for comparing the similarity of data sets. We adopt it to investigate the intersection rate of prevalent co-locations between regions. The similarity function helps us decide the priority of candidates to be merged. Finally, we give the following definition to identify the regions with rare events as anomalies.

Definition 7 (Significant NRG). *Given a ratio threshold α, an NRG $\mathcal{G}$ and its corresponding region are significant if $|V(\mathcal{G})| \geqslant \alpha|E|$, where $|E|$ is the total number of spatial events.* ∎

3.2 Problem Statement

With the above defi itions, we give a formal description of regional co-location pattern discovery in the following.

Given: (1) A spatial data set including a set of features F and a set of their spatial events E; (2) a prevalence threshold θ; (3) a distance variance threshold ϵ; and (4) a ratio threshold α.

Find: Regional co-location patterns, i.e., (1) a set of regions $\Gamma = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_m\}$ in which each element is represented as a kNNG, where $V(\mathcal{G}_i) \subset E$, $E(\mathcal{G}_i) =$ kNNG($V(\mathcal{G}_i)$) ($1 \leqslant i \leqslant m$); and (2) a set of prevalent co-locations C for each region $\mathcal{G}_i$ where $Pi(C) \geqslant \theta$.

Constraints: Consistent neighborhood distances and consistent co-locating information within regions, i.e., (1) $\Omega(\mathcal{G}_i) \leqslant \epsilon$ and $|V(\mathcal{G}_i)| \geqslant \alpha|E|$ ($1 \leqslant i \leqslant m$); and (2) minimizing the similarity between adjacent regions $\mathcal{R}(\mathcal{G}_i, \mathcal{G}_j)$ ($1 \leqslant i, j \leqslant m$).

4 Regional Co-location Mining Algorithm

In the next, we present our algorithm for mining regional co-location patterns. The algorithm is carried out by three steps as follows. (1) In Step 1, a set of initial regions are formed by assigning each of them an MkNN edge with $k = 1$ or a single event that does not participated in those mutual edges. (2) In Step 2, the algorithm iteratively merges the similar regions under the constraint of distance variation. Step 2 is achieved by two sub-steps as follows. (2.1) In Step 2.1, in each iteration, k is increased by one and every newly generated MkNN edge links two of the current regions which compose a merging candidate; and (2.2) in Step 2.2, we decreasingly sort the merging candidates by their similarity values and sequentially merge them if the constraint of distance variation is satisf ed. (3) In Step 3, the signif cant regions represented as kNNGs are returned together with their prevalent co-locations which are discovered by the second phase of join algorithm [1] in each region.

We illustrate the process of each iteration in Figure 2, in which Figure 2(a) shows four significan regions as indicated by circles each of which has a prevalent co-location in the ($k-1$)th iteration, and three MkNN edges found indicated by solid lines in the kth iteration; and Figure 2(b) shows a new region 5 which is the merge result of regions 1 and 3. Algorithm 1 presents the pseudo code of the mining process.

Step 1. Algorithm 1 fi st initializes the value of k as one, and sets the graph set Γ and the candidate set S empty (lines 1–2). It then f nds a set of MkNN edges with $k = 1$, which are disjoint to each other since any event has at most one mutual neighbor. For these mutual edges, we also require that the end points of them have distinct feature types (line 3). Naturally, a part of initial regions are formed by assigning a found edge

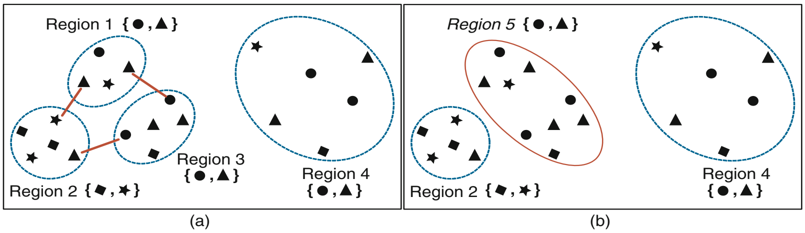


Fig. 2. An example of the mining process in each iteration

```
Inputs: A data set including $E$ and $F$, parameters $\theta$, $\epsilon$ and $\alpha$.
Outputs: A set of $k$NNGs $\mathit{\Gamma}$, a set of corresponding co-locations $C$.
Method:
1: $k = 1$;                                                          //$k$ of $k$NN
2: $\mathit{\Gamma} = \varnothing$; $S = \varnothing$;                //$S$ is a set of merging candidates
3: $N_k$ = M$k$NN($E, F, k$);                        //$N_k$ is a set of M$k$NN edges of $E$
4: Initialize each edge $\ell \in N_k$ as an individual graph $\mathcal{G}$ in $\mathit{\Gamma}$;
5: Initialize each $e \in E \backslash V(N_k)$ as an individual graph $\mathcal{G}$ in $\mathit{\Gamma}$;
6: while $|\{\mathcal{G}|\ \mathcal{G} \in \mathit{\Gamma}$ and $V(\mathcal{G}) \geqslant \alpha|E|\}|$ is changed
7:    $k = k + 1$;                                              // increase $k$ by one
8:    $N_k$ = M$k$NN($E, F, k$);                         //update the mutual $k$-nn edges
9:    for each $\ell_{e_1,e_2} \in N_k \backslash N_{k-1}$        //$e_1$ and $e_2$ are end points of edge $\ell$
10:      if $e_1 \in V(\mathcal{G}_1)$ and $e_2 \in V(\mathcal{G}_2)$                    //link two regions
11:      then $S = S \cup \mathcal{R}(\mathcal{G}_1, \mathcal{G}_2)$;     //append to candidate set without duplication
12:   end for
13:   sort $S$ decreasingly by similarity value $R$;
14:   for each $\mathcal{R}(\mathcal{G}_1, \mathcal{G}_2) \in \mathcal{S}$            //traverse candidates by sorted order
15:      $V(\mathcal{G}) = V(\mathcal{G}_1) \cup V(\mathcal{G}_2)$; $E(\mathcal{G}) = k$NNG($V(\mathcal{G})$);               //test merging
16:      if $\mathit{\Omega}(\mathcal{G}) \leqslant \epsilon$ or $(V(\mathcal{G}_1) < \alpha)$ and $(V(\mathcal{G}_2) < \alpha)$          //variation not large
17:      then $\mathit{\Gamma} = ((\mathit{\Gamma} \backslash \mathcal{G}_1) \backslash \mathcal{G}_2) \cup \mathcal{G}$        //replace $\mathcal{G}_1, \mathcal{G}_2$ with their merged region
18:   end for
19: end while
20: Calculate co-locations $C$ for each $\mathcal{G} \in \mathit{\Gamma}$ with prevalence threshold $\theta$;
21: return $\mathit{\Gamma}$ & $C$;                       //output significant regions and their co-locations
```

Algorithm 1. Discovery of regional co-location patterns

to each of them (line 4); whereas the rest part is formed by assigning each region with a single event that does not participate in any of those edges (line 5).

Step 2.1. It then starts to merge the current regions iteratively. The process does not terminate until the significant region can be merged, which means that the number of significant regions does not change (line 6). In each iteration, the algorithm increases k by one and updates the set of MkNN edges (lines 7–8). For each newly generated edge, a merging candidate is formed if it connects two of the current regions and the similarity is calculated (lines 9–12).

Step 2.2. Given a set of merging candidates, the algorithm decreasingly sorts them by their similarity values (line 13), and sequentially investigates each candidate whether

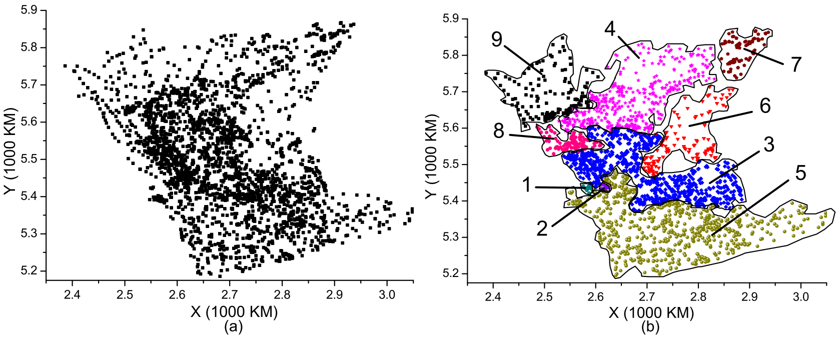


Fig. 3. An example of regional co-location pattern mining

its two regions have not yet been merged (lines 14–15). Each candidate is investigated by a test merge and an evaluation of the merged region with the constraint of distance variation. If the constraint is satisfied or both of two regions of the candidates are still insignificant, the algorithm replaces them with the newly merged region (lines 16–19).

Step 3. Based on a set of final regions, the algorithm calculates a set of co-location patterns for each of them if it is significant with small variation, and uses the second phase of join algorithm to find prevalent co-locations (line 20). Otherwise, the region is discarded since it has rare events which can be regarded as an anomaly. Finally, a set of significant regions and their prevalent co-location sets are returned to users (line 21).

5 Experimental Evaluation

Based on a real world data set, we evaluate the mining results of our approach against the conventional ones, and study the trends of several statistics in our mining process.

5.1 Description of Real Data Set

Figure 3(a) shows a real world data set (details shown in Table 1) we use for experiments. It is available at the Digital Chart of the World (DCW) Data Server [23] for research on spatial co-location discovery (see e.g.,[10,18]). Its spatial events have location information of latitudes and longitudes. Moreover, they are classified by distinct types of landmarks, such as drainage, land cover, and populated place. It is obvious that the data set includes various densities. In the data set, the geographic coordinates are transferred to projection coordinates using Universal Transverse Mercartor projection.

5.2 Mining Results

In this set of experiments, we run both our regional mining algorithm and the join algorithm on the DCW data set with parameters $\epsilon = 0.6$, $\theta = 0.6$ and $\alpha = 0.005$. Figure 3(b) illustrates our mining results (details shown in Table 2) for the data set shown in Figure 3(a), in which we identify nine regions and some regional co-locations.

Table 1. The data set of US Minnesota state in DCW

No.	Landmark Type	abbr.	# of events	No.	Landmark Type	abbr.	# of events
1	Aeronautical Point	Ap	86	5	Hypsography	Hy	72
2	Cultural Landmark	Cl	103	6	Hypsography Supplemental	Hs	687
3	Drainage	Dr	6	7	Land Cover	Lc	28
4	Drainage Supplemental	Ds	1338	8	Populated Place	Pp	517

Table 2. Information of found regions in DCW data set

Regions	$\boldsymbol{R}_1$	$\boldsymbol{R}_2$	$\boldsymbol{R}_3$	$\boldsymbol{R}_4$	$\boldsymbol{R}_5$	$\boldsymbol{R}_6$	$\boldsymbol{R}_7$	$\boldsymbol{R}_8$	$\boldsymbol{R}_9$
Number of events	15	14	866	430	727	128	94	96	157
Average neighborhood distance (km)	10.54	8.12	16.74	22.19	24.91	3.63	3.49	2.78	4.22
Distance variation	0.51	0.40	0.58	0.60	0.57	0.60	0.55	0.59	0.60
Number of co-locations (sizes 2, 3, and 4)	3, 1, 0	1, 0, 0	3, 1, 0	10, 10, 3	7, 3, 0	15, 13, 4	8, 3, 0	10, 5, 1	9, 2, 0

Table 3. Value of participation index of co-locations in DCW data set

Algorithms	$Pi(\{\text{Ds,Hs,Pp}\})$	$Pi(\{\text{Ap,Ds,Pp}\})$	$Pi(\{\text{Hs,Cl,Pp}\})$	$Pi(\{\text{Ap,Cl,Pp}\})$
join(30km)	0.65	Null	Null	Null
join(35km)	0.72	0.70	Null	Null
join(40km)	0.77	0.78	0.64	Null

RCMA	0.98(R_3),0.96(R_4)
	0.69(R_5),0.90(R_6)
		0.66(R_5),0.71(R_6)
		0.86(R_8),0.75(R_9)
			0.71(R_4)
			0.90(R_5)
				0.81(R_6)

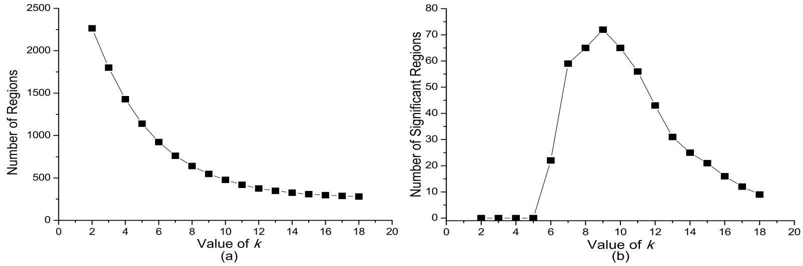


Fig. 4. Impact of k values to (a) the number of regions, and (b) the number of significant regions

In Table 3, we select four co-locations to demonstrate the difference between these two algorithms. The join algorithm discovers only the co-location {Ds, Hs, Pp} with a small distance threshold (30km). When the value of distance threshold increases, the other two co-locations ({Ap, Ds, Pp} and {Hs, Cl, Pp}) are detected. However, these regional co-locations are over-estimated to be globally prevalent. By contrast, our algorithm can detect them in their corresponding regions. We also find the co-location

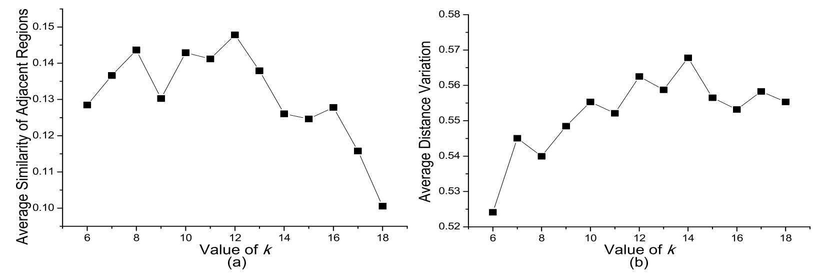


Fig. 5. Impact of k values to (a) the average similarity of adjacent regions, and (b) the average distance variation

{Ap, Cl, Pp} in a single region which is under-estimated by the conventional approaches. By comparing with the previous research results [18], our mining results include their discovered co-locations, and can assign them to the corresponding regions.

5.3 Evaluation of Regional Mining Process

In what follows, we study the trends of four statistics in our mining process on the DCW data set with the information of the number of regions, the number of significant regions, the average similarity of adjacent regions and the average distance variation.

Number of Regions. As can be seen from Figure 4(a), the number of regions decreases with the increase of k value. This is because the small regions are merged into larger ones. The decreasing is fast when k is small, while most of the regions are insignificant and the merging condition is easy to satisfy. The fast rate of decreasing also indicates that a relatively small k value can sufficiently describe the neighborhood relationship of spatial events, as well as the regional structure of space.

Number of Significant Regions. At the initial stage of mining process, there are hardly any significant regions as illustrated in Figure 4(b). Then the number of significant regions increases in a sudden when $k = 6$. This is because many insignificant regions are closing to the ratio line and become significant. The remaining set of insignificant regions naturally become anomalies which are hard to merge due to different neighborhood distance from their adjacent regions. After that, the number of significant regions decreases when the similar regions are sequentially merged.

Average Similarity of Adjacent Regions. To determine whether two regions are adjacent, we calculate each region a minimum bounding rectangle (MBR) and extend its width and height by 10%. Two regions are regarded as adjacent if they have an overlap between their MBRs. According to Definition 6, we calculate the average similarity of adjacent regions as shown in Figure 5(a). At the beginning of the process the similarity fluctuate , and then decreases rapidly. To explain the fluctuation, we tentatively test the relationship of k and the distance variation in Figure 6. We continually generate a set of kNNGs of the DCW data set by increasing k value from one to larger values, then calculate the value of distance variation for each generated graph. As can be concluded, the distance variation of the graph is large with small k. It decreases as k value

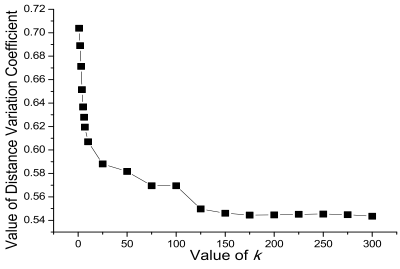


Fig. 6. Impact of k values to the distance variation of kNNG

increases. This is because small value of k indicates a large gap of the neighborhood distance between the events in dense and sparse areas. When k becomes larger, the edges with larger weights are involved and reduce the gap. Thus, at the beginning of the merging process, the regions which have different neighborhood distance are not merged even if they have consistent co-locating information. With larger k values, the distance variation of regions becomes smaller, while the consistency of co-locating information generally becomes a primary reference for merging. This explains the early fluctuatio because the impact of distance variation and co-locating information match each other, and the later on drop of the similarity because the co-locating information competes as the primary reference.

Average Distance Variation. Figure 5(b) shows the implication between the average distance variation of regions and k values. With slight fluctuation, the average distance variation generally becomes large. The trend of curve verifies our explanation that the impact of distance variation and co-locating information match each other at the initial stage, and then the latter becomes the primary reference for merging.

6 Conclusion

We have discussed the limitations of conventional approaches to mining spatial co-locations using distance thresholds, especially for data sets with various magnitudes of neighborhood distances. To get rid of those limitations, we have proposed a hierarchical mining framework to discover regional co-locations accounting for both varieties of neighborhood distances and spatial heterogeneity. By adopting kNNG instead of distance thresholds, we have proposed a novel "distance variation coefficient" to drive the mining process and determine an individual NRG for each region. With rigorous experiments on a real world data set, we have demonstrated that our framework has been effective for the discovery of regional co-location patterns.

Acknowledgement. This work is partly supported by National Key Technologies R&D Program of China under Grant No. 2011BAD21B02 and MOE-Intel IT Research Fund of China under Grant No. MOE-INTEL-11-06, in which Chiew's work is partly supported by National Natural Science Foundation of China under Grant No. 61272303.

References

1. Huang, Y., Shekhar, S., Xiong, H.: Discovering colocation patterns from spatial datasets: A general approach. IEEE Transactions on Knowledge and Data Engineering 16(12), 1472–1485 (2004)
2. Yoo, J.S., Shekhar, S.: A joinless approach for mining spatial colocation patterns. IEEE Transactions on Knowledge and Data Engineering 18(10), 1323–1337 (2006)
3. Xiao, X., Xie, X., Luo, Q., Ma, W.Y.: Density based co-location pattern discovery. In: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Irvine, USA, November 5-7, pp. 1–10 (2008)
4. Miller, H.J., Han, J.: Geographic Data Mining and Knowledge Discovery. CRC Press, New York (2009)
5. Eick, C.F., Parmar, R., Ding, W., Stepinski, T.F., Nicot, J.P.: Finding regional co-location patterns for sets of continuous variables in spatial datasets. In: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Irvine, USA, November 5-7, pp. 30:1–30:10 (2008)
6. Karypis, G., Han, E.H., Kumar, V.: CHAMELEON: Hierarchical clustering using dynamic modeling. IEEE Computer 32(8), 68–75 (1999)
7. Brito, M.R., Chavez, E.L., Quiroz, A.J., Yukich, J.E.: Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. Statistics & Probability Letters 35(1), 33–42 (1997)
8. Ding, C., He, X.: K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization. In: Proceedings of the ACM Symposium on Applied Computing, Nicosia, Cyprus, March 14-17, pp. 584–589 (2004)
9. Shekhar, S., Huang, Y.: Discovering spatial co-location patterns: A summary of results. In: Jensen, C.S., Schneider, M., Seeger, B., Tsotras, V.J. (eds.) SSTD 2001. LNCS, vol. 2121, pp. 236–256. Springer, Heidelberg (2001)
10. Zhang, X., Mamoulis, N., Cheung, D.W., Shou, Y.: Fast mining of spatial collocations. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, USA, August 22-25, pp. 384–393 (2004)
11. Yoo, J.S., Shekhar, S., Kim, S., Celik, M.: Discovery of co-evolving spatial event sets. In: Proceedings of the 6th SIAM International Conference on Data Mining, Bethesda, USA, November 20-22, pp. 306–315 (2006)
12. Munro, R., Chawla, S., Sun, P.: Complex spatial relationships. In: Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, USA, December 19-22, pp. 227–234 (2003)
13. Wang, L., Zhou, L., Lu, J., Yip, J.: An order-clique-based approach for mining maximal co-locations. Information Sciences 179(19), 3370–3382 (2009)
14. Huang, Y., Pei, J., Xiong, H.: Mining co-Location patterns with rare events from spatial data sets. GeoInformatica 10(3), 239–260 (2006)
15. Yoo, J.S., Bow, M.: Mining spatial colocation patterns: a different framework. Data Mining and Knowledge Discovery 24(1), 159–194 (2012)
16. Huang, Y., Zhang, P., Zhang, C.: On the relationships between clustering and spatial co-location pattern mining. International Journal on Artificial Intelligence Tools 17(1), 55–70 (2008)
17. Xiong, H., Shekhar, S., Huang, Y., Kumar, V., Ma, X., Yoo, J.S.: A framework for discovering co-location patterns in data sets with extended spatial objects. In: Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena, USA, April 22-24, vol. 89, pp. 78–89 (2004)

18. Sheng, C., Hsu, W., Li Lee, M., Tung, A.K.H.: Discovering spatial interaction patterns. In: Haritsa, J.R., Kotagiri, R., Pudi, V. (eds.) DASFAA 2008. LNCS, vol. 4947, pp. 95–109. Springer, Heidelberg (2008)
19. Celik, M., Kang, J.M., Shekhar, S.: Zonal co-location pattern discovery with dynamic parameters. In: Proceedings of IEEE International Conference on Data Mining, Omaha, USA, October 28-31, pp. 433–438 (2007)
20. Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G., Ng, A., Liu, B., Yu, P., Zhou, Z., Steinbach, M., Hand, D., Steinberg, D.: Top 10 algorithms in data mining. Knowledge and Information Systems 14(1), 1–37 (2008)
21. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Data Bases, Santiago de Chile, Chile, September 12-15, pp. 487–499 (1994)
22. Wang, J.: Spatial Analysis. Science Press, Beijing (2006)
23. Digital Chart of the World (2010), http://www.maproom.psu.edu/dcw/

Spectral Decomposition for Optimal Graph Index Prediction

Liyan Song[1], Yun Peng[1], Byron Choi[1], Jianliang Xu[1], and Bingsheng He[2]

[1] Department of Computer Science, Hong Kong Baptist University, Hong Kong
{lysong,ypeng,bchoi,xujl}@comp.hkbu.edu.hk
[2] School of Computer Engineering, Nanyang Technological University, Singapore
bshe@ntu.edu.sg

Abstract. There is an ample body of recent research on indexing for structural graph queries. However, as verified by our experiments with a large number of random and scale-free graphs, there may be a great variation in the performances of indexes of graph queries. Unfortunately, the structures of graph indexes are often complex and *ad-hoc*, so deriving an accurate performance model is a daunting task. As a result, database practitioners may encounter difficulties in choosing the optimal index for their data graphs. In this paper, we address this problem by proposing a spectral decomposition method for predicting the relative performances of graph indexes. Specifically, given a graph, we compute its spectrum. We then propose a similarity function to compare the spectrums of graphs. We adopt a classification algorithm to build a model and a voting algorithm for predicting the optimal index. Our empirical studies on a large number of random and scale-free graphs, using four structurally distinguishable indexes, demonstrate that our spectral decomposition method is robust and almost always exhibits an accuracy of 70% or above.

1 Introduction

Due to the flexibility of the graph model, it has a wide range of recent applications, such as biological databases, social networks and XML. To optimize query processing on graph data, many indexing techniques have recently been proposed. Unfortunately, graph data are often heterogeneous and the structures of their indexes are often complex and *ad-hoc*. As our experiments reveal, the performances of such graph indexes may vary greatly. This leads to a natural question for database practitioners: *Which index is the most efficient for a given graph?*

When compared to their relational counterparts, the structures of many graph indexes are far more complex. This causes a few unique problems. Firstly, it is sometimes time-consuming to construct the indexes. For example, our experiments on a commodity computer show that given a random graph of a modest size (∼3,000 vertices and a density of 0.02), the construction time for a graph index, namely `2-hop labeling` [12], is already 8.3 seconds. (For background details of the indexes discussed, please refer to our technical report [13].) While some other graph indexes can be constructed in less than a second, the most time-efficient index can only be identified after *all* indexes have been constructed and benchmarked. Furthermore, performance depends not only on the

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 187–200, 2013.

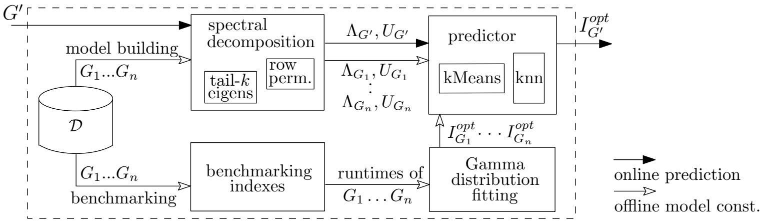


Fig. 1. Overview of our spectral decomposition prediction framework

algorithms but also the details and quality of the implementation, as there is not yet a well-received (*i.e.*, commercial) implementation in place. Secondly, graph indexes are sometimes several times larger than the graph itself. Using the example given above, the index generated by `2-hop labeling` is 19 times larger than the graph whereas that by `Interval labeling` [1] is almost as large as the graph itself. Hence, it is not space-efficie t to use and maintain multiple indexes simultaneously on the entire graph. Overall, it is clearly desirable to be able to predict the optimal index and construct it, and only it, for efficient query processing.

As a proof of concept, we focus on *reachability queries* — "given two vertices, is one reachable from the other?" — which is a fundamental query of graphs. However, our proposed technique does not depend on any specific type of graph query.

In this paper, we apply data mining techniques to predict the relative performances of different graph indexes. One of the core problems is to extract important features (or characteristics) from data graphs. While a large variety of features have been studied in graph theories [9], it is not yet clear which are the most relevant to index performances. Therefore, we propose to apply a general technique derived from spectral graph theories [5], namely spectral decomposition, to solve this problem. Spectral methods have been reported successful in many applications such as VLSI design and computer vision. To the best of our knowledge, this is the first work to empirically demonstrate the relationships between graph spectrums and index performances. In general, graph spectrums are known to be *characteristics of graphs* and to be related to many important graph properties. Another advantage is that they are supported by industrial-strength softwares, not to mention the availability of their advanced optimizations.

The second core problem is to represent the performance of a graph index. Our preliminary experiments show that the runtimes of 1,000 random queries on an index, even on the same graph, can often exhibit large variances. For example, we ran 1,000 random queries on each of 8,000 random graphs and indexed each using `2-hop` and `Prime labeling` [11]. The mean and standard deviation of `2-hop` are 14.1 seconds and 4.2 and those of `Prime labeling` are 11.6 seconds and 59.7, respectively. Moreover, the runtimes are often skewed and have a long tail at large values. (In a later section, we illustrate some of the runtime distributions in Figure 2.) We observe a similar phenomenon in our collection of scale-free graphs. A possible explanation is that a graph may contain many different sub-structures and the indexes are also complex structures. This leads to a wide range of runtimes. While average runtimes are often used to quantify index performances, it is desirable to propose a more flexible metric.

In this paper, we fit the runtimes of queries into a distribution. From our experiments on estimating its parameters, the goodness of fit of the Gamma distribution is always the best. By comparing the distributions of runtimes, we can obtain a more robust and flexible way to compare performances. While we may apply the research on the Gamma distribution for further analysis, in this paper, we apply the inverse cumulative distribution function to estimate the time when $y\%$ of queries finish. Depending on users' applications, they may specify a value for $y\%$ to express their "tolerance" of long query runtimes. For instance, some Internet connection providers (*e.g.* hotels and cafes) charge their users according to connection time and so long query runtimes can be undesirable. To cater for their needs, data practitioners may choose the index that is optimal at 98%, instead of the one that has the optimal average runtime.

Contributions. To our knowledge, this is the first investigation of graph spectrums in relation to index performances. We summarize the contributions of this work below and present an overview of these in Figure 1.

- Given a graph G_i in a database $\mathcal{D}$, we propose a spectral decomposer to determine G_i's Laplacian matrix and a set of eigenvalues Λ_{G_i} and eigenvectors U_{G_i}. These eigenvalues and eigenvectors are transformed into a unified representation for comparison purposes.
- We propose a spectral similarity function between two graphs.
- We propose to fit the runtimes of an index on a given graph into the Gamma distribution using a distribution fitter. Users may then evaluate this in terms of their desired optimal index.
- We adopt the k-Means algorithm and k-nearest neighbor with a voting method for predicting the optimal index $I_{G'}^{opt}$ of a given graph G'.
- We conduct experiments with a large number of random and scale-free graphs. The results show that our proposed technique can achieve accuracies that are almost always above 70% and very often above 80%.

The rest of the paper is organized as follows. Section 2 presents the backgrounds to graph spectral decomposition. We present our problem statement in Section 3. Section 4 presents the definition of the optimal index. A uniform spectral representation of graphs and a similarity function between graphs are proposed in Section 5. Our prediction method is detailed in Section 6 and our experimental evaluation is reported in Section 7. Section 8 discusses related work and Section 9 concludes this paper.

2 Background to Graph Spectral Decomposition

In this section, we provide a background to graph spectrums. Graph spectrums have been widely used to study many interesting properties of graphs, such as spectral partitioning and expansion [8], the cut problem [7] and graph drawing [14].

Graph spectrum is often defined using Laplacian matrix. Laplacian matrix L of a directed or undirected graph $G = (V, E)$ is defined as $L = D - A$, where D is the degree matrix of G and A is the adjacency matrix of G. More specifically, $A_{i,j} = 1$ if $(v_i, v_j) \in E$; and $A_{i,j} = 0$ otherwise, where i and i are the *ID*'s of the vertices v_i and v_j. The degree matrix is a diagonal matrix and $D_{i,i}$ is the outdegree of v_i

The Laplacian matrix L of G can be eigendecomposed as $L = U\Lambda U^{-1}$, where Λ is a diagonal matrix of eigenvalues of L, and U is a matrix of the corresponding eigenvectors. Specificall , let $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$, where $n = |V|$, denote the eigenvalues; $X_1, X_2, ..., X_n$ denote the corresponding eigenvectors, where $\Lambda_{i,i} = \lambda_i$; and the i-th column of U is X_i. We call U the *eigenvector matrix* and use Λ to refer to eigenvalues. We use a subscript in U_G and Λ_G to denote the U and Λ of graph G if needed.

Eigenvalues Λ are called *spectrums* in spectral graph theories. Since eigenvectors U are also known to be closely related to the characteristics of vertices, we include them in our algorithm. We use the Λ and U of the underlying undirected graphs of the data graphs, as they capture their structures and their properties are well-studied [5].

3 Problem Formulation

In this section, we formulate the optimal graph index prediction problem based on graph eigenvalues and eigenvectors.

We assume a graph database $\mathcal{D}$ containing a large number of directed graphs $\{G_1, G_2, ..., G_m\}$. The reachability query on the graphs is formally defined as follows.

Definition 3.1. *Given a directed graph $G = (V, E)$, u, $v \in G$, v is reachable from u, denoted as $u \rightsquigarrow v$ = true, if and only if there is a path from u to v in G.*

As discussed earlier, various types of indexes are available to support the reachability query on graphs in $\mathcal{D}$. However, it is desirable to predict the optimal one without building and benchmarking all the available indexes. This problem can be described as follows.

Problem Statement. *Given a set of indexes $\mathcal{I}$ = $\{I_1, I_2, ..., I_n\}$ and a graph database $\mathcal{D}$, we want to build a predictive model M using the eigenvalues and eigenvectors of graphs in $\mathcal{D}$, in order to efficiently determine the optimal index I_G^{opt} for a graph $G \notin \mathcal{D}$.*

There are two main sub-problems within the problem statement.

(P1) How can we represent the performance of an index on a graph?

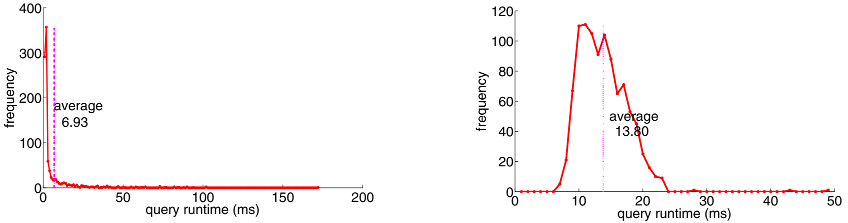
As motivated in Section 1, the query runtimes of a particular index on a particular graph may deviate from the average. Therefore, in Section 4, we investigate the more flexible notion of the *optimal index* in expressing desirable runtimes.

(P2) How do we compare spectrums of graphs?

As mentioned in Section 1, we propose to represent a graph G with eigenvalues Λ_G and eigenvectors U_G. A similarity function between the spectral graph representations is needed. In addition, the number of eigenvalues and the dimension of the eigenvectors of a graph G is the number of vertices of G, which are not uniform in a graph database. They are therefore transformed into a uniform representation for comparison purposes. Finally, while the eigenvalues Λ_G are invariants of G, the row vectors of the eigenvector matrix U_G are dependent on the permutations of the vertex IDs of G. There is hence a row permutation problem when comparing U's of graphs.

4 Performance Metric

We define the performance of an index using the query times of 1,000 random queries, as the query workloads are often not known when an index is chosen. To study performance, we plot the query time distribution, where the x-axis is the query time and the y-axis the number of queries finished at time x. For example, Figures 2(a) and (b) show the query runtime distributions of two indexes on a random data graph. We demonstrate that average runtime alone may lack the flexibility to describe the desired notion of performance. For example, Figure 2(a) shows that `Grail(1)` [18] has the smallest average time but has a long tail. In comparison, the rutntimes of `Interval` exhibit a relatively small variance, although `Interval` has a relatively large average time. Therefore, `Interval` could be a better choice in applications where long query times are unacceptable or commercially unfavorable.



(a) Distribution of runtimes of `Grail(1)` (b) Distribution of runtimes of `Interval`

Fig. 2. Some distributions of the runtimes of 1000 random queries on a random graph

Once the query runtimes are presented as a distribution, we fit this to some well-studied distributions, such as Normal, Poisson and Gamma distributions. To measure goodness of fit, we adopt the L_2-norm between the estimated and the real distributions. From our experiments on a large number of random and scale-free graphs, we observe that the Gamma distribution almost always yields the best fit. A possible reason for this is that it is often used to model the waiting time and the query runtime may be considered as the waiting time until queries finish. (The detailed experiment on the fitting is presented in Section 7.) Moreover, the parameters of the Gamma distribution can be efficiently estimated. Therefore, we use it to represent the query runtime.

While the optimal index is intuitively the most efficient on 1,000 random queries, we define it as the one with the smallest estimated runtime w. r. t. the user-defined parameter $y\%$ (*e.g.*, 98%). Once the parameters of the Gamma distribution have been estimated, the notion of the optimal index can be tuned and determined mathematically by adjusting y.

Definition 4.2. *Given a set of indexes $\mathcal{I} = \{I_1, I_2, ..., I_n\}$, a graph G, and a set of random queries Q, the* optimal index *in $\mathcal{I}$ of G is the index that finishes $y\%$ of queries of Q in the shortest time.*

An application of Definition 4.2 is that database practitioners may check the robustness of an index. For instance, given a graph database, we may use the estimated Gamma parameters to construct prediction models for a few y values, *e.g.*, 75%, 85% and 95%, *without rerunning the benchmarking queries*. An index can be considered robust if it is optimal for all those y values, instead of a specific one.

5 Spectral Similarity of Graphs

This section presents a similarity measure for the eigenvalues and eigenvectors of graphs. Specifically, we first transform these into a uniform representation and then permute the rows of the eigenvector matrix to allow a similarity comparison. Finally, we propose a spectral similarity function for graphs.

5.1 Unifying the Dimensionalities of Graphs

The first issue in using Λ and U to represent and compare graphs is that the dimensions of Λ and U of different graphs are different; that is, they cannot be directly compared. Therefore, we unify the dimensions of Λ and U as follows.

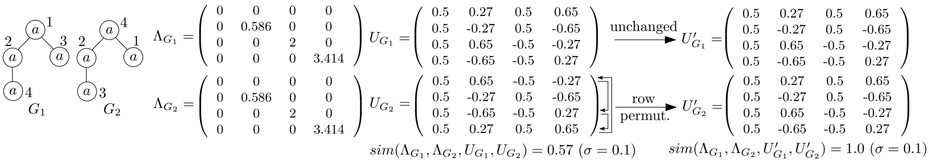
(i) We use the tail-k non-zero eigenvalues of Λ_G and the corresponding eigenvectors of U_G to represent a graph G. According to [15], the eigenvectors of the tail-k non-zero eigenvalues provide the best approximation of U_G. This unifies not only the number of eigenvalues Λ_G but also the column dimension of U_G.

(ii) Each row of U_G corresponds to a vertex in G. The row dimension of U_G of the graphs can be unified by adding rows of zeros, until the dimension of U_G matches the largest graph in the database. Using simple matrix theories, we have that adding zero rows to a matrix affect neither its eigenvalues nor the directions of its eigenvectors. In our context, a zero row vector corresponds to an isolated (virtual) vertex of a graph and does not affect the relative performance of indexes.

We further remark that the computation of the similarity between the eigenvector matrices involves determining the cosine similarity between *each pair of the eigenvectors* (to be detailed in Formula 1). The computation complexity is quadratic to the number of eigenvectors in the matrices. Due to this performance issue, we opt not to introduce eigenvectors to unify the column dimension of U_G. In contrast, the computation time of the similarity between U_G is linear to the size of the row dimension. Thus, our dimension unification does not lead to a significant increase in computation time while retaining the characteristics of the vertices.

5.2 Permutation of Vertex ID

While the eigenvalues of G are graph invariants [3], the eigenvectors (columns) in U_G are not. Since a row in U_G represents the characteristics of a vertex in G and the IDs of rows are directly related to the IDs of vertices, graphs with similar structures may have very different eigenvector matrices. This can be illustrated using the simple example shown in Figure 3. Here, graphs G_1 and G_2 are isomorphic and so are expected to have the same Λ's and U's. However, U_{G_1} and U_{G_2} are different, and a direct similarity computation (to be defined in Section 5.3) yields a low similarity score of 0.57. We reorder the rows in the eigenvector matrices of U_{G_1} and U_{G_2} to obtain U'_{G_1} and U'_{G_2},



$$\Lambda_{G_1} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.586 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3.414 \end{pmatrix} U_{G_1} = \begin{pmatrix} 0.5 & 0.27 & 0.5 & 0.65 \\ 0.5 & -0.27 & 0.5 & -0.65 \\ 0.5 & 0.65 & -0.5 & -0.27 \\ 0.5 & -0.65 & -0.5 & 0.27 \end{pmatrix} \xrightarrow{\text{unchanged}} U'_{G_1} = \begin{pmatrix} 0.5 & 0.27 & 0.5 & 0.65 \\ 0.5 & -0.27 & 0.5 & -0.65 \\ 0.5 & 0.65 & -0.5 & -0.27 \\ 0.5 & -0.65 & -0.5 & 0.27 \end{pmatrix}$$

$$\Lambda_{G_2} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.586 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3.414 \end{pmatrix} U_{G_2} = \begin{pmatrix} 0.5 & 0.65 & -0.5 & -0.27 \\ 0.5 & -0.27 & 0.5 & -0.65 \\ 0.5 & -0.65 & -0.5 & 0.27 \\ 0.5 & 0.27 & 0.5 & 0.65 \end{pmatrix} \xrightarrow[\text{permut.}]{\text{row}} U'_{G_2} = \begin{pmatrix} 0.5 & 0.27 & 0.5 & 0.65 \\ 0.5 & -0.27 & 0.5 & -0.65 \\ 0.5 & 0.65 & -0.5 & -0.27 \\ 0.5 & -0.65 & -0.5 & 0.27 \end{pmatrix}$$

$$sim(\Lambda_{G_1}, \Lambda_{G_2}, U_{G_1}, U_{G_2}) = 0.57 \ (\sigma = 0.1) \qquad sim(\Lambda_{G_1}, \Lambda_{G_2}, U'_{G_1}, U'_{G_2}) = 1.0 \ (\sigma = 0.1)$$

Fig. 3. Eigenvalues and eigenvectors of two isomorphic graphs

respectively, as shown in Figure 3. Using U'_{G_1} and U'_{G_2} for the similarity computation, we obtain a similarity score of 1.0.

Unfortunately, the number of possible permutations is $n!$, where n is the number of rows. Therefore, we propose three practical heuristic functions to reorder the rows. The aim is to be able to compare vertices with similar spectral characteristics.

(i) VP_rad: for each row in U_G, we compute its L_2-norm. We order the rows by their L_2-norms in descending order. Intuitively, the L_2-norm of a row vector denotes its distance (radius) from the original point in a vector space. Therefore, the row vectors that are far (or, respectively, close) to the original point are compared.

(ii) VP_coor: According to [14], each row of U_G can be considered as the coordinates of a vertex of G in a vector space. The rows can then be ordered in descending lexicographic order. It is worth-remarking here that such an ordering of rows is biased towards the first few dimensions, which correspond to smaller eigenvalues and hence are considered more important than the latter dimensions.

(iii) VP_W_rad: Since the eigenvalues indicate the importance of a dimension, we may integrate them with the heuristic function. Specificall , for each row vector, we compute its weighted L_2-norm, where the weight of the i-th entry of a row is $1/\lambda_i$, and then order the rows by their weighted L_2-norms in descending order.

In summary, the processing presented above can be applied to both the rows and columns of all graphs to obtain a unified representation for similarity computation. In subsequent discussions, we simply use Λ and U to refer to the matrices whose dimensions have been unified and ordered in this manner.

5.3 Spectral Similarity between Graphs

The central part of the prediction framework is the spectral similarity between graphs. This has two major components.

Firstly, we determine the most comparable eigenvectors between two graphs G_1 and G_2. Specificall , for each eigenvector X_i in U_{G_1}, we pair X_i with an eigenvector Y_j of U_{G_2} whose direction is the most similar to X_i, defined as follows:

$$p[i] = argmax_{j=1,\ldots,n}(cos\_sim(X_i, Y_j)), \tag{1}$$

where X_i in U_{G_1}, Y_j in U_{G_2} and cos_sim denotes the cosine similarity between vectors.

Secondly, the *spectral similarity between two graphs* G_1 and G_2 is define as the weighted sum of the cosine similarity between paired eigenvectors, where the weights are proportional to the difference between the corresponding eigenvalues:

$$sim(G_1, G_2) = \frac{\sum_{\substack{(\lambda_i, X_i) \in G_1, \\ (\lambda_{p[i]}, Y_{p[i]}) \in G_2}} (cos\_sim(X_i, Y_{p[i]}) \times e^{-\frac{(\lambda_i - \lambda_{p[i]})^2}{2\sigma^2}}))}{\sum_{\lambda_i \in G_1, \lambda_{p[i]} \in G_2} e^{-\frac{(\lambda_i - \lambda_{p[i]})^2}{2\sigma^2}}}, \qquad (2)$$

where σ is a parameter that controls the importance between eigenvalues and eigenvectors. In particular, the larger the value of σ, the greater the influence of the eigenvalues on the sim function. To compare two spectrums, (i) we compute the weight determined by a Gaussian function on the difference between the eigenvalues, $e^{-\frac{(\lambda_i - \lambda_{p[i]})^2}{2\sigma^2}}$. We assume such a difference of eigenvalues follows a normal distribution; (ii) the cosine similarity between the corresponding eigenvectors is multiplied by the weight; and (iii) the denominator normalizes the similarity function.

6 Prediction Algorithm

With the uniform representation of graphs in $\mathcal{D}$ and the spectral similarity function, we are ready to present our prediction algorithm. In this paper, the *label* of a graph G is its optimal index among a given set of indexes $\mathcal{I}$ = $\{I_1, I_2, ..., I_n\}$. We use the eigenvalues and eigenvectors of graphs in $\mathcal{D}$ and their labels to train a prediction model. To summarize the graphs in $\mathcal{D}$, we simply adopt classical k-Means clustering, while other clustering techniques may be applied. As we shall see from experiment, prediction with eigenvalues Λ alone is not as accurate as that achieved with both Λ and U. However, some other classical methods, such as neural networks or decision trees, require to cast U into some numerical values for training, while preserving their semantics. This does not appear trivial and so these are not adopted.

A trained model is then used to predict the label of a graph G', where $G' \notin \mathcal{D}$. Specifically, given a graph G', we determine its k-nearest cluster centers and apply a voting method to obtain the weighted majority of their labels. Such a label is returned as the optimal index of G'.

6.1 Clustering Algorithm

In this subsection, we highlight the adoption of k-Means clustering for training a prediction model. The overall algorithm (Procedure `kMeans`) for the construction is summarized in Figure 4. The label (*i.e.*, the optimal index) of each graph $G \in \mathcal{D}$ is determined by the definition presented in Section 4. Then, similar graphs in $\mathcal{D}$ are clustered by Procedure `kMeans`. The label of a cluster center represents the label of that cluster.

While Procedure `kMeans` follows the general framework of classical k-Means algorithm, we highlight this particular adoption, *i.e.*, Lines 06-09. Most importantly, the major modification is to the recalculation of the new cluster centers in Lines 08-09. Classical k-Means algorithms often use averaging of a distance function between data objects in a cluster to obtain a new center. However, the averages of the eigenvectors or eigenvalues do not correspond to any clear semantics. Therefore, in Line 08, we determine the sum of the spectral similarity between each graph and all other graphs in a cluster. The new cluster center is the graph with the highest similarity sum (Line 09).

```
Procedure kMeans
Input: labels, $\Lambda$'s and $U$'s of graphs in $\mathcal{D}$, cluster number $k$,
        and termination parameter $\delta$
Ouput: the $k$ clusters $\mathcal{C}$ and their centers

randomly choose $k$ centers for $k$ clusters $\mathcal{C}$
for each $G$ in $\mathcal{D}$
  $i_{max} = argmax_{i=1,...,k}(sim(G, \mathcal{C}[i].ctr))$   //$\mathcal{C}[i]$ is $i$th cluster
  assign $G$ to the cluster $\mathcal{C}[i_{max}]$
while the clusters $\mathcal{C}$ have changed
  for each $\mathcal{C}[i]$, where $\mathcal{C}[i] \in \mathcal{C}$, $\delta\%$ of graphs in $\mathcal{C}[i]$ changed
    // recalculate the center of $\mathcal{C}[i]$
    for each $G \in \mathcal{C}[i]$   $G.sim = \sum_{G' \in \mathcal{C}[i]}(cos\_sim(G, G'))$
    $\mathcal{C}[i].ctr = argmax_{G \in \mathcal{C}[i]}(G.sim)$   // $\mathcal{C}[i].ctr$ is center of $\mathcal{C}[i]$
  for each $G$ in $\mathcal{D}$
    $i_{max} = argmax_{i=1,...,k}(sim(G, \mathcal{C}[i].ctr))$
    assign $G$ to $\mathcal{C}[i_{max}]$
```

Fig. 4. Procedure kMeans

Optimization. The clustering algorithm recalculates cluster centers in each iteration. However, in later iterations of Procedure kMeans, the changes in clusters are often small. In other words, the algorithm may then recompute the spectral similarity of the same graphs many times. To optimize this, we store the spectral similarities between graphs when they are first computed and then retrieved in later iterations.

6.2 Prediction Algorithm

To predict the optimal index of a graph $G' \notin \mathcal{D}$, one may be tempted to use the label of the cluster center that is the most similar to G'. However, as discussed above, the cluster centers are data graphs themselves and sometimes may not be optimal. Accordingly, the prediction using one cluster center may be overly sensitive to the quality of the clusters. To enhance the robustness of the prediction, we adopt a simple k-NN algorithm, where k is a user-defined parameter. Specifically, given a graph G', we decompose it into Λ'_G and U'_G. We determine the k clusters $\mathcal{C}[i_1]$, $\mathcal{C}[i_2]$,..., $\mathcal{C}[i_k]$ from $\mathcal{C}$, whose centers are the most similar to G'. Suppose the label of $\mathcal{C}[i]$'s center is L. The vote of $\mathcal{C}[i]$ to L is defined as the spectral similarity between $\mathcal{C}[i]$'s center and G'. The optimal index of G' is the label with the largest sum of votes.

7 Experimental Evaluation

In this section, we report on an experiment conducted to verify the accuracy and efficiency of the spectral decomposition approach to predict the optimal graph index.

7.1 Experimental Setup

Implementation: We ran our implementation on a commodity PC with a Quad-core 2.4GHz CPU with 4G memory running Windows 7. We implemented our proposed technique using MATLAB R2011a. We used the functions provided by MATLAB as far as possible, such as those determining the eigenvalues and eigenvectors, and statistical distribution fittings.

Graph Collections: We used both random and scale-free graphs for our experiments, as they are popular classes used in graph analysis. Moreover, we controlled for their sizes and densities. The generators used were provided by Zhu et al. [19]. We generated 1,024 graphs of each type. For random graphs, the average number of vertices and fanout were 3.4k and 6.8, respectively. For scale-free graphs, we set $\alpha = 0.27$ and $\beta = 10$ and obtained 1,024 graphs with an average number of vertices of 3k and average fanout 7.2. We use `R` and `S` to denote experiments with random and scale-free graphs, respectively.

Reachability Query Time Collections: We ran the implementations of `Interval` [19], `Grail` [18], `2-hop` [2] and `Prime labeling` [11] on our graph collections. We ran 1,000 random reachability queries on each graph. The runtimes of these 1,000 queries on each index were then stored and fitte into Gamma distributions. The runtimes were obtained from warm runs. The estimated α and β of Gamma distributions were stored for determining the optimal index. The label of a graph is determined by the y value.

We observe that there were cases where the prediction problem was trivial, *e.g.*, one index was almost always more efficie t than the others. These cases were omitted as our prediction model was very accurate. In other words, neither of the indexes chosen in our experiments dominated another.

Table 1. Meanings & default values of parameters

parameter	meaning	default
k_knn	k in k-NN	7
k_kmeans	k in KMeans	64
$kmeans$	whether KMeans is used in prediction	yes
$\|T\|$	no. of samples used for testing	28
$\|D\|$	no. of samples used for both training and testing	256
k_tail	tail k eigenvalues and their corresponding eigenvectors used in graphs' similarity	32
σ	parameter in sim to balance the importance of eignvalues and eigenvectors	0.1

Table 2. Fitting error (L_2-norm)

fittin error	Poisson	Normal	Gamma
`R(Interval)`	0.16	0.09	0.06
`R(2-hop)`	0.17	0.16	0.14
`R(Grail(1))`	0.50	0.43	0.27
`R(Prime)`	0.43	0.49	0.26
`S(Interval)`	0.18	0.07	0.04
`S(2-hop)`	0.25	0.23	0.20
`S(Grail(1))`	0.52	0.42	0.29
`S(Prime)`	0.77	0.70	0.51

Default Parameter Settings: We conducted a set of experiments to show the effect of each parameter in our technique. Unless specifie otherwise, we used the default settings shown in Table 1. We used the VP_rad utility function for the vertex permutation in the sim computation by default. For ease of exposition, we predict the optimal index from two graph indexes. That is, the prediction label of the experiments was binary.

Performance Metric: Unless otherwise specified we ran each experiment 100 times and report here the average accuracies.

7.2 Experiments on Distribution Fittings

To verify that the distributions of the reachability query times on each index can be fitte to some well-known distributions, we tested their fitti g functions. We generated the runtime distributions of all of our random and scale-free graphs and all of our index

Table 3. Effects of $|D|$ on R

$\|D\|$	average accuracy (ours / [6])	
	2-hop vs Grail(1)	Interval vs Prime
64	84.18% / 85.06%	79.18% / 75.86%
128	87.57% / 87.35%	83.14% / 85.31%
256	87.11% / 88.18%	82.68% / 85.25%
512	85.18% / 92.03%	82.54% / 89.36%

Table 4. Effects of k in kMeans on R and S

k_kmeans	average accuracy (R)		average accuracy (S)	
	2-hop vs Grail(1)	Interval vs Prime	2-hop vs Grail(3)	2-hop vs Grail(5)
8	71.21%	70.43%	63.64%	65.11%
16	79.61%	77.29%	76.75%	79.68%
32	85.25%	80.50%	79.79%	81.18%
64	89.46%	83.75%	78.71%	80.00%
128	90.54%	84.21%	76.00%	76.89%

Table 5. Effects of k in knn on R and S

k_knn	average accuracy (R)		average accuracy (S)	
	2-hop vs Grail(1)	Interval vs Prime	2-hop vs Grail(3)	2-hop vs Grail(5)
1	83.57%	79.61%	67.86%	67.46%
3	86.79%	81.43%	73.68%	74.86%
5	88.82%	82.61%	77.36%	77.50%
7	89.79%	83.00%	79.86%	80.57%
9	89.29%	82.82%	81.07%	80.75%

Table 6. Effects of k in k_tail on R and S

k_tail	average accuracy (R)		average accuracy (S)		time
	2-hop vs Grail(1)	Interval vs Prime	2-hop vs Grail(3)	2-hop vs Grail(5)	(s)
8	81.68%	79.54%	69.43%	68.93%	3.65
16	84.00%	81.11%	73.75%	74.57%	10.00
32	88.29%	83.75%	77.79%	80.04%	30.10
64	88.43%	83.43%	80.25%	82.89%	105.88
128	88.82%	84.39%	81.71%	83.04%	413.09

implementations. We used y%=98% in our experiments. In Table 2, we show the L_2-norm between the actual and estimated distribution of Poisson, normal and Gamma distributions. We note that the Gamma distributions almost always clearly offered more accurate fittings and the fitting errors were often small. Therefore, in this work, we have adopted the Gamma distribution.

7.3 Prediction Accuracies

In this experiment, we tuned the parameters of our prediction model and studied their effects on prediction accuracies. Due to space limitations, we present only the results obtained from some pairs of indexes for each dataset: 2-hop vs Grail(1) and Interval vs Prime for random graphs, and 2-hop vs Grail(3) and 2-hop vs Grail(5) for scale-free graphs.

Effects of the Size of Training Dataset. We studied the effect of the dataset size on prediction accuracy for random graphs, as shown in Table 3. $kMeans$ was set to no in this experiment. From Table 3, we observe that our prediction accuracies were almost always above 80% on training sets of different sizes. We also note that the prediction accuracy first increased and then slightly declined as the training dataset grew larger. This was possibly because our model was overfitted by large training sets. Moreover compared our method with a previous work [6]. While our method is either comparable or slightly less accurate, it is less *ad-hoc* than [6]. As discussed, spectrums have known to be useful in many real applications. However, in [6], the features were chosen simply because they are verified useful by experiments.

Effects of k_kmeans. We studied the effects of k_kmeans on prediction accuracy as shown in Table 4. It can be seen that accuracy increased with the growth in k_kmeans for random graphs. This is because the clusters become more refined with larger k_kmeans. Thus, we had a higher probability of selecting similar cluster centers for prediction. However, accuracy may reduce slightly if each cluster is too fine as shown in the results for scale-free graphs.

Effects of k_knn. Table 5 presents the effects of k_knn on prediction accuracy. It can be seen that accuracy increases with the growth of k_knn. This is because that a large k results in more votings, which reduces the effects of outliers. Our accuracy was over 80% on both random and scale-free graphs when $k \geq 7$. The prediction accuracy was stable when $k \geq 9$.

Table 7. Effects of vertex permutation and σ on `R`

Vertex	σ (`2-hop` vs `Grail(1)`)					σ (`Interval` vs `Prime`)				
Permutation	0.01	0.1	1	10	100	0.01	0.1	1	10	100
VP_rad	89.43%	89.71%	82.25%	83.25%	71.21%	82.93%	83.21%	78.64%	74.14%	73.64%
VP_coor	91.00%	90.21%	69.25%	69.11%	68.93%	82.71%	85.96%	73.00%	73.00%	72.43%
VP_W_rad	89.39%	90.68%	70.61%	72.21%	71.14%	82.71%	84.89%	72.86%	74.36%	75.25%
VP_none	87.36%	80.86%	82.25%	69.50%	70.00%	81.89%	82.82%	78.61%	72.86%	72.14%

Effects of the Number of Tail-k Eigens. We used different number of eignvalues and eigenvectors in prediction and studied the effect on accuracy, as shown in Table 6. It can be seen that accuracy increases with more eignvalues and eignvectors, while the prediction time increased roughly linearly. We exclude the time taken to determine the spectrum of a graph as this mainly depends on the algorithm used. From our data, the MATLAB function ran from 2.6s to 173s with an average of 39s. However, this is often still more efficien than choosing the optimal index by constructing all candidate indexes and running a large number of benchmark queries.

Effects of Vertex Permutation and σ. In this experiment, we studied the effects of vertex permutation and σ on prediction accuracy. Table. 7 presents the results. It may be observed that while VP_coor, VP_W_rad and VP_none could all sometimes be accurate, they were sensitive to the choice of σ, in the similarity function. In comparison, VP_rad is both robust and accurate. Moreover, when σ increased, the relative importance of eigenvectors reduced, and accuracy decreased.

8 Related Work

To the best of our knowledge, there have been only few preliminary studies that use graph features to predict the relative query performances of graph indexes. Deng et al. [6]. extract features from data graphs and use neural networks for prediction. However, there have been many features in graph theories and it is unclear which of these are the principal ones. In contrast, we use the tail-k eigenvalues and eigenvectors for prediction. Moreover, the optimal index of [6] is defined by the best average runtimes. In comparison, we also allow users to fine-tune their notion of the optimal index. Another work by Zhu et al. [19] applies multiple graph indexes to partitioned subgraphs of a data graph. An analytical cost model is proposed and illustrated with `2-hop` and `Interval`. Our approach has been applied to various indexes. Spectral methods have been applied to produce k partitions of graphs *e.g.*, [10]. Our aim is not to produce exactly k partitions but to predict to the optimal index.

Finally, there is a large body of work on determining graph features, *e.g.*, [16], for query processing, *e.g.*, [17,4]. Due to space limitations, we cannot include a detailed survey on this area. However, in these studies, features are graphs (structures). It remains unclear how to exploit them to build a predictive model.

9 Conclusions

In this paper, we propose a spectral decomposition method for predicting the optimal graph index of a given graph. Specifically, we propose a uniform representation of a graph, a spectral similarity function and a prediction algorithm. We obtain the implementation of four structurally different graph indexes. One observation is that the runtime distributions of the indexes fit accurately into a Gamma distribution. This allows us to refine the notion of the optimal index, using the inverse cumulative distribution function. We report on detailed experiments on the parameters in our techniques on both random graphs and scale-free graphs. We note that our technique is robust and can achieve approximately 70% accuracy or higher. In future work, we will investigate methods for other subgraph queries.

Acknowledgment. This work is partly supported by FRG2/10-11/106, GRF HKBU210510 and GRF 211512.

References

1. Agrawal, R., Borgida, A., Jagadish, H.V.: Efficient management of transitive relationships in large data and knowledge bases. In: SIGMOD, pp. 253–262 (1989)
2. Bramandia, R., Choi, B., Ng, W.K.: Incremental maintenance of 2-hop labeling of large graphs. TKDE 22, 682–698 (2010)
3. Brouwer, A.E., Haemers, W.H.: Spectra of Graphs. Springer (2012)
4. Cheng, J., Ke, Y., Ng, W., Lu, A.: Fg-index: towards verification-free query processing on graph databases. In: SIGMOD, pp. 857–872 (2007)
5. Chung, F.: Spectral Graph Theory. Conference Board of the Mathematical Sciences (1997)
6. Deng, J., Liu, F., Peng, Y., Choi, B., Xu, J.: Predicting the optimal ad-hoc index for reachability queries on graph databases. In: CIKM, pp. 2357–2360 (2011)
7. Dhillon, I.S., Guan, Y., Kulis, B.: Weighted graph cuts without eigenvectors: A multilevel approach. TPAMI 29, 1944–1957 (2007)
8. Hendrickson, B., Leland, R.: An improved spectral graph partitioning algorithm for mapping parallel computations. SIAM J. Sci. Comput. 16(2), 452–469 (1995)
9. Yellen, J., Gross, J.L.: Handbook of Graph Theory. CRC Press (2004)

10. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: NIPS, pp. 849–856. MIT Press (2001)
11. Peng, Y., Choi, B., Xu, J.: Selectivity estimation of twig queries on cyclic graphs. In: ICDE, pp. 960–971 (2011)
12. Schenkel, R., Theobald, A., Weikum, G.: Efficient creation and incremental maintenance of the hopi index for complex xml document collections. In: ICDE, pp. 360–371 (2005)
13. Song, L., Peng, Y., Choi, B., Xu, J., He, B.: Spectral decomposition for optimal graph index prediction. Technique Report (2012), http://www.comp.hkbu.edu.hk/~ypeng/papers/TechRep2012.pdf
14. Spielman, D.A.: Spectral graph theory and its applications. In: FOCS, pp. 29–38 (2007)
15. Spielman, D.A.: Spectral graph theory. In: Combinatorial Scientific Computing, pp. 1–23. Chapman and Hall/CRC Press (2011)
16. Yan, X., Han, J.: gspan: Graph-based substructure pattern mining. In: ICDM, pp. 721–724 (2002)
17. Yan, X., Yu, P.S., Han, J.: Graph indexing: a frequent structure-based approach. In: SIGMOD, pp. 335–346 (2004)
18. Yildirim, H., Chaoji, V., Zaki, M.J.: Grail: scalable reachability index for large graphs. PVLDB 3(1-2), 276–284 (2010)
19. Zhu, L., Choi, B., He, B., Yu, J.X., Ng, W.K.: A uniform framework for ad-hoc indexes to answer reachability queries on large graphs. In: Zhou, X., Yokota, H., Deng, K., Liu, Q. (eds.) DASFAA 2009. LNCS, vol. 5463, pp. 138–152. Springer, Heidelberg (2009)

Patterns amongst Competing Task Frequencies: Super-Linearities, and the Almond-DG Model

Danai Koutra[1,*], Vasileios Koutras[2],
B. Aditya Prakash[3], and Christos Faloutsos[1]

[1] Computer Science Dept., Carnegie Mellon Univ.
{danai,christos}@cs.cmu.edu
[2] Dept. of Accounting & Finance, Athens Univ. of Econ. & Bus.
vkoutras@aueb.gr
[3] Computer Science Department, Virginia Tech.
badityap@cs.vt.edu

Abstract. If Alice has double the friends of Bob, will she also have double the phone-calls (or wall-postings, or tweets)? Our first contribution is the discovery that the relative frequencies obey a power-law (sub-linear, or super-linear), for a wide variety of diverse settings: tasks in a phone-call network, like count of friends, count of phone-calls, total count of minutes; tasks in a twitter-like network, like count of tweets, count of followees etc. The second contribution is that we further provide a full, digitized 2-d distribution, which we call the Almond-DG model, thanks to the shape of its iso-surfaces. The Almond-DG model matches all our empirical observations: super-linear relationships among variables, and (provably) log-logistic marginals. We illustrate our observations on two large, real network datasets, spanning $\sim 2.2M$ and $\sim 3.1M$ individuals with 5 features each. We show how to use our observations to spot clusters and outliers, like, e.g., telemarketers in our phone-call network.

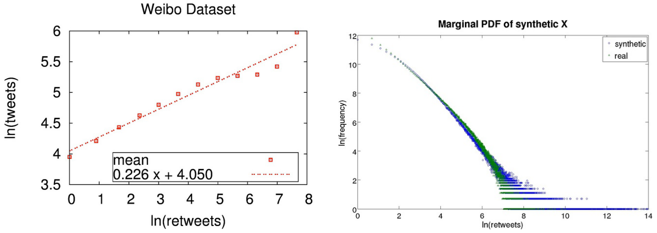
1 Introduction

If 'Alice' has 50 contacts and did 100 phonecalls to them, what should we expect for 'Bob', who has twice the contacts? One would expect a linear relationship (double the contacts, double the phonecalls). However, we show that in numerous settings, the relationship is a power law, being sub- or super-linear.

Useful as it may be for point estimates, the power-law relationship cannot give us any estimate for the variance. How would we model such joint distributions,

[*] We would like to gratefully acknowledge the organizers of KDD Cup 2012 as well as Tencent Inc. for making the datasets available. Funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088. This work is also partially supported by an IBM Faculty Award and a grant from the VT College of Engineering. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 201–212, 2013.



(a) Super-linear Rel. Frequency (b) log-logistic marginals

Fig. 1. Illustration of super-linearity and goodness of our proposed Almond-DG. (a) Power-law relationship between count of tweets and count of retweets for each user in Tencent-Weibo (log-log scales). (b) Marginal PDF of the retweets, in log-log scales: real (in green triangles); generated by Almond-DG (in blue circles).

like, say, the number of contacts vs. number of phone-calls? What can we say about their marginals? They are definitely *not* Gaussian. Do they follow a Pareto (power-law) distribution in their marginals? What about the joint distribution?

The questions we want to answer here are:

1. **Q1: Patterns**: if 'Alice' executes task x (say, phone call) for n_x times, how many times n_y does she do task y (say, send an sms)?
2. **Q2: Distribution estimate**: What is the appropriate 2-d distribution to fit real, 2-d points (like, say <# tweets, # retweets> in the twitter setting)? Multivariate Gaussian fails miserably, due to heavy tails in real data.
3. **Q3: Practical use**: Can we answer "what-if" scenarios, and find anomalies?

Our contributions are exactly the answers to the above questions:

- **A1: Patterns:** We observe power law relationships between tasks competing for a person's resources (e.g., time).
- **A2: Distribution Estimate:** We propose the Almond-DG distribution, which uses the lesser-known tool of *copulas*, and has all the properties we observed in real data: the super-linear relationship, and also, log-logistic marginals (which are prevalent in many real-world datasets).
- **A3: Practical use:** Our Almond-DG fits several, diverse real datasets. We show how to spot outliers, and how to answer what-if questions.

We report results from two large, real, diverse network datasets. The first spans $\sim 3.1M$ users and is on a phone-call dataset; for each customer, we study the count of distinct contacts, phonecalls, text messages, and the total minutes. The second is from Tencent-Weibo network, a Chinese version of Twitter$^{\mathrm{TM}}$, with count of tweets, re-tweets, followees etc. per user. Figure 1 illustrates our main ideas and discoveries: Figure 1(a) depicts the power-law relationship between count of tweets and count of retweets. Both axes are logarithmic; each red square is the conditional average of tweets, for the given count of retweets. The last few points are noisy, because of extreme-value effects (there are very few people with so many re-tweets, and they dominate the average). Figure 1(b) shows the marginal PDF of the retweets, again in log-log scales. The green triangles are the real distribution, while the blue circles correspond to synthetic data, generated by our proposed ALMOND-DG. Notice that (i) the real distribution has a power-law tail, but it tilts in the beginning (top concavity) and (ii) our synthetic data fit well. Table 1 gives the major symbols we use and their definitions.

Table 1. Symbols and definitions

Symbol	Definition
$F_X(\cdot)$, $F_D(\cdot)$	cumulative distribution function (CDF) for: (a) random variable X or (b) distribution D (e.g., F_{LL} = CDF of log-logistic)
a_x, a_y	location parameter of log-logistic random variables X & Y
b_x, b_y	scale parameter of log-logistic random variables X & Y
$C(\cdot, \cdot)$	copula: 2 variable dependence function $[0, 1] \times [0, 1] \to [0, 1]$
θ	parameter in Gumbel's copula that captures correlations between the random variables X & Y
SURF	Super-Linear Relative Frequency Observation
ALMOND	our 2-d continuous log-logistic distribution using Gumbel's copula
ALMOND-DG	our proposed distribution: the discretized and truncated version of ALMOND

2 Patterns and Observations

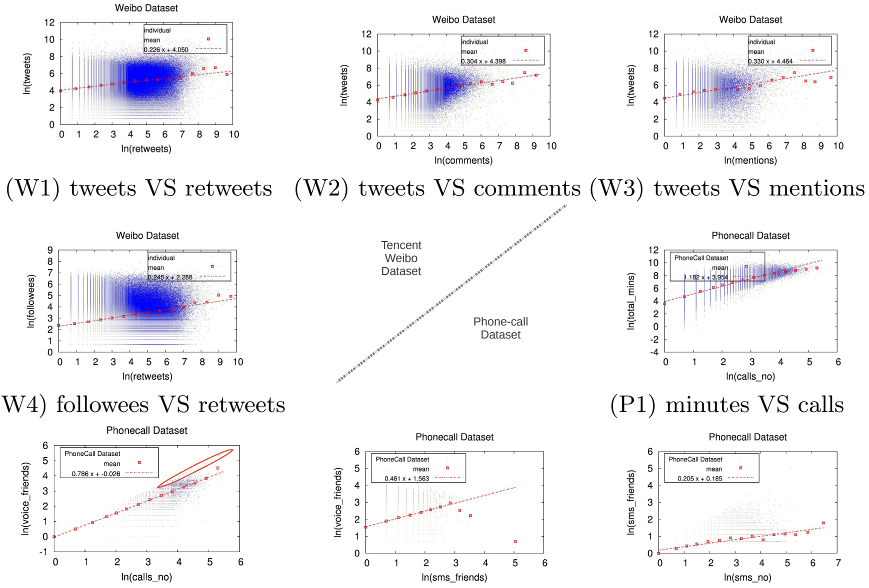
What happens to the number of tweets of a user if her re-tweets triple (Figure 1(a))? To answer such 'what-if' scenarios, we study two big, real networks, from which we extract 5 features for each user; each feature corresponds to the occurrences of a task:

- Tencent Weibo (W) [10]: one of the largest micro-blogging websites in China. For each of the $\sim$ 2.2 million users we extracted five quantities: the number of her tweets, retweets, comments, mentions and followees.
- Phonecall dataset (P): phone-call records from a mobile provider in a big Asian city. For each of the $\sim$ 3.1 million customers we obtained the number of her calls, messages, "voice" and "sms" friends, as well as the total minutes of her phone-calls.

In Fig. 2 we present the scatter plots of pairs of tasks. For each dataset we have $n = 5$ features. We are giving the plots for $n - 1$ pairs, instead of $\binom{n}{2}$, due to space limitations. Each user/customer is a blue point on the plane and is characterized by the number of times she did tasks 'A' and 'B'. All plots "suffer" from heavy over-plotting (not visible), especially for small values of occurrences. So, linear regression fails and we resort to the following solution: we group the points in logarithmic buckets and compute the mean (red points) of Y given X. The line $E[Y|X = x]$ is obtained by linear regression on the red points (ignoring the few last points, where the observations are extremely sparse, possibly due to the "horizon effect"). As we observe, in all cases the conditional expectation is a linear function of x. We call this sub- or super-linear relationship between the frequency of occurrence of the tasks SURF (Super-linear Relative Frequency).

Observation 1. *Deviations from the power-law pattern, as shown in Fig. 2(P2) are due to outliers, e.g. telemarketers.*

The customers within the red ellipse all have about 100 contacts, and 100 phone-calls, that is about one phone-call per contact. The rest of the population has many more phone-calls than contacts; thus, this difference in behavior leads to the suspicion that the former are telemarketers.



(W1) tweets VS retweets (W2) tweets VS comments (W3) tweets VS mentions

(W4) followees VS retweets (P1) minutes VS calls

(P2) voice friends VS calls (P3) voice VS sms friends (P4) sms friends VS sms

Fig. 2. SuRF patterns in real datasets (log-log scale, "W": Weibo, and "P": Phonecall network): power-law relationship between competing tasks. In plot (P2), the 'anomalous' customers in the red ellipse deviate from the power-law pattern, having called each of their contacts only once; they are probably telemarketers.

3 Almond-DG Distribution

In order to model the observed patterns of the previous section, we need a probabilistic model; this model must satisfy the properties found in real 2-d distributions, including the important property found in the previous section, namely, the conditional average seems to follow a power law (Figure 1(a)). The rest of the paper focuses on two questions:

- **Q2.1**: Can we find additional properties of such 2-d (and more ambitiously, higher-d) distributions?
- **Q2.2**: Can we build a probabilistic model (i.e., find a 2-d PDF) that will fit most real clouds of points? It is clear that the multivariate Gaussian is heavily violated, even visually from Figure 1(b), and, as we show later, from the marginals of the x and y axis.

In summary, the answers are as follows:

- **A2.1**: Yes, the marginals of almost any attribute in our real datasets has a skewed distribution, and can be modeled well by a (truncated, digitized) log-logistic distribution.
- **A2.2**: Subject to log-logistic marginals, and power-law conditional averages, a possible candidate 2-d distribution is our proposed 'Almond' distribution (digitized and truncated).

We want a 2-d distribution whose iso-surfaces will resemble the ones in Figure 3. Since they have 'almond'-shape (see Fig. 3(c)), we name our proposed distribution as the 'Almond' distribution, and, after digitization and truncation, Almond-DG.

The final answer is given by the discretized form of Eq. (1) in p. 207, which we repeat here for convenience:

$$F_{ALM}(x, y; a_x, b_x, a_y, b_y, \theta) = e^{-\left(\left[\ln\left(1+(x/a_x)^{-b_x}\right)\right]^{\theta}+\left[\ln\left(1+(y/a_y)^{-b_y}\right)\right]^{\theta}\right)^{1/\theta}},$$

where θ captures the correlation between the random variables (attributes) X and Y, while a_x (a_y) and b_x (b_y) determine respectively the location (≈ average/mode) and the spread (≈ variance).

As we discuss next, this distribution has all the desired properties (the super-linear relationship, and provably, log-logistic marginals). The approach we follow for Almond-DG has two steps: (a) modeling the marginal (univariate) distributions of the tasks, and (b) combining them with the use of the so-called *copula*. Before we see the train of thought, we first give a property of the marginals (in response to question Q2, above), and then some definitions.

For Q1, given the overwhelming number of real-world (univariate) datasets that exhibit skewed distributions, one would expect that the marginals follow power-law or log-normal distributions. We found that this is *almost* true: an even better fit is provided by the so-called *log-logistic* distribution, which also accounts for the top concavity (see Fig. 1(b) p. 202, Fig. 4(a,b) p. 209, Fig. 5 p. 212). We proceed with some definitions.

3.1 Definitions

Log-logistic distribution. All the marginals we report match the so-called log-logistic distribution. Thus, we remind its definition next.

Definition 1 (CDF of log-logistic). *The log-logistic CDF is given by*

$$F_{LL}(x; a, b) = (1 + (x/a)^{-b})^{-1}, x \geq 0$$

where $a > 0$ is the scale parameter and $b > 0$ is the shape parameter.

By definition, a (continuous) random variable X follows the log-logistic distribution, $\mathcal{LL}(a, b)$, iff its logarithm $\ln X$ follows the logistic distribution $\mathcal{L}(\ln a, 1/b)$. Intuitively, the CDF of the logistic distribution is the sigmoid function – exactly the one used for logistic regression, artificial neural networks, modeling product market penetration (Bass model), spread of epidemics (SI model), etc. A second property is that the odds-ratio of the log-logistic distribution, follows a power-law, and thus it is a straight line in log-log scales (see Fig. 5 in p. 212).

Moreover, the log-logistic distribution is related to the standard Pareto distribution: If $X \sim F_{LL}(x; 1, 1)$, then the shifted random variable $Z = X + 1$ follows the standard Pareto distribution.

Copulas for Modeling Dependence.

Our proposed 2-d log-logistic distribution (Almond-DG) is based on copulas. So, before we present the details of our distribution, we briefly give the main notions behind this powerful technique, which has been successfully used in survival models, financial risk management, and decision analysis.

In a nutshell, copulas are used for understanding and modeling dependence structures between random variables (e.g., $X = \#$ of phonecalls, $Y = \#$ of sms). More specifically, the copula function links the univariate margins (F_X, F_Y) with their full multivariate distribution. By construction, the latter (a) has the same marginals as X and Y, and (b) exhibits the correlation between them. Copulas have been proved very popular in statistical applications as they allow one to easily model and estimate the distribution of random vectors by estimating the marginals and copula separately. The major difference between the many parametric copula families cited in the literature, is their capability of assuming different dependence structures.

More formally, copulas are defined as follows:

Definition 2 (Copula). *Let X, Y be two random variables with marginal CDFs F_X and F_Y respectively. A copula $C(u, v)$ is a two-variable dependence function $C : [0, 1] \times [0, 1] \rightarrow [0, 1]$ that produces a joint CDF which captures the correlation between the F_X and F_Y variables, i.e., $F(x, y) = C(F_X(x), F_Y(y))$.*

The existence of such copula is guaranteed by Sklar's Theorem [17]. Note that if X, Y are independent variables, their joint CDF takes the form $F(x, y) = C(F_X(x), F_Y(y)) = F_X(x)F_Y(y)$. Hence the copula $C(u, v) = u\,v$ captures their independence.

The copulas are a powerful tool that can capture any type of dependence: positive, negative, none (independence). One of the prevailing families of copulas is the so-called Archimedean family; among its members we choose the so-called *Gumbel's* copula. It has been used successfully in several settings, for example, to model the dependence between indemnity payment (loss) and an allocated loss adjustment expense (e.g., lawyer's fees) in order to calculate reinsurance premiums [19]. Equally successfully, Gumbel's copula has been used to model the rainfall frequency as a joint distribution of rain characteristics (e.g., volume, peak, duration) [9]. Formally, it is defined as follows:

Definition 3 (Gumbel's Copula). *Gumbel's copula is given by $C(u, v) = e^{-\left[\phi(u)^{\theta}+\phi(v)^{\theta}\right]^{1/\theta}}$, where $\theta \geq 1$, and $\phi(t) = (-\ln t)^{\theta}$.*

3.2 Proposed Almond-DG Distribution

As we briefly mentioned in the previous section, the log-logistic distribution fits well the skewness of many real-world datasets. The reasons we pick the (digitized, truncated) log-logistic for the marginals, are the following:

- real data (#-mentions, #-phonecalls) have a linear log-odd plot (Fig. 5),
- the log-logistic is related to the Pareto distribution, and
- it captures the top concavity observed in numerous real-world 1-d datasets.

So, how can we model the distribution of pairs of tasks (e.g., number of likes and number of comments) competing for an individual's resources (e.g., time) now that we know that the distribution of each task is log-logistic or power-law-like? It turns out we have a lot of choices, since several 2d-logistic (not log-logistic) distributions with logistic marginals have been proposed in the literature. However, the majority of them (Malik and Abraham [12], Fang and Xu [7], etc) are not suitable in our case, since they are not flexible enough to capture the dependence between the variables that the real datasets exhibit.

Now we have all the ingredients (log-logistic marginals, a first step on how to combine them), and we only need to make sure that X and Y are positively correlated.

Notice that, in this section, we start from continuous distributions (like the log-logistic), and we use their digitized version ($floor()$), followed by truncation: whenever the result is "0" (say, zero phonecalls), we ignore it, since it won't register in our real datasets.

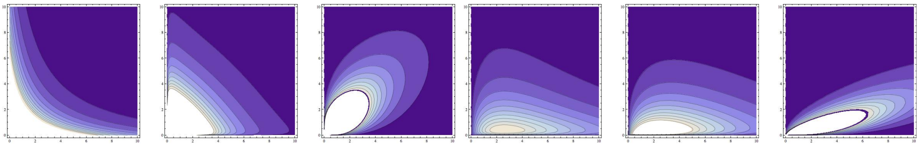
Consider two random variables X and Y (like 'number of phonecalls', and 'count of contacts' in our phonecall network). How can we model their joint PDF? Copulas provide a way to do that, when we know the marginals F_X and F_Y. Sklar's theorem [17] states that we can always find a 2-d function $C(u, v)$ that can model the joint CDF. We use Gumbel's copula with parameter θ, and log-logistic marginals with different parameters each. So, our proposed 'ALMOND' distribution has 5 parameters and is defined below:

Definition 4 ('Almond' Distribution). *The CDF of* ALMOND*, our proposed continuous 2-d log-logistic distribution, is given for $x, y \geq 0$ by (1), where θ is a parameter that captures the dependence between X and Y.*

$$F_{ALM}(x, y; a_x, b_x, a_y, b_y, \theta) = e^{-\left(\left[\ln\left(1+(x/a_x)^{-b_x}\right)\right]^{\theta}+\left[\ln\left(1+(y/a_y)^{-b_y}\right)\right]^{\theta}\right)^{1/\theta}} \tag{1}$$

For illustration purposes, in Fig. 3, we give some examples of contour plots of our ALMOND Distribution. The following observation is useful for estimating the parameter θ from the real data.

Observation 2. *For a pair of random variables (X, Y) that follows the* ALMOND *distribution, θ can be estimated by $\theta = (1 - \tau)^{-1}$, where τ is Kendall's tau rank correlation coefficient between X and Y. In practice, for efficiency, we use Spearman's coefficient ρ instead of τ.*



(a) $\theta = 1$ (b) $\theta = 1.67$ (c) $\theta = 3.33$ (d) $\theta = 1$ (e) $\theta = 1.26$ (f) $\theta = 3.33$

Fig. 3. Contour plots of the Almond distribution with parameters $a_x = a_y = 1$, $b_x = b_y = 1$ for (a)-(c); and $a_x = 6.5, a_y = 2.1$, $b_x = 1.6, b_y = 1.27$ – as in the "comments VS mentions" dataset – for (d)-(e)

Lemma 1. *The marginals of the* Almond *distribution are log-logistic distributions.*

Proof. By taking the limit of y to infinity, we obtain the marginal of X:

$$\lim_{y \to \infty} F(x, y) = F_X(x; a_x, b_x) = \frac{1}{1 + (x/a_x)^{-b_x}}$$

Hence, $X \sim \mathcal{LL}(a_x, b_x)$. Similarly, we can show that $Y \sim \mathcal{LL}(a_y, b_y)$. □

Observation 3 (Special case). *If $\theta = 1$, then $C(u, v) = uv$, and X, Y are independent log-logistic random variables. The CDF of* Almond *becomes then*

$$F(x, y; a_x, b_x, a_y, b_y) = \left(1 + (x/a_x)^{-b_x} + (y/a_y)^{-b_y} + (x/a_x)^{-b_x} (y/a_y)^{-b_y}\right)^{-1}.$$

The definition of our proposed digitized, truncated distribution follows:

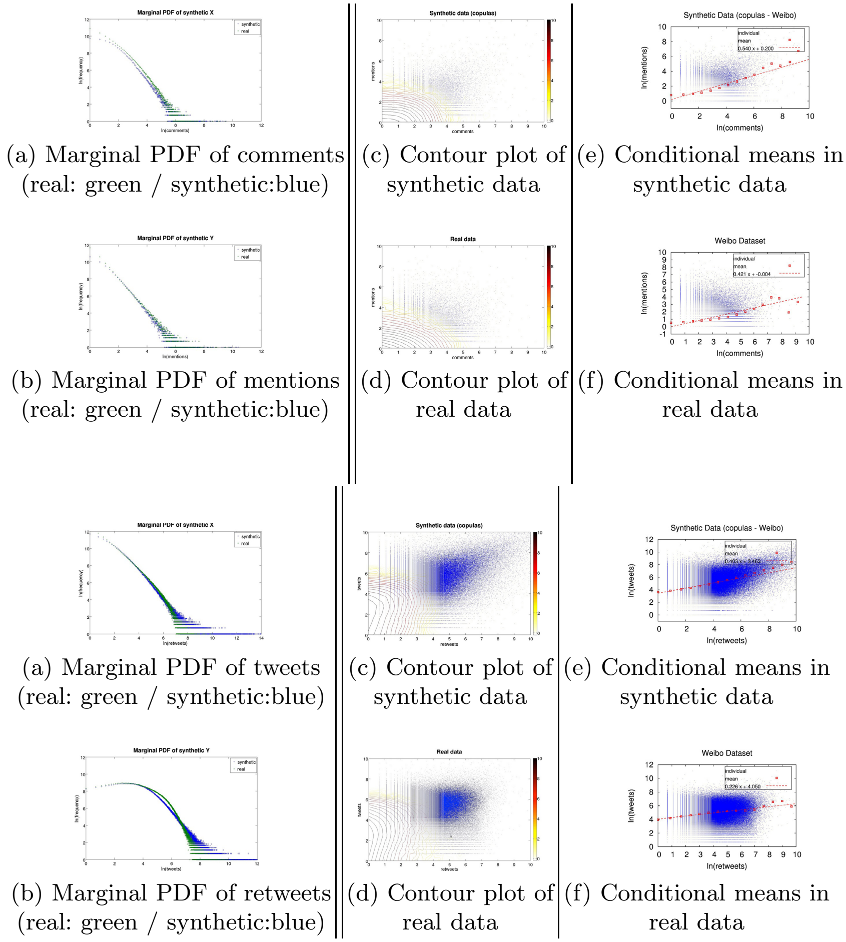
Definition 5 (Almond-DG Distribution). *If (X, Y) follows the* Almond *distribution, then the discrete bivariate random variable $(floor(X), floor(Y))$ given that $X \geq 1$ and $Y \geq 1$ follows the* Almond-DG *distribution.*

Essentially, Almond-DG is derived from Almond by discretizing its values and rejecting the pairs with either $X = 0$ or $Y = 0$.

4 Goodness of Fit

In this section, we show the goodness of fit of our Almond-DG distribution in a qualitative manner. The interested reader may refer to the Appendix for information about the parameter fitting and generation of data following the Almond-DG distribution.

Note: Evaluating the goodness of fit for skewed distributions is a rather challenging task and the majority of methods seem to fail in real data. In [8], Johnson et al. explore several methods for evaluating the goodness of fit for *univariate* Pareto distributions with no clear winner. The difficulty in the evaluation increases even more in the case of bivariate distributions, which we are addressing



(a) Marginal PDF of comments (real: green / synthetic:blue)

(c) Contour plot of synthetic data

(e) Conditional means in synthetic data

(b) Marginal PDF of mentions (real: green / synthetic:blue)

(d) Contour plot of real data

(f) Conditional means in real data

(a) Marginal PDF of tweets (real: green / synthetic:blue)

(c) Contour plot of synthetic data

(e) Conditional means in synthetic data

(b) Marginal PDF of retweets (real: green / synthetic:blue)

(d) Contour plot of real data

(f) Conditional means in real data

Fig. 4. Goodness of fit of ALMOND-DG to the "mentions VS comments" (above) and "tweets VS retweets" datasets. (i) the log-logistic distribution fits well the marginals of the mentions and comments in the real Tencent Weibo network (plots a,b); (ii) the contour plots of the real (plots d) and synthetic 2-d datasets (plots c) have the same shape; (iii) both datasets obey the same power-law pattern, as shown in plots e,f.

in this paper. As we see in Fig. 4 , our proposed ALMOND-DG distribution fares pretty well. In the first column we see the marginal distributions of X (e.g., comments) and Y (e.g., mentions), i.e. $\ln(frequency)$. The green points represent the real data, while the blue points correspond to the distribution of the generated data with the estimated parameters (see Appendix).

Observation 4. *The real marginal distributions are captured well by our* ALMOND-DG *distribution, even when they are power-law-like.*

The second column of the figures holds the contour plots of the synthetic (c), and the real data points (d), while the last column shows the conditional means in synthetic (e) and real data (f).

Observation 5. *The contours of our truncated and digitized* ALMOND-DG *distribution with estimated parameters resemble the real contour lines.*

Observation 6. *The* SURF *pattern is preserved in the synthetic data; the real and synthetic data have similar power-law slopes.*

All in all, ALMOND-DG captures well the patterns found in the real-world data.

5 Related Work

Power Laws: Power laws have been discovered in numerous cases [3], often in conjunction with fractals and self-similarities [15]. Some of the most famous power laws are the Zipf distribution [20] and the Pareto distribution [14]. They have negative slopes, though. Power laws with positive slopes have also been discovered (length of coastlines, number of quad-tree blocks versus granularity [6]), and more recently in graphs ([11], [18], [13]). Akoglu et. al. [1] proposed the Triple Power Law (3PL), a bivariate distribution, to model reciprocity in phone-call networks. However, the model is bound to power-law distributions, which is not always the case in real networks.

Logistic and Log-Logistic Distributions: They have been studied extensively, in the continuous, univariate setting. The multivariate setting has been studied for the logistic distribution [12]. The discretized version of the univariate case has been shown to be a good fit for the duration of phonecalls by real users [4]. Earlier work [2] tried to fit discretized lognormals, or the so-called doubly-Pareto Lognormal [16].

However, none of the above articles provided a solution to our setting, namely, a 2-d distribution, with an explanation for the super-linearity we observe, and validation on several, diverse datasets.

6 Conclusions

The contributions are the answers to the questions we posed in the introduction: Q1: what can we say about the relative frequency of two tasks that compete for an individual's resources (e.g., phone calls vs. number of sms)? Q2: how can we model the corresponding 2-d clouds of points? Q3: how can we put our observations and developments to practical use.

Specifically, our contributions are:

1. **A1 [Patterns]:** Discovery of power law (SURF) in several, real, diverse network datasets, on most of their n-choose-2 pairs of attributes/tasks.
2. **A2 [Distribution Estimate]:** A new distribution, the ALMOND distribution, that describes well the skewed multivariate distributions and explains super-linearity, marginals and conditionals in real, diverse network datasets, on most of their n-choose-2 pairs of attributes/tasks.
3. **A3 [Practical Use]:** Illustration that ALMOND can be used for anomaly detection (Fig. 2(P2)), clustering and what-if scenarios.

References

1. Akoglu, L., Vaz de Melo, P.O.S., Faloutsos, C.: Quantifying reciprocity in large weighted communication networks. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part II. LNCS (LNAI), vol. 7302, pp. 85–96. Springer, Heidelberg (2012)
2. Bi, Z., Faloutsos, C., Korn, F.: The "DGX" distribution for mining massive, skewed data. In: KDD (August. 2001)
3. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. SIAM Rev. 51(4), 661–703 (2009)
4. Vaz de Melo, P.O.S., Akoglu, L., Faloutsos, C., Loureiro, A.A.F.: Surprising patterns for the call duration distribution of mobile phone users. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010, Part III. LNCS (LNAI), vol. 6323, pp. 354–369. Springer, Heidelberg (2010)
5. Embrechts, P., Lindskog, F., McNeil, A.: Modelling dependence with copulas and applications to risk management. In: Handbook of Heavy Tailed Distributions in Finance, pp. 331–385 (2003)
6. Faloutsos, C., Gaede, V.: Analysis of the z-ordering method using the hausdorff fractal dimension. In: VLDB (September 1996)
7. Fang, K.-T., Xu, J.-L.: A class of multivariate distributions including the multivariate logistic. Journal of Mathematical Research and Exposition 9, 91–98 (1989)
8. Johnson, N., Kotz, S., Balakrishnan, N.: Continuous Univariate Distributions, 2nd edn. Wiley (1995)
9. Karmakar, S., Simonovic, S.: Bivariate flood frequency analysis: Part 1. determination of marginals by parametric and nonparametric techniques. Journal of Flood Risk Management 1, 190–200 (2008)
10. KDD-Cup. Tencent Weibo Dataset (2012), http://www.kddcup2012.org
11. Leskovec, J., Kleinberg, J.M., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: KDD, pp. 177–187 (2005)
12. Malik, H.J., Abraham, B.: Multivariate logistic distributions. Annals of Statistics 1, 588–590 (1973)
13. McGlohon, M., Akoglu, L., Faloutsos, C.: Weighted graphs and disconnected components: patterns and a generator. In: KDD, pp. 524–532 (2008)
14. Pareto, V.: Oeuvres Completes. Droz, Geneva (1896)
15. Schroeder, M.: Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise. W.H. Freeman and Company, New York (1991)
16. Seshadri, M., Machiraju, S., Sridharan, A., Bolot, J., Faloutsos, C., Leskovec, J.: Mobile call graphs: beyond power-law and lognormal distributions. In: KDD, pp. 596–604 (2008)
17. Sklar, A.: Fonctions de répartition à n dimensions et leurs marges. Publ. Inst. Statist. Univ. Paris 8, 229–231 (1959)
18. Tsourakakis, C.E.: Fast counting of triangles in large real networks without counting: Algorithms and laws. In: ICDM, pp. 608–617 (2008)
19. Valdez, E.A.: Understanding relationships using copulas. North American Actuarial Journal 2, 1–25 (1998)
20. Zipf, G.: Human Behavior and Principle of Least Effort: An Introduction to Human Ecology. Addison Wesley, Cambridge (1949)

Appendix: Fitting and Data Generation

Most of the estimation methods (e.g., MLE, MOM, etc.) fail when we have skewed, truncated, and digitized distributions with outliers. Fitting becomes even more difficult when fitting 2-d skewed distributions.

Fitting

We propose a fast method for fitting log-logistic (or power-law-like) data to the univariate log-logistic distribution. It is well known that if a random variable X follows the logistic distribution $\mathcal{L}(\mu, \sigma)$, then $-\ln\{odd\} = -\ln\left\{\frac{P(X \leq x)}{P(X > x)}\right\} = -\ln\left\{\frac{F_X(x)}{1-F_X(x)}\right\} = \frac{x-\mu}{\sigma}$. As depicted in Fig. 5, to fit the data via the "log-odd plot": (a) estimate the slope and intercept by applying linear regression on the "log-odd" plot, (b) solve $slope = \frac{1}{\sigma}$ and $intercept = -\frac{\mu}{\sigma}$ for μ and σ, and (c) compute the log-logistic parameters: $a_x = exp(\mu)$ and $b_x = \frac{1}{\sigma}$.

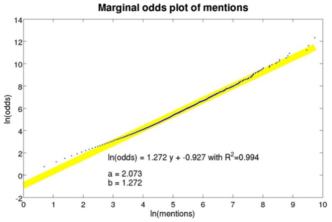


Fig. 5. "Log-odd plot" method for fitting

Generation of Synthetic Data

Step 1: Select $\theta = \frac{1}{1-\tau}$, where τ is the Kendall tau rank correlation coefficient.

Note: Since the computation of τ is prohibitive for the large datasets, being quadratic in the number of points, we use Spearman's ρ, which is a good substitute of τ, and robust to outliers in the data.

Step 2. Estimate the parameters using one of the traditional methods of parameter estimation (e.g., MLE, MOM) ; otherwise, use the "log-odd plot" method described above. In our experiments, we mainly used the MLE of the parameters.

Step 3: Generate two independent random variables s and u following the Uniform distribution $\mathcal{U}(0,1)$ and solve the equation $K(t) = u$ for t, where $K(t) = t - \frac{\phi(t)}{\phi'(t)} = t - \frac{1}{\theta} t \ln t$ (see Def. 3).

Step 4: Compute $x_1 = t^{s^{1/\theta}}$ and $y_1 = t^{(1-s)^{1/\theta}}$.

Note: Originally, $x_1 = \phi^{-1}(s\phi(t))$ and $y_1 = \phi^{-1}((1-s)\phi(t))$.

By starting from a general algorithm for copula-based data generation [5] and using the formulas related to Gumbel's copula, we obtain the formulas given in steps 3-5. Essentially, up to this point we have two uniform $\mathcal{U}(0,1)$ random variables correlated according to Gumbel's copula.

Step 5: Compute $x_0 = \ln a_x - \frac{1}{b_x} \ln\left(\frac{1}{x_1} - 1\right)$ and $y_0 = \ln a_y - \frac{1}{b_y} \ln\left(\frac{1}{y_1} - 1\right)$, which follow the logistic distribution $\mathcal{L}(\ln a_x, \frac{1}{b_x})$ and $\mathcal{L}(\ln a_y, \frac{1}{b_y})$ respectively.

Note: we use the fact that if $U \sim \mathcal{U}$, then $X = \mu - \sigma \ln(\frac{1}{U} - 1) \sim \mathcal{L}(\mu, \sigma)$.

Step 6: Find the "coupled", "digitized" log-logistic variables $x = floor\{e^{x_0}\}$ and $y = floor\{e^{y_0}\}$, and truncate them by keeping only the (x, y) pairs with $x > 0$ and $y > 0$.

Similar digitization process is traditionally practiced when one wants to shift from continuous to discrete power law distributions [3].

Node Classification in Social Network via a Factor Graph Model

Huan Xu[1], Yujiu Yang[1], Liangwei Wang[2], and Wenhuang Liu[1]

[1] Shenzhen Key Laboratory of Information Science and Technology,
Graduate School at Shenzhen, Tsinghua University, P.R. China
[2] Noah's Ark Laboratory, Huawei Technologies Co., Ltd., Shenzhen, P.R. China

Abstract. This paper attempts to addresses the task of node classification in social networks. As we know, node classification in social networks is an important challenge for understanding the underlying graph with the linkage structure and node features. Compared with the traditional classification problem, we should not only use the node features, but also consider about the relationship between nodes. Besides, it is difficult to cost much time and energy to label every node in the large social networks. In this work, we use a factor graph model with partially-labeled data to solve these problems. Our experiments on two data sets (DBLP co-author network, Weibo) with multiple small tasks have demonstrated that our model works much better than the traditional classification algorithms.

1 Introduction

With the success of many large-scale online social networks these years, such as Facebook, Tweeter, Weibo, social network has become a bridge between the virtual web world and our daily life. Weibo, one of the largest and most influential social networks in China, has more than 300 million active users in 2012. Consequently, Network ideas have been applied successfully in many areas such as Internet (pages) [3][8], coauthors[19], mobiles and mails[5]. Considerable research has been conducted on social network analysis, such as social influence analysis[6][18], community structure learning[18][1][14], and of course node classification in social network[4][15][7][10].

As is usual in machine learning, we first have to identify some features of nodes that can be used to guide the classification. The obvious features are properties of the node itself. For online social network like Weibo, information that may be known for all (or most) nodes, such as age, location, and some other profile information is usually considered first. For coauthor network, people may take authors profile, publish year as features. More than that, some latent features such as topic model become more and more popular in network analysis, these latent features reflect user character quite well. But in a network structure, the presence of an explicit link structure makes the node classification problem different from traditional machine learning classification tasks, where objects being classified are considered independent. In contrast to the traditional classification,

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 213–224, 2013.

we should first think about the network topology, the neighbors label information may be very important and decisive. The social sciences identify two important phenomena that can apply in social networks:

- **homophily**, when a link between individuals (such as friendship or other social connection) is correlated with those individuals being similar in nature. For example, friends usually have the similar age, education background.
- **co-citation**, regularity is a related concept, which holds when similar individuals tend to refer or connect to the same things. For example, when two people send microblogs with similar topics, it will be probably they have similar taste in some areas.

Macskassy and Provost used a simpler classification method based on taking a weighted average of the class probabilities in the neighborhood (wvRN)[11]. This classifier is based on a direct application of homophily and uses the immediate neighborhood of a node for classification. Another classifier, which is similar with wvRN, is called the Class-Distribution Relational Neighbor (CDRN)[12], it also considered on the distribution of the neighbors only. There were many works about using random walk on node classification, but most of them concerned about the labels of neighbors. Pennacchiotti and Popescu[15] used a machine learning algorithm with hundreds of features in twitter data, but they simply take network topology as some features in a traditional classifier.

In this work we address the task of node classification in social networks, our main contributions are the following:

- We employ a factor graph model to classify nodes in networks, using topic model as node features instead of profiles and consider multiple relationships in network.
- We conduct experiments on two data sets (DBLP coauthor and Weibo) and multiple tasks. Experimental results show that our model can be applied to the different scenarios and perform quite well than traditional classifiers.
- We provide an in-depth analysis of experiments on the partially-labeled data sets, results show that our model can do a good job with different ratio of labeled data.

The rest of this paper is organized as follows. Section 2 formally formulates the problem. Section 3 explains the factor graph model we used. And then in section 4 we discuss the experiments and evaluation. Finally, in Section 5 we draw final conclusions and outline future work.

2 Problem Definition

In this section, we first give several necessary definitions and then present the problem formulation.

Definition 1. ***Social network:*** *A social network can be represented as* $G = (V, E)$*, where* V *is a set of* $|V| = N$ *users.* $E \subset V \times V$ *is a set of* $|E| = M$ *relationships between* N *users.*

As we know, Social relationships might be directed in some networks (e.g., A follow B in Weibo) or undirected in others (A and B are coauthors in DBLP). In this work, we make the relationship as factor and ignore whether it is directed or undirected, the number of relationship types is what we only concern about. In some situations, the network can be defined in more than one way, such as in DBLP coauthor network: take each author as a node, coauthor may be a relationship; we can also take each paper as a node, two papers have common author may be a relationship. Which one should we choose is depend on the specific task, in this work, we choose the latter.

In real networks, it is difficult to cost much time and energy to label every node, the labeled data is always less than expected, so naturally, we define the input of our problem, a partially labeled network.

Definition 2. ***Partially labeled network:*** *A partially labeled network is an augmented social network represented as $G = (V_L, V_U, E, Y_L, W)$, where V_L is a set of labeled nodes while V_U is a set of unlabeled nodes with $V_L \bigcup V_U = V$; Y_L is a set of labels corresponding to the node classes in V_L; W is an weight matrix associated with users in V where each row corresponds to a user, each column an attribute, and an element w_{ij} denotes the value of the j^{th} attribute of user v_i.*

In our work, we choose a topic model called PLSA (Probabilistic Latent Semantic Analysis)[9] to analysis the text about each node as attribute. The PLSA model assumes that there are k topics in the corpora, where k is a fixed parameter, and every document in the corpora corresponds to one distribution of topics. This is a hierarchical model. We can describe its generative process as:

- *Select a document d with probability $P(d)$;*
- *Pick a latent topic z with probability $P(z|d)$;*
- *Generate a word w with probability $P(w|z)$.*

For example, we use PLSA to analysis the microblogs of every user in Weibo, and w_{ij} represents the probability of user v_i belongs to j^{th} topic.

Based on the above concepts, we can now define the problem of node classification in social network. Given a partially labeled network, the goal is to detect the classes (labels) of all unknown nodes in the network. Formally,

Target 1. ***Node classification in social network:*** *Given a partially labeled network $G = (V_L, V_U, E, Y_L, W)$, the objective is to learn a predictive function*

$$f : G = (V_L, V_U, E, Y_L, W) \rightarrow Y$$

The above formulation make our work very different from existing work on node classification. Macskassy, C. Perlich and Desrosiers[4][11][12] had done some constructive work about relational learning algorithm, but they only concern about the relationships in network. Pennacchiotti[15] tried many features about each node, even treated relationship as some features, but he ignored the network topology.

3 A Factor Graph Model

Actually in social network analysis, there are some works about utilizing graph model. For example, Tang[17] used a factor graph model to infer social ties; Yang[18] proposed a factor graph model too, their works focused on representative-user finding and community-structure discovery. In this section, we explain the factor graph model we use to classify nodes in social network.

3.1 Basic Domain Criterions

For inferring label of nodes in social network, we have three basic criterions from our specified domain knowledge.

First, users from different classes may have different topics while in same class may have similar topics. For example, if two users come from the same company, they may have similar microblogs in Weibo which talk about their works or about their company. Second, the relationships may have a correlation in classification. For example, in DBLP network, if two papers have at least one common author, they probably belong to the same research area (e.g., artificial intelligence, database and so on).

And last, different relationship types may lead to different kinds of effects. For example in Weibo, user A follows user B but B doesnt follow A back, this is a relationship type; user A follows user C and user B also follows C, this is another relationship type between A and B, obviously, these two relationship types are quite different, and of course it will influence the classification model.

3.2 A Factor Graph Model with Partially-Labeled Data

Based on the above observations, we then describe the proposed factor graph model in details.

As we can see in figure 1, it is a small network with 4 nodes v_1, v_2, v_3, v_4 and some relationships between these nodes. Corresponding to the factor graph model, we have 4 hidden vectors y_1, y_2, y_3, y_4 and the nodes attribute factor. To deal with multiple relationship types, we should have multiple relation factors. Actually, we use at most 2 relation factors in our experiments, but in theory, we can use as many relation factors as possible. The definitions of the factors are as follows:

- ***Attribute factor:*** $f(y_i, w_i)$ *represents the posterior probability of the relationship* y_i *given the attribute vector* w_i*;*
- ***Relation factor 1:*** $g(y_i, y_j)$ *denotes a relationship of node* y_i *and* y_j*;*
- ***Relation factor 2:*** $h(y_i, y_j)$ *denotes another relationship of node* y_i *and* y_j*.*

Given a partially-labeled network $G = (V_L, V_U, E, Y_L, W)$, based on the definition of factor graph model, we can have the joint distribution over Y as

$$p(Y|G) = \prod_i f(y_i, w_i) g(y_i, N_1(y_i)) h(y_i, N_2(y_i)) \qquad (1)$$

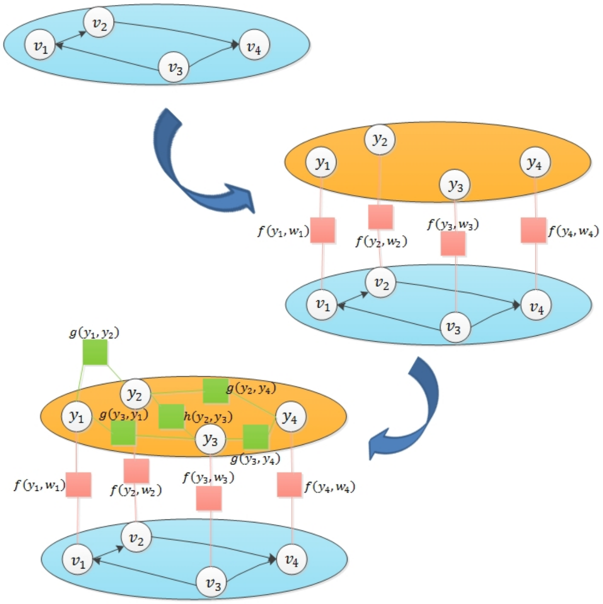


Fig. 1. A Factor Graph Model

where $N_1(y_i)$ and $N_2(y_i)$ are sets of neighbors of y_i. The three factors can be instantiated in different ways. In this work, we use exponential-linear functions. Particularly, we define the attribute factor as

$$f(y_i, w_i) = \frac{1}{Z_\lambda} exp\{\lambda^T \varPhi(y_i, w_i)\} \tag{2}$$

where λ is a weighting vector that will be learned in the model and $\varPhi$ is a vector of feature functions. Similarly, we denote the relation factor as

$$g(y_i, N_1(y_i)) = \frac{1}{Z_\alpha} exp\{\sum_{y_i \in N_1(y_i)} \alpha^T g(y_i, y_j)\} \tag{3}$$

$$h(y_i, N_2(y_i)) = \frac{1}{Z_\beta} exp\{\sum_{y_i \in N_2(y_i)} \beta^T h(y_i, y_j)\} \tag{4}$$

where α and β is similar with λ, g and h can be defined as a vector of indicator functions.

3.3 Model Learning and Parameter Inference

Learning this model is to estimate a series of parameters $\theta = (\lambda, \alpha, \beta)$ and maximize the log-likelihood of observation information (labeled nodes). For simple presentation, we concatenate all factor functions for y_i as

$$s(y_i) = (\varPhi(y_i, w_i)^T, \sum_{y_j} g(y_i, y_j)^T, \sum_{y_j} h(y_i, y_j)^T)^T \tag{5}$$

The joint probability can be written simply as

$$\begin{aligned} p(Y|G) &= \frac{1}{Z}\prod_i exp\{\theta^T s(y_i)\} \\ &= \frac{1}{Z}exp\{\sum_i \theta^T s(y_i)\} \\ &= \frac{1}{Z}exp\{\theta^T S\} \end{aligned} \tag{6}$$

where $Z = Z_\lambda Z_\alpha Z_\beta$ is a normalization factor, and S is the combination of factor functions of all nodes.

Since the input data of this model is partially-labeled, to calculate the normalization factor Z, we need to sum up the likelihood of possible states for all labeled and unlabeled nodes. Naturally, we think of using the labeled data to infer the label of unknown nodes. Then, we have the objective function as

$$\begin{aligned} O(\theta) &= log \sum_{Y|Y^L} \frac{1}{Z}exp\{\theta^T S\} \\ &= log \sum_{Y|Y^L} exp\{\theta^T S\} - logZ \\ &= log \sum_{Y|Y^L} exp\{\theta^T S\} - log\sum_{Y} exp\{\theta^T S\} \end{aligned} \tag{7}$$

Where $Y|Y^L$ denotes inferring label of Y from Y^L. To solve this problem, we use a gradient decent method to calculate the partial derivative of θ

$$\begin{aligned} \frac{\partial O(\theta)}{\partial \theta} &= \frac{\partial(log\sum_{Y|Y^L} exp\{\theta^T S\} - log\sum_Y exp\{\theta^T S\})}{\partial \theta} \\ &= \frac{\sum_{Y|Y^L} exp\{\theta^T S \cdot S}{\sum_{Y|Y^L} exp\{\theta^T S} - \frac{\sum_Y exp\{\theta^T S \cdot S}{\sum_Y exp\{\theta^T S} \\ &= E_{\theta(Y|Y^L,G)}S - E_{\theta(Y,G)}S \end{aligned} \tag{8}$$

Since the social network can be arbitrary graphical structure, our factor graph model may have many circles, so it is intractable to calculate the expectation in a directed and exact way. To alleviate the cost of computation, some kinds of approximate algorithms have been proposed such as LBP (Loopy Belief Propagation)[13] and MCMC (Markov Chain Monte Carlo)[16]. In this work, we utilize LBP to calculate the marginal probabilities.

LBP is simply to apply the sum-product algorithm even though there is no guarantee that it will yield good results[2]. It is possible because the message passing rules for the sum-product algorithm are purely local. However, because the graph now has cycles, information can flow many times around the graph. For some models, the algorithm will converge, whereas for others it will not.

After completing the calculation of the marginal probabilities, the gradient can be obtained by summing over all nodes, and then we update each parameter with a learning rate η and the gradient.

3.4 Infer Unknown-Label Nodes and Classify

Finally, we can infer the category of unknown-label node. Based on the learned parameters θ, we can predict the label of each node by finding a label configuration which maximizes the joint probability as

$$\tilde{Y} = argmax_{Y|Y^L} p(Y|G) \tag{9}$$

In the same way, we use LBP to calculate the marginal probability of each node $p(y_i|Y^L, G)$, then we can put one node into the class which has the maximum marginal probability. In other words, the marginal probability is taken as the prediction confidence.

4 Experiment Results

The model we use to classify nodes is general and can be used in many different situations. In this section, we present our experiment results on two different data sets with multiple tasks to evaluate the effectiveness of our model.

4.1 Data Sets

DBLP Coauthor Network. This benchmark data set contains more than 50000 papers published at 22 computer science conferences from 2008 to 2010. These conferences can be mainly divided into five research areas:

- AI: artificial intelligence, including IJCAI, AAAI, ICML, UAI and NIPS;
- DB: database, including EDBT, ICDT, ICDE, PODS, SIGMOD and VLDB;
- DP: distributed and parallel computing, including ICCP, IPDPS and PACT;
- GV: graphics, vision and HCI, including ICCV, CVPR and SIGGRAPH;
- NC: networks, communications and performance, including MOBICOM, INFOCOM, SIGMETRICS and SIGCOMM.

In this data set, the objective is to classify papers into the correct research area. We extract some subsets randomly for three tasks:

- 6000 papers with 3000 papers published in GV (positive set) and 3000 in others (negative set), 25319 edges totally;
- 6000 papers with 3000 papers published in NC (positive set) and 3000 in others (negative set), 18085 edges totally;
- 10000 papers with 5000 papers published in AI (positive set) and 5000 in others (negative set), 48083 edges totally.

Weibo Network. As our previous introduction, Weibo is a very large online social network with 300 million users. We extract some small experiment data from Weibo by crawler. In Weibo data, we have two binary classification tasks as:

- **Company Affiliation:** 600 users with 305 belong to a specific company (positive set) and 295 are not (negative set), 2393 relationships of following others, 2509 relationships of following common users. The positive users explicitly mention their company in tags, descriptions or screen names;

- **Domain Affiliation:** 3231 users with 1580 positive users whose study domain is about internet information technique and other 1651 users are negative with some different domains, 12194 relationships of following others, 2740 of following common users. The users we collect are explicitly mention their domain in tags.

Table 1. statistics of the data sets

Data set	Task	Users	Positive set	Negative set	Relationship (type1+type2)
DBLP	GV	6000	3000	3000	25319
	NC	6000	3000	3000	18085
	AI	10000	5000	5000	48083
Weibo	company	600	305	295	2393+2509
	domain	3231	1580	1651	12194+2740

4.2 Experiment Description and Evaluation

In the DBLP data set, we make each paper as a node, if two papers have at least one common author, we treat this relationship as an edge. In DBLP data, we simply use one relationship type. In Weibo data set, we treat each user as a node, if a user follows another user, we put an edge between them. More in Weibo data, we consider about the relationship of following common users (e.g. two users follow at least 5 common users) and make it as another edge factor.

In both two data sets, we use PLSA to get the topic model of each node as the node features. Actually, we set the number of topics as 20.

To compare our approach with the traditional methods, we carried out the representative algorithms such as wvRN, CDRN, LibSVM on the same data sets:

- **wvRN:** a simply neighbor-voted classifier, consider about the relationship only.
- **CDRN:** similar with wvRN, it uses the probability distribution of neighbors to classify nodes, consider about the relationship only.
- **libSVM:** a traditional classification algorithm, using node features to classify nodes. We use Weka 3.6 to implement libSVM.

To quantitatively evaluate the model we use, we consider three aspects:

- **Results Analysis (Basic Experiment):** we try to prove the effectiveness of our method with some basic experiments.
- **The Influence of Labeled Data Size:** since our method is semi-supervised, we use the different labeled data size to test the robustness and generality of our method.
- **Discussion on Multiple Relationship Types Setting:** we consider about multiple relationship types in our experiments and discover the extendibility of our method.

For the classification performance, we evaluate the approaches in terms of accuracy, precision, recall, and F1-score.

4.3 Accuracy Performance

Basic Experiment. Table 2 lists the accuracy performance of classifying nodes in DBLP data with three tasks by the different methods. All these three tasks, the labeled data is 10% of each subset.

Table 2. performance of node classification with different methods on three tasks in DBLP data

Positive set	Method	Accuracy	Precision	Recall	F1-score
GV	wvRN	0.580	**0.949**	0.469	0.627
	CDRN	0.584	0.935	0.479	0.633
	libSVM	0.860	0.818	0.925	0.868
	Our model	**0.943**	0.921	**0.962**	**0.944**
NC	wvRN	0.522	**0.984**	0.365	0.532
	CDRN	0.523	**0.984**	0.365	0.532
	libSVM	0.754	0.679	0.963	0.796
	Our model	**0.913**	0.891	**0.942**	**0.916**
AI	wvRN	0.448	**0.961**	0.272	0.424
	CDRN	0.451	**0.961**	0.273	0.426
	libSVM	0.834	0.814	0.868	0.840
	Our model	**0.897**	0.889	**0.907**	**0.898**

As we can see from the results, our method consistently outperforms other comparative methods on all the three tasks in DBLP data. In terms of F1-score, our model is 5% – 12% better than libSVM, 30% – 40% better than wvRN and CDRN. We notice that wvRN and CDRN, which are focus on the labels of neighbors, get very high precision scores, even better than our model, but their recall scores are really poor, and their F1-score are much worse than libSVM and our model, so wvRN and CDRN dont classify the nodes well. Actually, these two methods classify most of the nodes into negative set.

Different Size of Labeled Data. We have known that in DBLP data, our model perform much better than others and either wvRN or CDRN has a poor result. One of the reasons maybe the labeled data is only 10%. Based on the above assumption, we have some experiments on company affiliation using Weibo data, the classification objective is estimate whether a user is from a specific company or not. We test the size of labeled data from 10% to 90%, and the results are shown in figure 2.

From the results shown in figure 2, we can find some interesting points. When labeled data are 10%, our model is 11.9% better than libSVM, about 30% better than wvRN and CDRN in F1-score; on accuracy score, our model is also 10% better than others. When labeled data are 50%, on F1-score, our model is still 11.8% better than libSVM, and about 35% better than wvRN and CDRN; on accuracy score, our model is more than 20% than others. When labeled data

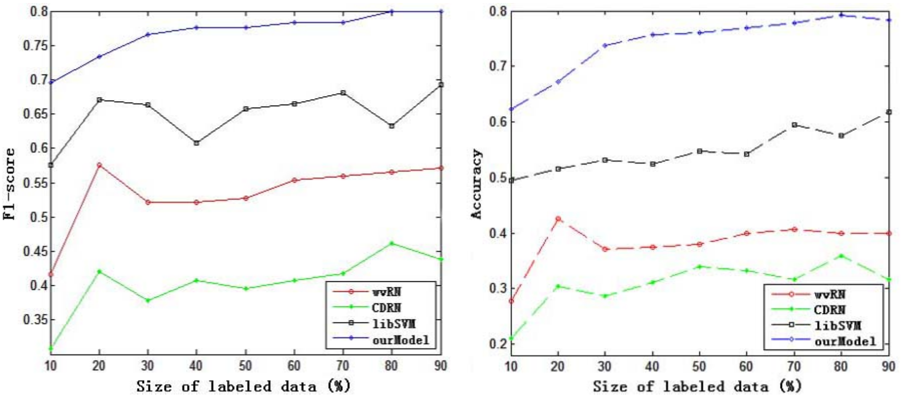


Fig. 2. Performance of node classification with different labeled size and different method on company affiliation using Weibo data

are 90%, our model is 11.7% better than libSVM and more than 25% better than wvRN and CDRN on F1-score; on accuracy score, our model is more than 15% better than others. More notably, either F1 or accuracy, the score of our method increases smoothly and almost monotonously when the size of labeled data increases, while other methods shake obviously. It denotes that our model is much stronger and more effective no matter how many data are labeled.

Multiple Relationship Types. We now evaluate the performance of multiple relationship types. Table 3 shows the result of our experiment on both company affiliation and domain affiliation in Weibo data, where our model(S) denotes using just single relationship type, Our model(M) denotes using multiple relationship types, the size of labeled data is set 10%.

As we can see, our method is much better than other methods, although in domain affiliation, libSVM has 1% better than our model in terms of accuracy

Table 3. performance of node classification with multiple relationship types on two tasks in WEIBO data

Task	method	Accuracy	Precision	Recall	F1-score
company affiliation	wvRN	0.278	0.908	0.270	0.416
	CDRN	0.211	**0.931**	0.185	0.308
	libSVM	0.557	0.560	0.593	0.576
	Our model(S)	0.623	0.590	0.842	0.695
	Our model(M)	**0.624**	0.586	**0.887**	**0.705**
domain affiliation	wvRN	0.189	0.776	0.150	0.251
	CDRN	0.468	0.600	0.590	0.593
	libSVM	**0.618**	**0.815**	0.283	0.420
	Our model(S)	0.590	0.547	**0.951**	0.694
	Our model(M)	0.604	0.556	0.947	**0.701**

score, the F1-score of our model is much higher than others (at least 10%). Considering the multiple relation types, the accuracy scores are improved by 0.1% and 1.4% in two tasks while the F1-scores are improved by 1% and 0.7%. It suggests that multiple edge features do add value to our classification model.

5 Conclusion and Future Work

In this paper, we study the problem of node classification in social network, which is an interesting but challenging research domain. We use a factor graph model with semi-supervised learning to infer the category of unlabeled data. In our model, each node in social network is modeled as variable node and various relationships are modeled as factor nodes. In this way, this model can take the advantages of both node features and graph information. Experiments on the different data sets validate the effectiveness of the model we use. It outperforms both model of pure node features (libSVM) and model of pure relationships (wvRN, CDRN).

Node classification in social network is a potential research direction in social network analysis. As future work, since many networks have multiple categories, extending our method to multi-class classification will be quite useful. Considering it is intractable to simply modify the objective function, we should use some indirect approach such as one-against-one to solve the multi-class classification, for example, if there are k categories, we need $k(k+1)/2$ classifiers and vote for each unlabeled node. In addition, the online social network becomes larger and larger, it is interesting to study some fast but effective method for huge networks.

Acknowledgment. The work was Supported by the Upgrading Plan Project of Shenzhen Key Laboratory (No. CXB201005250038A) and The Shenzhen Key Technology R&D Program (No. ZD201010210080A, No.JCYJ20120619153002947). In addition, we thank the anonymous reviewers for their very valuable comments and suggestions.

References

1. Asur, S., Parthasarathy, S.: A viewpoint-based approach for interaction graph analysis. In: Elder IV, J.F., Fogelman-Soulié, F., Flach, P.A., Zaki, M.J. (eds.) KDD, pp. 79–88. ACM (2009)
2. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks 30(1-7), 107–117 (1998)
4. Desrosiers, C., Karypis, G.: Within-Network Classification Using Local Structure Similarity. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009, Part I. LNCS (LNAI), vol. 5781, pp. 260–275. Springer, Heidelberg (2009)

5. Ebel, H., Mielsch, L.-I., Bornholdt, S.: Scale-free topology of e-mail networks (2002)
6. Gao, B., Liu, T.-Y., Wei, W., Wang, T., Li, H.: Semi-supervised ranking on very large graphs with rich metadata. In: KDD, pp. 96–104 (2011)
7. Heatherly, R., Kantarcioglu, M., Thuraisingham, B.M.: Social network classification incorporating link type values. In: ISI, pp. 19–24. IEEE (2009)
8. Henzinger, M.R., Chang, B.-W., Milch, B., Brin, S.: Query-free news search. In: WWW, pp. 1–10 (2003)
9. Hofmann, T.: Probabilistic latent semantic analysis. In: UAI, pp. 289–296 (1999)
10. Ji, M., Han, J., Danilevsky, M.: Ranking-based classification of heterogeneous information networks. In: KDD, pp. 1298–1306 (2011)
11. Macskassy, S.A., Provost, F.J.: A simple relational classifier. In: Proc. of the 2nd Workshop on Multi-Relational Data Mining, pp. 64–76 (2003)
12. Macskassy, S.A., Provost, F.J.: Classification in networked data: A toolkit and a univariate case study. Journal of Machine Learning Research 8, 935–983 (2007)
13. Murphy, K.P., Weiss, Y., Jordan, M.I.: Loopy belief propagation for approximate inference: An empirical study. In: UAI, pp. 467–475 (1999)
14. Nadakuditi, R.R., Newman, M.E.J.: Graph spectra and the detectability of community structure in networks. CoRR, abs/1205.1813 (2012)
15. Pennacchiotti, M., Popescu, A.-M.: A machine learning approach to twitter user classification. In: Adamic, L.A., Baeza-Yates, R.A., Counts, S. (eds.) ICWSM. The AAAI Press (2011)
16. Spall, J.C.: Estimation via markov chain monte carlo. IEEE Control Systems Magazine 23, 34–45 (2003)
17. Tang, W., Zhuang, H., Tang, J.: Learning to infer social ties in large networks. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part III. LNCS (LNAI), vol. 6913, pp. 381–397. Springer, Heidelberg (2011)
18. Yang, Z., Tang, J., Li, J., Yang, W.: Social community analysis via a factor graph model. IEEE Intelligent Systems 26(3), 58–65 (2011)
19. Zaïane, O.R., Chen, J., Goebel, R.: Mining research communities in bibliographical data. In: Zhang, H., Spiliopoulou, M., Mobasher, B., Giles, C.L., McCallum, A., Nasraoui, O., Srivastava, J., Yen, J. (eds.) WebKDD/SNA-KDD 2007. LNCS, vol. 5439, pp. 59–76. Springer, Heidelberg (2009)

Fast Graph Stream Classification Using Discriminative Clique Hashing

Lianhua Chi[1,2], Bin Li[1], and Xingquan Zhu[1]

[1] Centre for Quantum Computation & Intelligent Systems, FEIT,
University of Technology, Sydney, NSW 2007, Australia
[2] School of Computer Science & Technology,
Huazhong University of Science & Technology, Wuhan 430074, China
{lianhua.chi,bin.li-1,xingquan.zhu}@uts.edu.au

Abstract. As many data mining applications involve networked data with dynamically increasing volumes, graph stream classification has recently extracted significant research interest. The aim of graph stream classification is to learn a discriminative model from a stream of graphs represented by sets of edges on a complex network. In this paper, we propose a fast graph stream classification method using DIscriminative Clique Hashing (DICH). The main idea is to employ a fast algorithm to decompose a compressed graph into a number of cliques to sequentially extract clique-patterns over the graph stream as features. Two random hashing schemes are employed to compress the original edge set of the graph stream and map the unlimitedly increasing clique-patterns onto a fixed-size feature space, respectively. The hashed cliques are used to update an "in-memory" fixed-size pattern-class table, which will be finally used to construct a rule-based classifier. DICH essentially speeds up the discriminative clique-pattern mining process and solves the unlimited clique-pattern expanding problem in graph stream mining. Experimental results on two real-world graph stream data sets demonstrate that DICH can clearly outperform the compared state-of-the-art method in both classification accuracy and training efficiency.

Keywords: Graph classification, graph streams, cliques, hashing.

1 Introduction

The emergence of complex networks has led to a surge of research in graph data mining [1]. Graph classification is an important graph data mining task that aims to learn a discriminative model from training examples to predict class labels of test examples, where both training and test examples are graphs. Many real-world applications involve graph-represented data, such as chemical compounds, XML documents, and program flows. The essential challenge for graph classification is to extract features from graphs and represent graph data in instance-feature format to support model training. A variety of studies on substructure extraction (e.g., walks [2], paths [3], and subtrees [4,5]) for describing graphs have been proposed in the past decade. However, most of them only

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 225–236, 2013.

consider the learning problem of graph classification in batch mode (all data are available for training), which limits their applicability to large-scale and stream scenarios.

Due to the streaming nature of many real-world complex networks, such as social networks and sensor networks, graph stream classification has recently attracted increasing research interest [6,7,8,9]. Graph stream classification is defined on a complex network which comprises a massive universe of nodes, where the stream of graphs are represented as sets of edges on the underlying network. For example, co-authorships of research works continuously form graphs on a coauthor network (e.g., DBLP), dynamic communities of interest continuously form graphs on a social network (e.g., Facebook), and traffic flows continuously form graphs on a transportation network. Graph stream classification on a complex network with massive nodes is challenging, because

- **Subgraph Feature Generation:** Graph stream is defined on a massive universe of nodes, enumerating subgraph-patterns from such a large node set as features is time consuming and memory intensive. We need fast and inexpensive feature generation method for graph stream classification.
- **Increasing Stream Volumes:** The volumes of graph data are continuously growing, so graph streams can usually be accessed only once. Graph stream classification must be able to tackle dynamically increasing graph volumes and generate discriminative model with high speed.
- **Changing Feature Distributions:** The marginal distributions of subgraph-patterns (features) may continuously change over the graph stream (i.e., the concept-drift problem [10]), a dynamic updating scheme is required to update the discriminative model.

Few studies have investigated the graph stream classification problem. To the best of our knowledge, only two works [9,11] may be applied to the considered problem. Both of them employ hashing techniques to sketch the graph stream for saving computational cost and controlling the size of the subgraph-pattern set. In [11], the authors proposed a hash kernel to project arbitrary graphs onto a compatible feature space for similarity computing, but it can only be applied to node-attributed graphs. Recently, Aggarwal [9] proposed a 2-D hashing scheme to construct an "in-memory" summary for sequentially presented graphs and used a simple heuristic to select a set of most discriminative frequent patterns to build a rule-based classifier. Although [9] has exhibited promising performance on graph stream classification, there are two inherent limitations: (1) The selected subgraph-patterns are composed with disconnected edges, which may have less discriminative capability than connected subgraph-patterns due to a lack of semantic meaning. (2) The computational cost is high because an additional frequent pattern mining procedure is required to perform on the summary table which comprises massive transactions.

In this paper, we propose a fast graph stream classification method using DIscriminative Clique Hashing (DICH) to address the aforementioned challenges. The main idea is to decompose a compressed graph into a number of cliques (fully connected subgraphs) to sequentially extract clique-patterns over the graph stream as features. Two random hashing schemes are employed to compress the original edge set of the graph stream and map the unlimitedly increasing clique-patterns onto a fixed-size feature space, respectively. The hashed cliques are then used to update an "in-memory" fixed-size pattern-class table, which will be finally used to generate a rule-based classifier. *Since DICH adopts connected subgraphs as features and needs no additional frequent pattern mining procedure, it can achieve very fast training speed for graph stream classification.* The experimental results on two real-world graph stream data sets clearly show that DICH outperforms the compared state-of-the-art [9] in both classification accuracy and training efficiency.

The remainder of this paper is organized as follows: Section 2 introduces the related work. The proposed framework for graph stream classification will be described in Section 3 and the detailed method of DICH will be presented in Section 4. We empirically evaluate our method to show its effectiveness and efficiency in Section 5 and conclude the paper in Section 6.

2 Related Work

The considered problem is closely related to graph classification. Most existing works focus on designing effective yet efficient kernels for measuring graph similarity. A large number of graph kernels have been proposed in the last decade, most of which are based on the similar idea of extracting substructures from graphs to compare their co-occurrences. Typical substructures for describing graphs include walks [2,12], paths [3], subtrees [4,5,13], and subgraphs (usually based on a frequent subgraph mining technique, e.g., [14]). In this paper, we extract cliques as features for describing a graph.

Our problem is also related to data stream mining. Mining high-speed data streams was first studied in [15] and the idea of using ensemble learning for data stream classification was proposed soon after [16]. A classical ensemble learning framework for addressing the concept-drift problem in data stream mining was proposed in [10]. In our graph stream classification problem, we employ a rule-based classification approach rather than the ensemble learning framework for faster processing speed and easier model updating.

The most relevant work to this paper is [9], which also considers graph stream classification on a complex network. It employs a 2-D hashing scheme to construct an "in-memory" summary for the sequentially presented graphs. The first random-hash scheme is used to reduce the size of the edge set. The second min-hash scheme is used to dynamically update a number of hash-codes (i.e., corresponding to random sorting samples), which is able to summarize the frequent patterns of co-occurrence edges in the graph stream observed thus far. Finally, a simple heuristic is used to select a set of most discriminative frequent patterns to build a rule-based classifier. In this paper, we propose a clique-based hashing scheme for solving the same problem with a better performance in both classification accuracy and training efficiency as well as avoiding the the limitations of [9] discussed in Section 1.

3 Framework

We first introduce the problem setting for graph stream classification. Suppose there is a complex network which comprises a massive universe of nodes. The edges connecting these nodes are denoted by the edge set $\mathcal{E}$. The stream of graphs $\{G_1, G_2, \ldots, G_i, \ldots\}$ are presented continuously as subsets of $\mathcal{E}$, where the subscript i denotes the receiving order in the graph stream. In particular, the edge set of G_i are denoted by $\{E_1, \ldots, E_e\} \subset \mathcal{E}$, where e denotes the number of edges in G_i. Each graph G_i has a class label $L_i \in \{1, \ldots, M\}$. We assume G_i is received in the form $\langle i, E_1, \ldots, E_e, L_i \rangle$. In this paper, we assume that each edge has a default weight 1 for simplicity. The underlying graph stream can only be accessed once and our **goal** is to learn a discriminative model from $\{G_1, G_2, \ldots, G_i, \ldots\}$ at a high efficiency to accurately predict the class label of a test graph G_{test} in the future graph stream.

We next give an overview of DICH for graph stream classification. The corresponding framework, illustrated in Figure 1, comprises three modules. The graphs in the stream are received and processed one by one. The first module is for clique detection from each graph in the stream. The incoming edges of G_i are first randomly hashed to a compressed edge set and then we adopt a fast algorithm to decompose the compressed graph into a number of cliques (fully connected subgraphs) as the features of G_i. Since the number of clique-patterns will unlimitedly increase as new graphs are fed in, the underlying feature space will keep expanding accordingly. Thus, in the second module, a clique hashing scheme is performed to map the unlimitedly emerged clique-patterns onto a fixed-size clique-pattern set. In the last module, an “in-memory” fixed-size pattern-class table is updated using the clique-pattern and class label information of G_i; and a rule-based classifier is constructed based on the pattern-class table by identifying frequent and discriminative clique-patterns associated to each class. To test a graph G_{test} in the future graph stream, G_{test} is processed in the first two modules and the obtained hashed clique-patterns are input to the rule-based classifier for class label prediction. The detailed approaches to the three modules are described in the following section.

4 Graph Stream Classification

4.1 Graph Clique Detection

As shown in Figure 1, instead of relying on expensive frequent subgraph mining to discover graph features, we propose to use frequent and discriminative clique-patterns for clique-based classifier construction. So our first step is to detect all the cliques from each graph in the graph stream. Since the edge set of the graph stream on the complex network can be extremely large, it is necessary to sketch the graph stream as a preprocessing step. In particular, we use a random hash function to map the original edge set $\mathcal{E}$ onto a significantly compressed edge set $\bar{\mathcal{E}}$ of size N. That is to say, the edges in the compressed graph of G_i will be indexed by $\{1, \ldots, N\}$. If multiple edges in G_i are hashed onto the same index, the weight

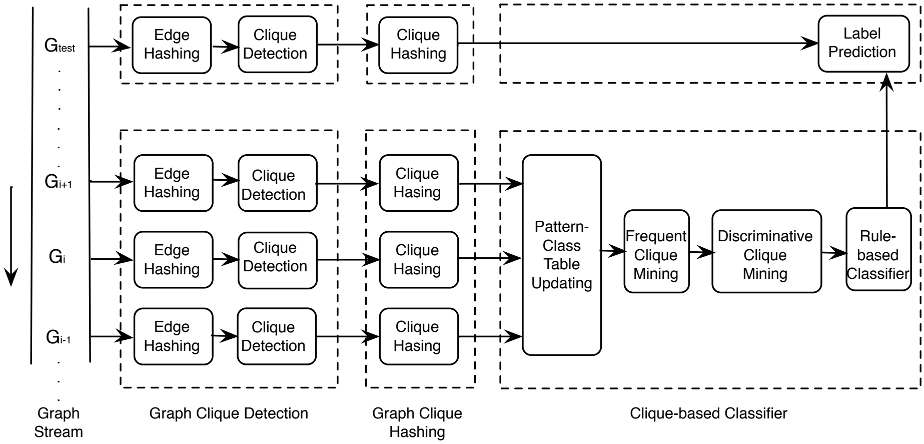


Fig. 1. The framework of DICH for graph stream classification

Algorithm 1. Clique Detection

```
Clique-Detect(): Detect the clique set $C_i$ from graph $G_i$.
begin
  $\bar{G}_i$ := Edge-Hash$(G_i, N)$;
  $C_i := \emptyset$;
  for $t := \max(\bar{G}_i)$ to $\min(\bar{G}_i)$ step 1 do
    $\bar{G}_i^{(t)} := \mathbf{1}(\bar{G}_i \geq t)$;
    $C_i^{(t)}$ := Bron-Kerbosch$(\bar{G}_i^{(t)})$;
    $C_i := C_i \bigcup C_i^{(t)}$;
  od
end
```

of the compressed edge is set to the number of the edges that get the same index. After hashing all the edges in G_i, we obtain a compressed graph $\bar{G}_i$ for clique detection. We define this edge hashing scheme using $\bar{G}_i :=$ Edge-Hash(G_i, N). The leftmost two columns in Figure 2 illustrates this procedure.

Next we will employ a fast algorithm to detect cliques in each compressed graph. We adapt the graphlet basis estimation algorithm used in [17] to this end. The first step for clique detection is to threshold the compressed graph $\bar{G}_i$ at a number of weight levels in descending order, say $t = \{\max(\bar{G}_i), \ldots, \min(\bar{G}_i)\}$, where $\max(\bar{G}_i)$ and $\min(\bar{G}_i)$ denote the largest and the smallest edge weights in $\bar{G}_i$, respectively. We define this operation using $\bar{G}_i^{(t)} := \mathbf{1}(\bar{G}_i \geq t)$, where $\mathbf{1}(\cdot)$ denotes the indicator function and the resulting graph is denoted by $\bar{G}_i^{(t)}$. We use the Bron-Kerbosch algorithm [18] to identify all the cliques from $\bar{G}_i^{(t)}$ at each threshold. The union set of the cliques found in $\{\bar{G}_i^{(t)}\}_{t=\min(\bar{G}_i)}^{\max(\bar{G}_i)}$ is represented as the clique set for G_i. This procedure is detailed in Algorithm 1.

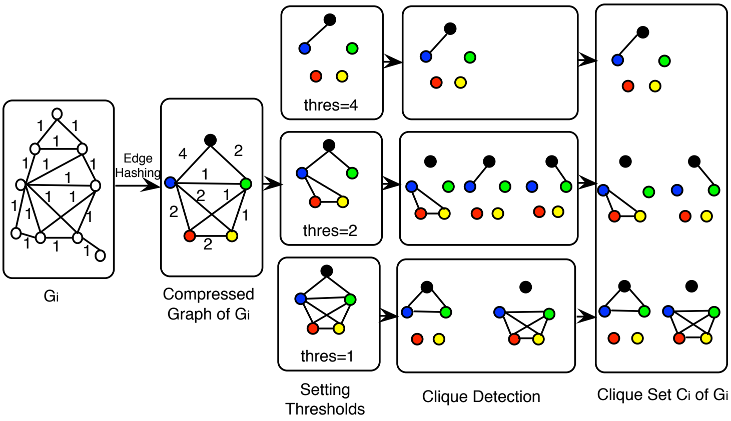


Fig. 2. Clique detection in a compressed graph

An example of clique detection is illustrated in Figure 2. After edge hashing, we obtain the compressed graph of G_i (2nd column). Then, four weight thresholds $\{4, 3, 2, 1\}$ are set to generate three graphs $\{\bar{G}_i^{(1)}, \bar{G}_i^{(2)}, \bar{G}_i^{(4)}\}$ (3rd column). Note that $\bar{G}_i^{(3)}$ is an empty graph which is not shown. The Bron-Kerbosch algorithm is applied to the three graphs and detect a set of cliques from each graph (4th column). Finally, the cliques detected at all weight thresholds are merged to form the clique set C_i for G_i as its feature representation (5th column).

4.2 Graph Clique Hashing

The cliques extracted from each graph are used to represent its features. To learn a classifier from the graph stream, it is required to make the features of all graphs be in the same feature space. In other words, we should count the occurrences of the same set of clique-patterns in all graphs in the stream. Since the number of clique-patterns will increase as new graphs are continuously fed in, the induced feature space will keep expanding accordingly. To address this problem, we adopt a feature hashing scheme used in [11] to randomly map the unlimitedly emerged clique-patterns onto a fixed-size set. In particular, we use an "in-memory" $P \times M$ pattern-class table Δ, which can be dynamically updated, to count clique-pattern and class label information from the graph stream. In Δ, P rows correspond to the indices of hashed clique-patterns while M columns correspond to all the classes of the graphs.

Given G_i in the graph stream, we first use Algorithm 1 to collect the clique set C_i. Then, for each clique in C_i, say $C_{i,j}$, we apply a random hash function $\hbar(\cdot)$ to the string of ordered edges in $C_{i,j}$ to generate an index $H_{i,j} \in \{1, \ldots, P\}$. If a clique with class label L_i is hashed to an index $H_{i,j}$, we add 1 to the entry $\Delta[H_{i,j}, L_i]$, which means clique-pattern $C_{i,j}$ has a contribution to class L_i. This fixed-size pattern-class table is continuously updated as new cliques are detected over the graph stream. This procedure is detailed in Algorithm 2.

Algorithm 2. Clique Hashing

```
Clique-Hash(): Hash the cliques in $G_i$ and update the pattern-class table.
begin
  for $j := 1$ to size$(C_i)$ step 1 do
      $H_{i,j} := \hbar(C_{i,j})$;
      $\varDelta[H_{i,j}, L_i] := \varDelta[H_{i,j}, L_i] + 1$;
  od
end
```

4.3 Clique-Based Classifier

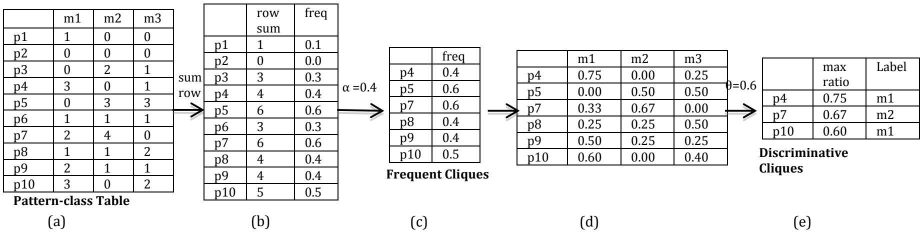
Given the "in-memory" pattern-class table $\varDelta$, we can construct a rule-based classifier by identifying frequent yet discriminative clique-patterns from $\varDelta$. To identify frequent clique-patterns, we first sum up the counts in each row of $\varDelta$ and divide them by the number of graphs received thus far. The result for each row indicates the occurrence frequency of a set of cliques with the same hash value in the graph stream. Then we sort them in a descending order and set a threshold parameter α to select the clique-patterns whose frequencies $\geq \alpha$. These selected cliques are frequent clique-patterns which are also the candidates for the subsequent discriminative clique-pattern selection.

Next we can determine whether a frequent clique-pattern is also a discriminative one by comparing its occurrence ratios on the M classes (corresponding to the M columns in $\varDelta$). For a candidate clique-pattern, the ratio in column j represents the probability that the clique-pattern belongs to class j. A higher probability on a certain class indicates a better discriminative capability. Similarly, we can set a threshold parameter θ to select the clique-patterns whose maximum ratios $\geq \theta$. Figure 3 gives a toy example for selecting the frequent and discriminative clique-patterns from a pattern-class table $\varDelta$.

Finally, based on the selected clique-patterns, we can classify a test graph using majority voting based on the detected cliques in the test graph. In particular, given a test graph G_{test}, we detect its cliques C_{test} using Algorithm 1 and hash its cliques $\{H_{test,j}\}_{j=1}^{\text{size}(C_{test})}$ to index its clique-patterns. Each clique corresponding to a discriminative clique-pattern will contributes a class label $L_{test,j} :=$ Find-Rule$(\varDelta, H_{test,j})$. The class label of the test graph L_{test} is determined by the majority of class labels $\{L_{test,j}\}_{j=1}^{\text{size}(C_{test})}$. This procedure is detailed in Algorithm 3.

5 Experimental Results

In this section, we will test the proposed DICH method for graph stream classification on two real-world data sets. In particular, we will evaluate the effectiveness and efficiency of DICH by comparing it with the 2-D hash compressed stream classifier [9], which is the only state-of-the-art method applicable to graph stream classification. We use the following data sets in our experiments.



	m1	m2	m3
p1	1	0	0
p2	0	0	0
p3	0	2	1
p4	3	0	1
p5	0	3	3
p6	1	1	1
p7	2	4	0
p8	1	1	2
p9	2	1	1
p10	3	0	2

Pattern-class Table

(a)

	row sum	freq
p1	1	0.1
p2	0	0.0
p3	3	0.3
p4	4	0.4
p5	6	0.6
p6	3	0.3
p7	6	0.6
p8	4	0.4
p9	4	0.4
p10	5	0.5

(b)

	freq
p4	0.4
p5	0.6
p7	0.6
p8	0.4
p9	0.4
p10	0.5

Frequent Cliques

(c)

	m1	m2	m3
p4	0.75	0.00	0.25
p5	0.00	0.50	0.50
p7	0.33	0.67	0.00
p8	0.25	0.25	0.50
p9	0.50	0.25	0.25
p10	0.60	0.00	0.40

(d)

	max ratio	Label
p4	0.75	m1
p7	0.67	m2
p10	0.60	m1

Discriminative Cliques

(e)

Fig. 3. A toy example of frequent and discriminative clique-pattern mining. (a) A pattern-class table with 10 clique-patterns $\{p_1, \ldots, p_{10}\}$ and 3 classes $\{m_1, m_2, m_3\}$. (b) The sums of individual clique-patterns in rows and the corresponding occurrence frequencies (i.e., the sums divided by the number of graphs, say 10 here). (c) The selected frequent clique-patterns whose frequencies are larger than the frequent pattern threshold $\alpha = 0.4$. (d) The occurrence ratios of the selected clique-patterns on the M classes. (e) The selected discriminative clique-patterns whose maximum ratios are larger than the discriminative pattern threshold $\theta = 0.6$.

Algorithm 3. Graph Classification

```
Graph-Classify(): Predict the class label of a test graph $G_{test}$ in the stream.
begin
  $C_{test}$ = Clique-Detect($G_{test}$);
  for $j := 1$ to size($C_{test}$) step 1 do
    $H_{test,j} := \hbar(C_{test,j})$;
    $L_{test,j}$ := Find-Rule($\Delta, H_{test,j}$);
  od
  $L_{test}$ := Majority-Vote($\{L_{test,j}\}_{j=1}^{\text{size}(C_{test})}$);
end
```

- DBLP Data Set[1]: In this data set, authors are nodes and co-authorship forms edges, and a graph is constituted by the co-authors of a paper. There are three classes in the data set: 1) Database related conferences, 2) Data mining related conferences, and 3) All remaining conferences. Our goal is to classify a test paper into one of three classes. The final data set contains over 5×10^5 authors, 9.75×10^5 edges, and 3.55×10^5 different graphs as the training data. We divide the data set into five splits and choose four splits as the training data and the remaining split as the test data.
- IBM Sensor Data Set[2]: This data set records the information from local traffic constituted by each graph on a sensor network. The IP-addresses are nodes and local traffic flows are edges. Each graph is associated with a particular intrusion type and there are over 300 different intrusion types (classes) in the data set. Our goal is to classify a traffic flow pattern into one

[1] http://www.charuaggarwal.net/dblpcl/

[2] http://www.charuaggarwal.net/sens1/gstream.txt

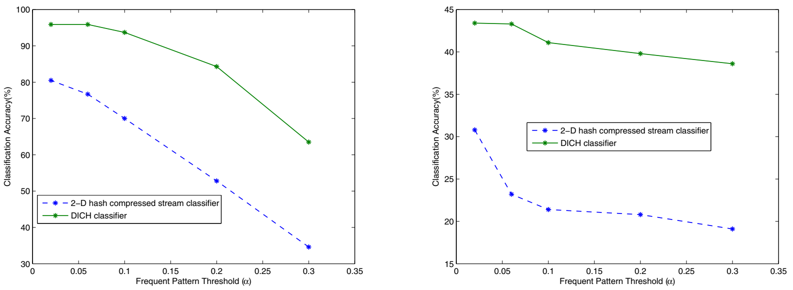


Fig. 4. Effectiveness evaluation w.r.t. α on the DBLP data set (left) and the IBM sensor data set (right), $N = 5000$ and $\theta = 0.4$

of intrusion types. Because the number of classes is extremely large (> 300), we expect the overall accuracy to be relatively low. The data set contains more than 1.57×10^6 edges. We choose 90% of the data as the training data and the remaining 10% as the test data.

5.1 Effectiveness Evaluation

In this experiment, we evaluate the effectiveness of DICH by comparing it with the 2-D hash compressed stream classifier proposed in [9]. We will investigate the classification performance and sensitivity of the two methods by varying 1) the frequent pattern threshold α, 2) the discriminative pattern threshold θ, and 3) the size of the compressed edge set N.

First, we adjust the frequent pattern threshold[3] α for performance evaluation and fix the other two parameters by setting $N = 5000$ and $\theta = 0.4$. Figure 4 plots the classification accuracy curves (y-axis) w.r.t. α (x-axis) on the two data sets. We can see that the classification performance of DICH is much higher than the 2-D hash compressed stream classifier on both data sets and in all values of α. The performance of both classifiers trends to decline as α becomes larger since more graph features will be eliminated and such information loss will affect classification performance. By comparing the curve slopes of two classifiers, the 2-D hash compressed stream classifier is more sensitive to α. In the case of $\alpha = 0.3$, the classification accuracy of the 2-D hash compressed stream classifier is much lower. From this experiment, we can validate the effectiveness of DICH, which can clearly outperform the 2-D hash compressed stream classifier and is more insensitive to the frequent pattern threshold.

Second, we adjust the discriminative pattern threshold θ for performance evaluation and fix the other two parameters by setting $N = 5000$ and $\alpha = 0.05$. Figure 5 plots the classification accuracy curves (y-axis) w.r.t. θ (x-axis) on the two data sets. On the DBLP data set, DICH was significantly superior to the 2-D hash compressed classifier in classification accuracy. The classification performance of both methods was insensitive to θ, which may be due to the fact

[3] In [9], the frequent pattern threshold α is used to screen out subgraph patterns.

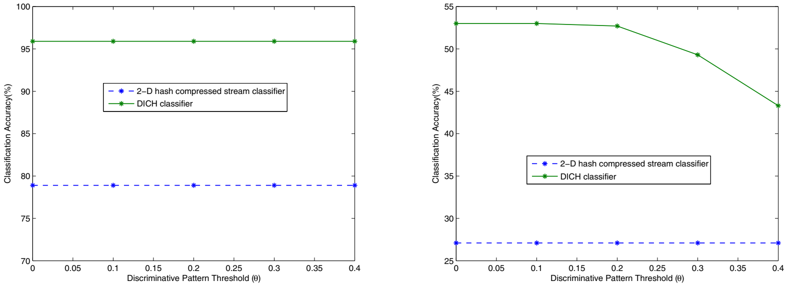


Fig. 5. Effectiveness evaluation w.r.t. θ on the DBLP data set (left) and the IBM sensor data set (right), $N = 5000$ and $\alpha = 0.05$

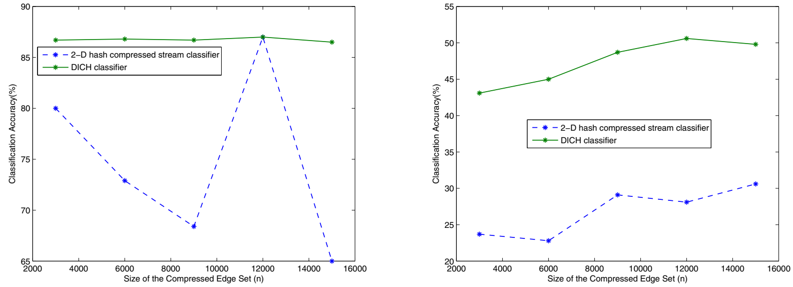


Fig. 6. Effectiveness evaluation w.r.t. the edges compression size N on the DBLP data set (left) and the IBM sensor data set (right), $\alpha = 0.06$ and $\theta = 0.3$

that the two classes (Database related conferences and Data mining related conferences) in DBLP data are extremely rare, the identified frequent patterns have already had relatively high discriminative capability. On the IBM sensor data set, although DICH is somewhat more sensitive to θ than the 2-D hash compressed classifier, it has much higher classification accuracy in all cases. This experiment further demonstrates that DICH has higher effectiveness than the 2-D hash compressed classifier in terms of the discriminative pattern threshold.

Third, we adjust the size of the compressed edge set N for performance evaluation and fix the other two parameters by setting $\alpha = 0.06$ and $\theta = 0.3$. Intuitively, the classification performance of both classifiers will increase at the expense of more space. Figure 6 plots the classification accuracy curves (y-axis) w.r.t. N (x-axis) on the two data sets. We can see that the 2-D hash compressed classifier is very sensitive to N, especially on the DBLP data set; while DICH is more steady on the DBLP data set but no clear performance improvement can be observed as N becomes larger. On the IBM sensor data set, the performance of both classifiers is improved as N becomes larger. Again, DICH outperforms the 2-D hash compressed classifier in all cases.

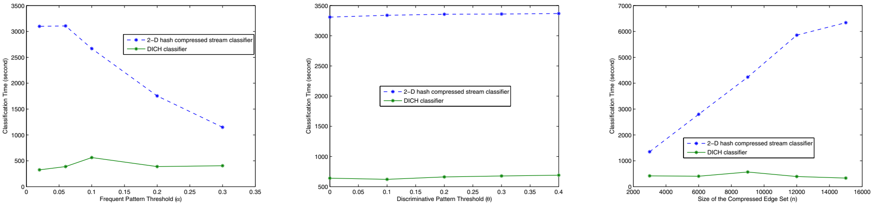


Fig. 7. Efficiency evaluation (1) w.r.t. α, $N = 5000$ and $\theta = 0.4$ (left); (2) w.r.t. θ, $N = 5000$ and $\alpha = 0.05$ (middle); and (3) w.r.t. N, $\alpha = 0.06$, $\theta = 0.3$ (right), on the DBLP data set

5.2 Efficiency Evaluation

In this experiment, we evaluate the efficiency of the two compared methods on the DBLP data set by adjusting the frequent pattern threshold α, the discriminative pattern threshold θ, and the size of the compressed edge set N. The settings of these parameters are the same as those in the above effectiveness evaluation. All the experiments are conducted on a Linux Cluster which comprises 24 nodes with 3.33GHz Intel Xeon CPU (64bit). Both DICH and the 2-D hash compressed stream classifier are implemented using R studio.

Figure 7 plots the training time curves of the two compared methods w.r.t. α, θ, and N on the DBLP data set. We can see that the training time of DICH is significantly less than the compressed hash-based classifier in all cases. The computational cost of the 2-D hash compressed classifier is much higher because it requires an additional frequent pattern mining procedure to perform on the edge co-occurrence table which comprises massive transactions. In contrast, DICH employs a fast clique detection algorithm, which can directly find cliques (connected subgraphs) from the graph stream as features for classifier construction, such that no additional frequent pattern mining procedure is required to find connected subgraph patterns. This experiment shows that DICH clearly outperforms the 2-D hash compressed classifier in not only classification accuracy but also training efficiency.

6 Conclusion

In this paper, we propose a fast graph stream classification method using DIscriminative Clique Hashing (DICH). The main idea is to employ a fast algorithm to decompose a compressed graph into a number of cliques to sequentially extract clique-patterns over the graph stream as features. Two random hashing schemes are employed to speed up the discriminative clique-pattern mining process and address the unlimitedly clique-pattern expanding problem. The hashed cliques are used to update an "in-memory" fixed-size pattern-class table, which is finally used to construct a rule-based classifier. We test DICH on two real-world graph stream data sets. Because DICH directly extracts cliques (connected subgraphs) from the graph stream as features for classifier training, rather than mining unconnected co-occurrence edge sets as that in the compared state-of-the-art method [9], DICH can significantly outperform [9] in both classification accuracy and learning efficiency.

Acknowledgements. This work was supported in part by Australian Research Council (ARC) Discovery Project DP1093762, Australian Research Council (ARC) Future Fellowship FT100100971, and a UTS Early Career Researcher Grant.

References

1. Aggarwal, C.C., Wang, H.: Managing and Mining Graph Data. Springer, New York (2010)
2. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: ICML, pp. 321–328 (2003)
3. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: ICDM, pp. 74–81 (2005)
4. Mahé, P., Vert, J.P.: Graph kernels based on tree patterns for molecules. Machine Learning 75(1), 3–35 (2009)
5. Shervashidze, N., Borgwardt, K.: Fast subtree kernels on graphs. In: NIPS, pp. 1660–1668 (2009)
6. Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., Zhang, J.: Graph distances in the data-stream model. SIAM Journal on Computing 38(5), 1709–1727 (2008)
7. Aggarwal, C.C., Zhao, Y., Yu, P.S.: On clustering graph streams. In: SDM, pp. 478–489 (2010)
8. Aggarwal, C.C., Li, Y., Yu, P.S., Jin, R.: On dense pattern mining in graph streams. In: PVLDB, pp. 975–984 (2010)
9. Aggarwal, C.C.: On classification of graph streams. In: SDM, pp. 652–663 (2011)
10. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: KDD, pp. 226–235 (2003)
11. Li, B., Zhu, X., Chi, L., Zhang, C.: Nested subtree hash kernels for large-scale graph classification over streams. In: ICDM, pp. 399–408 (2012)
12. Vishwanathan, S., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph kernels. Journal of Machine Learning Research 11, 1201–1242 (2010)
13. Hido, S., Kashima, H.: A linear-time graph kernel. In: ICDM, pp. 179–188 (2009)
14. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: ICDM, pp. 721–724 (2002)
15. Domingos, P., Hulten, G.: Mining high speed data streams. In: KDD, pp. 71–80 (2000)
16. Street, W.N., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: KDD, pp. 377–382 (2001)
17. Soufiani, H.A., Airoldi, E.: Graphlet decomposition of a weighted network. Journal of Machine Learning Research – Proceedings Track 22, 54–63 (2012)
18. Bron, C., Kerbosch, J.: Algorithm 457: Finding all cliques of an undirected graph. Communications of the ACM 16(9), 575–577 (1973)

Mining Interesting Itemsets in Graph Datasets

Boris Cule, Bart Goethals, and Tayena Hendrickx

Department of Mathematics and Computer Science
University of Antwerp
{firstname.lastname}@ua.ac.be

Abstract. Traditionally, pattern discovery in graphs has been mostly limited to searching for frequent subgraphs, reoccurring patterns within which nodes with certain labels are frequently interconnected in exactly the same way. We relax this requirement by claiming that a set of labels is interesting if they often occur in each other's vicinity, but not necessarily always interconnected by exactly the same structures. Searching for such itemsets can be done both in datasets consisting of one large graph, and in datasets consisting of many graphs. We present novel methods dealing with both possible settings. Our algorithms benefit from avoiding computationally costly isomorphism checks typical for subgraph mining, as well as from a greatly reduced search space, as we consider only itemsets, and not all possible edges and paths that can connect them.

1 Introduction

Traditionally, searching for patterns in graphs has been almost exclusively limited to searching for frequent subgraphs. A frequent subgraph is a structure within a graph where nodes with certain labels are frequently interconnected in exactly the same way. We propose to relax this requirement. We claim that a pattern, a set of node labels, is interesting if these labels often occur in the graph near each other, not necessarily with exactly the same edges between them.

Consider the graph given in Fig. 1. It can be very easily observed that pattern *abcd* clearly stands out. However, traditional approaches, even with a frequency threshold as low as 2, will not find a single subgraph of size larger than 2, since nodes labelled *a*, *b*, *c* and *d* are always interconnected differently. This demonstrates the need for a new approach.

On top of being more flexible, and thus capable of finding previously undetected patterns, our method also greatly reduces the search space by looking for itemsets alone, rather than considering all possible combinations of edges and paths connecting them. However, if the dataset does contain a frequent subgraph, our method will find the equivalent itemset consisting of the same labels, but without the edges connecting them.

We consider two different problem settings, as the dataset can consist either of a (very large) single graph, or of a set of (smaller) graphs. In the single graph setting, the goal is to find itemsets that reoccur within the graph. We propose two methods to achieve this goal. The first is based on the traditional approaches to

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 237–248, 2013.

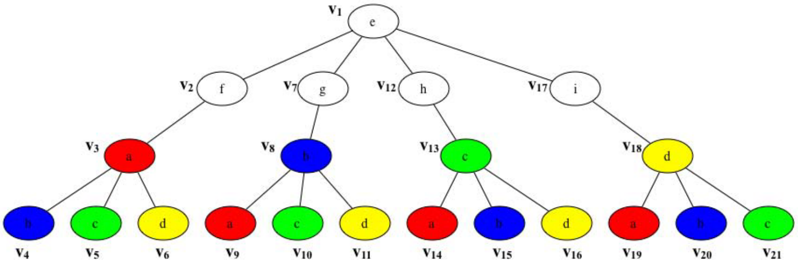


Fig. 1. A graph containing a pattern not discovered by subgraph mining

mining frequent itemsets in transaction databases. Our main contribution here consists of a way to transform a graph into a transaction database, after which any existing itemset mining algorithm can be used to find the frequent itemsets. In our second approach, we look for *cohesive* itemsets, whereby we insist that for an itemset to be considered interesting, it should not only appear often, but its items should also never appear far from each other. In the multiple graph setting, the goal is to find itemsets that occur in many of the input graphs. Here, the dataset cannot be transformed into a typical transaction database. However, the cohesive itemset approach proves perfectly adaptable to this setting.

An interesting property of the cohesive itemset approach is that the proposed interestingness measure is not anti-monotonic, which, at first glance, might represent an obstacle when generating large candidate itemsets. However, in both the single and the multiple graph setting we managed to come up with an efficient pruning method, which enables us to mine cohesive itemsets in a reasonable amount of time.

Even though the problem setting of subgraph mining is very different to that of mining itemsets in graphs, and therefore mostly incomparable, we do note that our algorithm results in a massive reduction in output.

The paper is organised as follows. In Section 2, we present the main related work. In Section 3 we propose two methods of identifying interesting itemsets in the single graph setting. In Section 4, we extend these approaches to be able to handle datasets consisting of multiple graphs. In Section 5, we give a sketch of our algorithms, before presenting the results of our experiments in Section 6. We end the paper with our conclusions in Section 7.

2 Related Work

The problem of discovering subgraphs that occur frequently in a dataset consisting of either one or multiple graphs has been very popular in data mining research. A good survey of the early graph based data mining methods is given by Washio and Motoda [18]. The first studies to find subgraph patterns were conducted by Cook and Holder [3] for a single graph, and by Motoda and Indurkhya [21] for multiple graphs, both using a greedy scheme to find some of the most prevalent subgraphs. Although this greedy search may miss some significant subgraphs, it avoids the high complexity of the graph isomorphism problem.

Inokuchi et al. [9] and Kuramochi and Karypis [13] proposed the AGM and FSG algorithms, respectively, for mining all frequent subgraphs, using a level-wise approach, similar to that of APRIORI [1]. They suffer from two additional drawbacks, however: costly subgraph isomorphism testing and an enormous number of candidates that are generated along the way (due to the fact that both edges and nodes must be added to the pattern). Yan and Han [19] proposed GSPAN, which performs a depth-first search, based on a pattern growth principle similar to the one used in FP-GROWTH [7] for mining itemsets. Nijssen et al. proposed a more efficient frequent subgraph mining tool, called GASTON, which finds the frequent substructures in a number of phases of increasing complexity [15]. More specifically, it first searches for frequent paths, then frequent free trees and finally cyclic graphs. Further attempts at mining frequent subgraphs, both in one or many input graphs, have been made by Bringmann and Nijssen [2], Kuramochi and Karypis [14], Huan et al. [8], Yan et al.[20] and Inokuchi et al. [10].

Up to now, however, most of the research in graph mining has gone into finding frequent subgraphs. We focus on mining interesting itemsets in graphs, thus relaxing the underlying structure of the pattern. By doing so, we avoid the costly isomorphism testing, and by using a depth-first search algorithm, we avoid the pitfalls of APRIORI-like algorithms. A similar approach has been proposed by Khan et al. [12], where nodes propagate their labels to other nodes in the neighbourhood, according to given probabilities. Labels are thus aggregated, and can be mined as itemsets in the resulting graph. Khan et al. also propose to solve the problem of query-based search in a graph using a similar method [11]. Silva et al. [16, 17] and Guan et al. [6] introduce methods to identify correlation between node labels and graph structure, whereby the subgraph constraint has been loosened, but some structural information is still present in the output.

3 Single Graph Setting

Formally, we define a graph G as the set of nodes $V(G)$ and the set of edges $E(G)$, where each node $v \in V(G)$ has a label $l(v)$. We assume that the graph is connected, and that each node carries at most one label. In this setting, a pattern is an itemset X, a set of node labels that frequently occur in graph G in each other's neighbourhood. In Sections 3.1 and 3.2 we propose two ways of defining and finding interesting patterns in a graph. Note that we can also handle input graphs that contain nodes with multiple labels, by transforming each such node into a clique of nodes, each carrying one label.

3.1 Frequent Itemsets Approach

One way of defining an interesting itemset is by simply looking at its frequency. An interesting itemset is one that occurs often in the dataset. Our challenge therefore consists of taking a graph and converting it into a transaction database.

In essence, we are looking for items that often appear near each other in the graph, so we propose to create a transaction database in which each transaction would correspond to the neighbourhood of a single node in the graph. For a node v, we define the corresponding transaction as $t(v) = \{l(w)|w \in G, d(v,w) \leq n\}$, where $d(v,w)$ is the distance from v to w and n is the neighbourhood size, a user-defined parameter, indicating how far w can be from v and still be considered a part of v's neighbourhood. Typically, n should be relatively small if meaningful results are to be obtained. In a well-connected graph, with a large n, all nodes will be in each other's neighbourhoods.

We define a *transaction database of graph* G as $T(G) = \{t(v)|v \in G\}$. The *frequency* of an itemset X in graph G, $fr_G(X)$, is defined as the number of transactions in $T(G)$ that contain X. An itemset is considered frequent if its frequency is greater than or equal to a user-defined frequency threshold *min_freq*. Once we have converted our original dataset in this way, we can apply any one of a number of existing frequent itemset algorithms to obtain the desired results.

Let us return to our example in Fig. 1. Given a neighbourhood size of 1, the resulting transaction database is given in Table 1. We see that itemset *abcd* will now be discovered as interesting. In fact, only its subsets of size 1 or 2 will score higher.

Table 1. A transaction database obtained from the graph in Fig. 1 with a neighbourhood size of 1

v_{id}	items	v_{id}	items	v_{id}	items
1	*efghi*	8	*abcdg*	15	*bc*
2	*afe*	9	*ab*	16	*cd*
3	*abcdf*	10	*bc*	17	*edi*
4	*ab*	11	*bd*	18	*abcdi*
5	*ac*	12	*ech*	19	*ad*
6	*ad*	13	*abcdh*	20	*bd*
7	*beg*	14	*ac*	21	*cd*

3.2 Cohesive Itemset Approach

Another possible approach is to look for cohesive, rather than simply frequent, itemsets. This idea was inspired by the approach Cule et al. applied in order to find interesting itemsets in event sequences [5]. The idea is not to find items that only occur near each other often, but items that imply the occurrence of each other nearby with a high enough probability.

Consider, for example, the graph given in Fig. 2. We see that patterns *de* and *bc* are both frequent, though *de* will score higher than *bc*, both in subgraph mining and in frequent itemset mining as defined in Section 3.1. However, it can be argued that the value of itemset *de* should diminish due to the fact that a *d* appears in the graph without an *e* nearby, while each *b* has a *c* right next to it, and vice versa. Here, we present an approach that takes this into account.

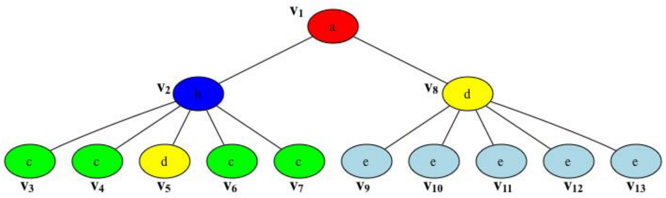


Fig. 2. A graph illustrating the intuition behind cohesive patterns

To start with, we introduce some notations and definitions. In a graph G, the set of nodes is denoted $V(G)$. The number of nodes in G is denoted $|V(G)|$. For an itemset X, we denote the set of nodes labelled by an item in X as $N(X) = \{v \in G | l(v) \in X\}$. The number of such nodes is denoted $|N(X)|$. Finally, we define the probability of randomly encountering a node in G labelled by an item in X as the *coverage* of X in G, or $P(X) = \frac{|N(X)|}{|G|}$.

For each occurrence of an item of X, we must now look for the nearest occurrence of all other items in X. For a node v, we define the sum of all these smallest distances as $W(v, X) = \sum_{x \in X} min_{w \in N(\{x\})} d(v, w)$. We then compute the average of such sums for all occurrences of items making up itemset X, $\overline{W}(X) = \frac{\sum_{v \in N(X)} W(v,X)}{|N(X)|}$. This allows us to define the *cohesion* of an itemset X in G as $C(X) = \frac{|X|-1}{\overline{W}(X)}$. The cohesion is a measure of how near to each other the items in X are in G on average. If they are always right next to each other, the sum of these distances for each occurrence of an item in X will be equal to $|X| - 1$, as will the average of such sums, and the cohesion of X will therefore be equal to 1.

Finally, the *interestingness* of an itemset X is defined as the product of its coverage and its cohesion, $I(X) = P(X)C(X)$. An itemset is considered interesting if its interestingness is greater than or equal to a user-defined interestingness threshold, *min_int*. Unlike with frequent itemsets, where we could tell nothing about the cohesion of the itemsets, we are now able to say that having encountered an item from an interesting itemset, there is a high probability of encountering the rest of the itemset nearby.

Let us now return to the example given in Fig. 2. If we apply these measures to itemsets bc and de, we first note that $P(bc) = 5/13$ and $P(de) = 7/13$. In order to compute the cohesion of the two itemsets, we first have to compute $\overline{W}(bc)$ and $\overline{W}(de)$, which are respectively 1 and $\frac{10}{7}$ (note that all relevant minimal windows are of size 1, except $W(v_5, de) = 4$). Therefore, $C(bc) = 1$ and $C(de) = \frac{7}{10}$, and $I(bc) = 0.385$ and $I(de) = 0.377$. We see that the value of itemset de has indeed diminished due to a d occurring far from any e.

Finally, note that while our method allows us to find more flexible patterns than subgraph mining, we do not miss out on any pattern subgraph mining can discover. If a graph contains many occurrences of a subgraph consisting of nodes labelled a, b and c, then we will find abc as an interesting itemset.

4 Multiple Graph Setting

In many applications, the dataset does not consist of a single graph, but of a collection of graphs. Formally, as before, we define a graph G as a set of nodes $V(G)$ and a set of edges $E(G)$, where each node $v \in G$ has a label $l(v)$. We also define $\mathcal{G}$ as a set of graphs $\{G_1, ..., G_n\}$, and assume that each graph in $\mathcal{G}$ is connected. A pattern is now an itemset X, a set of node labels that frequently occur in the set of graphs $\mathcal{G}$ in each other's neighbourhood. In other words, for a pattern to be interesting, it needs to appear in a cohesive form in many graphs. Unlike the single graph setting, the multiple graph setting does not allow us to transform the dataset into a transaction database. However, the cohesive itemset approach, presented in Section 3.2, can be generalised to the multiple graph setting in a relatively straightforward manner.

We first revisit the notations introduced in Section 3.2. Given a set of graphs $\mathcal{G}$, the number of graphs in $\mathcal{G}$ is denoted $|\mathcal{G}|$. We denote the set of all graphs that contain itemset X as $N_m(X) = \{G \in \mathcal{G} | \forall x \in X \; \exists v \in V(G) \text{ with } l(v) = x\}$. The number of such graphs is denoted as $|N_m(X)|$. Finally, we define the probability of encountering a graph in $\mathcal{G}$ containing the whole of itemset X as the *coverage* of X in $\mathcal{G}$, or $P_m(X) = \frac{|N_m(X)|}{|\mathcal{G}|}$.

Given a graph G_j in $N_m(X)$, we must now look for the most cohesive occurrence of X. To find such an occurrence, we will look for a node in the graph, labelled by an item in X, from which the sum of the distances to all other items in X is the smallest. Given a graph G_j containing an itemset X, we define this lowest sum as $W(X, j) = \min_{v \in N(X, G_j)} W(v, X)$, where $N(X, G_j)$ is the set of nodes in G_j labelled by an item in X, while $W(v, X)$ is defined as in Section 3.2.

We now compute the average of such smallest sums for all graphs in $\mathcal{G}$ containing the whole of itemset X, $\overline{W}_m(X) = \frac{\sum_{j \in N_m(X)} W(X,j)}{|N_m(X)|}$. We can then define the *cohesion* of an itemset X in $\mathcal{G}$ as $C_m(X) = \frac{|X|-1}{\overline{W}_m(X)}$. Once again, a fully cohesive itemset will have cohesion equal to 1.

Finally, the *interestingness* of an itemset X in $\mathcal{G}$ is defined as the product of its coverage and its cohesion, $I_m(X) = P_m(X) C_m(X)$.

5 Algorithms

Mining frequent itemsets in graphs can be done by transforming a graph into a transaction database, and then using an existing itemset miner to generate the output. However, the cohesive itemset approach is less straightforward, and we now present our algorithms, Grit and Mug, for solving this problem in the single graph setting and the multiple graph setting, respectively.

5.1 Single Graph Setting

The fact that our interestingness measure is not anti-monotonic clearly represents a problem. We will sometimes need to search deeper even when we encounter an uninteresting itemset, as one of its supersets could still prove interesting. Traversing the complete search space is unfeasible, so we will need a different pruning technique to speed up our algorithm. We adapt the approach introduced by Cule et al. for mining itemsets in sequences [5] to our setting.

We approach the problem using depth-first search, and the pseudocode of the main Grit algorithm is provided in Algorithm 1. The first call to the algorithm is made with X empty and Y containing all possible items. At the heart of the algorithm is the *UBI* pruning function, used to decide when to prune a complete branch of the search tree, and when to proceed deeper. Essentially, we can prune a complete branch if we are certain that no itemset generated within this branch can be interesting. To be able to ascertain this, we compute an upper bound for the interestingness of all these itemsets, and prune the branch if this upper bound is smaller than the interestingness threshold. We begin by noting that, for each Z, such that $X \subseteq Z \subseteq X \cup Y$, it holds that $|N(Z)| \leq |N(X \cup Y)|$, $|Z| \leq |X \cup Y|$ and $\sum_{v \in N(X)} W(v, X) \leq \sum_{v \in N(Z)} W(v, Z)$. Expanding the definition of the interestingness, we get that $I(Z) = \frac{|N(Z)| \times |N(Z)| \times (|Z|-1)}{\sum_{v \in N(Z)} W(v,Z) \times |G|}$. It therefore follows that $I(Z) \leq \frac{|N(X \cup Y)| \times |N(X \cup Y)| \times (|X \cup Y|-1)}{\sum_{v \in N(X)} W(v,X) \times |G|}$. We have thus found an upper bound for the interestingness of all itemsets Z, that can be generated in a branch of the search tree starting off with itemset X, and reaching as deep as itemset $X \cup Y$.

However, while this upper bound is theoretically sound, it is also computationally very expensive. Note that we would need to compute $\sum_{v \in N(X)} W(v, X)$ at each node in our search tree. This would require traversing the whole graph searching for the minimal distances between all items in X for all relevant nodes. This, too, would be infeasible. Luckily, if we express these sums differently, we can avoid these computationally expensive database scans. Adapting the approach introduced by Cule and Goethals [4] to the graph setting, we first note that the sum of the minimal distances between items making up an itemset X and the remaining items in X can also be expressed as a sum of separate sums of such distances for each item individually, $\sum_{v \in N(X)} W(v, X) = \sum_{x \in X} \sum_{v \in N(\{x\})} W(v, X)$. We then note that each such sum for an occurrence of an item $x \in X$ is equal to the sum of individual minimal distances between the same occurrence of x and any other item $y \in X$. For the sum of such distances,

Algorithm 1. GRIT($\langle X, Y \rangle$) finds interesting itemsets

```
if $UBI(\langle X,Y \rangle) \geq min\_int$ then
  if $Y = \emptyset$ then
    output $X$
  else
    Choose $a$ in $Y$
    GRIT($\langle X \cup \{a\}, Y \setminus \{a\} \rangle$)
    GRIT($\langle X, Y \setminus \{a\} \rangle$)
  end if
end if
```

it holds that $\sum_{v \in N(\{x\})} W(v, X) = \sum_{v \in N(\{x\})} \sum_{y \in X \setminus \{x\}} W(v, xy)$. Naturally, it also holds that $\sum_{v \in N(\{x\})} W(v, X) = \sum_{y \in X \setminus \{x\}} (\sum_{v \in N(\{x\})} W(v, xy))$. To simplify our notation, from now on we will denote $\sum_{v \in N(\{x\})} W(v, xy)$ by $s(x, y)$. Finally, we see that $\sum_{v \in N(X)} W(v, X) = \sum_{x \in X} \sum_{y \in X \setminus \{x\}} (s(x, y))$, giving us a much more elegant way to compute the sum of distances between an occurrence of an item in X and the rest of X for all nodes labelled by an item in X. Finally, we are ready to define our upper-bound-based pruning function: $\mathit{UBI}(\langle X, Y \rangle) = \frac{|N(X \cup Y)|^2 \times (|X \cup Y| - 1)}{\sum_{x \in X} \sum_{y \in X \setminus \{x\}} (s(x,y)) \times |G|}$.

This pruning function is easily evaluated, as all it requires is that we store $s(x, y)$, the sum of minimal distances between x and y over all occurrences of x, for each pair of items (x, y), so we can look them up when necessary. This can be done as soon as the dataset has been read, and all entries can be computed efficiently. Note that if Y is empty, then $\mathit{UBI}(\langle X, Y \rangle) = I(X)$, so if we reach a leaf node in the search tree, we can immediately output the itemset.

5.2 Multiple Graph Setting

As in the single graph setting, the interestingness measure based on cohesive itemsets in multiple graphs is also not anti-monotonic. Here, however, the coverage alone is anti-monotonic. Given itemsets X and Y, such that $X \subset Y$, it is clear that any graph that contains Y will also contain X. Keeping in mind that the interestingness of an itemset is never greater than its coverage, this allows us to prune even more candidates from our search space. For any itemset Z, such that $X \subseteq Z$, it holds that $I(Z) \leq P(Z) \leq P(X)$. Therefore, if we encounter an itemset X, such that its coverage is lower than the interestingness threshold *min_int*, we can safely discard all its supersets from the search space.

On top of this pruning criterion, we can develop a pruning function *MUBI* in much the same way as we did in the single graph setting above. Once again, it would be infeasible to compute all necessary sums of minimal windows at each node in our search tree. However, this time we cannot express this sum using similar sums for pairs of items as we did in Section 5.1. The reason for this is the fact that we now define $W(X, j)$ as the minimal occurrence of X in graph G_j, while in Section 3.2 we defined $W(v, X)$ as a sum of individual minimal distances between items. Given a graph G_j and an itemset X, knowing the individual minimal distances between the items of X in G_j would bring us no closer to knowing the size of the minimal occurrence of X in G_j. We have therefore decided to perform all our experiments without *MUBI* — pruning less, but faster. As a result, our Mug algorithm for mining interesting itemsets in multiple graphs is exactly the same as the one given in Algorithm 1, with line **if** $P(X) \geq \mathit{min\_int}$ **then** replacing line 1.

6 Experiments

For our single graph setting experiments, we generated a number of synthetic datasets. To start with, we generated a graph with 10 000 nodes, randomly allocating labels ranging from 1 to 20. We made sure that labels 0 and 1 were more probable than all the others, followed by labels 2 and 3, while the remaining labels were all equally probable. We build the graph by, at each step, adding either a new node and connecting it to a random existing node, or adding a new edge between two existing nodes. In our first set of experiments, we set the probability of adding a new node to 60%, and the probability of adding a new edge to 40%, resulting in a relatively sparse graph, with around 1.7 edges per node on average. The characteristics of the input graph and the results of the experiments for both the frequent itemset approach and the cohesive itemset approach are presented in the top quarter of Table 2. For the frequent itemset approach, the reported runtime is the sum of the time needed to transform the dataset into a transaction database added to the time needed to run an implementation[1] of the classical Fp-growth algorithm on the transformed dataset.

Table 2. Experimental results on four different single graph datasets. The runtime is measured in milliseconds.

Input Graph			Frequent Itemset Approach			Cohesive Itemset Approach		
nodes	edges	items	*min_freq*	runtime	itemsets found	*min_int*	runtime	itemsets found
10 000	16 853	20	2 000	873	4	0.37	2 545	1
			1 000	905	25	0.35	2 796	35
			100	966	737	0.33	3 236	704
			10	1 202	15 644	0.30	4 480	8 193
			1	2 513	250 181	0.20	36 314	430 961
10 000	16 853	30	2 000	853	4	0.37	823 263	1
			500	892	71	0.35	892 480	31
			50	960	1 793	0.33	1 013 627	1 672
			5	1 225	37 349	0.31	1 253 782	28 147
10 000	68 190	20	4 000	1 238	11	0.55	3 327	35
			1 000	1 720	4 449	0.50	4 084	2 228
			250	6 279	462 933	0.35	23 178	245 668
			50	10 953	1 048 575	0.20	61 519	950 411
100 000	166 446	20	20 000	2 994	4	0.37	21 425	1
			5 000	3 192	63	0.35	21 856	17
			500	3 575	1 980	0.30	24 502	8 092
			50	4 137	31 424	0.25	31 559	89 617

To examine how our methods react to different types of input, we created three more graphs, each time changing one of the settings. In the second set of experiments, we introduced some noise into the dataset, by randomly choosing 100 nodes in the original graph and changing their labels to a random item ranging from 21 to 30. For the third set of experiments, we created a denser graph, by setting the probability of adding a new node to 15%, and the probability of adding a new edge to 85%. In the fourth set of experiments, we used a larger graph, creating 100 000 nodes, keeping the other settings as in the original dataset. The characteristics of these three input graphs and the results of our experiments can be seen in the bottom three quarters of Table 2.

In all four sets of experiments, we note that frequent itemset mining works very fast once the transformation of the dataset has been done, while the cohesive itemset approach suffers from having to compute $s(x, y)$ for each pair of

[1] Taken from `http://adrem.ua.ac.be/~goethals/software`

items x and y, regardless of the threshold. Grit struggles with noisy data, as expensive computations need to be done even for very rare items and itemsets, while transforming the graph into a transaction database takes a lot more time when dealing with both a denser or a larger input graph. Finally, all experiments on the four datasets produced expected results in terms of the discovered itemsets — items 0 and 1 were ranked top amongst items, while itemset $\{0, 1\}$ was the highest ranked pair of items. Both methods discovered the same itemsets of size 3 and 4 (itemsets made up of items 0, 1, 2 and 3), while results differed for itemsets of size 5 or more. However, due to the anti-monotonicity of the traditional frequency measure, Fp-growth ranked singletons higher than their supersets, while Grit considered itemset $\{0, 1, 2, 3\}$, for example, more interesting than each individual item alone. We conclude that both approaches have their respective merits, with the frequent itemset approach being faster, and the cohesive itemset approach more intuitive. Grit will first find large, informative itemsets, while Fp-growth first finds singletons, and the really interesting large itemsets can be buried beneath a pile of smaller itemsets.

In the multiple graph setting, we experimented on a dataset consisting of 340 chemical compounds, made up of 66 different atom types. However, we discovered five unconnected graphs in the dataset and removed them, as the cohesive itemset approach can only be applied to connected graphs, leaving us with 335 input graphs. We compare our algorithm for finding cohesive itemsets in multiple graphs with the Gaston tool developed by Nijssen and Kok [15]. Both the dataset and the implementation are available online[2]. We present the results of our experiments in Tables 3 and 4.

Table 3. Results of the Mug algorithm on the Chemical_340 dataset

min_int	runtime	itemsets
0.50	17	5
0.20	173	17
0.10	772	53
0.05	2 196	172
0.01	21 869	1 122

Table 4. Results of the Gaston algorithm

min_freq	runtime	itemsets
200	10	11
100	16	68
50	26	427
10	376	19 237
5	9 318	447 879

By comparing our Mug algorithm to a frequent subgraph miner, we are in fact comparing apples and oranges. While Mug uses an interestingness measure, Gaston uses a frequency threshold. Mug searches for itemsets, Gaston discovers subgraphs. Mug discards structure and focuses on items in each other's neighbourhoods, while Gaston focuses on structure. Clearly, Mug results in a massive reduction in output, as, for a typical interesting itemset discovered by Mug, Gaston will find a number of combinations of edges (and their subsets) that are frequent. Furthermore, the actual patterns discovered by the two algorithms differ greatly. To name but one example, the most interesting itemset of

[2] http://www.liacs.nl/~snijssen/gaston/download.html

size 4 that we discovered was $\{0, 1, 4, 5\}$, while the most frequent subgraph of size 4 was $0-0-0-0$. On the other hand, if we compare only patterns of size 2, Gaston finds graph $1-9$ as the most frequent, while we also found $\{1, 9\}$ as highest ranked itemset of size 2. However, it is also important to note that itemset $\{0, 1, 4, 5\}$ ranked as one of the best itemsets overall in our output, while, due to the nature of frequent subgraph mining, graph $0-0-0-0$ was ranked below all its subgraphs. As with Grit, Mug has the advantage of ranking the larger interesting itemsets above all other, while Gaston will always rank a large pattern below a number of smaller patterns.

7 Conclusion

In this paper we presented a number of new methods to identify interesting itemsets in one or many input graphs. For one graph, such itemsets consist of items that often occur close to each other. Unlike previous approaches that typically look for structures connecting these items, we only look at the distances between the items themselves. This enables us to avoid the typical pitfalls of subgraph mining — costly isomorphism checks and a huge number of candidates. On top of the classical frequent itemset approach that we adapted to mining itemsets in a large graph, we propose a second method, mining cohesive itemsets, consisting of items that appear close to each other frequently enough. This second approach proved perfectly adaptable to the multiple graph setting, too.

References

[1] Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proc. of the 20th Int. Conf. on Very Large Data Bases, pp. 487–499 (1994)

[2] Bringmann, B., Nijssen, S.: What is frequent in a single graph? In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 858–863. Springer, Heidelberg (2008)

[3] Cook, D.J., Holder, L.B.: Substructure discovery using minimum description length and background knowledge. Journal of Artificial Intelligence Research 1, 231–255 (1994)

[4] Cule, B., Goethals, B.: Mining association rules in long sequences. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS (LNAI), vol. 6118, pp. 300–309. Springer, Heidelberg (2010)

[5] Cule, B., Goethals, B., Robardet, C.: A new constraint for mining sets in sequences. In: Proc. of the 9th SIAM Int. Conf. on Data Mining, pp. 317–328 (2009)

[6] Guan, Z., Wu, J., Zhang, Q., Singh, A., Yan, X.: Assessing and ranking structural correlations in graphs. In: Proc. of the 2011 ACM SIGMOD Int. Conf. on Management of Data, pp. 937–948. ACM (2011)

[7] Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proc. of the 2000 ACM SIGMOD Int. Conf. on Management of Data, vol. 29, pp. 1–12 (2000)

[8] Huan, J., Wang, W., Prins, J., Yang, J.: Spin: mining maximal frequent subgraphs from graph databases. In: Proc. of the 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 581–586 (2004)

[9] Inokuchi, A., Washio, T., Motoda, H.: An apriori-based algorithm for mining frequent substructures from graph data. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 13–23. Springer, Heidelberg (2000)

[10] Inokuchi, A., Washio, T., Motoda, H.: Complete mining of frequent patterns from graphs: Mining graph data. Machine Learning 50, 321–354 (2003)

[11] Khan, A., Li, N., Yan, X., Guan, Z., Chakraborty, S., Tao, S.: Neighborhood based fast graph search in large networks. In: Proc. of the 2011 ACM SIGMOD Int. Conf. on Management of Data, pp. 901–912. ACM (2011)

[12] Khan, A., Yan, X., Wu, K.-L.: Towards proximity pattern mining in large graphs. In: Proc. of the 2010 ACM SIGMOD Int. Conf. on Management of Data, pp. 867–878. ACM (2010)

[13] Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: Proc. of the 2001 IEEE Int. Conf. on Data Mining, pp. 313–320 (2001)

[14] Kuramochi, M., Karypis, G.: Finding frequent patterns in a large sparse graph. Data Mining and Knowledge Discovery 11, 243–271 (2005)

[15] Nijssen, S., Kok, J.N.: The gaston tool for frequent subgraph mining. Electronic Notes in Theoretical Computer Science 127, 77–87 (2005)

[16] Silva Jr., A., Meira, W., Zaki, M.J.: Structural correlation pattern mining for large graphs. In: Proc. of the 8th Workshop on Mining and Learning with Graphs, pp. 119–126. ACM (2010)

[17] Silva Jr., A., Meira, W., Zaki, M.J.: Mining attribute-structure correlated patterns in large attributed graphs. Proc. of the VLDB Endowment 5(5), 466–477 (2012)

[18] Washio, T., Motoda, H.: State of the art of graph-based data mining. ACM SIGKDD Explorations Newsletter 5, 59–68 (2003)

[19] Yan, X., Han, J.: gspan: Graph-based substructure pattern mining. In: Proc. of the 2002 IEEE Int. Conf. on Data Mining, pp. 721–724 (2002)

[20] Yan, X., Zhou, X., Han, J.: Mining closed relational graphs with connectivity constraints. In: Proc. of the 11th ACM SIGKDD Int. Conf. on Knowledge Discovery in Data Mining, pp. 324–333 (2005)

[21] Yoshida, K., Motoda, H., Indurkhya, N.: Graph-based induction as a unified learning framework. Journal of Applied Intelligence 4, 297–316 (1994)

Robust Synchronization-Based Graph Clustering

Junming Shao[1], Xiao He[2], Qinli Yang[2], Claudia Plant[3], and Christian Böhm[2]

[1] University of Mainz, 55122 Mainz, Germany
[2] University of Munich, 80538 Munich, Germany
[3] Florida State University, FL 32306-4120 Tallahassee, USA

Abstract. Complex graph data now arises in various fields like social networks, protein-protein interaction networks, ecosystems, etc. To reveal the underlying patterns in graphs, an important task is to partition them into several meaningful clusters. The question is: how can we find the natural partitions of a complex graph which truly reflect the intrinsic patterns? In this paper, we propose *RSGC*, a novel approach to graph clustering. The key philosophy of *RSGC* is to consider graph clustering as a dynamic process towards synchronization. For each vertex, it is viewed as an oscillator and interacts with other vertices according to the graph connection information. During the process towards synchronization, vertices with similar connectivity patterns tend to naturally synchronize together to form a cluster. Inherited from the powerful concept of synchronization, *RSGC* shows several desirable properties: (a) it provides a novel perspective for graph clustering based on proposed interaction model; (b) *RSGC* allows discovering natural clusters in graph without any data distribution assumption; (c) RSGC is also robust against noise vertices. We systematically evaluate *RSGC* algorithm on synthetic and real data to demonstrate its superiority.

Keywords: Graph Clustering, Synchronization, Kuramoto Model.

1 Introduction

As a data format, graphs are characterized as a set of interconnected units. These units, often called nodes or vertices, are linked to each other by edges expressing their relationships. In recent years, the study of graph clustering has attracted a huge attentions and many techniques have been developed based on different partitioning criteria, e.g. betweenness, modularity and clique. Although established approaches have already achieved some success, finding the real and intrinsic clusters in graphs is still a big challenge [7]. Moreover, previous studies of graph clustering mainly focused on unweighted graphs. A fresh and increasingly challenging care is to study weighted graphs where each link is associated with a large heterogeneity in the capacity and intensity.

In view of these challenges, in this paper we consider graph clustering from a different perspective: *synchronization*. Synchronization is a prevalent phenomenon in nature that a group of events spontaneously come into co-occurrence with a common rhythm through mutual interactions. A paradigmatic example

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 249–260, 2013.

of synchronization phenomena is the synchronous flashing of fireflies observed in some South Asian forests. Briefly, synchronization means adjustment of states of oscillators due to weak interaction so that their states can coincide. To better illustrate the concept of synchronization, let us take social networks as an example. People in a social network with similar characteristics, e.g. common interest, friends, or similar calling behaviors from phone companies tend to group together. In such network, for a certain problem, in the beginning, each person may has his/her own opinion. As time evolves, people tend to be affected by their friends and change opinion gradually. In principle, the closer relationship they are, the higher influence between each other. Through the discussion, finally people with similar characteristics tend to achieve the same opinion. The dynamic process of opinion formation in the social network can be reviewed as a common synchronization phenomenon. From this example, what makes us interested is that the process of opinion formation in the social networks (a dynamic process towards synchronization) is very similar to a dynamic clustering process. More importantly, the interactions among vertices during the process of synchronization are completely governed by the intrinsic structural of the graph.

Therefore, inspired by natural synchronization phenomena and established models, for graph clustering, a new intuitive idea is to consider it as a dynamic process towards synchronization. We consider each vertex as an oscillator and it interacts with other vertices relying on its intrinsic connection information. The graph clustering is thus transformed into investigating the dynamics of vertices during the process towards synchronization. A graph cluster is finally defined as the vertices which can finally group together after synchronization.

The remainder of this paper is organized as follows: in the following section, we briefly survey related work. Section 3 presents our algorithm in detail. Section 4 contains an extensive experimental evaluation and we finally conclude the paper in Section 5.

2 Related Work

During the past several decades, many approaches have been proposed for graph clustering. Due to space limitation, we only provide a very brief survey on graph clustering related to our work.

Spectral Clustering: These approaches refer to a class of well-known techniques which rely on the Eigenvector decomposition of a similarity matrix to partition objects into disjoint clusters. The algorithm proposed by [14] allows detecting arbitrarily shaped clusters by considering the clustering problem from a graph-theoretic perspective. A cluster is obtained by removing the weakest edges between highly connected subgraphs, which is formally defined by the normalized cut or similar objective functions. To overcome the difficulty of parametrization, Zelnik-Manor and Perona [18] proposed a new method to estimate the number of clusters by investigating the structure of the Eigenvectors.

Multi-Level Clustering. Metis is a set of serial of multi-level partitioning techniques proposed by Karypis and Kumar [11]. For graph partitioning, an initial clustering is performed on the coarsest graph, and then, a sequence of successively finer graphs is constructed level by level. At each level, an iterative refinement algorithm such as Kernighan-Lin (KL) or Fiduccia-Mattheyses (FM) is used to further improve the bisection. In all, these methods are fast and give high-quality partitions in most cases. However, like spectral clustering, a suitable number of k clusters has to be set for the algorithm. In addition, these multilevel algorithms restrict to detect clusters of nearly equal size.

Markov Clustering. The Markov Cluster algorithm (MCL) is a popular algorithm used in life sciences for fast clustering of weighted graphs. MCL [6] simulates a flow on the graph by calculating successive powers of the associated adjacency matrix. At each iteration, an inflation step is applied to enhance the contrast between regions of strong or weak flow in the graph. The process converges towards a partition of the graph, with a set of high-flow regions (the clusters) separated by boundaries with no flow. The value of the inflation parameter strongly influences the number of clusters.

MDL-Based Clustering. The key idea of these methods is to detect clusters by using a model of probability density functions (PDFs) to describe the data structure and link the clustering problem to data compression. One of fundamental techniques in this line is Cross-Association [5] approach, which finds groups in unweighted graphs by loss-less compression with Minimum Description Length (MDL). Similar to Cross-Association, the algorithm called PaCCo [13], is proposed for weighted graph, which combines the MDL principle with a bisecting k-Means strategy. The MDL principle provides a good way to qualify the clustering results and thus avoids the parameter setting.

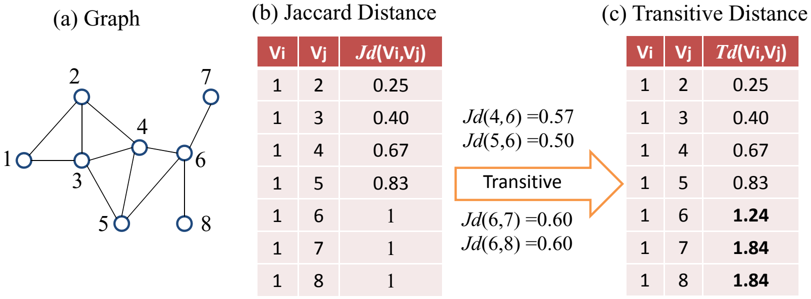
Synchronization. Arenas et al. [1] apply the Kuramoto model for network analysis, and study the relationship between topological scales and dynamic time scales in complex networks. From bioinformatics, Kim et.al. [4] propose a strategy to find groups of genes by analyzing the cell cycle specific gene expression with a modified Kuramoto model. Recently, Shao et.al proposed an extension of the Kuraomto model for clustering and outlier detection [15], [2], [16] on vector data based on the concept of synchronization and MDL principle.

3 Synchronization-Based Graph Clustering

In this section, we introduce *RSGC* algorithm, which consider graph clustering as a dynamic process. In the following, we start with the vertex feature representation and then introduce an interaction model for graph clustering. In Section 3.3 we discuss the algorithm *RSGC* in detail.

3.1 Vertex Feature Representation

Given an undirected graph G, the only information we can gain is its connectivity patterns. In this study, we first compute the proposed *Transitive distance* of any two vertices and then transform them into a feature vector space. Each vertex



(b) Jaccard Distance

Vi	Vj	*Jd*(Vi,Vj)
1	2	0.25
1	3	0.40
1	4	0.67
1	5	0.83
1	6	1
1	7	1
1	8	1

(c) Transitive Distance

Vi	Vj	*Td*(Vi,Vj)
1	2	0.25
1	3	0.40
1	4	0.67
1	5	0.83
1	6	**1.24**
1	7	**1.84**
1	8	**1.84**

Fig. 1. Illustration of computing the dissimilarity among vertices

is finally mapped as a feature vector to represent its initial phase. Before that, let us first introduce some necessary definitions.

Definition 1 (Jaccard distance). Given any two vertices u and v in Graph G, the Jaccard distance $Jd(u, v)$ of two vertices u and v is defined as:

$$Jd(u, v) = 1 - \rho(u, v) \tag{1}$$

where $\rho(u, v)$ is the Jaccard coefficient [10] between vertices u and v,

$$\rho(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{2}$$

Here $\Gamma(\cdot)$ is neighbors of one vertex. Relying on Eq. (1), the Jaccard distance of the pairs of non-neighbor vertices is always 1, which is insufficient to represent the similarity of any two vertices effectively. Therefore, we further define *Transitive distance* as follows.

Definition 2 (Transitive distance). Given two vertices u and v have no common neighbors, $\{S_1, \cdots, S_N\}$ are all shortest paths between vertices u and v and $S_k = \{u, w_1^k, \cdots, w_{|S_k-2|}^k, v\}, k \in (1, \cdots, N)$. The Transitive distance between vertices u and v is defined as the minimal Jaccard distance of these paths.

$$Td(u, v) = \begin{cases} \min\limits_{k \in (1,\cdots,N)} \left(Jd(u, w_{|S_k-2|}^k) + Jd(w_{|S_k-2|}^k, v)\right) & Jd(u, v) = 1 \text{ AND} \\ & Jd(u, w_{|S_k-2|}^k) \neq 1 \\ Jd(u, v) & else \end{cases} \tag{3}$$

Figure 1 gives an example to illustrate the Transitive distance computation from vertex 1 to all other vertices. Based on Transitive distance, the distance matrix for all vertices can be computed. Finally, we map it into points in a feature vector space and each vertex is represented as a feature vector. Here, we apply the well-known method, called *FastMap* [8], to transform the distance matrix in a metric space into a feature vector space. In this study, we simply map each vertex into a 2-dimensional feature space. In principle, the higher dimensional feature space is selected, the less connection information among vertices is lost. However, the difficulty of clustering in high dimensional space is also increased.

3.2 Interaction Model

Currently, one of the most successful approaches to explore the synchronization phenomena is Kuramoto Model [12]. It describes the dynamics of a large set of phase oscillators by coupling the sine of their phase differences. Formally, the *Kuramoto model* (KM) consists of N phase oscillators where the phase of the *i-th* unit, denoted by θ_i, evolves in time according to the following dynamics:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_j - \theta_i), (i = 1, \ldots, N), \tag{4}$$

where ω_i stands for its natural frequency and K describes the coupling strength between units. $sin(\cdot)$ is the coupling function.

The KM describes the global synchronization behavior of all coupled phase oscillators. However, in real-world graphs or networks, the connectivity among vertices are often not full but partial, which indicates that only a local ensemble of vertices are connected. Therefore, it is necessary to extensively reformulate *Eq.*(4). The natural and intuitive way is to model the interactions in graph with its intrinsic connection patterns.

Definition 3 (Interaction Model). Let u be a vertex in the graph G. $\varGamma(u)$ are the neighbors of vertex u and u_i is the $i-th$ feature of vertex u. The interaction range R_u of vertex u is the maximal distance to these neighbors, according to Eq.(4), the dynamics of $i-th$ phase u_i of vertex u is governed by:

$$\frac{du_i}{dt} = \omega_i + \frac{K}{\sum_{v\in\varGamma(u)} W(u,v)}\sum_{v\in\varGamma(u)} W(u,v)\cdot\varPhi(u,v)\cdot\sin(v_i - u_i). \tag{5}$$

where $W(v,u)$ is the edge weight between vertices u and v. The $\varPhi(u,v)$ is used to check whether the vertex v should interact with the vertex u, which is defined as:

$$\varPhi(u,v) = \begin{cases} 0 \; dist(v,u) > R_u \\ 1 \; else \end{cases}$$

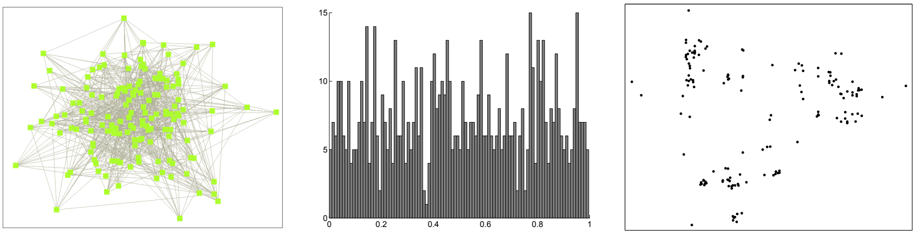
Then, let $dt = \varDelta t$, the Equation (5) can be further written as:

$$u_i(t+1) = u_i(t) + \varDelta t\cdot\omega_i + \frac{\varDelta t\cdot K}{\sum_{v\in\varGamma(u)} W(v,u)}\sum_{v\in\varGamma(u)} W(v,u)\cdot\varPhi(u(t),v(t))\cdot\sin(v_i(t) - u_i(t)) \tag{6}$$

Here, without knowing external knowledge, all vertices (oscillators) are assumed having the same frequency w. The term $\varDelta t\cdot\omega_i$ is thus the same for each vertex and ignored. $\varDelta t\cdot K = C$ is a constant and fixed as 1. Finally the dynamics of $i-th$ phase u_i of the vertex u over time is provided by:

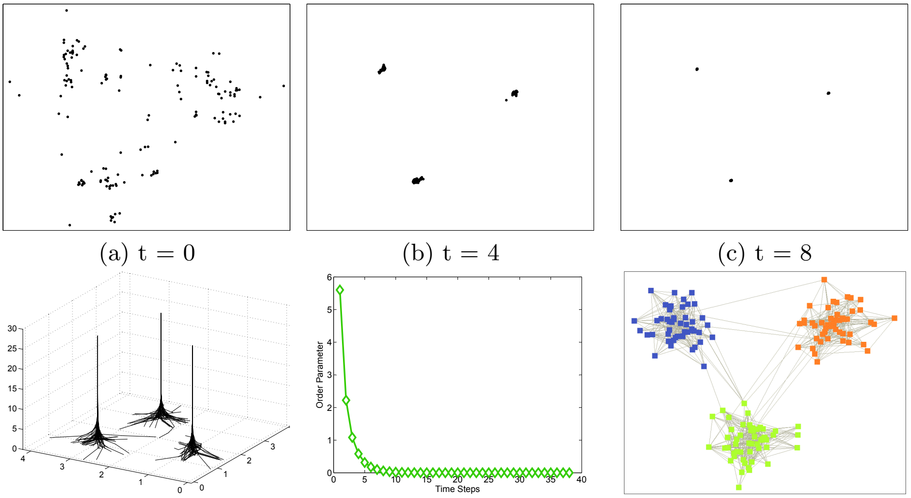
$$u_i(t+1) = u_i(t) + \frac{1}{\sum_{v\in\varGamma(u)} W(v,u)}\sum_{v\in\varGamma(u)} W(v,u)\cdot\varPhi(u(t),v(t))\cdot\sin(v_i(t) - u_i(t)) \tag{7}$$

The vertex u at time step $t = 0$: $u(0)$ represents the initial phase of the vertex (original feature vector). The $u_i(t+1)$ describes the renewal phase value of i-th phase of vertex u at the $t = (0, \ldots, T)$ time evolution.



(a) Weighted Graph (b) Edge Weights Distribution (c) Feature Representation

Fig. 2. Vertices Feature Representation



(a) t = 0 (b) t = 4 (c) t = 8

(d) Dynamics of Vertices (e) Graph Order Parameter (f) Graph partitioning result

Fig. 3. The detail states of vertices during the process towards synchronization

In order to investigate local dynamic effects so that the clusters of synchronized vertices can be discovered, we define a graph order parameter r_g, measuring the coherence of the local oscillator population in graphs.

Definition 4 (Graph Order Parameter) The graph order parameter r_g characterizing the degree of local synchronization in graphs is provided by:

$$r_g = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{\sum_{v \in \Gamma(u)} W(v,u)} \sum_{v \in \Gamma(u)} W(v,u) \cdot \Phi(u,v) \cdot ||v-u|| \right) \tag{8}$$

The more vertices are synchronized together, the value of r_g will become much smaller. The dynamics of all vertices will terminate when the r_g converges, which indicates that the vertices in clusters synchronize together (in phase).

3.3 The RSGC Algorithm

Generally, the process of the graph clustering based on synchronization involves three steps: (1) Vertices Feature Representation; (2) Dynamics on Graph and (3)

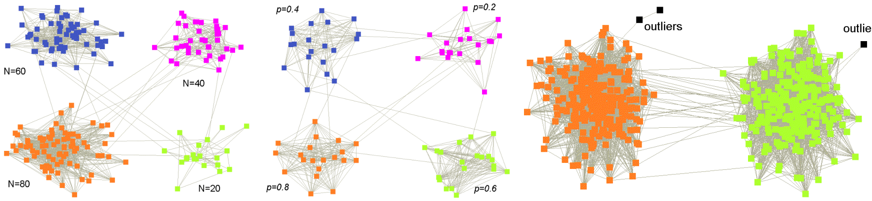
Clusters Discovering. For illustration, we introduce a simple weighted graph in Figure 2(a). Figure 2 (b) plots the edge weights distribution of the graph. With Transitive distance and FastMap, each vertex is projected into a feature space, which is indicated in Figure 2(c). After that, the dynamics of all vertices can be simulated according to *Eq.*(7). Figure 3(f) displays the dynamic movement of all vertices during the process towards synchronization. Figure 3 (a)-(c) further depict the detailed states of vertices at time step $t = 0$ to $t = 8$. The process of synchronization will be terminated when the graph order parameter r_g converges, which indicates in Figure 3(e). The result of graph partitioning is shown in Figure 3(f). Finally, the pseudocode of *RSGC* algorithm is further described in Algorithm 1.

Algorithm 1. RSGC algorithm

```
Input: Graph $G(V, E, W)$

Compute matrix $A$ with Jaccobi Coefficient ;
Obtain dissimilarity matrix $D$ with transitive distance & $A$;
Transform $D$ into features vectors $F$ using FastMap;

for (Each $u \in F$) do
  $u(0) = F_u(0)$
end for
while (loopFlag = True) do
  for (Each $u \in F$) do
    Obtain new phase $u(t+1)$ using $Eq.(7)$;
  end for
  Compute graph order parameter $r_g$;
  $F(t+1) = \sum_1^N \bigcup u(t+1)$;
  if $r_g$ converges then
    loopFlag = False;
    $C = synCluster(F(t+1))$;
  end if
end while

return $C$;
```

4 Experiments

To extensively study the performance of *RSGC*, we conduct experiments on several synthetic and real-world data sets. We compare *RSGC* to representatives of various graph clustering paradigms: *Metis*, *MCL* and two parameter-free weighted graph clustering approaches: information-theoretic clustering *PaCCo* [13] and the spectral clustering approach [18] (named *Spectral* in the following). All experiments have been performed on a workstation with 2.0 GHz CPU and 8.0 GB RAM.



(a) Different Sizes (b) Various Densities (c) Outliers Handling

Fig. 4. Performance of *RSGC*

4.1 Evaluation Criteria

To provide an objective comparison of effectiveness, we evaluate the graph clustering algorithms in two ways. If class label information is available, three information-theoretical measures: normalized mutual information (NMI), adjusted mutual information (AMI) and adjusted variation information (AVI) [17] are directly applied for clustering comparison. For the comparison of different algorithms on the real data sets without class-label information, we evaluate them based on a measure called *modularity*[9], which is applied to quantifies the quality of a division of a network into modules or communities.

4.2 Proof of Concept

Intrinsic Cluster Structure Discovery. We first evaluate the performance of *RSGC* to discover natural graph partitioning in difficult settings, starting with subgraphs of arbitrary size. The data set displayed in Figure 4(a) consists of 4 clusters with different sizes, ranging from 20 to 80. The intra-connection is approximately 40% and the distribution of edge weights is Gaussian. *RSGC* successfully detects all clusters without any edge weights distribution assumptions. Moreover, we generate four clusters with different densities. For each cluster, it includes 20 vertices and the probabilities of intra-connection of each cluster vary from 20% to 80%, see Figure 4(b).

Outliers Handling. Inherit from the concept of synchronization, *RSGC* algorithm allows detecting outliers, where these vertices in graphs are difficult to synchronize with other vertices and have different dynamics. As displayed in Figure 4(c), the outliers are found by exploring these vertices which are out of synchronization.

4.3 Synthetic Data

For comparison, we further create a graph consisting of four clusters with different settings to evaluate their effectiveness. The number of vertices per cluster varied from 25 to 100. Meanwhile, the probabilities of intra-connection of each cluster ranging from 20% to 80% and the inter-connection among clusters is randomly interlinked with 10% . The weights of all connections are generated

Table 1. Performance of graph clustering algorithms on the simple synthetic data set

Algorithms	RSGC	Metis	MCL	PaCCo	Spectral
NMI	1	0.686	0.922	0.924	0.963
AMI	1	0.680	0.920	0.922	0.962
AVI	1	0.684	0.936	0.953	0.975

with Gaussian distribution. In addition, 6 nodes are randomly inserted into the graph with very few connections. The quality of clustering results based on different clustering approaches is illustrated in Table 1. *RSGC* successfully find all 4 clusters and irregular nodes automatically. The experiment shows that *RSGC* outperforms Metis and is also comparable to algorithms of *MCL*, *Spectral* and *PaCCo*.

4.4 Real World Data

In this section, we evaluate the performance of *RSGC* on several real-world data sets. Due to space limitation and difficult parametrization, we limit the comparison to the parameter-free graph clustering algorithms *RSGC*, *Spectral* and *PaCCo*. We obtain five author-collaboration networks from different communities: Network Theory (Netsci), PhD Student Network in Computer Science (CS-PhD), Computational Geometry (Geom), Arxiv General Relativity (CA-GrQc), Erdos Research (Erdos) [1] [2] ; three Protein-Protein Interaction networks from three species, which are *S. cerevisiae (Scere)*, *Escherichia coli (Ecoli)* and *C.elegans (Celeg)* and a Autonomous Systems network of routers comprising the Internet (As) [3].

Table 2 shows the clustering results in terms of *modularity score*. It is obvious that *RSGC* perform well on all these data sets, which obtain the best modularity scores for all experiments except the *Spectral* algorithm on *As* data set. For the algorithm of *PaCCo*, it can not yield good partition results for most data sets, especially for the unweighted graphs. The reason behind it is that *PaCCo* tends to fail if the weight distribution does not correspond to the cluster model. Like *PaCCo*, *Spectral* also obtain relatively few clusters except for the *CS-PhD* and *As* data sets.

Case Study: To further evaluate the performance of *RSGC*, we illustrate it on a case study on a protein-protein interaction (PPI) network. Here, we use the latest version of PPI network of C.elegans (Celeg), which contains 2880 proteins and 4812 known interactions. We analyze this interaction network with *RSGC* and also compare its performance to *PaCCo* and *Spectral*. *RSGC* discovers 32 clusters, while *PaCCo* and *Spectral* produce only 2 clusters, respectively. In the context of biology, we can evaluate the biological significance of obtained clusters with the help of the Gene Ontology database, which provides the ontology

[1] http://vlado.fmf.uni-lj.si/pub/networks/data/default.htm

[2] http://www-personal.umich.edu/~mejn/netdata,
http://snap.stanford.edu/data/

[3] http://dip.doe-mbi.ucla.edu/dip/Main.cgi,http://snap.stanford.edu/data/

Table 2. Performance of graph clustering algorithms on real world data sets

Data Set	PaCCo		Spectral		RSGC	
	#cluster	Modularity	#cluster	Modularity	#cluster	Modularity
NetSci	11	0.542	4	0.696	19	**0.779**
CS-Phd	2	0.077	40	0.531	36	**0.715**
Geom	54	0.327	6	0.067	44	**0.465**
CA-GrQc	2	0.352	4	0.144	46	**0.627**
As	2	0.151	19	**0.439**	39	0.382
Erdos	2	0.049	2	0.001	24	**0.422**
Scere	2	0.085	3	0.053	32	**0.196**
Ecoli	2	0.052	2	0.002	25	**0.224**
Celeg	2	0.019	2	0.005	32	**0.309**

Table 3. Biological significant clusters detected by different clustering algorithms

Molecular Function Annotations		P-value
RSGC	structural constituent of ribosome	1.2e-17
	acetylcholine receptor activity	3.1e-6
	protein binding	0.002
PaCCo	structural constituent of ribosome	2.3e-9
Spectral	structural constituent of ribosome	1.3e-19

Biological Processing Annotations		P-value
RSGC	embryo development	2.1e-24
	reproduction	2.9e-11
	growth	0.004
	multicellular organismal reproductive pro.	0.006
	protein localization	0.007
	morphogenesis of an epithelium	0.007
	germ cell development	0.009
	translation	0.011
	inductive cell migration	0.045
PaCCo	reproduction	3.7e-20
	embryo development ending in birth	6.8e-18
Spectral	reproduction	3.1e-37

of defined terms representing gene product properties on three vocabularies of annotations: Molecular Function, Biological Process and Cellular Component. Researchers can apply P-value to demonstrate the biological significance, which is defined as the probability to observe by chance at least x elements at the intersection between the query set and the reference set [3].

Under the evaluation with Molecular Function annotations, *RSGC* finds three clusters which are enriched for three molecular functions. In contrast, *PaCCo* and *Spectral* only obtain one biological significance cluster for molecular functions. In addition, for all three approaches, they find a significant cluster enriched for the function *structural constituent of ribosome*, where the P-values are 1.2e-17, 2.3e-9 and 1.3e-19 for *RSGC*, *PaCCo* and *Spectral* respectively. In addition,

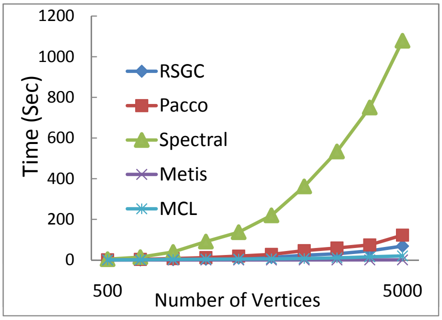


Fig. 5. The runtime of the different graph clustering algorithms

RSGC finds another two clusters enriched for *protein binding* (P-value = 2.5e-03), *acetylcholine receptor activity* (P-value = 3.1e-06). Therefore, *RSGC* can detect more clusters which make sense biologically. Similarly, we also evaluate the clusters using biological processing annotations. Here, *RSGC* successfully obtains 9 significant clusters which are enriched for biological processing while *PaCCo* and *Spectral* have two and one significant clusters respectively. All of *RSGC*, *PaCCo* and *Spectral* methods discover one significant cluster which is enriched for the term *reproduction* with the P-values of 2.8e-11, 3.7e-20, 3.1e-37 respectively. Moreover, *RSGC* also reveals another 8 significant clusters enriched for different biological process, such as *embryo development*, *multicellular organismal reproductive process* , *morphogenesis of an epithelium*, etc. Please refer to Table 3 for details.

4.5 Runtime

For runtime comparisons, we generated several synthetic data sets, where the number of clusters k varied ranges from 10 to 50 and each cluster contained 100 vertices. Approximately 30 % of the intra cluster edges were connected and 5% inter cluster edges were linked. To obtain more accurate runtime results, for each method, each data set was processed for 10 times and then found the mean of the 10 rounds. Fig. 5 clearly shows that *RSGC* is faster than *Spectral* and *PaCCo*. However, *RSGC* is slightly slower than the parameter dependent approach MCL and $Metis$.

5 Conclusions

In this paper, we introduce *RSGC*, a natural graph clustering algorithm based on synchronization. The key idea is to consider the graph clustering as a dynamic process towards synchronization. The extensive experiments demonstrate that *RSGC* algorithm has several desirable properties: *RSGC* provides a natural way for graph clustering, where the proposed interaction model well fits the real-world networks, such as the interaction weights and range. Relaying on the proposed interaction model, *RSGC* allows discovering graph clusters with arbitrary size and density without any data distribution assumption. *RSGC* is robust against noise vertices or outliers.

Acknowledgments. Junming Shao is supported by the Alexander von Humboldt-Foundation.

References

1. Arenas, A., Diaz-Guilera, A., Perez-Vicente, C.J.: Synchronization reveals topological scales in complex networks. Phys. Rev. Lett. 96(11), 1–4 (2006)
2. Böhm, C., Plant, C., Shao, J., Yang, Q.: Clustering by synchronization. In: KDD, pp. 583–592 (2010)
3. Brohee, S., Faust, K., Lima-Mendez, G., Vanderstocken, G., van Helden, J.: Network analysis tools: from biological networks to clusters and pathways. Nat. Protoc. 3, 1616–1629 (2008)
4. Bae, C.S., Kim, C.S., Tcha, H.J.: Synchronization clustering algorithm for identifying interesting groups of genes from cell cycle expression data. BMC Bioinformatics 9(56) (2008)
5. Chakrabarti, D., Papadimitriou, S., Modha, D.S., Faloutsos, C.: Fully automatic cross-associations. In: KDD, New York, pp. 79–88 (2004)
6. Dongen, S.: A cluster algorithm for graphs. Technical report, CWI (Centre for Mathematics and Computer Science), The Netherlands (2000)
7. Evans, T.: Clique graphs and overlapping communities. Journal of Statistical Mechanics, P12037 (2010)
8. Faloutsos, C., Lin, K.: Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In: SIGMOD (1995)
9. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. PNAS 99(12), 7821–7826 (2002)
10. Jaccard, P.: Distribution de la flore alpine dans la Bassin de Dranses et dans quelques regions voisines. Bulletin de la Société Vaudoise des Sciences Naturelles 37, 241–272 (1901)
11. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal on Scientific Computing 20, 359–392 (1998)
12. Kuramoto, Y.: Chemical oscillations, waves, and turbulence. Springer, Berlin (1984)
13. Mueller, N., Haegler, K., Shao, J., Plant, C., Böhm, C.: Weighted graph compression for parameter-free clustering with pacco. In: SDM (2011)
14. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: NIPS 14, pp. 849–856 (2001)
15. Shao, J., Plant, C., Yang, Q., Boehm, C.: Detection of Arbitrarily Oriented Synchronized Clusters in High-Dimensional Data. In: ICDM 2011, pp. 607–616 (2011)
16. Shao, J., Böhm, C., Yang, Q., Plant, C.: Synchronization based outlier detection. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010, Part III. LNCS (LNAI), vol. 6323, pp. 245–260. Springer, Heidelberg (2010)
17. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: is a correction for chance necessary? In: ICML 2009, New York, NY, USA, pp. 1073–1080 (2009)
18. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS, vol. 17, pp. 1601–1608 (2004)

Efficient Mining of Combined Subspace and Subgraph Clusters in Graphs with Feature Vectors

Stephan Günnemann, Brigitte Boden, Ines Färber, and Thomas Seidl

RWTH Aachen University, Germany
{guennemann,boden,faerber,seidl}@cs.rwth-aachen.de

Abstract. Large graphs are ubiquitous in today's applications. Besides the mere graph structure, data sources usually provide information about single objects by feature vectors. To realize the full potential for knowledge extraction, recent approaches consider both information types simultaneously. Thus, for the task of clustering, combined clustering models determine object groups within one network that are densely connected and show similar characteristics. However, due to the inherent complexity of such a combination, the existing methods are not efficiently executable and are hardly applicable to large graphs.

In this work, we develop a method for an efficient clustering of combined data sources, while at the same time finding high-quality results. We prove the complexity of our model and identify the critical parts inhibiting an efficient execution. Based on this analysis, we develop the algorithm EDCAR that approximates the optimal clustering solution using the established GRASP (Greedy Randomized Adaptive Search) principle. In thorough experiments we show that EDCAR outperforms all competing approaches in terms of runtime and simultaneously achieves high clustering qualities. For repeatability and further research we publish all datasets, executables and parameter settings on our website[1].

1 Introduction

In recent years, real world networks have become bigger and also more numerous. Their growing availability motivated researchers and practitioners to analyze and use them for several purposes. One aim is the cluster analysis of graph data, which can be done in various ways [1] including the task of mining densely connected subgraphs hidden in one large graph. This task is useful for, e.g., social network analysis. Besides partitioning approaches [10, 9] some methods assume that the given graph naturally divides into (possibly overlapping) subgraphs of certain patterns, e.g. cliques or γ-quasi-cliques [20, 8].

Restricting the considerations to the nodes' relations only, however, does not realize the full potential for knowledge extraction. Usually for all objects a variety of additional information is available in form of attribute data (cf. Fig. 1).

[1] http://dme.rwth-aachen.de/EDCAR

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 261–275, 2013.

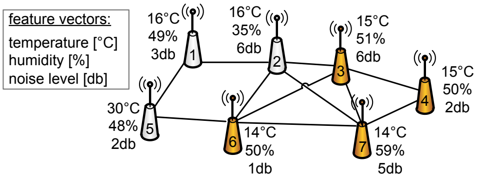


Fig. 1. Exemplary graph with feature vectors and a combined subspace and dense subgraph cluster (located in subspace $\{\text{temperature}, \text{humidity}\}$)

This information allows for finding homogeneous node sets. In order to gain more informative patterns it is preferable to consider relationships together with shared characteristics. As shown, e.g., in [6, 18], clustering methods using both information sources can outperform methods using just a single one. Recently introduced techniques aim at combining traditional clustering (using attributes) and dense subgraph mining (using relationships). They group objects based on a high connectivity as well as on a high similarity concerning their attribute values. This responds to the requirements of many applications: To reduce energy consumption in sensor networks, the long distance reports of connected sensors with similar measurements can be accumulated and transfered by just one representative. In systems biology, functional modules can be determined, which are groups of highly interacting genes with similar expression levels. In social networks, closely related friends with similar interests are useful for target marketing.

While the domain's data usually represents a multitude of different recorded characteristics, not all of them need to be relevant for each cluster. In, e.g., social networks it is very unlikely that people are similar within all of their characteristics. In Fig. 1 the sensors $3, 4, 6, 7$ are highly connected and they show similarity in two of their three measurements. In such scenarios, applying full-space clustering leads to questionable clustering results since irrelevant dimensions strongly obfuscate the clusters. Subspace clustering methods solve this problem by finding clusters in their locally relevant subspace projections of the attribute data [7]. Consequentially, recent approaches [12, 2–4] combine the paradigms of *dense subgraph mining* and *subspace clustering.*

These methods enable us to detect more meaningful clusters in the data, like, e.g., the sensor group $3, 4, 6, 7$ in Fig. 1. However, combining the paradigms of dense subgraph mining and subspace clustering poses several efficiency challenges. First, analyzing subspace projections is inherently hard since the number of subspaces grows exponentially in the number of attributes. Second, as shown in [2], to obtain high quality clusterings, an unbiased synthesis of both paradigms has to be conducted. Thus, the clustering process has to realize a complex optimization to fairly trade off the cluster properties 'size', 'density', and 'dimensionality'. Last, often an overlap between clusters is reasonable since objects can belong to multiple clusters when regarding different attribute subsets. Musicians of an orchestra, e.g., may share similar musical interests but probably will practice sports with different persons. However, if the clustering model allows clusters to overlap, it is indispensable to avoid redundancy induced by highly overlapping clusters. As known from usual subspace clustering, redundancy elimination is highly complex [11, 13].

As we have seen so far, for a proper combination of subspace clustering and dense subgraph mining, a model has to handle numerous aspects. Although mostly not accommodating all requirements, previous approaches already have high runtime and space consumptions. Thus, an execution on large datasets (if possible at all) is not efficient. In our work we deal with all the aforementioned aspects, but lay special focus on the *efficiency challenges*.

We start by taking the idea of GAMer [2] to the next level. While GAMer restricts the underlying clustering model to just greedily select good clusters for the result, which does not necessarily result in the most interesting clustering, we aim for a globally optimizing clustering model. Since even the previous models are rarely efficiently computable, it is not surprising that such a model, aiming at a global optimization, has a high complexity. We therefore analyze our model's complexity to identify the most critical parts, which inhibit an efficient execution. We substitute these critical parts through highly efficient heuristics that, however, influence the clustering quality only marginally. Thorough experiments demonstrate that our algorithm not only is far superior to all other approaches in terms of runtime but also shows better quality in nearly all experiments. Our main contributions are: (a) We develop a novel clustering model for a result having globally maximal quality, allowing clusters to overlap in general, and avoiding redundancy (b) We propose the efficient algorithm EDCAR exploiting the GRASP principle and approximating the optimal result.

2 Related Work

Recently, clustering methods have been introduced analyzing *graph data in combination with attribute data.* [6] transforms the network into a distance and combines it with the original feature distance. Afterwards any distance-based clustering method can be applied. The clusters are difficult to interpret since they do not have to obey a certain graph structure. In [19] the attribute information is transformed into a graph and densely connected subgraphs are mined by combining this novel graph with the original one. The work of [18] uses a combined objective function extending the modularity idea. All three approaches [6, 19, 18] perform full-space clustering on the attributes. [21] enriches the graph by further nodes corresponding to (categorical) attribute values and connects them to nodes showing this value. The clustered objects are only pairwise similar and no specific relevant dimensions can be defined. Furthermore, the previous methods determine disjoint clusters.

Only a few approaches deal with *subspace clustering and dense subgraph mining.* CoPaM's [12] combination of both paradigms, however, is not sound since it solely maximizes the number of nodes; the density of subgraphs and the subspace dimensionality are incidental. Furthermore, CoPaM does not eliminate redundancy, which fast leads to an overwhelming result size. The GAMer approach [2] simultaneously considers the density, the size, and the dimensionality of clusters by trading off these characteristics. Furthermore, GAMer uses a redundancy model to confine the result to a manageable size. A disadvantage, however, is the simple determination of the final clustering: GAMer does not globally examine the result but simply successively adds (in a greedy manner) clusters to the result. Thereby, the resulting clustering does not necessarily correspond to the most interesting one. In [3, 4] a cluster definition has been introduced for finding arbitrarily shaped subspace clusters in graphs with feature vectors. The work uses the same redundancy model as proposed in [2].

The major drawback of all methods is their high runtime and large space requirement, which prevents an application on larger datasets.

3 Maximum Quality Clustering

EDCAR (**E**fficient **D**etermination of **C**lusters regarding **A**ttributes and **R**elationships) realizes a novel clustering model. The model is based on the cluster definition introduced and already verified for its effectiveness in GAMer [2]. The input of our model is a vertex-labeled graph $G = (V, E, l)$ with vertices V, edges $E \subseteq V \times V$ and a labeling function $l : V \to \mathbb{R}^d$ where $Dim = \{1, \ldots, d\}$ is the set of dimensions. We assume an undirected graph without self-loops. We use $l(O) = \{l(o) \mid o \in O\}$ to denote the set of vectors that is associated to the set of vertices $O \subseteq V$.

3.1 Clustering Model

Our method combines objectives from subspace clustering and dense subgraph mining. Thus, the desired clusters are sets of objects $O \subseteq V$ that are meaningful subspace clusters in the attribute space and also form dense subgraphs within the input graph. For identifying subspace clusters, we adapt the cell-based model of DOC [15]. According to this definition the values of all objects in a subspace cluster vary at most by a threshold w in the relevant dimensions. For identifying dense subgraphs, we use the definition of quasi-cliques [8]. The density of a quasi-clique is determined by $\gamma(O) = \frac{\min_{v \in O} deg^O(v)}{|O|-1}$, where $deg^O(v) = |\{o \in O \mid (v, o) \in E\}|$ is the vertex degree restricted to the set O.

Definition 1. (Twofold cluster [2]) *A twofold cluster $C = (O, S)$ is a set of vertices $O \subseteq V$ and a set of dimensions $S \subseteq Dim$ with the following properties*

- *$(l(O), S)$ is a subspace cluster with dimensionality $|S| \geq s_{min}$*
- *O is a quasi-clique with density $\gamma(O) \geq \gamma_{min}$*
- *the induced subgraph of O is connected and $|O| \geq n_{min}$*

The resulting clusters are meaningful in the attribute space as well as in the graph. For example in Fig. 2 (choosing $w = 0.5$, $n_{min} = 3$, $\gamma_{min} = 0.4$ and $s_{min} = 2$) the vertex set $C_1 = \{v_1, v_2, v_4, v_5, v_6, v_7\}$ is a valid twofold cluster with

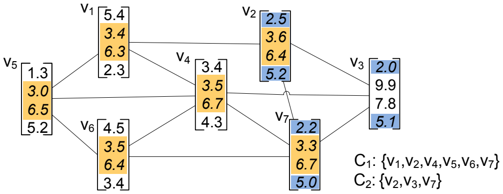


Fig. 2. Exemplary twofold clusters

the relevant dimensions 2 and 3 (marked in orange). The set $C_2 = \{v_2, v_3, v_7\}$ is a twofold cluster with the relevant dimensions 1 and 4 (marked in blue).

Based on the above definition, the number of node sets fulfilling this definition is potentially very large and probably many clusters will overlap (e.g. C_1 and C_2 in Fig. 2). Furthermore, some subsets of the clusters are twofold clusters as well, e.g. $\{v_1, v_4, v_5, v_6\}$. Though it makes sense to allow overlapping clusters in general since one node can belong to several meaningful groups, clusters that are too similar to each other often contain nearly the same information.

Since these redundant clusters are not beneficial but obstructing, they should be excluded from the result. To identify a redundant cluster C w.r.t. another cluster C', several properties have to apply. First, the structural information of the corresponding clusters has to be similar, i.e. they have to share a large portion of their vertices and their dimensions. Second, the cluster C should be less interesting than the cluster C'; otherwise one would prefer C. Formally, the redundancy of C w.r.t. C' is based on the following relation:

Definition 2. (Redundancy relation) *Given the redundancy parameters $r_{obj} \in [0,1]$ and $r_{dim} \in [0,1]$, the binary redundancy relation $\prec_{red}$ is defined by:*

For all twofold clusters $C = (O, S), C' = (O', S')$:

$$C \prec_{red} C' \Leftrightarrow Q(C) < Q(C') \wedge \frac{|O \cap O'|}{|O|} \geq r_{obj} \wedge \frac{|S \cap S'|}{|S|} \geq r_{dim}$$

Using the parameters r_{obj} and r_{dim} the user can determine to which extent two clusters may overlap without being defined as redundant. For example, with $r_{obj} = r_{dim} = 0.5$ the cluster C_2 would not be redundant w.r.t. C_1. Although many of C_2's nodes are covered by C_1, the relevant dimensions of the two clusters do not overlap. With the values $r_{obj} = 0.5$ and $r_{dim} = 0$, C_2 would be redundant w.r.t. C_1.

Cluster selection based on global optimization. Our goal in selecting the final clustering is a solution, that (a) does not contain clusters that are redundant to each other, i.e. it has to be redundancy-free, and (b) is most interesting. While the GAMer method greedily selects clusters according to their quality, we perform a more sophisticated selection. Instead of deciding locally which cluster to select next for the result, we perform a global optimization to get the most interesting clustering. We, thus, do not prefer the selection of *single* interesting *clusters*, since that carries the risk of selecting only uninteresting clusters afterwards, but we select the *overall* most interesting *clustering*. Correspondingly, we require of our *Result* that the sum of its clusters' qualities is maximal compared

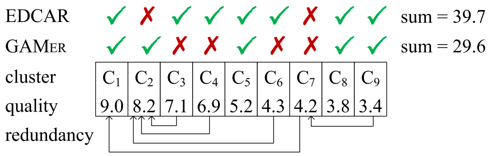

EDCAR ✓ ✗ ✓ ✓ ✓ ✓ ✗ ✓ ✓ sum = 39.7
GAMER ✓ ✓ ✗ ✗ ✓ ✗ ✗ ✓ ✓ sum = 29.6
cluster
C1 C2 C3 C4 C5 C6 C7 C8 C9
quality
9.0 8.2 7.1 6.9 5.2 4.3 4.2 3.8 3.4
redundancy

Fig. 3. Global optimization in EDCAR vs. greedy selection in GAMER

to all other possible clusterings. Formally, the maximum quality clustering is defined as follows:

Definition 3. (Maximum quality clustering) *Given the set of all twofold clusters Clusters, the maximum quality clustering Result* $\subseteq$ *Clusters fulfills*

- *(redundancy-freeness)* $\neg\exists C_i, C_j \in Result : C_i \prec_{red} C_j$
- *(maximum quality sum)* $\neg\exists Res' \subseteq Clusters : Res'$ *fulfills the redundancy-free property and* $\sum_{C_i \in Res'} Q(C_i) > \sum_{C_i \in Result} Q(C_i)$

Fig. 3 shows an example for the final clusterings of GAMER and EDCAR: Nine clusters, their quality values, and the redundancy relation are illustrated. The quality sums of the overall clusterings are depicted on the right. GAMER selects the cluster C_2 since it is not redundant w.r.t. any other cluster. A greedy selection according to the quality values is performed. In EDCAR, cluster C_2 is not selected for the final clustering. While C_2 has a high quality itself, its admittance would prohibit the clusters C_3, C_4, and C_6. However, by including these clusters and excluding C_2, our final clustering has a higher quality (39.7 vs. 29.6). As the example illustrates, EDCAR optimizes the interestingness of the overall clustering, which can yield better results but is computationally more challenging.

3.2 Complexity Analysis

The complexity of our clustering model is given by the following two theorems (proofs on the web). First, the overall complexity of our model, i.e. of generating the twofold clusters *and* selecting the maximum quality clustering, is #P-hard.

Theorem 1. *Given a vertex-labeled graph* $G = (V, E, l)$*, determining the maximum quality clustering according to Def. 3 is #P-hard w.r.t.* $|V|$.

Second, even if the set of twofold clusters *Clusters* is given, selecting the maximum quality clustering $Result \subseteq Clusters$ (cf. Def. 3) is NP-complete w.r.t. the input size.

Theorem 2. *Given a set of twofold clusters Clusters, selecting the maximum quality clustering according to Def. 3 is NP-complete w.r.t.* $|Clusters|$.

Conclusions. From Theorem 1 we can infer that the input size $|Clusters|$ can be exponential in $|V|$. Overall we can identify *two parts* that, especially in combination, *prevent an efficient determination* of the optimal solution: the tremendous amount of clusters used as candidates for the optimal clustering and the complexity of the selection process for a final subset of these clusters.

4 The Efficient EDCAR Algorithm

As shown, efficiently determining a maximum quality clustering is not possible. Thus, we develop the heuristic algorithm EDCAR to ensure an efficient execution. We have to tackle two major challenges: First, we have to reduce the number of result candidates (Section 4.1). We cannot use the whole set *Clusters* of exponentially many candidates as the input for the selection procedure. Second, we have to resolve the NP-hardness of the selection process itself (Section 4.2).

4.1 Reduce the Number of Result Candidates

This first phase generates the cluster candidates among which the subsequent process chooses the final clustering. The goal is to *efficiently* determine a set of twofold clusters that is of *manageable size* and of *high quality*.

To analyze sets of vertices whether they are twofold clusters, we enumerate them using the set enumeration tree [17]. An exemplary tree for a graph with four vertices is shown in Fig. 4. Each node of the tree represents a set of vertices $O \subseteq V$. Each node O is associated with a candidate set $cand_O$. A child node O' extends its parent node O through one of the vertices in $cand_O$. Thus, the subtree of a node O represents all potential clusters X with $O \subset X \subseteq O \cup cand_O$. By pruning a vertex v from the candidate set of a node O, the search space can be reduced. If we were able to remove e.g. the vertex v_3 from the set $cand_{\{v_1\}}$, the highlighted subsets in Fig. 4 would disqualify themselves as clusters without further analysis. EDCAR employs all pruning methods of [2].

We want to avoid analyzing *each* node along *each* (non-pruned) path in the set enumeration tree since this could lead to an exponential number of twofold clusters which are used as candidates for the final clustering. To reduce the number of candidates we implement two different strategies. In the first step, we avoid analyzing all paths of the tree by *systematically determining single paths* along which interesting clusters can be expected. By selecting a polynomial number of paths we will also only get a polynomial number of candidates. This

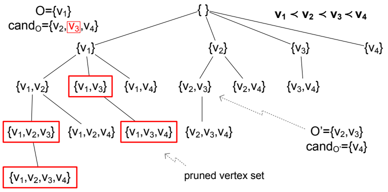


Fig. 4. Set enumeration tree

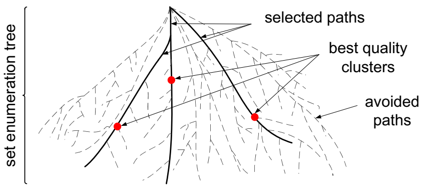


Fig. 5. Reducing the number of candidates

Algorithm 1. generateCandidates()

```
$Cands \leftarrow \{\}$
for( $1 \ldots maxClus$ )
  $\alpha \leftarrow$ rand($[0 \ldots 1]$) // trade off greedy & randomized
  $O \leftarrow \{\}$ // root of set enumeration tree
  while( $true$ ) // generate path $P$
    determine $cand_O$ and prune (cf. [2])
    if( $cand_O = \emptyset$ ) break;
    determine $RCL_O$ as in Equation 1
    $v \leftarrow$ rand($RCL_O$)
    $O \leftarrow O \cup \{v\}$ // extend path $P$ by $v$
  select cluster $C^+$ along path $P$ with highest quality
  $Cands \leftarrow Cands \cup \{C^+\}$
return $Cands$
```

path selection is illustrated in Fig. 5 by the solid lines. Even though this method reduces the number of clusters considerably, this set is still unnecessarily large and will be reduced in a second step. Since along each selected path the object sets successively grow by one vertex (cf. Fig. 4), the clusters along this path are most likely redundant to each other. Using all these clusters as the input for the selection step is needless since most clusters will be discarded anyway. Based on the definition of our redundancy relation we know that clusters with high quality are preferred. Thus, instead of using all clusters, we select along each path just the cluster with the highest quality (cf. dots in Fig. 5). Overall, we realize by our two strategies that only few candidates are used as the input for the cluster selection step. In the remaining of this section, we present more details on our path selection technique.

GRASP for efficient path selection. To systematically (and efficiently) achieve a selection of interesting paths, we adapt the GRASP principle (Greedy Randomized Adaptive Search Procedure) [14, 16], Naively, one could randomly determine a path. This, however, does not assure to generate high quality clusters. Alternatively, one could decide at node O which successor $v \in cand_O$ (potentially) leads to a good cluster and one descends in the subtree with the highest potential. We use a function $g(v|O)$ to estimate the potential of each node $v \in cand_O$ w.r.t. the current set O. The definition of $g(v|O)$ will be derived in the next subsection. This approach corresponds to a greedy construction of the path. The huge advantage of this greedy method is that the graph structure and the cluster definition can be incorporated into the estimation function g to rate the potential of the path. Thus, it corresponds to an *informed search* and high quality clusters can be expected. Disadvantageously is the risk of reaching only local maxima and generating always very similar paths. These problems do not hold for the randomized approach, which is able to generate a *diversity of paths* in an *uninformed* fashion.

To exploit the advantages of both methods (informed and randomized search), we use the GRASP principle, which acts as a metaheuristic to combine them. Several studies show that this principle often leads to optimal or nearly optimal solutions [14]. According to the GRASP principle, we first construct a restricted candidate list (RCL) corresponding to a *set* of potentially meaningful vertices for expanding the path. Afterwards, we randomly select one vertex $v \in RCL_O$ to descend into a subtree. Formally,

$$RCL_O = \{v \in cand_O \mid g(v|O) \geq g_m + \alpha \cdot (g_M - g_m)\} \tag{1}$$

with $g_m = \min\limits_{v \in cand_O} \{g(v|O)\}$ and $g_M = \max\limits_{v \in cand_O} \{g(v|O)\}$.

By choosing α=1 we can simulate the greedy approach whereas α=0 corresponds to a completely randomized selection. The determination of a single path is shown in Algorithm 1, line 5 to 10: In line 6 we determine the candidate set $cand_O$ for the current vertex set O, which is then reduced using the pruning techniques from [2]. Next we determine the RCL as described above and add a randomly selected node to O. This procedure is repeated until $cand_O = \emptyset$, i.e. no more nodes can be added to the path.

As depicted in Fig. 5 we want to descend in several paths. This is done by line 2 and we use a randomly determined α to trade off the two GRASP principles for each novel path, leading to more stable results [16]. Overall, we efficiently generate different paths containing high quality clusters based on the estimated potential.

Potential of Paths. At last, we have to determine the potential of a subtree. Since our goal is to maximize the sum of qualities, we want to use the quality as our estimation function $g(v|O)$. If efficiency was not required, one could use $g'(v|O) = \gamma(O \cup \{v\})^a \cdot |O \cup \{v\}|^b \cdot |S(O \cup \{v\})|^c$ where $S(X)$ denotes the subspace of the corresponding vertex set. However, efficiency is crucial in our case. While the size and the subspace can be efficiently determined, the exact density $\gamma(O \cup \{v\})$ is computationally expensive. Keep in mind that $g(v|O)$ has to be evaluated for each $v \in cand_O$. Therefore, we approximate the density $\gamma(O \cup \{v\})$ of the potential cluster $O \cup \{v\}$ by the lower bound $\overline{\gamma}(O, v) := \frac{\min\{\min_{o \in O} deg^O(o), deg^O(v)\}}{|O|} \leq \gamma(O \cup \{v\})$. The density approximation $\overline{\gamma}(O, v)$ can be efficiently computed for different vertices v because it is mostly independent of v. We only have to compute the term $deg^O(v)$. Our overall estimation function is $g(v|O) = \overline{\gamma}(O, v)^a \cdot (|O| + 1)^b \cdot |S(O \cup \{v\})|^c$

4.2 GRASP for Efficient Clustering Selection

So far, we reduced the number of candidates used as the input for the cluster selection step. Now we approximate the maximum quality clustering (Def. 3)) itself since its determination is NP-hard. We again use the GRASP principle. Therefore, we relax Def. 3 by only demanding the resulting clustering Res to be redundancy-free and maximal: $(\neg\exists C, C' \in Res : C \prec_{red} C') \wedge (\forall C \in Cands \backslash Res : \exists C' \in Res : C \prec_{red} C' \vee C' \prec_{red} C)$. Starting with an empty result Res, we now successively add further clusters $C \in Cands$ based on the GRASP principle. Since our goal is to find clusterings with a high quality sum, we instantiate the estimation function $h(C|Res)$, which assesses the potential of adding C to Res, by the quality of clusters: $h(C|Res) = Q(C)$. This value is already given at this time. Thus, no additional computation has to be done. The

Algorithm 2. selectClustering(...)

```
1:  input: set of clusters $Cands$
2:  $Res \leftarrow \{\}$ // (preliminary) result
3:  $\alpha \leftarrow$ rand([0 . . . 1])
4:  while( $Cands \neq \emptyset$ )
5:    $g_m = \min_{C \in Cands} h(C|Res)$, $g_M = \max \ldots$
6:    $RCL = \{s \in Cands \mid h(C|Res) \geq g_m + \alpha(g_M - g_m)\}$
7:    $C \leftarrow$ rand($RCL$)
8:    $Res \leftarrow Res \cup \{C\}$
9:    $Cands \leftarrow \{C \in Cands \mid \neg\exists C' \in Res : C \prec_{red} C' \vee C' \prec_{red} C\}$
10: // improve clustering by local search
11: for( $NewRes \in Neighborhood(Res)$ )
12:   if( $Q(NewRes) > Q(Res)$ ) // higher quality sum
13:     $Res \leftarrow NewRes$
14:     goto line 10 // efficient first-improving strategy
15: return $Res$
```

Algorithm 3. EDCAR algorithm

```
$Res \leftarrow \{\}$ // preliminary result
do
  $Cands \leftarrow$ generateCandidates()
  $Tmp \leftarrow Res \cup Cands$
  $Res \leftarrow$ selectClustering($Tmp$)
while( $Cands \neq \emptyset$ )
return $Res$
```

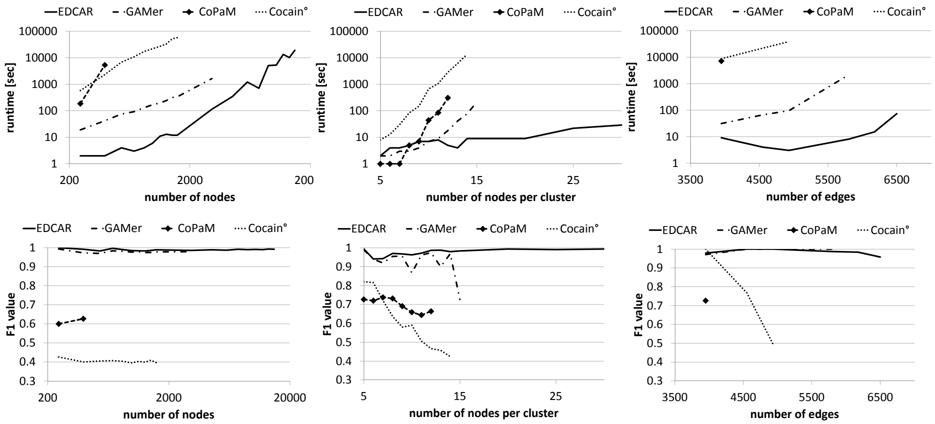
pseudo code for selecting the subset is given in Algorithm 2. Since our model requires redundancy-freeness, we are able to remove in line 9 all clusters that induce such a redundancy. These clusters can no longer be added to Res.

To further increase the overall quality, we conduct in line 10-14 a local search on the set of valid clusterings. The idea is to replace a cluster $C \in Res$ by a set of not yet selected clusters New_C to get the potentially better clustering $Res \backslash \{C\} \cup New_C$. The set New_C is built by collecting clusters from $Cands \backslash Res$, in decreasing order w.r.t. their quality values, as long as the redundancy-freeness property of the overall result $Res \backslash \{C\} \cup New_C$ is not violated. Thus, an efficient greedy approach can be used to generate New_C. Formally, for each cluster $X \in Cands \backslash Res$ *not* selected for New_C it holds: $X \notin New_C \Leftrightarrow \exists C' \in New_C : (X \prec_{red} C') \vee \exists C' \in Res \backslash \{C\} : (X \prec_{red} C' \vee C' \prec_{red} X)$. The overall neighborhood of a clustering Res is the whole set of such generated alternatives $Neighborhood(Res) = \{Res \backslash \{C\} \cup New_C \mid \forall C \in Res\}$. If no better clustering in the neighborhood exists, we have reached a local maximum and the cluster selection in this iteration is finished.

4.3 Overall Processing Scheme

The two phases of our method, generating a small number of candidates and selecting the resulting clustering based on these candidates, lead to an overall efficient execution. Since we select just the one cluster with the highest quality along each path, however, lower quality clusters do not get the chance to be selected for the result. Nevertheless, also clusters with lower qualities can contribute to the overall result, as C_9 in Fig. 3. To give these low-quality but valuable clusters the chance to be considered as result candidates, we repeat both phases recurrently (cf. Algorithm 3).

In the first iteration no information is given ($Res = \emptyset$) and we select the clusters with highest quality along each path. In subsequent iterations Res is used to avoid considering redundant candidates. We thus block redundant parts of a path and select the most interesting cluster among the remaining non-redundant ones as additional candidate. Overall, we generate in each iteration only candidates



(a) w.r.t. database size (b) w.r.t. cluster size (c) w.r.t. cluster density

Fig. 6. Scalability and Quality w.r.t. different data or ground truth characteristics

fulfilling $\forall C \in Cands : \neg\exists C' \in Res : C \prec_{red} C'$. It is very likely that some of these clusters can be added to the final clustering. Thus, we perform the clustering selection phase on the enriched set $Cands \cup Res$ to get the novel preliminary result $Res^* \subseteq Cands \cup Res$. Overall, our processing interweaves the generation and the selection of clusters by cyclically invoking both phases. The method automatically terminates if no further non-redundant candidates can be found.

5 Experiments

We compare EDCAR with GAMER [2] and CoPaM [12]; both consider subspaces and dense subgraphs. As a further competitor we use the extension Cocain° of Cocain [20] as described in [2]. Efficiency is measured by the approaches' runtime. All experiments were conducted on Opteron 2.3GHz CPUs using Java6 64 bit. Methods that did not finish within two days were aborted. Clustering quality is calculated via the F1 value [5]. We use several public real world datasets and synthetic data, by default with 80 clusters, each with 15 nodes, a density of 0.6 and 5-10 relevant dimensions out of 20 dimensions. 6% of the clusters' nodes overlap. *We provide all datasets with descriptions, executables, and parameter settings on our website.*

Database size. First, we vary the number of vertices in the graph by increasing the number of clusters and keeping the number of objects per cluster fixed. As depicted in Fig. 6(a) (top), EDCAR is several orders of magnitude faster than all competing approaches (note the logarithmic scale on both axes). EDCAR is the only method applicable on large data sets. Especially CoPaM is no longer executable for these settings since its limited redundancy model leads to an impracticably large amount of clusters. GAMER scales worse than EDCAR, too. It has to analyze the complete set of all twofold clusters, leading to a high runtime.

Although EDCAR uses an even more complex clustering model, the runtime is lower since we systematically generate and select only the most interesting clusters.

Besides EDCAR's high efficiency, we observe in Fig. 6(a) (bottom) its high effectiveness. Despite the used approximations, the quality of EDCAR is similar or even higher than that of GAMer. The remaining approaches CoPaM and Cocain° achieve only low qualities, as also shown in [2]. This experiment has shown that EDCAR is also applicable on large datasets. Though, for the following experiments we chose medium-sized datasets to enable a comparison with the other algorithms.

Cluster size. In this experiment, we keep the number of clusters fixed but increase the number of vertices per cluster. This setting is more challenging since larger clusters correspond to longer paths in the set enumeration tree. In Fig. 6(b) (top) we observe only a slow increase in runtime for EDCAR. Since the number of hidden clusters is fix, the number of required iterations in EDCAR is almost constant (about 15). All competing approaches show heavily increasing runtimes and are not applicable at an early stage. Fig. 6(b) (bottom) shows nearly perfect quality for EDCAR. The qualities of the other methods decrease to different extents. The advantage of our novel clustering model becomes apparent.

Graph density. In Fig. 6(c) we increase the graph's density by adding edges between the clustered nodes. Again, EDCAR is orders of magnitudes faster confirming the usefulness of our solution.

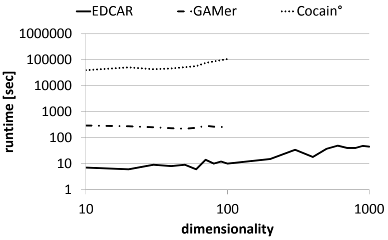


Fig. 7. Data dimensionality

Data dimensionality. In Fig. 7 we increase the data's dimensionality. Though the runtimes of the competing methods do not increase significantly, they are not applicable for larger datasets due to the extreme memory usage. CoPaM is not applicable at all. The other methods have to manage a tremendous amount of clusters whereas our algorithm generates incrementally a small set of clusters. Overall, all experiments indicate that EDCAR achieves far better runtimes than all competitors. EDCAR is the only method applicable to large datasets. At the same time the results of our approximation achieve high effectiveness.

Real world data. We use *gene data*[2] and their interactions (3548 nodes; 8334 edges; 115d), an extract of the *Arxiv database*[3] (27769; 352284; 300d), *patent information*[4] (492007; 528333; 5d), and a co-author graph extracted out of the *DBLP database*[5] (133097; 631384; 2695d). Since for real world data no hidden clusters are given, we analyze in Fig. 8 different properties of the clustering results (runtime, avg. number of vertices, density, dimensionality of the detected clusters). For all datasets EDCAR is orders of magnitude faster than the other methods. GAMer is only applicable on three of the datasets. CoPaM can only be executed on the gene data and achieves extremely high runtimes. Cocain° finished on none of the datasets within 2 days. The clusters identified by EDCAR and GAMer have nearly similar properties. Thus, our approximations do not impair the clustering quality.

	Gene			Arxiv		Patent		DBLP
	EDCAR	GAMer	CoPam	EDCAR	GAMer	EDCAR	GAMer	EDCAR
runt. [s]	49	76	33060	179	945	282	574	25752
∅ vertices	9	8.8	9.67	7.7	8.2	12	11.7	9.3
∅ density	0.74	0.72	0.24	0.69	0.6	0.6	0.6	0.83
∅ dim.	15.8	15.47	12.21	5	5	3.0	3.0	3.02

Fig. 8. Clustering properties for real world data

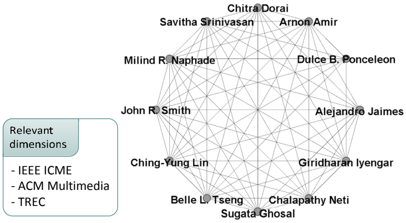


Fig. 9. Exemplary cluster for DBLP

An exemplary cluster from EDCAR's clustering result on the DBLP co-author graph is shown in Fig. 9. Here, each node represents an author, each edge corresponds to a co-authorship, and the 2695 attributes of a node indicate the conferences which an author has attended. Fig. 9 illustrates a cluster consisting of 12 authors who jointly published papers at the conferences IEEE ICME, ACM Multimedia, and TREC (i.e., the cluster is located in a 3d subspace). Please note that the cluster does not form a clique (its quasi-clique density is 0.64), thus not all authors collaborated together. EDCAR is the only method that can handle this data set; all competing methods fail due to their high runtime and space complexity.

6 Conclusion

We introduced the method EDCAR for efficiently detecting clusters showing high density in graphs as well as feature similarity in subspace projections. Our model combines subspace clustering with dense subgraph mining and performs

[2] `http://thebiogrid.org,http://genomebiology.com/2005/6/3/R22`

[3] `http://www.cs.cornell.edu/projects/kddcup/datasets.html`

[4] `http://www.nber.org/patents/`

[5] `http://dblp.uni-trier.de`

an overall optimization of the result to get the most interesting, redundancy-free clustering. Based on the proven complexity of our model, we developed the algorithm EDCAR to efficiently calculate an approximate solution. By interweaving the process of cluster generation and cluster selection, which both make use of the GRASP principle, EDCAR determines high quality clusters and ensure low runtimes. Thorough experiments demonstrate that EDCAR has high effectiveness and at the same time constantly outperforms all competing approaches in terms of efficiency.

Acknowledgments. This work has been supported by the UMIC Research Centre, RWTH Aachen University, Germany.

References

1. Aggarwal, C., Wang, H.: Managing and Mining Graph Data. Springer (2010)
2. Günnemann, S., Färber, I., Boden, B., Seidl, T.: Subspace clustering meets dense subgraph mining: A synthesis of two paradigms. In: ICDM, pp. 845–850 (2010)
3. Günnemann, S., Boden, B., Seidl, T.: DB-CSC: A density-based approach for subspace clustering in graphs with feature vectors. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part I. LNCS (LNAI), vol. 6911, pp. 565–580. Springer, Heidelberg (2011)
4. Günnemann, S., Boden, B., Seidl, T.: Finding density-based subspace clusters in graphs with feature vectors. Data Min. Knowl. Discov. 25(2), 243–269 (2012)
5. Günnemann, S., Färber, I., Müller, E., Assent, I., Seidl, T.: External evaluation measures for subspace clustering. In: CIKM, pp. 1363–1372 (2011)
6. Hanisch, D., Zien, A., Zimmer, R., Lengauer, T.: Co-clustering of biological networks and gene expression data. Bioinformatics 18, 145–154 (2002)
7. Kriegel, H.P., Kröger, P., Zimek, A.: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. TKDD 3(1), 1–58 (2009)
8. Liu, G., Wong, L.: Effective pruning techniques for mining quasi-cliques. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 33–49. Springer, Heidelberg (2008)
9. Long, B., Wu, X., Zhang, Z.M., Yu, P.S.: Unsupervised learning on k-partite graphs. In: KDD, pp. 317–326 (2006)
10. Long, B., Zhang, Z.M., Yu, P.S.: A probabilistic framework for relational clustering. In: KDD, pp. 470–479 (2007)
11. Moise, G., Sander, J.: Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering. In: KDD, pp. 533–541 (2008)
12. Moser, F., Colak, R., Rafiey, A., Ester, M.: Mining cohesive patterns from graphs with feature vectors. In: SDM, pp. 593–604 (2009)
13. Müller, E., Assent, I., Günnemann, S., Krieger, R., Seidl, T.: Relevant subspace clustering: Mining the most interesting non-redundant concepts in high dimensional data. In: ICDM, pp. 377–386 (2009)
14. Pitsoulis, L., Resende, M.: Greedy randomized adaptive search procedures. In: Handbook of Applied Optimization, pp. 168–183. Oxford University Press, New York (2002)

15. Procopiuc, C., Jones, M., Agarwal, P., Murali, T.: A Monte Carlo algorithm for fast projective clustering. In: SIGMOD, pp. 418–427 (2002)
16. Resende, M.G.C., Ribeiro, C.C.: Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications. Int. Series in Op. Research & Management Science, pp. 283–320 (2010)
17. Rymon, R.: Search through systematic set enumeration. In: KR, pp. 539–550 (1992)
18. Shiga, M., Takigawa, I., Mamitsuka, H.: A spectral clustering approach to optimally combining numerical vectors with a modular network. In: KDD, pp. 647–656 (2007)
19. Ulitsky, I., Shamir, R.: Identification of functional modules using network topology and high-throughput data. BMC Systems Biology 1(1) (2007)
20. Zeng, Z., Wang, J., Zhou, L., Karypis, G.: Coherent closed quasi-clique discovery from large dense graph databases. In: KDD, pp. 797–802 (2006)
21. Zhou, Y., Cheng, H., Yu, J.X.: Graph clustering based on structural/attribute similarities. PVLDB 2(1), 718–729 (2009)

Exploiting Temporal Information in a Two-Stage Classification Framework for Content-Based Depression Detection

Yu-Chun Shen[1], Tsung-Ting Kuo[1], I-Ning Yeh[2], Tzu-Ting Chen[3], and Shou-De Lin[1,*]

[1] Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
{r98922071,d97944007,sdlin}@csie.ntu.edu.tw
[2] St. Joseph Hospital, Yunlin, Taiwan
yehining@gmail.com
[3] National Taiwan University Hospital Yun-Lin Branch, Yunlin, Taiwan
tzuting.chen@gmail.com

Abstract. Depression has become a critical illness in human society as many people suffer from the condition without being aware of it. The goal of this paper is to design a system to identify potential depression candidates based on their write-ups. To solve this problem, we propose a two-stage supervised learning framework. The first stage determines whether the user possesses apparent negative emotion. Then the positive cases are passed to the second stage to further evaluate whether the condition is clinical depression or just ordinary sadness. Our training data are generated automatically from Bulletin Board Systems. The content and temporal features are designed to improve the classification accuracy. Finally we develop an online demo system that takes a piece of written text as input, and outputs the likelihood of the author currently suffering depression. We conduct cross-validation and human study to evaluate the effectiveness of this system.

Keywords: Depression Detection, Time Information, Text Classification.

1 Introduction

Depression has gradually become a common mental illness in the modern era. According to World Health Organization, 121 million people are affected by depression, but less than 25% of those people receive adequate treatment (Saraceno, B. 2002).

Depression is a type of mental disease without apparent symptoms, especially during the early stage (Feightner et al. 1990). Sometimes the patients do not understand their drastic mood swings are caused by depression. With the goal to combat this illness, this paper presents an early-detection mechanism that is capable of identifying potential depression cases given the written materials of the subjects.

Depression detection has been studied for a while but most of the existing depression detection systems are not fully automatic. Existing systems usually require the potential subjects to take online evaluation tests. Then a simple rule-based system

[*] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 276–288, 2013.

or a more complicated learning-based system can be designed. Such process has significant limitations in real-world usage. First, the potential candidates need to be manually identified (usually by another human). Second, even if the potential candidates can be identified, the validity of the test results remain to be questioned as the test takers may not take the test seriously.

To address the above concerns, we design a supervised-learning system for text-based depression detection. Our system does not need the potential candidates to undergo an evaluation, as it scans through their write-ups on public platforms (e.g. on blogs or BBS) to make a decision. To design such a system, there are several issues to address:

Q1. What kind of classifier should be designed? A binary classification tool that separates depression from non-depression? A multi-class classifier that distinguishes multiple mental diseases including depression? Or something else?

Q2. Given the answer from Q1, how to obtain the labeled training data?

Q3. What kinds of features are useful for this task?

Q4. What kind of evaluation procedure is considered as a sound mechanism to assess the quality of the system?

In the following sections, we will discuss the above issues to design a text-based depression detection framework.

2 Methodology

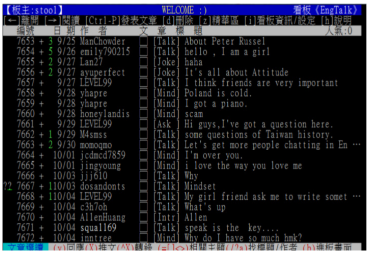
We investigate how to design a learning-based system that is capable of determining whether the author of a piece of writing is suffering or on the verge of suffering depression. Furthermore, we investigate how the temporal information can be exploited in such task.

Our training data is collected from the most popular bulletin board system (BBS) in Taiwan, the PTT. Figure 1 is a screenshot of the user interface of PTT. Due to the anonymous property of the cyber world, users are more willing to express their true feelings on web platforms than they do in the real world. This makes such public-sharing platform a good source of data for depression detection, and consequently enables our study on depression detection through text mining.

In PTT, there are more than twenty thousand boards focusing on wide range of topics, such as politics, sports, life and game. Over 1 million registered users post tens of thousands of new posts every day. The main reason we use data from PTT for depression analysis is twofold. First, there is an existing depression board, the *Prozac* board, on PTT for depressed persons to express their feelings and thoughts. Second, besides this board, there are other boards that allow users to express their feelings that can be exploited as the negative training data or as the testing platform. For example, there is a *Sad* board for sad people to share their feelings, and there is a *Diary* board that allows users to write down what are on their minds.

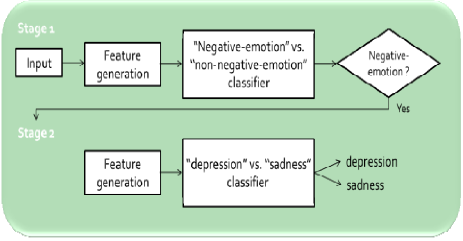
2.1 Methodology Overview

To solve Q1 as described previously, one conventional solution for depression detection is to treat this task as a binary classification problem. One can collect some positive and negative samples to train a classifier. However, it is non-trivial to choose the source of negative data for learning in this case, as there are many different kinds of non-depression content to choose from (e.g. technical papers, news, announcements, etc.). If, say, a bunch of technical papers are chosen as negative samples, then it is very possible all non-technical manuscripts (e.g. poetry) will be classified as 'depression articles'. Similar drawbacks happen to cases when news, novel, or other kinds of writings are used as negative examples. If a variety of different corpora are used as negative samples, then we will inevitably face a serious data-imbalance problem as there are significantly more negative instances than positive ones. Treating this task as a multi-class classification problem brings up similar concern as there are too many classes to be distinguished from depression.



```
【板主:stool】                WELCOME :)                看板《EngTalk》
[←]離開 [→]閱讀 [Ctrl-P]發表文章 [d]刪除 [z]精華區 [i]看板資訊/設定 [h]說明
   編號    日 期 作  者       文  章  標  題                        人氣:0
   7653 + 3 9/25 ManChowder    □ [Talk] About Peter Russel
   7654 + 5 9/26 emily790215   □ [Talk] hello , I am a girl
   7655 + 2 9/27 Lan27         □ [Joke] haha
   7656 + 2 9/27 ayuperfect    □ [Joke] It's all about Attitude
   7657 +   9/27 LEVEL99       □ [Talk] I think friends are very important
   7658 +   9/28 yhapre        □ [Mind] Poland is cold.
   7659 +   9/28 yhapre        □ [Mind] I got a piano.
   7660 +   9/28 honeylandis   □ [Mind] scam
   7661 +   9/29 LEVEL99       □ [Ask ] Hi guys,I've got a question here.
   7662 + 1 9/29 M4smss        □ [Talk] some questions of Taiwan history.
   7663 + 2 9/30 momoqmo       □ [Talk] Let's get more people chatting in En …
   7664 +  10/01 jcdmcd7859    □ [Mind] I'm over you.
   7665 +  10/01 jingyoung     □ [Mind] i love the way you love me
   7666 +  10/03 jjj610        □ [Talk] Why
?? 7667 + 110/03 dosandonts    □ [Talk] Mindset
   7668 + 110/04 LEVEL99       □ [Talk] My girl friend ask me to write somet …
   7669 +  10/04 c3h7oh        □ [Talk] What's up
   7670 +  10/04 AllenHuang    □ [Intr] Allen
   7671 +  10/04 squal169      □ [Talk] speak is the  key....
   7672 +  10/04 inntree       □ [Mind] Why do I have so much hmk?
 文章選讀  (y)回應(X)推文(^X)轉錄 (=[]<>)相關主題(/?a)找標題/作者 (b)進板畫面
```

Fig. 1. PTT bulletin board



Stage 1
Input
Feature generation
"Negative-emotion" vs. "non-negative-emotion" classifier
Negative-emotion ?
Yes
Stage 2
Feature generation
"depression" vs. "sadness" classifier
depression
sadness

Fig. 2. Methodology overview

To further investigate the above issues, we first discuss two practical usage scenarios of our system. First, as we believe our system will be most useful in situation when somebody intents to know whether a target manuscript is written by a person suffering depression. For such manuscript to be identified in the first place, it should, to some extent, already carry certain negative information. However, not all write-ups with negative emotion are written by depression patients and it is generally hard for a non-expert to distinguish between subjects suffering 'potential depression' or 'simply carry some negative thoughts'. Therefore we believe it is very important to design an automatic system for such task. That says, this usage scenario suggests that an effective depression detection engine should be able to distinguish writings that contain depressed emotion from those that contain quasi-depressed emotions.

The second usage scenario is to allow the detection system to scan through a large amount of writings (most likely crawled from the WWW) and highlight the potential candidates for early treatment. Combining with a Web Crawler, the depression detection system can act as an active detector that automatically identifies suspicious candidates for further treatment. In this case, the system needs to distinguish depression articles from many other types of writings.

To handle these two scenarios, we design a two-stage classification framework (as shown in Figure 2). In the first stage, we train an emotion classifier to classify the input manuscript into "negative-emotion" or "non-negative-emotion". Inputs being classified as negative-emotion are then passed to the second stage, which contains a "depression" or "sadness" classifier to determine whether the author should be considered as a depression candidate or simply a user with sad emotion. The proposed framework satisfies both scenarios as we need a sadness/depression detector for the first scenario and an additional positive-negative emotion detection system can be used to filter non-negative posts in the second scenario. Intuitively, the first classifier deals with a simpler task than the second classifier, because depression and sadness are closely related concepts that cannot be easily distinguished.

2.2 Feature Generation

In the first stage, we exploit only content information for classification. We chose TF-IDF values of words as the feature representation (i.e. term features). Shown in Equation 1 and Equation 2, $n_{i,j}$ is the frequency of term t_i occurring in a person p_j's written articles. In equation 2, the denominator in idf equation represents the number of people whose articles contain the term t_i.

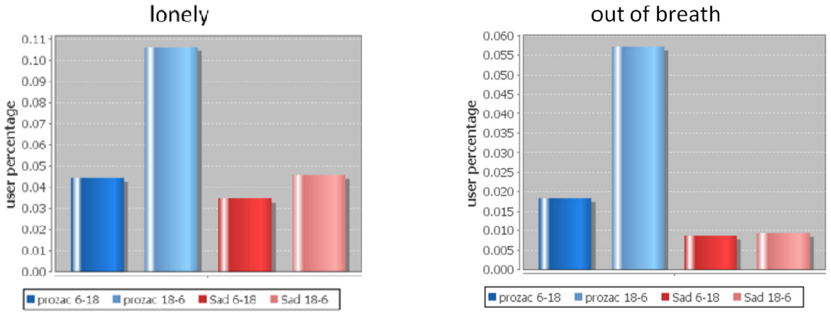
$$tf_{i,j} = \frac{n_{i.j}}{\sum_k n_{k,j}} \quad (1)$$

$$idf_i = \log \frac{1}{|\{j: t_i \in p_j\}|} \quad (2)$$

In the second stage, we try to exploit not only content-based unigram features but also temporal information to handle a more difficult problem in distinguishing depression and sadness. The intuition is that people suffering from depression differ from normal people not only in the written content, but also on the timing of such content being produced. Below we will describe several of our observations in this aspect.

We first select two boards from PTT for observation: *Prozac* and *Sad* board, whose members consist of mostly people suffering clinical depression and people with ordinary sadness emotion (more detailed descriptions of these two boards will be stated in section 3.1). Figure 3 shows two of many examples we observed that contain time clues.

In Figure 3(a), we observe that people in both boards use the term "lonely". However, on the *Prozac* board, the term is used significantly more frequently during night time (from 18:00 to 6:00 next day), but on the *Sad* board, such difference is not significant. Similarly in Figure 3(b), the clinically depressed persons use "out of breath" much more frequently during the night time, while sad persons do not show such trend. Both observations motivate us to bring the temporal information into play to better recognize depression cases.



(a). Frequency of the word 'lonely' being used. In *Prozac* board, the frequency of usage at night is much higher than that of daytime. In *Sad* board, the difference is not significant.

(b). Frequency of the phrase 'Out of breath' being used. In *Prozac* board, the frequency of usage at night is higher than that of daytime. In *Sad* board, the difference is not significant.

Fig. 3. Examples of the data observation

The posting time of a post on BBS, blogs, or micro-blogs is usually fine-grained, accurate to the minutes or even seconds. To incorporate this information into our model, we define the temporal feature as a pair containing the term itself and the time of it being posted. We encode the posting time into different spans as shown in Table 1. Then we can apply TF-IDF in equation 1 and 2 to generate the value of temporal features. The only difference is that now $n_{i,j}$ stands for the frequency of a " timeslot-term" pair.

Table 1. Predefined temporal categories

Category	#	Description	Feature (tired)
12Hour(6-18)	2	6-18, 18-6	18-6 tired
12Hour(8-20)	2	8-20, 20-8	20-8 tired
12Hour(10-22)	2	10-22, 22-10	10-22 tired
6Hour(6-12)	4	6-12, 12-18, 18-24, 24-6	18-24 tired
6Hour(8-14)	4	8-14, 14-20, 20-2, 2-8	20-2 tired
6Hour(10-16)	4	10-16, 16-22, 22-4, 4-10	16-22 tired
24Hour	24	24 hours	20 tired
12Month	12	12 months	March tired
4Season	4	spring, summer, fall, winter	Spring tired
2Season	2	spring-summer, fall-winter	Spring-summer tired
Season change	2	April+May+September+October, others	Others tired
Workday-Weekend	2	workday, weekend	Workday tired

Each of these temporal categories has its own meaning. The first three categories divide a day into two sections, “daytime” vs. “night” or “working hours” vs. “non-working hours”. The next 3 categories further separate a day into four sections. Next, we use more fine-grained slots such as “24Hour” and “12Month”. Units such as “4Season”, “2Season” and “Season change” are motivated by the observation that weather may affect human beings’ emotions, in particular for people suffering depression. Finally, we define the category “Workday-Weekend” to reflect the observation that depressed people sometimes suffer occupational function impairment, which leads to different mental conditions or behaviors between workday and weekend.

There are 12 different categories in Table 1. Thus, one single term feature can be converted to 12 different temporal features. Assuming there is a message posted on *Mon Mar 21 20:16:11* with a term *tired*, the resulting temporal features are listed in the right-most column in Table 1.

3 Experiment

3.1 How the Training Data Can Be Obtained?

We conduct experiment on PTT data, and use 5-fold cross validation to obtain the accuracy. As in the first stage, we hope to train a classifier that distinguishes messages with negative emotions from those with non-negative emotions. We choose posts in *Gossiping* and *Happy* boards on PTT to represent messages with non-negative emotions and posts from *Prozac* and *Sad* boards to represent the ones with negative emotions. The Gossiping board is chosen to be the data source not only because it is the most popular board on PTT but also due to the variety of posts with different purposes and emotions that we believe are general enough to represent many different types of write-ups.

In the second stage, we train a *depression vs. sadness* classifier. We use the posts from *Prozac* and *Sad* boards as the positive and negative examples respectively, while the former contains mostly depression posts and the latter contains mostly sad (but not necessarily depressed) messages. *Prozac* board is a place for people with clinical depression or potential candidates to interact with each other, share their experiences and express their emotions. The word Prozac is the name of medicine for treating depressive disorder. The categories of posts in *Prozac* board are clearly specified, as listed in Table 2. Note that we use only the articles from the *cloudy day* category, which contain posts created by people with depression to express their thoughts and feelings. Other categories such as *information* or *sunny* day are excluded while generating the training data to ensure the quality of the training data. Table 3 shows the basic statistics of the training data.

Table 2. Categories of posts in Prozac board

Self-induction	For the new users to introduce themselves
Medical	For the discussion of hospital, doctor, counseling, medicine and symptoms
Experience	For the discussion of experience, treatment situation
Information	For news, information, research and books about depression
Question	For asking questions
Transcription	For transcription of posts from other boards
Cloudy day	For self-introduced users to vent bad emotions
Sunny day	For self-introduced users to share good emotions
Chat	For self-introduced users to use when other categories are not applicable

Table 3. Basic statistics of training data

Statistics	Gossiping	Happy	Prozac	Sad
Time period	08/01~10/12	10/01~10/12	08/01~10/12	10/01~10/12
Posts number	6505	11209	6015	4900
Users number	1699	2695	1027	1652

3.2 Stage 1 – Negative vs. Non-negative Classifier

Most of the posts in PTT are in Traditional Chinese. Therefore we focus on Chinese posts in this experiment, though the proposed technique is language universal. We first use Yahoo's segmentation API service to perform word-segmentation on the posts because of its stability and efficiency. We then filter out single-character terms in Chinese as they are more likely to be s top-words without apparent meaning. Besides, terms used by too few people (< 25) are removed to avoid over-fitting. In the end, 5622 unigram terms are chosen.

Here we collect all pos ts of a user to ex tract the unigram features. Note th at the subject is persons not posts, because in practice it is more critical to know whether a person is a potential depression patient. We exploit linear SVM using Liblinear(Fan et al. 2008) for classification. Note that each feature is scaled to [0, 1]. The 5 fold cross validation accuracy reaches 96.17%, which means negative-posts and non-negative-posts can be easily separated based on content.

3.3 Stage 2 – Depression vs. Sadness Classifier

The feature generation process is the same as mentioned in previous section. Similar to what we did in stage 1, we first extract 2460 unigrams for learning. The results are shown in Table 4. The 5-fold cross validation accuracy is 81.86%, which signifies that depression and sadness posts are indeed separable, but doing so is much harder than separating negative vs. non-negative write-ups.

Table 4. Experiment results in stage 2 (term features only)

CV accuracy(5 fold)	81.86%
CV AUC(5 fold)	0.8863

To verify the quality of training data and learning process, we also examine the weight of each feature. Table 5 shows the top 40 depression features, and Table 6 shows the top 40 sad features. All features have been translated into English.

Table 5. Top 40 depression features and weights

Take medicine	2.17	Normal people	1.34	World	1.08	Day after tomorrow	0.98
Doctor	1.96	Horrific	1.27	Prozac	1.06	Crowd	0.98
Depression	1.79	Boyfriend	1.26	Yesterday	1.05	Dark night	0.98
School	1.78	Medicine	1.24	Squeeze out	1.02	Subsist	0.98
Suicide	1.62	Emotion	1.23	Clinic	1.02	Tight Chest	0.97
Counseling	1.50	Pain	1.20	Destroy	1.02	Agree	0.97
Sick	1.44	Psychologist	1.18	Patient	1.02	State	0.96
They	1.44	Afraid	1.17	Thought	1.00	Stable	0.95
Happy	1.42	Bedtime	1.14	Ward mate	0.99	Appetite	0.95
Trembling	1.39	What if	1.09	Can't hold out	0.98	Social	0.94

Table 6. Top 40 sad features and weights

Miserable	-1.70	Cant' let go	-1.24	Day	-1.01	Opportunity	-0.92
Cheer up	-1.37	Hesitate	-1.15	Can't see	-1.00	Immediately	-0.92
We	-1.37	Mood	-1.15	Put	-1.00	Misunderstand	-0.90
Hope	-1.31	Feel	-1.12	Memory	-0.99	Time	-0.90
Torture	-1.30	Quietly	-1.10	Really	-0.98	A little	-0.90
Sad	-1.27	Think	-1.08	And	-0.97	Relieved	-0.90
Broken-heart	-1.27	Actual	-1.03	Tear	-0.96	Sorry	-0.88
Such	-1.26	Future	-1.02	Slowly	-0.95	Popular feeling	-0.87
Obviously	-1.26	Always	-1.02	Children	-0.94	Try one's best	-0.86
Exaggerate	-1.24	Affair	-1.01	Future	-0.94	Outcome	-0.85

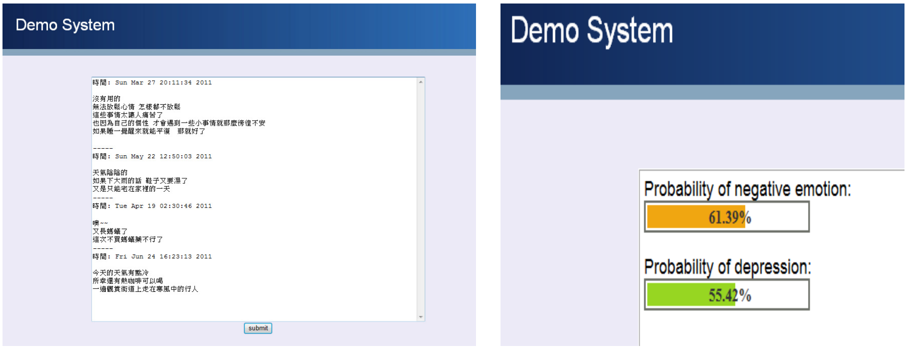
From the top 50 depression features, we can see that some of them are related, such as *depression, suicide* and *pain*; some are related to somatic symptoms like *chest tightness* and *appetite*; and some provide more insights into the mental world of people with depression such as terms including *normal people*, *squeeze out*, *social* and *crowd*. The term *knife* might indicate the most common tool they used or imagined to hurt themselves with. Furthermore, *dark night, destroy, unable to hold out* and *terror* show the darkness and fear inside their minds. On the other hand, the top keywords for *sad* posts are more general.

Table 7. Results of using temporal features only

Feature	#	Accuracy	AUC
Term	2460	81.86%	0.8863
12Hour(6-18)	4920	81.97%	0.8817
12Hour(8-20)	4920	80.55%	0.8817
12Hour(10-22)	4920	80.55%	0.8790
6Hour(6-12)	9833	81.34%	0.8897
6Hour(8-14)	9812	80.55%	0.8746
6Hour(10-16)	9812	80.55%	0.8746
24Hour	49644	76.15%	0.8109
12Month	28759	78.09%	0.8439
4Season	9839	80.03%	0.8711
2Season	4920	80.81%	0.8797
Season change	4920	80.93%	0.8825
Workday	4920	80.89%	0.8821
All	147237	84.47%	0.9167

Table 8. Results of using both term features and temporal features

Feature	#	Accuracy	AUC
Term	2460	81.86%	0.8863
12Hour(6-18)	7380	82.34%	0.8931
12Hour(8-20)	7380	82.34%	0.8940
12Hour(10-22)	7380	81.71%	0.8892
6Hour(6-12)	12290	82.79%	0.8987
6Hour(8-14)	12293	83.58%	0.9089
6Hour(10-16)	12272	82.79%	0.8970
24Hour	52104	82.38%	0.8980
12Month	31219	83.20%	0.9033
4Season	12299	82.98%	0.8941
2Season	7380	81.71%	0.8887
Season change	7380	81.49%	0.8915
Workday	7380	81.34%	0.8882
All	149697	84.51%	0.9176



```
時間: Sun Mar 27 20:11:34 2011

沒有用的
無法放鬆心情 怎樣都不放鬆
這些事情太讓人痛苦了
也因為自己的個性 才會遇到一些小事情就那麼徬徨不安
如果睡一覺醒來就能平復   那就好了


------
時間: Sun May 22 12:50:03 2011

天氣陰陰的
如果下大雨的話 鞋子又要濕了
又是只能宅在家裡的一天
------
時間: Tue Apr 19 02:30:46 2011

噢~~
又長螞蟻了
這次不買螞蟻藥不行了
------
時間: Fri Jun 24 16:23:13 2011

今天的天氣有點冷
所幸還有熱咖啡可以喝
一邊觀賞街道上走在寒風中的行人
```

Probability of negative emotion: 61.39%

Probability of depression: 55.42%

(a). The input screen

(b). System output

Fig. 4. Screenshots of the demo system

We then conduct experiments to observe the performance of using only temporal features. The results are shown in Table 7. Note that the accuracies are not significantly higher than that without temporal features. We believe it might due to the lack of sufficient data to train such fine-grained feature set, as the experiments also show that the more fine-grained a temporal category is (e.g. 24 hours, 12 month), the lower the accuracy it produces. However, when both term features and temporal features are exploited (see Table 8), the performance does show consistent improvement. In most of the temporal categories, we receive higher accuracy than using only term features, and the highest accuracy lies in "6Hour(8-14)" category. If we include all 12 temporal features, the accuracy can be boosted to 84.51%, which is significantly higher than using only term features.

4 System Demo and Manual Evaluation

To further confirm the effectiveness of our model, we construct a real-time system for demo and manual evaluation. The user interface of the system can be seen below. People can simply copy/paste user posts into the window and submit for evaluation. There is no need to fill in any personal information or questionnaires.

Figure 4 shows the screenshots of our demo system. Once the post content and timestamp are submitted, the system evaluates the posts and outputs the probabilities of the subject possessing negative emotion and depression.

4.1 Manual Evaluation

We extract posts from another board on PTT, the *Diary* board, to evaluate whether our system can identify some potential depression candidates based on their diary writings. *Diary* board is a place for PTT users to write things about their lives, feelings and thoughts. People have various reasons to post on the *Diary* board. Nevertheless, we believe that among the users who post on the *Diary* board, there might be a small fraction of them suffering from depression, and we want to test whether our system can identify some of them. We collect 15374 posts from the *Diary* board, total 2071 users. Figure 5 illustrates the flowchart of manual evaluation.

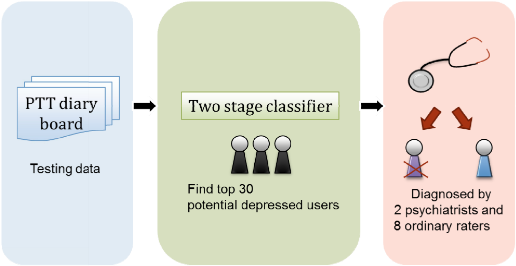

PTT diary board
Testing data
Two stage classifier
Find top 30 potential depressed users
Diagnosed by 2 psychiatrists and 8 ordinary raters

Fig. 5. Flowchart of manual evaluation

We feed the posts from the *Diary* board into our system, and rank the users using our two-stage classifier. We then asked 2 psychiatrists and 8 ordinary evaluators to diagnose (based on the writings) whether the top suspicious candidates our system identified are truly potential depression candidates. Since ordinary evaluators lack the knowledge to diagnose depression, before making the evaluation they are asked to read a brief document about the diagnosis principle released in the fourth edition of Diagnostic and Statistical Manual of Mental Disorders (DSM IV) published by American Psychiatric Association (2000). After reading the posts, evaluators have to choose from four options: non-diagnosable, major depression, moderate or minor depression, and no depression.

Table 9 shows the diagnosis results of two psychiatrists. Note that we regard a case as “major depression” if at least one of the doctors thinks so, or “moderate or minor depression” if at least one of the doctors thinks so. The results show that out of the 30 top suspects our system identified, 18 of them do possess some depression symptoms.

The agreement is even higher (21/30) for non-expert evaluators (see Table 10). The manual evaluation demonstrates that our system can effectively identify potential subjects who are suffering depression but are unaware of it from their writings.

Table 9. Diagnosis of 2 doctors

30 detected users	
major depression	9
moderate and minor depression	9
no depression	12
non-diagnosable	0

Table 10. Diagnosis of 8 ordinary raters

30 detected users	
major depression	7
moderate and minor depression	14
no depression	9
non-diagnosable	0

5 Related Work

5.1 Automated Depression Detection

Neuman and Kedma (2010) propose a system called Pedesis to automatically detect users with depression on the web, specifically on blogs. Their main idea is that depression can be hidden in some metaphors, which implies that obvious depressed terms are not the only indicators. By the help of a search engine, they extract sentences in the form of "depression is like *", and identify some popular metaphors for depression such as *black hole* or *dark cloud*. These metaphors are then used as keywords to identify users with depression. This knowledge-driven method is fundamentally different from our learning-based method.

There are researches applying machine learning methods for depression detection. We can classify them into four categories based on the sources used for detection: text, speech, facial expression and electroencephalogram (EEG).

Text. Jarrold et al. (2010) investigate whether language features can be used to diagnose depression. They use only binary classification and did not include temporal information in the features. Besides, the corpus they use takes lots of time and effort to create, while we use PTT which contains a gradually increasing resource whose data as well as labels can be extracted automatically with limit amount of efforts. Aamodt et al. (2010) develop a decision support system for depression diagnosis using case-based reasoning. However, to compare a new case with past cases, the system requires the patient to fill out a questionnaire first, thus the usage and coverage is limited.

Speech. Based on the theory that people with depression have slow and monotonous voice, Low et al. (2010) use speech characteristics as features for depression detection and compare performances of different feature combinations. Their experiments show MFCC and short time energy outperform other features. Sanchez et al. (2011) successfully exploit prosodic and spectral speech features to decide whether the speaker is depressed.

Facial expression. Cohn et al. (2009) consider not only verbal features but also facial actions to perform depression detection. They found both manual FACS coding and active appearance modeling (AAM) are correlated with depression. Maddage et al. (2009) also have similar idea of using facial features for depression detection.

EEG. Hosseinifard et al. (2011) use EEG signal as the features for depression classification and achieve high accuracy in the experiments. Unfortunately, a depression detection system like this can hardly be used as screening tool for large amount of candidates.

The related work shows that different sources of information can serve as the clue for depression detection. Our work is novel because we have not yet found any work that proposes similar two-stage online screening framework with empirical demonstration on the effectiveness of the designed temporal features.

5.2 Temporal Information in Depression

Early studies have suggested that depression might be a seasonal variation disorder. Eastwood and Stiasny (1978) conduct an analysis of hospital admission of neurotic and endogenous depression, and reported significant peaks in spring and fall. Morken et al. (2002) focus on monthly variation of depression, with gender information also considered. They find that depression admission for women reaches highest peak in November, and for men, in April. Both studies have confirmed a correlation between incidence of depression and time. Kerkhofs et al. (1991) investigate the 24 hour sleep patterns of 22 people, with 12 of them having major depressive disorder and 10 being normal people. The result shows a difference of sleeping behavior for the two groups. Normal people tend to take a nap in the early afternoon, and depressed people have no consistent sleeping period. If a person naps in the morning more often, indicating that the person lacks sleep at night, the probability of depression might be higher. These researches indicate the existence of time clues for detecting depression and non-depression candidates, which strengthen our proposal to introduce the temporal information as features.

6 Conclusion

The main contributions of this paper are listed below:

1. We propose two practical real-world usage scenarios for depression detection, and design a novel two-stage learning framework based on such scenarios.
2. We design a strategy to incorporate temporal information into the content feature and improve the detection accuracy significantly.
3. We identified an important resource, namely BBS, which allows us to automatically extract data and labels for training and testing. Furthermore, the data grows daily, which means one can obtain more training data to enhance the performance of the system. Such resources can potentially be used for other classification tasks because there are thousands of boards in this system, and one can easily obtain labeled data of specific purpose to facilitate training and testing.
4. We developed an online real-time depression detection engine which does not require the users to fill out any questionnaires or reveal their identities. For evaluation, we conduct cross-validation on the PTT dataset and find experts and normal people to judge the usability of the system. The results show that our system can indeed discover potential or hidden depression candidates that are otherwise hard to find.

The framework, data, and features we have proposed can easily be applied to design detection engines for other kinds of mental diseases such as delusional disorder, schizophrenia, anxiety disorder, etc. In the future, we plan to integrate other dimension of information into our system including the weather and geographical information, which we believe are also useful clues for depression detection.

Acknowledgements. This work was supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC101-2911-I-002-001, NSC101-2628-E-002-028-MY2 and NTU102R7501.

References

1. Aamodt, A., Gundersen, O.E., Loge, J.H., Wasteson, E., Szczepanski, T.: Case-Based Reasoning for Assessment and Diagnosis of Depression in Palliative Care. In: The International Symposium on Computer-Based Medical Systems, pp. 480–285 (2010)
2. American Psychiatric Association, Diagnostic and Statistical Manual of Mental Disorders, 4th edn., Text Revision. American Psychiatric Association, Washington, DC (2000)
3. Cohn, J.F., Kruez, T.S., Matthews, I., Yang, Y., Nguyen, M.H., Padilla, M.T., Zhou, F., De la Torre, F.: Detecting Depression from Facial Actions and Vocal Prosody. In: International Conference on Affective Computing and Intelligent Interaction (2009)
4. Eastwood, M.R., Stiasny, S.: Psychiatric Disorder, Hospital Admission, and Season. Archives of General Psychiatry 35, 769–771 (1978)
5. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A Library for Large Linear Classification. Journal of Machine Learning Research 9, 1871–1874 (2008)
6. Feightner, J.W., Worrall, G.: Early Detection of Depression by Primary Care Physicians. Can. Med. Assoc. J. 142, 1215–1220 (1990)
7. Hosseinifard, B., Moradi, M.H., Rostami, R.: Classifying Depression Patients and Normal Subjects Using Machine Learning Techniques. In: Iranian Conference on Electrical Engineering, pp. 1–4 (2011)
8. Jarrold, W.L., Peintner, B., Yeh, E., Krasnow, R., Javitz, H.S., Swan, G.E.: Language Analytics for Assessing Brain Health: Cognitive Impairment, Depression and Pre-symptomatic Alzheimer's Disease. Brain Informatics, 299–307 (2010)
9. Kerkhofs, M., Linkowski, P., Lucas, F., Mendelwicz, J.: Twenty-Four-Hour Patterns of Sleep in Depression. Sleep 14, 501–506 (1991)
10. Low, L.A., Maddage, N.C., Lech, M., Sheeber, L., Allen, N.: Influence of Acoustic Low-Level Descriptors in the Detection of Clinical Depression in Adolescents. In: ICASSP, pp. 5154–5157 (2010)
11. Maddage, N.C., Senaratne, R., Low, L.A., Lech, M., Allen, N.: Video-based Detection of the Clinical Depression in Adolescents. In: International Conference on Engineering in Medicine and Biology Society, pp. 3723–3726 (2009)
12. Morken, G., Lilleeng, S., Linaker, L.M.: Seasonal Variation in Suicides and in Admissions to Hospital for Mania and Depression. Journal of Affective Disorders 69, 39–45 (2002)
13. Neuman, Y., Kedma, G., Cohen, Y., Nave, O.: Using Web-Intelligence for Excavating the Emerging Meaning of Target-Concepts. In: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, pp. 22–25 (2010)

14. PTT, http://www.ptt.cc/index.html (retrieved June 27, 2011)
15. Sanchez, M.H., Vergyri, D., Ferrer, L., Richey, C., Garcia, P., Knoth, B., Jarrold, W.: Using Prosodic and Spectral Features in Detecting Depression in Elderly Males. In: INTERSPEECH, pp. 3001–3004 (2011)
16. Saraceno, B.: T he WHO World Health Report 2001 on mental health. Epidemiol. Psychiatr. Soc. 11, 83–87 (2002)

EEG-Based Person Verification Using Multi-Sphere SVDD and UBM

Phuoc Nguyen[1], Dat Tran[1], Trung Le[2], Xu Huang[1], and Wanli Ma[1]

[1] University of Canberra, ACT 2601 Australia
[2] HCM City University of Education, Vietnam
dat.tran@canberra.edu.au

Abstract. The use of brain-wave patterns extracted from electroencephalography (EEG) brain signals for person verification has been investigated recently. The challenge is that the EEG signals are noisy due to low conductivity of the human skull and the EEG data have unknown distribution. We propose a multi-sphere support vector data description (MSSVDD) method to reduce noise and to provide a mixture of hyperspheres that can describe the EEG data distribution. We also propose a MSSVDD universal background model (UBM) to model impostors in person verification. Experimental results show that our proposed methods achieved lower verification error rates than other verification methods.

Keywords: Support vector data description, support vector machine, person verification, EEG, universal background model.

1 Introduction

The use of brain-wave patterns for person verification has been investigated at IDIAP in Switzerland [1]. It has been shown that the brain-wave pattern of every individual is unique and that the electroencephalography (EEG) brain signal can be used for person identification [2]. Traditional biometrics such as fingerprint, voice, and retina can be damaged or missing for some people, however EEG biometric exists in every person. EEG has advantageous such as unobtrusive, requiring living person recording, spontaneous signal, individual uniqueness due to different brain configurations [3].

Person verification is different from person identification. Person identification is to match the user biometric data against all the records in a database, while person verification is to accept or to reject a user claimed identity providing his biometric data. EEG-based biometry is an emerging research topic that can open new research directions and applications. Most EEG-based biometry work was focusing mainly on person identification, for example [4] and very little work has been done on person verification [1]. In [5], Manhattan distances on autoregressive (AR) coefficients with PCA were used to compute thresholds for determining test patterns were clients or impostors, the person verification task from 5 subjects were done in 2 stages. In [6], Independent Component Analysis (ICA) was used to

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 289–300, 2013.

determine dominating brain regions to extract AR features, then a Naive Bayes probabilistic model is employed for person authentication of 7 subjects with Half Total Error Rate (HTER) of 2.2%. In [1], Gaussian mixture models has been applied for person verification task on EEG signal from 9 subjects. Half total error rate of 6.6 % was achieved for imagination left task. Those results were satisfactory but the numbers of subjects were very small.

Most features in BCI research have high dimensionality due to the number of channels. In addition, the training sets are usually small since the training process is time consuming and demanding [7]. As a result, models based on density estimation do not have enough data for training. Usually, maximum a posteriori adaptation is used to overcome this problem as seen in [1]. On the other hand, models based on kernel aim at determining the boundaries of the data instead of probability density, hence they do not require a lot of data for training. In support vector data description (SVDD), a spherically shaped boundary around a normal data set is used to separate this set from abnormal data. The volume of this data description is minimized to reduce the chance of accepting abnormal data. In [8], SVDD was used to model 70 individuals using energy features and attained 0.9913 area under curve (AUC). However SVDD does not guarantee that the single spherically shaped boundary can best describe the EEG data due to complex data distributions and much noise and outliers. A better description is the use of multiple spheres, however there is currently no investigation available for EEG-based person verification.

In speaker verification, the task can be stated as a hypothesis testing between the two hypotheses: the input utterance is from the hypothesis speaker, client, ($H0$) or not from the hypothesis speaker, impostors, ($H1$). The difficulties are usually in modelling the hypothesis $H1$ since it should represent all possible alternatives speakers. There are two approaches to model the alternatives speakers [9]. The first approach uses a set of other speaker models to cover the space of the alter native hypothesis, this is called likelihood ratio sets, cohorts or background speakers. The selection, size and combination of the background speakers have been the subject of much research [9]. The second approach pools speech from several speakers and train a single model, this is called the world model or universal background model (UBM). This approach has become the predominate and has been focused on selection and composition of the speakers [9]. The advantage of this approach is that a single background model can be trained once and shared for all hypothesis testing of individual speakers.

GMM UBM has been the state-of-the-art high performance probabilistic model for representing speaker model and alternative speaker models because of its capability to approximate arbitrary densities [10]. Recently, Support Vector Machine (SVM) has been used for speaker verification. Because SVM draws an optimal hyper-plane to separate two classes, its application to speaker verification is to separate the client from impostors. The score of a vector is its the distance to the hyperplane and the threshold can be changed by adding some amount to the decision value causing the hyperplane moves nearer or further the class $H0$. In [11] SVM was combined with GMM for speaker verification by stacking the means of the adapted mixture components into a GMM super-vectors. Then a linear kernel is derived based on an approximation to KL divergence between two GMM models. In [12] and [8] the authors use SVDD for speaker verification task the distance to the sphere centers are used as scores, the former author convert score to probability and the later author use percentage of rejection data as threshold.

In this paper, we propose a multi-sphere SVDD (MSSVDD) person verification in which an optimisation problem and an iterative algorithm are proposed to determine model parameters for MSSVDD to provide a better data description to EEG data for a person. An SVM UBM model is proposed to represent the person and background models as a set of sphere distributions, each sphere can represent some characteristics of a person with well-trained decision boundaries. Hence the collection of spheres can hopefully represent the varieties of feature space.

Experimental results on 4 large data sets show that the proposed multi-sphere SVDD can perform as the GMM UBM and the SVM UBM out perform GMM UBM.

2 Multi-Sphere SVDD

2.1 Problem Formulation

Consider a set of m hyperspheres $S_j(c_j, R_j)$with center c_j and radius R_j, $j = 1, \dots, m$. This hypershere set is a good data description of the normal data set $X = \{x_1, x_2, \dots, x_n\}$ if each of the hyperspheres describes a distribution in this data set and the sum of all radii $\sum_{j=1}^{m} R_j^2$ should be minimised.

Let matrix $U = [u_{ij}]_{n\times m}$, $i = 1, \dots, n$, $j = 1, \dots, m$ where u_{ij} is the hard membership representing the belonging of data point x_i to hypersphere S_j, $u_{ij} = 0$ if x_i is not in S_j and $u_{ij} = 1$ if x_i is in S_j. The optimisation problem of multi-sphere SVDD can be formulated as follows

$$\min_{R,c,\xi} \Big(\sum_{j=1}^{m} R_j^2 + C \sum_{i=1}^{n} \xi_i \Big) \tag{1}$$

subject to

$$\begin{aligned} &\sum_{j=1}^{m} u_{ij} ||\phi(x_i) - c_j||^2 \leq \sum_{j=1}^{m} u_{ij} R_j^2 + \xi_i \qquad i = 1, \dots, n \\ &\xi_i \geq 0, \quad i = 1, \dots, n \end{aligned} \tag{2}$$

where $R = [R_j]_{j=1,\dots,m}$ is vector of radii, C is a constant, $\xi = [\xi_i]_{i=1,\dots,n}$ is vector of slack variables, $\phi(.)$ is the nonlinear function related to the symmetric, positive definite kernel function $K(x_1, x_2) = \phi(x_1)^T \phi(x_2)$, and $c = [c_j]_{j=1,\dots,m}$ is vector of centres.

Minimising the function in (1) over variables R, c and ξ subject to (2) will determine radii and centres of hyperspheres and slack variables if the matrix U

is given. On the other hand, the matrix U will be determined if radii and centres of hyperspheres are given. Therefore an iterative algorithm will be applied to find the complete solution. The algorithm consists of two alternative steps: 1) Calculate radii and centres of hyperspheres and slack variables, and 2) Calculate membership U.

For classifying a data point x, the following decision function is used

$$f(x) = sign\Big(\max_j \Big\{R_j^2 - ||\phi(x) - c_j||^2\Big\}\Big) \tag{3}$$

The unknown data point x is normal if $f(x) = +1$ or abnormal if $f(x) = -1$.

2.2 Calculating Radii, Centres and Slack Variables

The Lagrange function for the optimisation problem in (1) subject to (2) is as follows

$$L(R, c, \xi, \alpha, \beta) = \sum_{j=1}^{m} R_j^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i\Big(||\phi(x_i) - c_{s(i)}||^2 - R_{s(i)}^2 - \xi_i\Big) - \sum_{i=1}^{n}\beta_i\xi_i \tag{4}$$

where $s(i)$ is index of the hypersphere to which data point x_i belongs and satisfies $u_{is(i)} = 1$ and $u_{ij} = 0 \ \forall j \neq s(i)$.

Setting derivatives of $L(R, c, \xi, \alpha, \beta)$with respect to primal variables to 0, we obtain m individual optimisation problems as follows

$$\min\Big(\sum_{i \in s^{-1}(j)} \alpha_i K(x_i, x_i) - \sum_{i,i' \in s^{-1}(j)} \alpha_i \alpha_{i'} K(x_i, x_{i'})\Big) \quad j = 1, \ldots, m \tag{5}$$

subject to

$$\sum_{i \in s^{-1}(j)} \alpha_i = 1 \quad \text{and} \quad 0 \leq \alpha_i \leq C \quad j = 1, \ldots, m \tag{6}$$

After solving all of these individual optimization problems, we can calculate the updating radii $R = [R_j]$ and centres $c = [c_j]$, $j = 1, \ldots, m$ using the equations in SVDD.

2.3 Calculating Membership *U*

We use radii and centres of hyperspheres to update the membership matrix U. The following algorithm is proposed:

```
For $i = 1$ to $n$ do
    If $x_i$ is misclassified then
        Let $j_0 = \arg\min_j \Big\{||\phi(x_i) - c_j||^2 - R_j^2\Big\}$
        Set $u_{ij_0} = 1$ and $u_{ij} = 0$ if $j \neq j_0$
    End if
    Else
        Denote $J = \{j : x_i \in S(c_j, R_j)\}$
        Let $j_0 = \arg\min_{j \in J} \left\{ ||\phi(x_i) - c_j||^2 \right\}$
        Set $u_{ij_0} = 1$ and $u_{ij} = 0$ if $j \neq j_0$
    End Else
End For
```

2.4 Iterative Learning Process

The proposed iterative learning process for multi-sphere SVDD will run two alternative steps until a convergence is reached as follows

```
Initialise $U$ by clustering the normal data set in the input space
Repeat the following
    Calculate $R$, $c$ and $\xi$ using $U$
    Update $U$ using $R$ and $c$
Until a convergence is reached.
```

3 MSSVDD UBM

The MSSVDD UBM models the background space by using multi-hypersphere distribution approach. The set of K background people is trained using multi-sphere SVDDs, each person is represented by n spheres resulting nK spheres in total. Then this single background model will be shared for all hypothesis testing of individuals. When testing hypothesis for a person, his/her model will be removed from the background model.

Below are widely used normalisation methods in person verification [13]:

$$L_0(x) = \log P(x|\lambda_0) \tag{7}$$

$$L_1(x) = \frac{P(x|\lambda_0)}{P(x|\lambda)} \tag{8}$$

$$L_1(x) = \log P(x|\lambda_0) - \log P(x|\lambda) \tag{9}$$

$$L_2(x) = \log P(x|\lambda_0) - \max_{\lambda \neq \lambda_0} \log P(x|\lambda) \tag{10}$$

The simplest method of scoring is to use the absolute likelihood score or in its log domain (7). The score (8) and its log domain (9) uses normalisation, the term $\log P(X|\lambda)$ is called the normalisation term and requires calculation of all impostors likelihood functions. An approximation of this method is to use only the closest impostor model for calculating the normalisation term (10)

Let x be a feature vector, $S = S(c, R)$ be a hyper-sphere, we can consider the probability of x belongs to the sphere S as

$$P(x|\lambda_S) = e^{-\|x-c\|}$$

Then the above scores for SVM will become

$$L_0(x) = \log P(x|\lambda_S) = -\|x - c_S\| \tag{11}$$

$$L_1(x) = \frac{P(x|\lambda_S)}{P(x|\lambda)} = \frac{e^{-\|x-c_S\|}}{\sum_{T \neq S} e^{-\|x-c_T\|}} \tag{12}$$

$$L_1(x) = \log P(x|\lambda_S) - \log P(x|\lambda) = -\|x - c_S\| - \log \sum\nolimits_{T \neq S} e^{-\|x-c_T\|} \tag{13}$$

$$L_2(x) = \log P(x|\lambda_S) - \max_{\lambda \neq \lambda_S} \log P(x|\lambda) = -\|x - c_S\| + \max_{T \neq S} \|x - c_T\| \tag{14}$$

The above scores for SVM have simple interpretations. The score (11) with a radius threshold R_S checks whether x is inside or outside sphere S. The score (12), (13) and (14) check whether x is nearer to the sphere S than other sphere T.

In this paper, we define the probability of x belonging to the sphere S as $P(x|\lambda_S) = e^{-\|x-c\|-R}$ to incorporate the sphere size.

4 Gaussian Mixture Model (GMM)

Since the distribution of feature vectors in X is unknown, it is approximately modelled by a mixture of Gaussian densities, which is a weighted sum of K component densities, given by the equation

$$P(X|\lambda) = \prod_{t=1}^{T} P(x_t|\lambda) = \prod_{t=1}^{T} \sum_{i=1}^{K} w_i N(x_t, \mu_i, \Sigma_i) \tag{15}$$

where λ denotes a prototype consisting of a set of model parameters $\lambda = \{w_i, \mu_i, \Sigma_i\}$, w_i , $i = 1, \ldots, K$, are the mixture weights and $N(x_t, \mu_i, \Sigma_i)$,$i = 1, \ldots, K$, are the d-variate Gaussian component densities with mean vectors μ_i and covariance matrices Σ_i.

5 GMM-UBM

The background model for GMM-UBM is trained in a similar way to MSSVDD-UBM. The set of K background people is trained using GMMs, each person is represented by a mixture of n Gaussian resulting nK Gaussian in total. Then this single background model will be shared for all hypothesis testing of individuals. When testing hypothesis for a person, his/her model will be removed from the background model. The score (14) will be used for testing.

6 Hypothesis Testing

The verification task can be stated as a hypothesis testing between the two hypotheses: the input is from the hypothesis person, ($H0$) or not from the hypothesis person ($H1$) [13].

Let λ_0 be the claimed person model and λ be a model representing all other possible people, i.e. impostors. For a given input X and a claimed identity, the choice is between the hypothesis $H0$: X is from the claimed person λ_0, and the alternative hypothesis $H1$: X is from the impostors λ. A claimed person's score $L(X)$ is computed to reject or accept the person claim satisfying the following rules

$$L(X) \begin{cases} > \theta_L \text{accept} \\ \leqslant \theta_L \text{reject} \end{cases} \tag{16}$$

where θ_L are the decision thresholds.

7 Experiments

7.1 Datasets

The Australian EEG Database used in this research consists of EEG recordings of 40 patients. This database consists of EEG records recorded at the John Hunter Hospital [14], near University of Newcastle, over an 11-year period. The recordings were made by 23 electrodes placed on the scalp sampled at 167 Hz for about 20 minutes.

The EEG motor movement/imagery (EEGMMIDB) dataset contain recordings of 109 subjects performed different motor/imagery tasks[1] [15]. Each subject performed 14 experimental runs in one or three minutes. The tasks include opening and closing fists or imagine opening and closing fists. The recordings used 64 electrodes 10-10 system.

The Alcoholism datasets come from a study to examine EEG correlates of genetic predisposition to alcoholism. The datasets contain EEG recordings of control and alcoholic subjects. Each subject was exposed to either a single stimulus (S1) or two stimuli (S1 and S2) which were pictures of objects chosen from the 1980 Snodgrass and Vanderwart picture set. When two stimuli were shown, they were presented in either a matched condition where S1 was identical to S2 or in a non-matched condition where S1 differed from S2. The 64 electrodes placed on the scalp sampled at 256 Hz for 1 second. The Alcoholism large dataset contains training and test data for 10 alcoholic and 10 control subjects. The Alcoholism full dataset contains 120 trials for 122 subjects. The summary of those datasets is listed in Table 1

[1] Available online at `http://www.physionet.org/pn4/eegmmidb/`

Table 1. Dataset descriptions

Dataset	#subjects	#tasks	#trials	length (seconds)
Australian EEG	40	free	1	1200
EEGMMIDB	109	4	14	60 or 180
Alcoholism (large)	20	2	120	1
Alcoholism (full)	122	2	120	1

Table 2. Identification rate vs number of Gaussians in GMM cross validation training

#Gaussians	1	2	4	8	16	32	64	128	256
Australian EEG	73.82%	86.11%	90.05%	91.60%	90.98%	91.86%	92.07%	92.12%	91.60%
EEGMMIDB	66.49%	71.92%	71.96%	71.71%	70.97%	70.63%	71.05%	70.01%	70.58%
Alcoholism (large)	86.72%	87.57%	85.73%	82.63%	79.94%	70.62%	75.85%	78.39%	77.54%
Alcoholism (full)	50.13%	50.55%	50.16%	49.17%	47.72%	45.72%	45.89%	45.41%	45.68%

7.2 Preprocessing and Feature Extraction

EEG signals were first divided into 15-millisecond segments. The Alcoholism large and full datasets are downsampled to 128 Hz. Then the raw EEG signals were spatially filtered using Surface Laplacian (SL) with spline interpolation constant of 4. This filter can represent better the cortical activity of local sources below the electrodes [1]. Then the power spectral density (PSD) in the band 8-30 Hz was estimated. The Welch's averaged modified periodogram method was used for spectral estimation. Hamming window was set to 1 second and 50% overlap. 12 power components in the frequency band 8-30 Hz were extracted.

Besides PSD features, autoregressive (AR) model parameters were extracted. In AR model, each sample is considered linearly related with a number of its previous samples. The AR model has the advantage of low complexity and has been used for person identification and verification [16] [17] [5]. Burg's lattice-based method was used with the AR model order 21, as a previous study [17] suggested when there were many subjects and epochs.

The electrodes C3, C4, Cz, P3, P4 and Pz were selected to extract PSD and AR features that result in 6*(12+21)=198 features. Those electrodes are available in both datasets and were used in previous study [1] based on expert knowledge.

7.3 Experimental Results

All experiments were conducted using 3-fold cross validation training and the best parameters found were used to train models on the whole training set and test on a separate test set. Table 2 shows the identification rate of GMM training. The best parameter number of Gaussian is 128 for Australian EEG dataset and 4 for EEGMMIDB dataset.

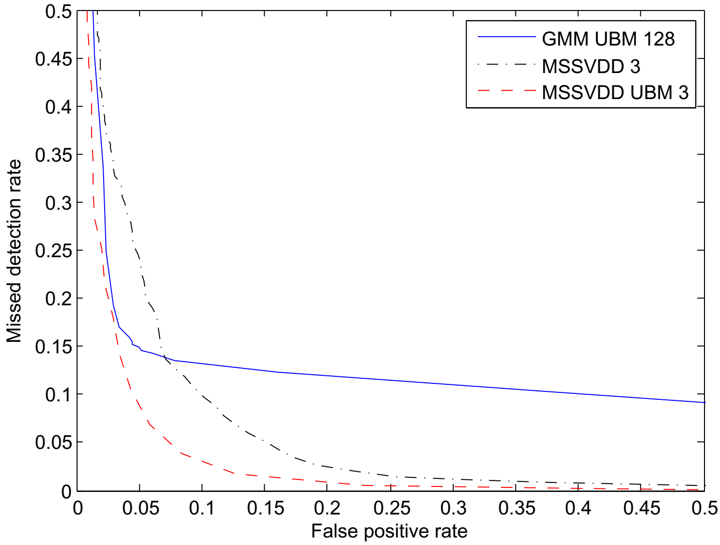


Fig. 1. DET curve of GMM UBM, 3-sphere SVDD, 3-sphere SVDD UBM on the Australian EEG dataset

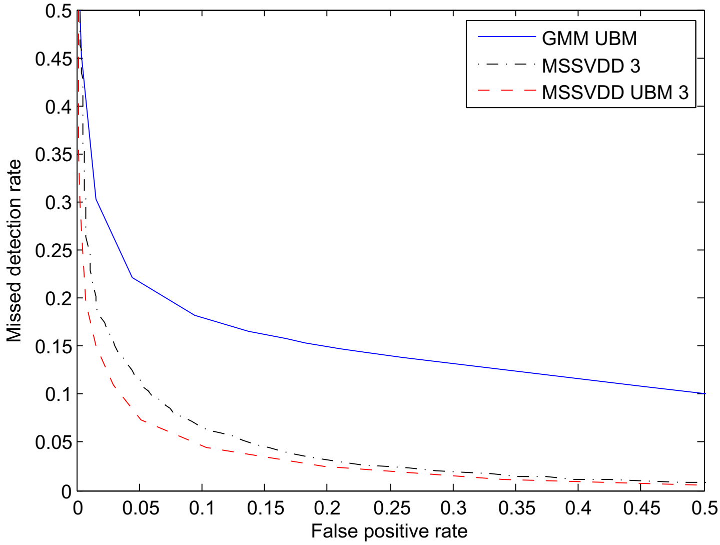


Fig. 2. DET curve of GMM UBM, 3-sphere SVDD, 3-sphere SVDD UBM on the EEGMMIDB dataset

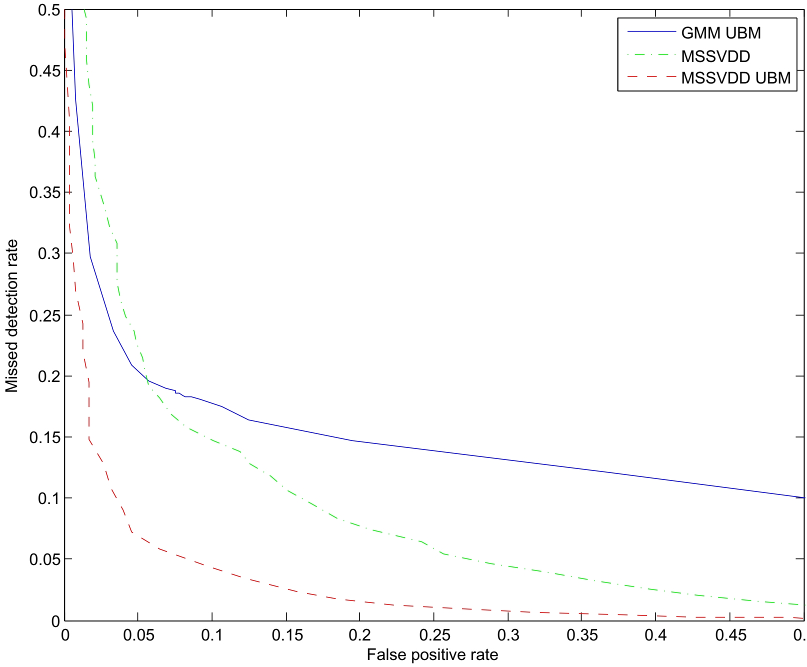


Fig. 3. DET curve of GMM UBM, 3-sphere SVDD, 3-sphere SVDD UBM on the Alcoholism EEG dataset

For MSSVDD training the RBF kernel function $K(x, x') = e^{-\gamma||x-x'||^2}$ was used. The parameter for MSSVDD training is γ and ν. The parameter γ was searched in $\{2^k : k = 2l + 1, l = -8, -7, \ldots, 2\}$. The parameter ν is considered proportional to the rejection percentage of the positive data when training the smallest hypersphere enclosing the positive data. The best parameters found for the first two datasets are $\gamma = 0.031$ and $\nu = 0.031$ with EER=0.221 and the latter two datasets are $\gamma = 0.125$, $\nu = 0.125$ with EER=0.25 and 0.35 respectively.

Figures 1 and 2 show DET curves of GMM-UBM, MSSVDD, MSSVDD-UBM on Australian EEG dataset and EEGMMIDB datasets, respectively. Over all, the MSSVDD outperforms SVDD, and the MSSVDD-UBM outperforms MSSVDD and GMM-UBM.

8 Conclusion

We have presented a Multi-Sphere Support Vector Data Description approach to solving person verification problem. Experiments on the Australian EEG, EEG motor movement/imagery and Alcoholism datasets show a lower verification error rates comparing with SVDD and Gaussian mixture model-based universal background model (GMM-UBM).

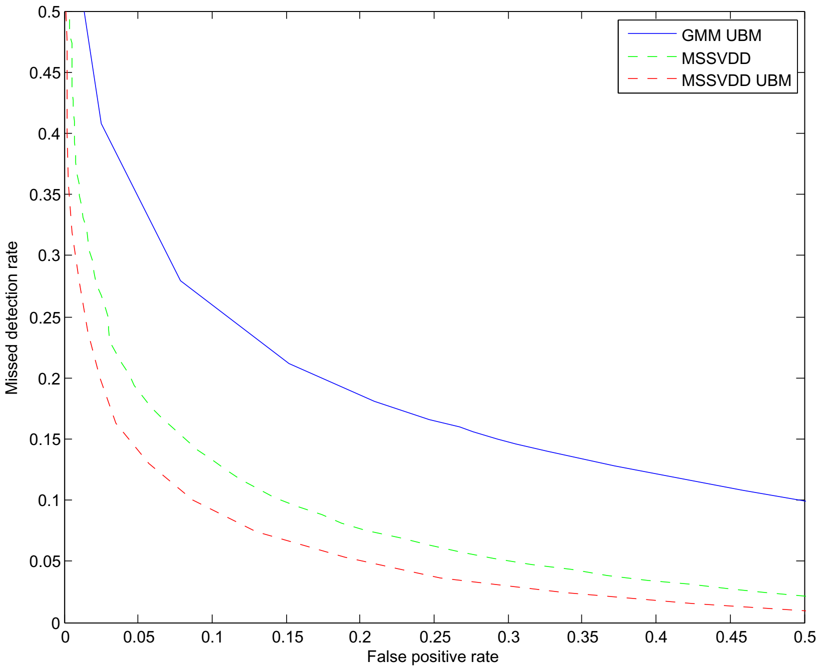


Fig. 4. DET curve of GMM UBM, 3-sphere SVDD, 3-sphere SVDD UBM on the full Alcoholism dataset

References

1. Marcel, S., Millán, J.R.: Person authentication using brainwaves (EEG) and maximum a posteriori model adaptation. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(4), 743–752 (2007)
2. Xu, Q., Zhou, H., Wang, Y., Huang, J.: Fuzzy support vector machine for classification of EEG signals using wavelet-based features. Medical Engineering & Physics 31(7), 858–865 (2009)
3. Riera, A., Soria-Frisch, A., Caparrini, M., Grau, C., Ruffini, G.: Unobtrusive biometric system based on electroencephalogram analysis. EURASIP Journal on Advances in Signal Processing 2008, 18 (2008)
4. Poulos, M., Rangoussi, M., Alexandris, N., Evangelou, A.: Person identification from the EEG using nonlinear signal classification. Methods of Information in Medicine 41(1), 64–75 (2002)
5. Palaniappan, R.: Two-stage biometric authentication method using thought activity brain waves. International Journal of Neural Systems 18(1), 59 (2008)
6. He, C., Wang, J.: An independent component analysis (ICA) based approach for EEG person authentication. In: 3rd International Conference on Bioinformatics and Biomedical Engineering, ICBBE 2009, pp. 1–4 (2009)
7. Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., Arnaldi, B.: A review of classification algorithms for EEG-based brain computer interfaces. Journal of Neural Engineering 4, R1 (2007)

8. Zúquete, A., Quintela, B., Cunha, J.P.S.: Biometric authentication using brain responses to visual stimuli. In: Proceedings of the International Conference on Bio-inspired Systems and Signal Processing, pp. 103–112 (2010)
9. Bimbot, F., Bonastre, J.F., Fredouille, C., Gravier, G., Magrin-Chagnolleau, I., Meignier, S., Merlin, T., Ortega-García, J., Petrovska-Delacrétaz, D., Reynolds, D.A.: A tutorial on text-independent speaker verification. EURASIP Journal on Applied Signal Processing 2004, 430–451 (2004)
10. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. Digital Signal Processing 10(1), 19–41 (2000)
11. Campbell, W.M., Sturim, D.E., Reynolds, D.A.: Support vector machines using GMM supervectors for speaker verification. IEEE Signal Processing Letters 13(5), 308–311 (2006)
12. Brew, A., Grimaldi, M., Cunningham, P.: An evaluation of one-class classification techniques for speaker verification. Artificial Intelligence Review 27(4), 295–307 (2007)
13. Tran, D.T.: Fuzzy Approaches to Speech and Speaker Recognition. PhD thesis, University of Canberra (2000)
14. Hunter, M., Smith, R.L., Hyslop, W., Rosso, O.A., Gerlach, R., Rostas, J.A.P., Williams, D.B., Henskens, F.: The australian EEG database. Clin. EEG Neurosci. 36(2), 76–81 (2005)
15. Schalk, G., McFarland, D.J., Hinterberger, T., Birbaumer, N., Wolpaw, J.R.: BCI2000: a general-purpose brain-computer interface (BCI) system. IEEE Transactions on Biomedical Engineering 51(6), 1034–1043 (2004)
16. Poulos, M., Rangoussi, M., Chrissikopoulos, V., Evangelou, A.: Person identification based on parametric processing of the EEG. In: Proceedings of the 6th IEEE International Conference on Electronics, Circuits and Systems, ICECS 1999, vol. 1, pp. 283–286 (1999)
17. Paranjape, R.B., Mahovsky, J., Benedicenti, L., Koles, Z.: The electroencephalogram as a biometric. In: Canadian Conference on Electrical and Computer Engineering, vol. 2, pp. 1363–1366 (2001)

Measuring Reproducibility of High-Throughput Deep-Sequencing Experiments Based on Self-adaptive Mixture Copula

Qian Zhang[1], Junping Zhang[1,⋆], and Chenghai Xue[2,⋆]

[1] Shanghai Key Lab of Intelligent Information Processing
School of Computer Science, Fudan University, China
[2] Cold Spring Harbor Laboratory, NY
qianzhang.fdu@gmail.com, jpzhang@fudan.edu.cn, xuec@cshl.edu

Abstract. Measurement of the statistical reproducibility between biological experiment replicates is vital first step of the entire series of bioinformatics analysis for mining meaningful biological discovery from mega-data. To distinguish the real biological relevant signals from artificial signals, irreproducible discovery rate (IDR) employing Copula, which can separate dependence structure and marginal distribution from data, has been put forth. However, IDR employed a Gaussian Copula which may cause underestimation of risk and limit the robustness of the method. To address the issue, we propose a Self-adaptive Mixture Copula (SaMiC) to measure the reproducibility of experiment replicates from high-throughput deep-sequencing data. Simple and easy to implement, the proposed SaMiC method can self-adaptively tune its coefficients so that the measurement of reproducibility is more effective for general distributions. Experiments in simulated and real data indicate that compared with IDR, the SaMiC method can better estimate reproducibility between replicate samples.

1 Introduction

During the past years, the biological technology revolution, next-generation high-throughput deep-sequencing, has produced mountains of data of DNA-Seq, RNA-Seq and ChIP-Seq. This mega-data allows biologists to observe the signals from tens of thousands of genes or related genomic elements in a single experiment, a way that was not possible before. One question arises here: how many of these signals are real biologically relevant? Generally, to avoid experiment noise or error, two experiment replicates of one biological sample should be produced, each of which has a collection of individual elements or signals such as a list of genes or transcripts. Then we need to verify the reproducibility of each individual signal. Only the individual signals with high reproducibility are considered as reliable results for further analysis such as differential gene expression identification or GO analysis. Here reproducibility of a signal's two observations in

[⋆] Corresponding authors.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 301–313, 2013.

two replicates is a measure of the confidence that the two observations are consistent with each other. Hereinafter it is shorted as "*reproducibility of signal*". We propose a posterior probability to characterize the confidence and also define irreproducibility = 1 - reproducibility. By choosing a specific critical value, each signal can be determined whether or not to be confident. It is worth noting that we only have two replicates, two lists of observations of individual signals but need to detect the reproducibility of each individual signal. Such extremely small samples also lead to a big and difficult challenging task to traditional data mining where data are generally of remarkably larger size and density estimation can be effectively calculated based on these data.

IDR (Irreproducible Discovery Rate) which measures the reproducibility in high-throughput experiments has been put forth by Li [1]. They proposed to use copula, which can separate the dependency structure of random variables, and a measurement based on copula to detect the high reproducible signals. A remarkable advantage of copula is that it provides an effective way to infer the dependency structure between biological signals without knowing their respective marginal distribution, which is difficult to obtain from real biological signals. Reproducibility measure has been adopted as the standard of ENCODE (The Encyclopedia of DNA Elements) project and has been carried on each signal of all samples before these data are submitted to public database. The strategy of IDR has been generalized to use on other data types such as RNA-Seq.

Although IDR has shown its ability of distinguishing the bona fide signals from artificial signals, it employed the Gaussian Copula, which assumes that the dependence structure of random variables follows multivariate Gaussian distribution. This assumption causes that the Gaussian Copula is sensitive to extreme events and can not capture the asymmetric dependence structure [2]. In fact, despite its simplicity, Gaussian Copula often leads to an underestimation of the risk of the occurrence of joint extreme events [3, 4].Therefore, it is necessary to develop a novel and efficient approach to measure the reproducibility of replicate data without stronger assumption to data distribution.

In this paper, we propose a **S**elf-**a**daptive **Mi**xture **C**opula, called SaMiC, to measure the reproducibility of the high-throughput deep-sequencing experiments. Unlike IDR, SaMiC doesn't assume the dependence structure of random variables to follow Gaussian distribution. SaMiC mixes several copulas and automatically determines the mixture coefficients based on the fitness of the data and the copulas. We prove theoretically that the new mixture copula is still a copula so that it can be used to measure the reproducibilities. Simple and easy to implement, SaMiC is effective and suitable for general distributions. Experiments in both simulated data and real biological data of RNA transcripts expression from human cells show that compared with IDR, SaMiC attains better performance in distinguishing bona fide signals from artificial signals.

The remainder of this paper is organized as following. In Section 2, we introduce the development and preliminary of copulas. In Section 3 we detail our proposed self-adaptive mixture copula and a novel measurement to detect the reproducibility of experiments. In Section 4, we perform experiments in simulated data and real data. In Section 5, we conclude the paper.

2 Related Work and Preliminary of Copula

As a tool of extracting the dependence structure from joint distributions of random variables, copula was first proposed by Sklar [5]. In his work, copula is obtained by a two-stage procedure, *i.e.*, estimating the marginal distribution of each random variable followed by measuring the dependence structure between different random variables. Deheuvels proposed several empirical functions, *i.e.*, the empirical copula of samples, to estimate the copula of population and constructed different non-parametric dependence tests from samples [6]. However, a well-recognized definition of copula hasn't been given until Nelsen's work [7].

There are two major categories in studying copulas: parameter estimation and test of goodness of fit. In the former category, Oakes and Genest proposed a common strategy to estimate the parameters of bivariate copula [8,9]. Later, Joe investigated the maximum likelihood estimation of parametric marginal distribution and parametric copula [10]. Furthermore, Chen studied two stage semi-parametric maximum likelihood estimation [11], and Abegaz derived asymptotic properties of the marginal and copula parameter estimators [12].

In the aspect of test of goodness of fit, an important goal is to measure how well a copula describes the dependence structure among random variables since it is closely related to the correctness of the proposed copula. According to the copula model, test of goodness of fit can be transformed into test of univariate distribution. Then Kolmogorov-Smirnov test can be used to test the goodness of fit of copula. In this manner, Klugman used Q-Q plot to measure the rationality of copula model [13]. Hu introduced M-statistics, which follows the chi-square distribution, to measure goodness of fit of copula model [14]. Engle proposed a test method named "Hit" [15], and Patton expanded the test method "Hit" to the nonlinear density model for checking the goodness of fit [16]. Theses methods can evaluate both the copula and the marginal distribution.

Since the function is useful to obtain the dependence structure of multivariate random variables with few assumptions, it has been applied in financial field. Embrechts employed copula for financial risk management [17]. Hu proposed to use mixed-copula to analyze the financial data [14]. However, such a mixture is not automatic and depends on experts' experience. Recently, copula was also employed in bioinformatics field. For example, Kim discussed the application in genetic data [18], Zhang used copula model to analyze ChIP-Seq data [19], and Li proposed a new method based on copula model to measure reproducibility of bioinformatics data [1]. A main reason of using copula in financial and bioinformatics fields is that it can obtain the dependence structure without knowing marginal distribution of random variables in advance. It is worth noting that the above-mentioned copulas still require more or less assumptions to data distribution, which may limit its effectiveness and extension. It is also noticeable that in bioinformatics field, copula is still a new tool of data analysis. For better understanding, we introduce some preliminaries of copula as follows.

Theorem 1 (Sklar's Theorem). *[5] Let H be a joint distribution function with margins F_1 and F_2. Then there exists a copula C such that*

$$H(x, y) = C(F_1(x), F_2(y)), \quad \forall x, y \in \boldsymbol{R} \tag{1}$$

If F_1 and F_2 are continuous, then C is unique; otherwise, C is uniquely determined on $RanF_1 \times RanF_2$. Here $RanF$ refers to the range of F. Conversely, if C is a copula and F_1 and F_2 are distribution functions, then the function H is a joint distribution function with margins F_1 and F_2.

Sklar's Theorem shows that a joint distribution function can be divided into each variable's marginal distribution function and a copula which present their statistical consistence. Therefore, it is possible to calculate copulas from joint distribution function and its marginal distribution functions. From Sklar's Theorem we can get the follow properties, which are important to measure the reproducibility of high-throughput deep-sequencing experiments:

Property 1. Let $G(X_1, X_2, \cdots, X_n)$ be a joint distribution function of n random variables $X_1, X_2, \cdots, X_n$, $F_1(x_1), F_2(x_2), \cdots, F_n(x_n)$ are marginal distribution functions of these random variables and $C(u_1, u_2, \cdots, u_n)$ is the corresponding copula, then for every $\mathbf{u} = (u_1, u_2, \cdots, u_n) \in [0, 1]^n$ it satisfies that

$$C(u_1, u_2, \cdots, u_n) = G(F_1^{-1}(u_1), F_2^{-1}(u_2), \cdots, F_n^{-1}(u_n)), \tag{2}$$

where $F_i^{-1}(u_i)$ is the right-continuous inverse of F_i, defined as $F_i^{-1}(u_i) = \inf\{z : F_j(z) \geq u_i\}$.

Property 2. Let $G(X_1, X_2, \cdots, X_n)$ be a joint distribution function of n random variables $X_1, X_2, \cdots, X_n$, $C(u_1, u_2, \cdots, u_n)$ be a copula, $c(u_1, u_2, \cdots, u_n)$ be its density function and $F_1, F_2, \cdots, F_n$ are marginal distribution functions of the random variables, we can get that

$$g(X_1, X_2, \cdots, X_n) = c(F_1(X_1), F_2(X_2), \cdots, F_n(X_n)) \prod_{i=1}^{n} f_i(X_i), \tag{3}$$

where $c(u_1, u_2, \cdots, u_n) = \frac{\partial C(u_1, u_2, \cdots, u_n)}{\partial u_1 \partial u_2 \cdots \partial u_n}$, and $f_i(X_i)$ and $g(X_1, X_2, \cdots, X_n)$ are density functions of $F_i(X_i)$ and $G(X_1, X_2, \cdots, X_n)$, respectively.

Currently, there are two main types of commonly-used copulas: Elliptical copulas and Archimedean copulas. Elliptical copulas are a kind of copulas with contoured elliptical distributions, such as Gaussian copula and t-copula. Easy to construct, Elliptical copulas don't have a closed form of function expression, and all of them are radially symmetric and hard to extend to high-dimensional situation. As a result, it is difficult to use Elliptical copulas to describe unpredictable dependence structures or adapt to complex situations.

Different from Elliptical copulas, Archimedean copulas are an associative class of copulas which satisfy the following equations:

$$C(u_1, u_2, \cdots, u_n) = \varphi^{-1}(\varphi(u_1) + \varphi(u_2) + \cdots + \varphi(u_n)) \tag{4}$$

where $\varphi(\cdot)$ is usually called the generator of Archimedean copula. Among a lot of Archimedean copulas, three frequently used Archimedean copulas are Frank

Copula, Clayton Copula and Gumbel Copula. Specifically, the reproducibility structures of Frank Copula and the variables drawn from Frank Copula are symmetric in both tails of their distributions. So any asymmetric consistence between random variables can't be captured using Frank Copula. Clayton Copula is sensitive to the low tail dependence of random variables, and can easily capture the changes around the low tail. Finally, Gumbel Copula is sensitive to the upper tail dependence of random variables.

3 The Proposed SaMiC Approach

It is obvious that each copula has its respective pros and cons in different distributions. To deal with more general distributions, we propose self-adaptive mixture copula, which is a linear combination of several copulas. For simplification, we discuss the proposed copula in two-dimensional situation.

Definition 1 (Mixture Copula). *A function is called mixture copula if it satisfies:* $C_M(u,v) = \sum_{i=1}^{m} \alpha_i C_i(u,v)$*, where* $0 \leq \alpha_i \leq 1$, $\sum_{i=1}^{m} \alpha_i = 1$*. Here* $C_1(u,v), C_2(u,v), \cdots, C_m(u,v)$ *are copulas and* $\alpha_1, \alpha_2, \cdots, \alpha_m$ *are linear coefficients of* $C_M(u,v)$*.*

We prove that $C_M(u,v)$ is a copula, which will be introduced in a extended version due to the length limitation. To self-adaptively estimate the linear coefficients of the proposed mixture copulas, we utilize Pearson χ^2 statistic proposed by Hu to measure the goodness of fit of each copula [14], i.e., $M = \sum_i^k \sum_j^k \frac{(A_{i,j}-B_{i,j})^2}{B_{i,j}}$, where $A_{i,j}$ and $B_{i,j}$ denote the number of observed and predicted frequencies in cell (i,j) of a contingency table, respectively. The details on the contingency table can be referred to as Hu [14]. M follows χ^2 distribution with $(k-1)^2$ degree of freedom. Given a total of m base copulas, $C_1, C_2, \cdots, C_m$, we can attain m results, $M_{C_1}, M_{C_2}, \cdots, M_{C_m}$. Because M_{C_i} follows χ^2 distribution, we can get probability β_i from M_{C_i}. In fact β_i is the probability that there is no significant difference between copula C_i and the data. Because of the additivity of chi-squared distribution, let $\alpha_i = \frac{\beta_i}{\sum_{i=1}^{m} \beta_i}$ be the mixing coefficient of C_i, then the proposed self-adaptive mixture-copula is

$$C_M(u,v) = \sum_{i=1}^{m} \alpha_i C_i(u,v) = \frac{1}{\sum_{i=1}^{m} \beta_i} \sum_{i=1}^{m} \beta_i C_i(u,v) \tag{5}$$

Since our self-adaptive method chooses coefficients automatically, it can deal with more general distributions. By contrast, the ordinary mixture copula manually selects the coefficients, heavily depending on human experience and need lots of tuning for each new group of data [14]. Consequently, it is only applicable to some specific distributions.

To measure the statistical consistency or reproducibility based on the proposed self-adaptive mixture copula, we here consider the situation with two rows of observations for simplification. The reason is that when observations subject to independent identically distribution, it is not difficult to expand to multivariate situation if we use a pairwise analysis to them.

Formally, let the two rows of observations, $(x_{1,1}, x_{1,2}), \cdots, (x_{n,1}, x_{n,2})$, be two replicates of n random signals. We assume that the observations consist of a more reproducible group and a less reproducible one, and π_0 and π_1 denote the proportion of the less reproducible group and the more reproducible group, respectively. Let parameter K_i be an indicator to identify whether or not a signal belong to the more or less reproducible group. $K_i = 1$ if the *i-th* signal belong to the more reproducible group, and $K_i = 0$ if it is in the less reproducible group.

Obviously, the signals in the more or less reproducible group have different probability distributions. We assume that the dependence structures of the two observations of signals in the more and less reproducible groups are induced by $\boldsymbol{z_1} = (z_{1,1}, z_{1,2})$ and $\boldsymbol{z_0} = (z_{0,1}, z_{0,2})$, respectively. According to Sklar's Theorem, any multivariate probability distribution can be divided into its marginal probability distributions and its copula. In other words, it provides a way to infer the reproducibility among several random variables without knowing their marginal distributions in advance. Thus, we construct our parametric model as follows:

Let $K_i \sim Bernoulli(\pi_1)$ and $(z_{i,1}, z_{i,2})$ be distributed as $(z_{i,1}, z_{i,2}) \mid K_i = k \sim S_k(u, v), k = 0, 1$, and

$$S_k(z_{i,1}, z_{i,2}) = C(F_1(z_{i,1}), F_2(z_{i,2}); \lambda_k), \quad k = 0, 1 \tag{6}$$

where $F_1(z_{i,1})$ and $F_2(z_{i,2})$ are the marginal distributions of $z_{i,1}$ and $z_{i,2}$, respectively. And λ_k denotes the relevant parameter of copula. Then considering (6), the total distribution function is

$$\begin{aligned} S(z_{i,1}, z_{i,2}) = P\{Z_{i,1} \le z_{i,1}, Z_{i,2} \le z_{i,2}\} &= \pi_0 S_0(z_{i,1}, z_{i,2}) + \pi_1 S_1(z_{i,1}, z_{i,2}) \\ &= \sum_{k=0,1} \pi_k C(F_1(z_{i,1}), F_2(z_{i,2}); \lambda_k) \end{aligned} \tag{7}$$

In this equation, our actual observations $(x_{i,1}, x_{i,2})$ are used to estimate the cumulative distribution function of $(z_{i,1}, z_{i,2})$. From (3) we can get the density functions of $S_0(z_{i,1}, z_{i,2})$ and $S_1(z_{i,1}, z_{i,2})$ as follows:

$$s_k(z_{i,1}, z_{i,2}) = c(F_1(z_{i,1}), F_2(z_{i,2}); \lambda_k) f_1(z_{i,1}) f_2(z_{i,2}), \quad k = 0, 1, \tag{8}$$

where $f_1(z_{i,1})$ and $f_2(z_{i,2})$ are the density functions of $F_1(z_{i,1})$ and $F_2(z_{i,2})$, respectively. From (7) and (3), therefore, the density function of $S(z_{i,1}, z_{i,2})$ is

$$s(z_{i,1}, z_{i,2}) = \sum_{k=0,1} \pi_k c(F_1(z_{i,1}), F_2(z_{i,2}); \lambda_k) f_1(z_{i,1}) f_2(z_{i,2}) \tag{9}$$

Up to now, our model is parametrized by $\boldsymbol{\theta} = (\pi_0, \lambda_0, \lambda_1)$ and F_1, F_2. The parameters can be attained using maximum likelihood estimation as:

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} s(x_{i,1}, x_{i,2}) = \prod_{i=1}^{n} (f_1(x_{i,1}) f_2(x_{i,2}) \sum_{k=0,1} \pi_k c(F_1(x_{i,1}), F_2(x_{i,2}); \lambda_k)). \tag{10}$$

Note that selecting different Archimedean copulas in (6) will lead to different forms of $S(z_{i,1}, z_{i,2})$. Since

$$C_M(u, v; \boldsymbol{\theta}) = \pi_0 C(u, v; \lambda_0) + (1 - \pi_0) C(u, v; \lambda_1) \tag{11}$$

is also a copula, the formula (7) can be rewritten as $S(z_{i,1}, z_{i,2}) = C_M(F_1(z_{i,1}), F_2\ (z_{i,2}); \boldsymbol{\theta})$, where the form of the final copula $C_M(u, v)$ is determined by the selection of Archimedean copulas in (6).

When several Archimedean copulas $C_{(1)}(u, v)$, $C_{(2)}(u, v)$, $\cdots$, $C_{(m)}(u, v)$ are substituted into (6), the corresponding parameters $\boldsymbol{\theta}_{(1)}, \boldsymbol{\theta}_{(2)}, \cdots, \boldsymbol{\theta}_{(m)}$ can be estimated from (10), where $\boldsymbol{\theta}_{(i)} = (\pi_{(i),0}, \lambda_{(i),0}, \lambda_{(i),1})$. After that, we substitute the obtained copulas $C_{M(1)}(u, v; \boldsymbol{\theta}_{(1)})$, $C_{M(2)}(u, v; \boldsymbol{\theta}_{(2)})$, $\cdots$, $C_{M(m)}(u, v; \boldsymbol{\theta}_{(m)})$ into the method of generating self-adaptive mixture-copula. Because $C_{M(i)}(u, v; \boldsymbol{\theta}_{(i)})$ is also a mixture-copula, we get the linear coefficients $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ and attain the final mixture-copula as:

$$C_R(u, v) = \sum_{i=1}^{m} \alpha_i C_{M(i)}(u, v; \boldsymbol{\theta}_{(i)}) \tag{12}$$

where α_i is the linear coefficient. Since $C_{M(i)}(u, v; \boldsymbol{\theta}_{(i)})$ is also a mixture-copula, $C_R(u, v)$ can be decomposed into

$$\begin{aligned} C_R(u, v) &= \sum_{i=1}^{m} \alpha_i \pi_{(i),0} C_{(i)}(u, v; \lambda_{(i),0}) + \sum_{i=1}^{m} \alpha_i (1 - \pi_{(i),0}) C_{(i)}(u, v; \lambda_{(i),1}) \\ &= C_{R,0}(u, v) + C_{R,1}(u, v) \end{aligned} \tag{13}$$

where $C_{R,0} = \sum_{i=1}^{m} \alpha_i \pi_{(i),0} C_{(i)}(u, v; \lambda_{(i),0})$ and $C_{R,1} = \sum_{i=1}^{m} \alpha_i (1 - \pi_{(i),0}) C_{(i)}(u, v; \lambda_{(i),1})$ are the more and less reproducible groups, respectively.

Once $C_R(u, v)$ is determined, it is easy to update the total distribution function of the data as

$$S_R(z_{i,1}, z_{i,2}) \equiv S_{R,0}(z_{i,1}, z_{i,2}) + S_{R,1}(z_{i,1}, z_{i,2}), \tag{14}$$

and thus

$$\begin{aligned} &s_R(z_{i,1}, z_{i,2}) = \\ &c_{R,0}(F_1(z_{i,1}), F_2(z_{i,2})) f_1(z_{i,1}) f_2(z_{i,2}) + c_{R,1}(F_1(z_{i,1}), F_2(z_{i,2})) f_1(z_{i,1}) f_2(z_{i,2}) \end{aligned} \tag{15}$$

where s_R ,c_R and $s_{R,k}$ are density functions of S_R, C_R and $s_{R,k}$, respectively.

Finally, we can estimate the irreproducibility of each signal based on:

$$P\{K_i = 0 \mid (x_{i,1}, x_{i,2})\} = \frac{\pi_{R,0} s_{R,0}(x_{i,1}, x_{i,2})}{\sum\limits_{k=0,1} \pi_{R,k} s_{R,k}(x_{i,1}, x_{i,2})} \tag{16}$$

$P\{K_i = 0 \mid (x_{i,1}, x_{i,2})\}$ describes the probability that a signal's two observations $(x_{i,1}, x_{i,2})$ are irreproducible, i.e., the irreproducibility of $(x_{i,1}, x_{i,2})$. To estimate

Algorithm 1. The Proposed SaMic Approach

```
Input: data $(x_{i,1}, x_{i,2})$, size $n$, copulas $c_j(u, v; \lambda)$ and size $m$
Get $\hat{F}_1(x_{i,1})$ and $\hat{F}_2(x_{i,2})$
for $j = 1$ to $m$ do
  Initialize $\pi_0^{(0)}, \lambda_1^{(0)}$, $k = 0$
  repeat
    Initialize $noChange = false$
    $\pi_0^{(k+1)} = \arg\min_{\pi_0} \prod_{i=1}^{n}\{\pi_0 + (1 - \pi_0) * c_j(\hat{F}_1(x_{i,1}), \hat{F}_2(x_{i,2}), \lambda_1^{(k)})\}$
    $\lambda_1^{(k+1)} = \arg\min_{\lambda_1} \prod_{i=1}^{n}\{\pi_0^{(k+1)} + (1 - \pi_0^{(k+1)}) * c_j(\hat{F}_1(x_{i,1}), \hat{F}_2(x_{i,2}), \lambda_1)\}$
    if $|\pi_0^{(k+1)} - \pi_0^{(k)}| < \varepsilon$ and $|\lambda_1^{(k+1)} - \lambda_1^{(k)}| < \varepsilon$ then $noChange = true$ end if
  until $noChange = true$
  let $M_j$ be Pearson $\chi^2$ statistic of $c_j$
end for
get $C_R$ from (5), (11) and (12), and irreproducibility from (14), (15) and (16)
```

the parameters $\boldsymbol{\alpha}$, $\boldsymbol{\theta}_{(i)}$ and the indicators K_i of each signal, we propose an effective two-stage one-dimensional optimization method. It's worth noting that we use empirical marginal CDFs $\hat{F}_1(x_{i,1})$ and $\hat{F}_2(x_{i,2})$ instead of using raw data directly, where $\hat{F}_j(x_{i,j}) = \frac{rank_j(x_{i,j})}{n+1}$ $j = 0, 1$, since it makes experiments comparable and the rank statistic tends to cope better with real-world systematic biases and errors. The pseudo-code of our estimation procedure is shown in Alg. 1.

Compared with the IDR proposed by Li [1], a remarkable advantage of our algorithm is that it is more effective since it only needs to do one-dimensional optimization search for no more than $2km$ times, where k is the threshold of iterations. So its asymptotic time complexity is $O(mn)$. The actual running time of our algorithm is also affected by the selected threshold of precision and iterations.

4 Experiments

To evaluate the effectiveness of our proposed SaMiC approach, we compare it with IDR proposed by Li [1] in two simulated data with remarkably different marginal distributions and reproducibility structures and one real biological data. Note that although IDR has four values to be initialized, we found that they have less influence to the analysis of the final results. We chose Frank Copula, Clayton Copula and Gumbel Copula that all from Archimedean family as base copulas since these copulas have high potential of extending from bivariate Archimedean copulas to multivariate ones.

4.1 Simulated Data

In the first experiment, we generate two rows of 10,000 numbers which follow normal distributions $N(0, 1)$ and $N(2, 12)$, respectively. Then we consider these numbers as 10,000 signals' two observations to detect their (ir)reproducibilities. For the two rows of numbers generated from different distributions, there's little

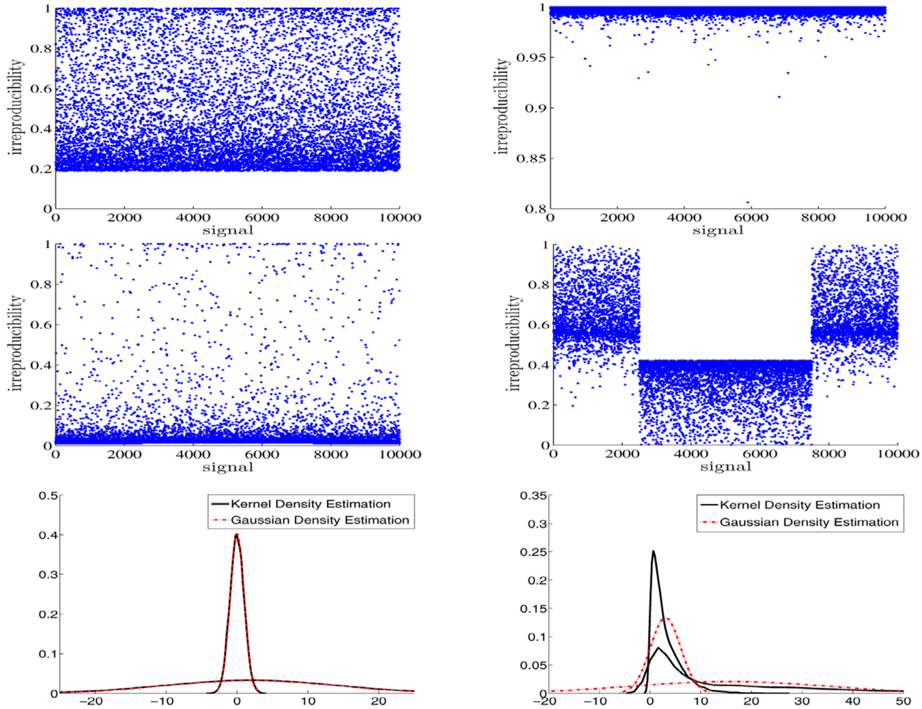


Fig. 1. Irreproducibilities (dot) of each signal observed by IDR (left) and SaMiC (right) in experiment 1 (Top) and experiment 2 (Middle). Bottom: Density Estimations of signals in experiment 1 (Left) and experiment 2 (Right).

chance that these signals' two observations are confidently consistent. So we expected the (ir)reproducibilities are low (high). Then we use both IDR and SaMiC to measure the (ir)reproducibilities of these signals. Note that both IDR and SaMiC output the irreproducibilities in $[0, 1]$. The results shown in Fig. 1 indicate that IDR has a lower recognition rate to discover the irreproducible signals. For example, if we regard those signals whose irreproducibilities are less than 0.5 are reproducible, then many irreproducible signals will be classified to be reproducible. In contrary, our SaMiC approach can correctly classify most irreproducible signals even when the cutoff value is set to be 0.9.

In the second experiment, we wish to test whether our SaMiC can be suitable for a more general distribution. Thus, we generated two rows of 10,000 random numbers combined from two different types of distributions. Firstly, we generated 5,000 random numbers $(t_1, t_2, \cdots, t_{5000})$ from Chi-square distribution $T \sim \Gamma(2, 14)$, and 10,000 random numbers $(a_1, a_2, \cdots, a_{10000})$ from beta distribution $A \sim \beta(3, 3)$. Let the first row be $(a_1, a_2, \cdots, a_{10000})$ and the second row be $(t_1, t_2, \cdots, t_{2500}, a_{2501}, a_{2502}, \cdots, a_{7500}, t_{2501}, t_{2502}, \cdots, t_{5000})$. From Fig. 1 it's obvious that IDR failed to distinguish the irreproducible signals. In

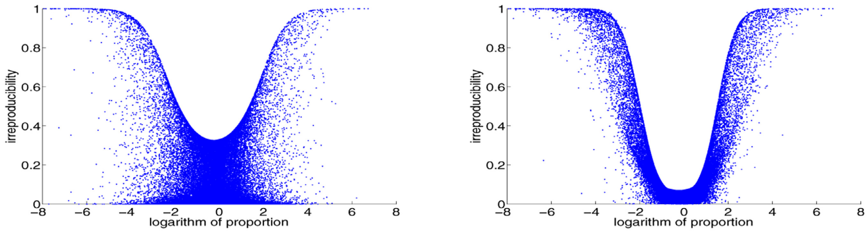


Fig. 2. HeLa-S3: The log-proportion vs. irreproducibility figure of IDR (**left**) and SaMiC (**right**)

contrast, SaMiC can estimate the (ir)reproducibilities of signals in experiment 2 with high confidence. The reason is that SaMiC makes less assumption to the dependence structure of observations, and the self-adaptive mixture copula is helpful to be suitable for general distributions.

We also use both kernel and Gaussian density estimation on these data of the two above experiments. As demonstrated in Fig. 1, the results from density estimation can show the differences between two rows of numbers only in an overall perspective. So it's hard to decide whether or not to trust some specific signals that are reproducible by using density estimation. In contrast, our method attains the (ir)reproducibility of each signal, which can distinguish bona fide signals from artificial signals.

4.2 Real Data

We also use real biological data to test the performance of SaMiC. The data can be downloaded from "http://genome.ucsc.edu/cgi-bin/hgFileUi?db=hg19 &g=wg EncodeCshlLongRnaSeq" (selected categories: Cell Line = HeLa-S3, Location = cell, RNA Extract = Long PolyA+ RNA, View: Transcript Gencode V7). This data was generated by ENCODE project [20] and they are biological experiments to detect the expression level of HeLa-S3 cell's long RNA transcripts, which were sequenced by RNA-Seq. The downloaded data file contains 161,999 annotated transcript individuals' expression values — the normalized RPKM values. As need, each transcript has two values from different experiment replicates respectively. They are estimated by SaMiC and IDR to test their performance. Different from some classical data mining problems, it's difficult to verify the experiment results on real data because of lacking test data or labels. So we need to analyze the results in some indirect ways.

Intuitively, the larger (or smaller) the proportion of a signal's two observations is, the smaller probability that the signal is reproducible. In fact, SaMiC scores are different from proportions because proportions only consider local information while SaMiC scores rely on both the entire distribution and the dependence structure of the observations. Nevertheless, it still can demonstrate some differences between IDR and SaMiC by using figure of proportions versus

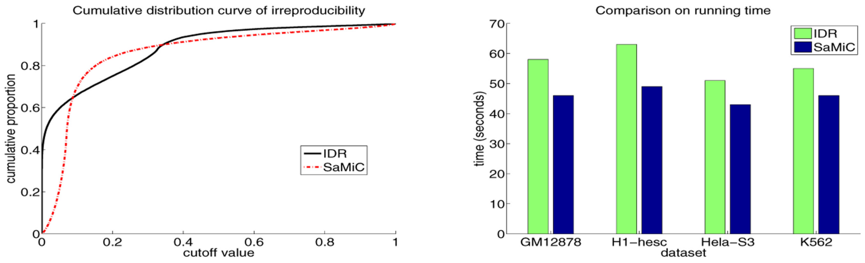


Fig. 3. Cumulative distribution curve (**left**) and comparison on running time (**right**)

irreproducibilities. As shown in Fig. 2, we draw this figure by putting logarithm of proportion on x-axis and irreproducibility on y-axis. From Fig. 2 we can see that SaMiC is more sensitive and has a stronger recognizing ability. Take signals in $(-\infty, -2] \cup [2, \infty)$ with irreproducibilities lower than 0.2 for example. The number of irreproducible signals estimated by IDR is remarkably larger than that estimated by SaMiC. It shows that SaMiC is more sensitive to (ir)reproducibility and can identify the irreproducible signals which are ignored by IDR.

For the convenience of subsequent data analysis, such as keeping specific proportion of data or choosing different critical values, we expect the irreproducibilities to be smooth. In order to compare IDR and SaMiC from this viewpoint, we produce the cumulative distribution curves of the results from both IDR and SaMiC. From Fig. 3 we observe that the curve of SaMiC is smoother than that of IDR. Besides, compared with IDR, SaMiC can provide more detailed data for keeping specific proportion of data. For example, if we want to get the signals with the lowest 20% irreproducibilities, it's easy while using SaMiC but unavailable while using IDR. This is because that there are almost 40% irreproducibilities that are 0 in the result of IDR, which means IDR fails to discriminate the signals with 40% lowest irreproducibilities while SaMiC succeeds. So SaMiC performs better on selecting the most reliable signals with a specific proportion.

Furthermore, we perform more experiments on another three different types of cells including GM12878, H1-hesc and K562, which are downloaded from the same website as HeLa-S3. For saving space, the detailed results can be found in future extended version. Based on these experiments, we give a comparison on running time. As shown in Fig. 3, SaMiC works faster than IDR on all of the four data. The computing environment is Intel Core2 2.93GHz with 4G memory.

5 Conclusion

In this paper, we have proposed a Self-adaptive Mixture Copula to measure the reproducibility of high-throughput deep-sequencing experiments, which is a difficult and challenging data mining problem since the number of samples is extremely small. The proposed SaMiC can effectively separate the dependence structure from joint distribution of signals without *priori* assumption. Compared with IDR, SaMiC can discover the irreproducible signals in a more reliable way.

SaMiC features no parameters that need to be tuned and can calculate the (ir)reproducibilities in an automatic way. It can self-adaptively choose the most suitable parameters for given data and is thus robust for different datasets. Besides, SaMiC works faster than IDR on all data we tested.

SaMiC can be used in all high-throughput deep-sequencing experiments that produce over one replicate to avoid reducing the confidence of experimental results. Actually, the reproducibility issue exists for a great number of researches so that the method of estimating reproducibility has a wide application.

In the future, we will compare SaMiC with other methods such as FDR. Furthermore, we will do more experiments with labeled data and investigate more application fields of SaMiC.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (NSFC No. 61273299, 60975044) and Ministry of Education (No. 20120071110035).

References

1. Li, Q., Brown, J.B., Huang, H., Bickel, P.: Measuring reproducibility of high-throughput experiments. The Annals of Applied Statistics 5(3), 1752–1779 (2011)
2. Kole, E., Koedijk, K., Verbeek, M.: Selecting copulas for risk management. Journal of Banking & Finance 31(8), 2405–2423 (2007)
3. Frey, R., McNeil, A.: Dependent defaults in models of portfolio credit risk. Journal of Risk 6, 59–92 (2003)
4. Trivedi, P., Zimmer, D.: Copula modeling: an introduction for practitioners, vol. 1. Now Pub. (2007)
5. Sklar, A.: Fonctions de répartition à n dimensions et leurs marges. Publ. Inst. Statist. Univ. Paris 3, 229–231 (1959)
6. Deheuvels, P.: A Kolmogorov-Smirnov type test for independence and multivariate samples. Rev. Roumaine Math. Pures Appl. 26(2), 213–226 (1981)
7. Nelsen, R.B.: An introduction to copulas. Springer, New York (1999)
8. Oakes, D.: Multivariate survival distributions. Nonparametric Statistics 3(3-4), 343–354 (1994)
9. Genest, C., Ghoudi, K., Rivest, L.P.: A semiparametric estimation procedure of dependence parameters in multivariate families of distributions. Biometrika 82(3), 543–552 (1995)
10. Joe, H.: Asymptotic efficiency of the two-stage estimation method for copula-based models. Journal of Multivariate Analysis 94(2), 401–419 (2005)
11. Chen, X., Fan, Y.: Estimation of copula-based semiparametric time series models. Journal of Econometrics 130(2), 307–335 (2006)
12. Abegaz, F., Naik-Nimbalkar, U.V.: Modeling statistical dependence of markov chains via copula models. Journal of Statistical Planning and Inference 138(4), 1131–1146 (2008)
13. Klugman, S.A., Parsa, R.: Fitting bivariate loss distributions with copulas. Insurance: Mathematics and Economics 24(1-2), 139–148 (1999)
14. Hu, L.: Dependence patterns across financial markets: a mixed copula approach. Applied Financial Economics 16(10), 717–729 (2006)

15. Engle, R.F., Manganelli, S.: Caviar. Journal of Business and Economic Statistics 22(4), 367–381 (2004)
16. Patton, A.J.: Modelling asymmetric exchange dependence. International Economic Review 47(2), 527–556 (2006)
17. Embrechts, P., McNeil, A., Straumann, D.: Correlation: pitfalls and alternatives. RISK Magazine 12, 69–71 (1999)
18. Kim, J.M., Jung, Y.S., Sungur, E., Han, K.H., Park, C., Sohn, I.: A copula method for modeling directional dependence of genes. BMC Bioinformatics 9(225) (2008)
19. Zhang, Y., Liu, T., Meyer, C., Eeckhoute, J., Johnson, D., Bernstein, B., Nussbaum, C., Myers, R., Brown, M., Li, W., et al.: Model-based analysis of ChIP-Seq (MACS). Genome Biol. 9(9), R137 (2008)
20. Myers, R., Stamatoyannopoulos, J., Snyder, M., Dunham, I., Hardison, R., Bernstein, B., Gingeras, T., Kent, W., Birney, E., et al.: A user's guide to the encyclopedia of dna elements (ENCODE project consortium). PLoS Biol. 9(4), e1001046 (2011)

Mining Representative Movement Patterns through Compression

Phan Nhat Hai[1], Dino Ienco[1], Pascal Poncelet[2], and Maguelonne Teisseire[1]

[1] IRSTEA Montpellier, UMR TETIS - 34093 Montpellier, France
{nhat-hai.phan,dino.ienco,maguelonne.teisseire}@teledetection.fr
[2] LIRMM CNRS Montpellier - 34090 Montpellier, France
pascal.poncelet@lirmm.fr

Abstract. Mining trajectories (or moving object patterns) from spatio-temporal data is an active research field. Most of the researches are devoted to extract trajectories that differ in their structure and characteristic in order to capture different object behaviors. The first issue is constituted from the fact that all these methods extract thousand of patterns resulting in a huge amount of redundant knowledge that poses limit in their usefulness. The second issue is supplied from the nature of spatio-temporal database from which different types of patterns could be extracted. This means that using only a single type of patterns is not sufficient to supply an insightful picture of the whole database.

Motivating by these issues, we develop a Minimum Description Length (MDL)-based approach that is able to compress spatio-temporal data combining different kinds of moving object patterns. The proposed method results in a rank of the patterns involved in the summarization of the dataset. In order to validate the quality of our approach, we conduct an empirical study on real data to compare the proposed algorithms in terms of effectiveness, running time and compressibility.

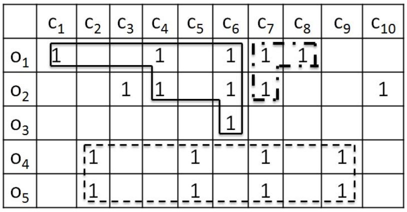
Keywords: MDL, moving objects, spatio-temporal data, top-k, compressibility.

1 Introduction

Nowadays, the use of many electronic devices in real world applications has led to an increasingly large amount of data containing moving object information. One of the objectives of spatio-temporal data mining [5] [10] [6] is to analyze such datasets for interesting moving object clusters. A moving object cluster can be defined as a group of moving objects that are physically closed to each other for at least some number of timestamps. In this context, many recent studies have been define such as flock [5], convoy queries [7], closed swarms [10], group patterns [15], gradual trajectory patterns [6], traveling companions [13], gathering patterns [16], etc...

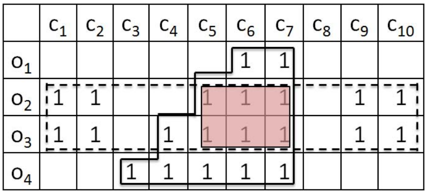
Nevertheless, after the extraction, the end user can be overwhelmed by a huge number of movement patterns although only a few of them are useful. However, relatively few researchers have addressed the problem of reducing movement pattern redundancy. In another context, i.e. frequent itemsets, the Krimp algorithm [14], using the minimum description length (MDL) principle [4], proposes to reduce the amount of itemsets by

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 314–326, 2013.



	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}
o_1	1			1		1	1	1		
o_2			1	1		1	1			1
o_3						1				
o_4		1			1		1		1	
o_5		1			1		1		1	

Fig. 1. An example of moving object database. Shapes are movement patterns, o_i, c_i respectively are objects and clusters.



	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}
o_1						1	1			
o_2	1	1			1	1	1		1	1
o_3	1	1		1	1	1	1		1	1
o_4			1	1	1	1	1			

Fig. 2. An example of pattern overlapping, between closed swarm (dashed line rectangle) and $rGpattern^{\geq}$ (step shape), overlapping clusters are c_5, c_6 and c_7

using an efficient encoding and then provide the end-user only with a set of informative patterns.

In this paper, we adapt the MDL principle for mining representative movement patterns. However, one of the key challenges in designing an MDL-based algorithm for moving object data is that the encoding scheme needs to deal with different pattern structures which can cover different parts of the data. If we only consider different kinds of patterns individually then it is difficult to obtain an optimal set of compression patterns.

For instance, see Figure 1, we can notice that there are three different patterns, with different structures, that cover different parts of the moving object data. If we only keep patterns having a rectangular shape then we lose the other two patterns and viceversa.

Furthermore, although patterns express different kinds of knowledge, they can overlap each other as well. Thus, enforcing non-overlapping patterns may result in losing interesting patterns. For instance, see Figure 2, there are two overlapping patterns. Krimp algorithm does not allow overlapping patterns then it has to select one and obviously loses the other one. However, they express very different knowledge and thus, by removing some of them, we cannot fully understand the object movement behavior. Therefore, the proposed encoding scheme must to appropriately deal with the pattern overlapping issue.

Motivated by these challenges, we propose an overlapping allowed multi-pattern structure encoding scheme which is able to compress the data with different kinds of patterns. Additionally, the encoding scheme also allows overlapping between different kinds of patterns. To extract compression patterns, a naive greedy approach, named NAIVECOMPO, is proposed. To speed up the process, we also propose the SMARTCOMPO algorithm which takes into account several useful properties to avoid useless computation. Experimental results on real-life datasets demonstrate the effectiveness and efficiency of the proposed approaches by comparing different sets of patterns.

2 Preliminaries and Problem Statement

2.1 Object Movement Patterns

Object movement patterns are designed to group similar trajectories or objects which tend to move together during a time interval. In the following, we briefly present the definitions of different kinds of movement patterns.

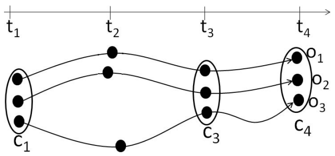


Fig. 3. An example of closed swarm

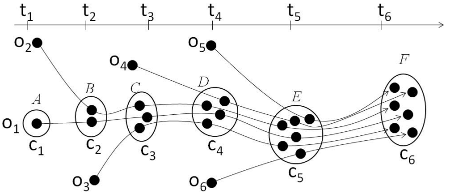


Fig. 4. An example of rGpattern

Database of clusters. Let us consider a set objects occurring at different timestamps. A database of clusters, $C_{DB} = \{C_{t_1}, C_{t_2}, \ldots, C_{t_m}\}$, is a collection of snapshots of the moving object clusters at timestamps $\{t_1, t_2, \ldots, t_m\}$. Given a cluster $c \in C_{t'}(\in C_{DB})$, $t(c)$ and $o(c)$ are respectively used to denote the timestamp that c is involved in and the set of objects included in c. For brevity sake, we take clustering as a preprocessing step.

After generating C_{DB}, the moving object database (O_{DB}, T_{DB}) is define such as each object $o \in O_{DB}$ contains a list of clusters (i.e. $o = c_1 c_2 \ldots c_m$) and T_{DB} stands for the associated timestamp. For instance, Figure 1 presents the database O_{DB} and object o_1 can be represented as $o_1 = c_1 c_4 c_6 c_7 c_8$.

From this set different patterns can be extracted. In an informal way, a closed swarm is a list of clusters $cs = c_1 \ldots c_n$ such that they share at least ε common objects, cs contains at least min_t clusters and cs cannot be enlarged in terms of objects and clusters. Note that there are no pairs of clusters which are in the same timestamps involved in cs. Then a closed swarm can be formally define as follows:

Definition 1. *ClosedSwarm[10]. A list of clusters $cs = c_1 \ldots c_n$ is a closed swarm if:*

$$\begin{cases} (1): |O(cs)| = |\bigcap_{i=1}^{n} c_i| \geq \varepsilon. \\ (2): |cs| \geq min_t. \\ (3): \nexists i, j \in \{1, \ldots, n\}, i \neq j, t(c_i) = t(c_j). \\ (4): \nexists cs' : cs \subset cs', cs' \textit{ satisfies the conditions (1), (2) and (3).} \end{cases} \tag{1}$$

For instance, see Figure 3, $cs = c_1 c_3 c_4$ is a closed swarm with $min_t = 2, \varepsilon = 2$. Similarly, in Figure 1, we also have $cs = c_2 c_5 c_7 c_9$ is a closed swarm. A convoy is a group of objects such that these objects are closed each other during at least min_t *consecutive* time points. Another pattern is group pattern which essentially is a set of disjointed convoys which are generated by the same group of objects in different time intervals. In this paper, we only consider closed swarm instead of convoy and group pattern since closed swarm is more general [10].

A gradual trajectory pattern [6], denoted *rGpattern*, is designed to capture the gradual object moving trend. More precisely, a rGpattern is a maximal list of moving object clusters which satisfy the graduality constraint and integrity condition during at least min_t timestamps. The graduality constraint can be the increase or decrease of the number of objects and the integrity condition can be that all the objects should remain in the next cluster. A rGpattern can be defined as follows:

Definition 2. *rGpattern [6]. Given a list of clusters $C^* = c_1 \dots c_n$. C^* is a gradual trajectory pattern if:*

$$C^* = C^{\geq} \begin{cases} (1) : |C^*| \geq min_t. \\ \forall i \in \{1, \dots, n-1\}, \\ (2) : o(c_i) \subseteq o(c_{i+1}). \\ (3) : |c_n| > |c_1|. \\ (4) : \nexists c_m : C^* \cup c_m \; is \; a \; C^{\geq}. \end{cases} \qquad C^* = C^{\leq} \begin{cases} (1) : |C^*| \geq min_t. \\ \forall i \in \{1, \dots, n-1\}, \\ (2) : o(c_i) \supseteq o(c_{i+1}). \\ (3) : |c_n| < |c_1|. \\ (4) : \nexists c_m : C^* \cup c_m \; is \; a \; C^{\geq}. \end{cases}$$

Essentially, we have two kinds of rGpatterns, $rGpattern^{\geq}$ and $rGpattern^{\leq}$. For instance, see Figure 1, $rGpattern^{\geq} = c_1c_4c_6$ and $rGpattern^{\leq} = c_7c_8$.

2.2 Problem Statement

Eliminating the number of uninteresting patterns is an emerging task in many real world cases. One of the proposed solutions is the MDL principle [4]. Let us start explaining this principle in the following definition:

Definition 3. *(Hypothesis). A hypothesis $\mathcal{P}$ is a set of patterns $\mathcal{P} = \{p_1, p_2, \dots, p_h\}$.*

Given a scheme S, let $L_S(P)$ be the description length of hypothesis $\mathcal{P}$ and $L_S(O_{DB}|P)$ be the description length of data O_{DB} when encoded with the help of the hypothesis and an encoding scheme S. Informally, the MDL principle proposes that the best hypothesis always compresses the data most. Therefore, the principle suggests that we should look for hypothesis $\mathcal{P}$ and the encoding scheme S such that $L_S(O_{DB}) = L_S(\mathcal{P}) + L_S(O_{DB}|\mathcal{P})$ is minimized. For clarity sake, we will omit S when the encoding scheme is clear from the context. Additionally, the description length of O_{DB} given $\mathcal{P}$ is denoted as $L_{\mathcal{P}}(O_{DB}) = L(\mathcal{P}) + L(O_{DB}|\mathcal{P})$.

In this paper, the hypothesis is considered as a dictionary of movement patterns $\mathcal{P}$. Furthermore, as in [9], we assume that any number or character in data has a fixed length bit representation which requires a unit memory cell. In our context, the description length of a dictionary $\mathcal{P}$ can be calculated as the total lengths of the patterns and the number of patterns (i.e. $L(\mathcal{P}) = \sum_{p \in \mathcal{P}} |p| + |\mathcal{P}|$). Furthermore, the length of the data O_{DB} when encoded with the help of dictionary $\mathcal{P}$ can be calculated as $L(O_{DB}|\mathcal{P}) = \sum_{o \in O_{DB}} |o|$.

The problem of finding compressing patterns can be formulated as follows:

Definition 4. *(Compressing Pattern Problem). Given a moving object database O_{DB}, a set of pattern candidates $F = \{p_1, p_2, \dots, p_m\}$. Discover an optimal dictionary $\mathcal{P}^*$ which contains at most K movement patterns so that:*

$$\mathcal{P}^* = \arg\min_{\mathcal{P}} \big(L^*_{\mathcal{P}}(O_{DB})\big) = \arg\min_{\mathcal{P}} \big(L^*(\mathcal{P}) + L^*(O_{DB}|\mathcal{P})\big), \mathcal{P}^* \subseteq F \quad (2)$$

A key issue in designing an MDL-based algorithm is: how can we encode data given a dictionary? The fact is that if we consider closed swarms individually, Krimp algorithm can be easily adapted to extract compression patterns. However, the issue here is that we have different patterns (i.e. closed swarms and rGpatterns) and Krimp algorithm has not been designed to deal with rGpatterns. It does not supply multi-pattern types in the dictionary that may lead to losing interesting ones. Furthermore, as mentioned before, we also have to address the pattern overlapping issue. In this work, we propose a novel overlapping allowed multi-pattern structures encoding scheme for moving object data.

3 Encoding Scheme

3.1 Movement Pattern Dictionary-Based Encoding

Before discussing our encoding for moving object data, we revisit the encoding scheme used in the Krimp algorithm [14]. An itemset I is encoded with the help of itemset patterns by replacing every non-overlapping instance of a pattern occurring in I with a pointer to the pattern in a code table (dictionary). In this way, an itemset can be encoded to a more compact representation and decoded back to the original itemset.

Table 1. An illustrative example of database and dictionary in Figure 1. $\bar{0}$, $\bar{1}$ and $\bar{2}$ respectively are pattern types: closed swarm, $rGpattern^{\geq}$ and $rGpattern^{\leq}$.

O_{DB}	Encoded O_{DB}	Dictionary $\mathcal{P}$
$o_1 = c_1c_4c_6c_7c_8$	$o_1 = [p_1, 0][p_3, 1]$	$p_1 = c_1c_4c_6, \bar{1}$
$o_2 = c_3c_4c_6c_7c_{10}$	$o_2 = c_3[p_1, 1][p_3, 0]c_{10}$	$p_2 = c_2c_5c_7c_9, \bar{0}$
$o_3 = c_6$	$o_3 = [p_1, 2]$	$p_3 = c_7c_8, \bar{2}$
$o_4 = c_2c_5c_7c_9$	$o_4 = p_2$	
$o_5 = c_2c_5c_7c_9$	$o_5 = p_2$	

In this paper we use a similar dictionary-based encoding scheme for moving object database. Given a dictionary consisting of movement patterns $\mathcal{P} = \{p_1, \ldots, p_m\}$, an object $o \in O_{DB}$ containing a list of clusters is encoded by replacing instances of any pattern p_i in o with pointers to the dictionary. An important difference between itemset data and moving object data is that there are different kinds of movement patterns which have their own characteristic. The fact is that if a closed swarm cs occurs in an object o then all the clusters in cs are involved in o. While an object can involve in only a part of a rGpattern and viceversa.

For instance, see Figure 1, we can consider that o_2 joins the $rGpattern^{\geq} = c_1c_4c_6$ at c_4c_6. While, the closed swarm $cs = c_2c_5c_7c_9$ occurs in o_4 and o_5 entirely.

Property 1. (Encoding Properties). Given an object o which contains a list of clusters and a pattern $p = c_1 \ldots c_n$. p occurs in o or o contributes to p if:

$$\begin{cases} (1) : \textit{p is a rGpattern}^{\geq}, \exists i \in [1, n] \big| \forall j \geq i, c_j \in o. \\ (2) : \textit{p is a rGpattern}^{\leq}, \exists i \in [1, n] \big| \forall j \leq i, c_j \in o. \\ (3) : \textit{p is a closed swarm}, \forall j \in [1, n], c_j \in o. \end{cases} \tag{3}$$

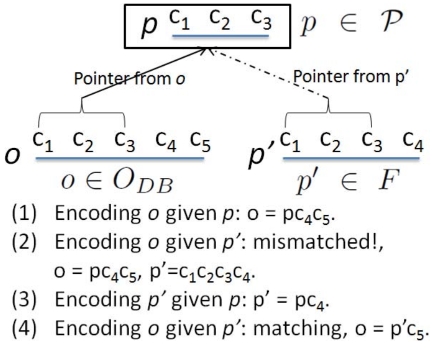
Proof. Case (1): after construction we have $o(c_i) \subseteq o(c_{i+1}) \subseteq \ldots \subseteq o(c_n)$. Additionally, $o \in o(c_i)$. Consequently, $o \in o(c_{i+1}), \ldots, o(c_n)$ and therefore $\forall j \geq i, c_j \in o$. Furthermore, in Case (2): we have $o(c_1) \supseteq o(c_2) \supseteq \ldots \supseteq o(c_{i-1})$. Additionally, $o \in o(c_{i-1})$. Consequently, $o \in o(c_1), \ldots, o(c_{i-1})$ and therefore $\forall j \leq i, c_j \in o$. In Case (3), we have $o \in O(cs) = \bigcap_{i=1}^{n} c_i$ and therefore $\forall j \in [1, n], c_j \in o$.

For instance, see Table 1, we can see that for each pattern, we need to store an extra bit to indicate the pattern type. Regarding to closed swarm, by applying Property 1, in the object o we only need to replace all the clusters, which are included in closed swarm, by a pointer to the closed swarm in the dictionary. However, in gradual trajectories (i.e. $rGpattern^{\geq}$, $rGpattern^{\leq}$), we need to store with the pointer an additional index to indicate the cluster c_i. Essentially, c_i plays the role of a starting involving point (resp. ending involving point) of the object o in a $rGpattern^{\geq}$ (resp. $rGpattern^{\leq}$).

As an example, consider dictionary $\mathcal{P}$ in Table 1. Using $\mathcal{P}$, o_1 can be encoded as $o_1 = [p_1, 0][p_3, 1]$ where 0 (in $[p_1, 0]$) indicates the cluster at index 0 in p_1, (i.e. c_1) and 1 (in $[p_3, 1]$) indicates the cluster at index 1 in p_3, i.e. c_8. While, o_4 can be encoded as $o_4 = p_2$, i.e. p_2 is a closed swarm.

3.2 Overlapping Movement Pattern Encoding

Until now, we have already presented the encoding function for different patterns when encoding an object o given a pattern p. In this section, the encoding scheme will be completed by addressing the pattern overlapping problem so that overlapped patterns can exist in the dictionary $\mathcal{P}$.



(1) Encoding o given p: o = pc$_4$c$_5$.
(2) Encoding o given p': mismatched!, o = pc$_4$c$_5$, p'=c$_1$c$_2$c$_3$c$_4$.
(3) Encoding p' given p: p' = pc$_4$.
(4) Encoding o given p': matching, o = p'c$_5$.

Fig. 5. An example of the approach

See Figure 5, a selected pattern $p \in \mathcal{P}$ and a candidate $p' \in F$ overlap each other at $c_1c_2c_3$ on object o. Assume that o is encoded given p then $o = pc_4c_5$. As in Krimp algorithm, p' is still remained as origin and then p' cannot be used to encode o despite of p' occurs in o. This is because they are mismatched (i.e. $o = pc_4c_5, p' = c_1c_2c_3c_4$). To solve the problem, we propose to encode p' given p so that o and p' will contain the same pointer to p (i.e. $p' = pc_4$). Now, the regular encoding scheme can be applied to encode o given p' (i.e. $o = p'c_5$). We can consider that p and p' are overlapping but both of them can be included in the dictionary $\mathcal{P}$. **Note:** in our context, overlapped clusters are counted only once.

Main idea. Given a dictionary $\mathcal{P}$ and a chosen pattern p (i.e. will be added into $\mathcal{P}$), a set of pattern candidates F. The main idea is that we first encode the database O_{DB} given pattern p. Secondarily, we propose to encode all candidates $p' \in F$ given p in order to indicate the overlapping clusters between p and p'. After that, there are two kinds of pattern candidates which are encoded candidates and non-encoded candidates. Next, the best candidate in F will be put into $\mathcal{P}$ and used to encode O_{DB} and F. The process will be repeat until obtaining $top\text{-}K$ patterns in the dictionary $\mathcal{P}$.

Let us consider the correlations between a pattern $p \in \mathcal{P}$ and a candidate $p' \in F$ to identify whenever encoding p' given p is needed. The correlation between p and p' is illustrated in Table 2. First of all, we do not allow overlap between two patterns of the same kind since they represent the same knowledge that may lead to extracting redundant information.

Table 2. Correlations between pattern p and pattern p' in F. O, Δ and X respectively mean *”overlapping allowed, regular encoding”*, *”overlapping allowed, no encoding”* and *”overlapping not allowed”*.

		p		
		cs	$rGpattern^{\geq}$	$rGpattern^{\leq}$
p'	cs	X	O	O
	$rGpattern^{\geq}$	Δ	X	O
	$rGpattern^{\leq}$	Δ	O	X

Next, if p is a closed swarm then p' do not need to be encoded given p. This is because there are objects which contribute to gradual trajectories p' but not closed swarm. These objects cannot be encoded using p and therefore p' needs to be remained the same and the regular encoding scheme can be applied. Otherwise, p' will never be chosen later since there are no objects in O_{DB} which match p'. For instance, see Figure 2, the objects o_1 and o_4 do not contribute to the closed swarm p. Thus, if the gradual trajectory p' is encoded given p to indicate the overlapping clusters $c_5c_6c_7$ then that leads to a mismatched statement between o_1, o_4 and the gradual trajectory p'.

Until now, we already have two kinds of candidates $p' \in F$ (i.e. non-encoded and encoded candidates). Next, some candidates will be used to encode the database O_{DB}. To encode an object $o \in O_{DB}$ given a non-encoded candidate p', the regular encoding scheme mentioned in Section 3.1 can be applied. However, given an encoded candidate p', we need to perform an additional step before so that the encoding scheme can be applied regularly. This is because the two pointers referring to the same pattern $p \in \mathcal{P}$ from o (e.g. $[p, k]$) and from p' (e.g. $[p, l]$) can be different (i.e. $k \neq l$) despite the fact that p' is essentially included in o. That leads to a mismatched statement between o and p' and thus o cannot be encoded given p'.

For instance, see Figure 2, given a gradual trajectory pattern $rGpattern^{\geq}$ $p = c_3c_4c_5c_6c_7$, a closed swarm $p' = c_1c_2c_5c_6c_7c_9c_{10}$, the object $o_3 = c_1c_2c_4c_5c_6c_7c_9c_{10}$. We first encodes o_3 given p such that $o_3 = c_1c_2[p, 1]c_9c_{10}$. Then, p' is encoded given p, i.e. $p' = c_1c_2[p, 2]c_9c_{10}$. We can consider that the two pointers referring to p from o (i.e. $[p, 1]$) and from p' (i.e. $[p, 2]$) are different and thus o_3 and p' are mismatched. Therefore, o cannot be encoded given p' despite the fact that p' essentially occurs in o.

To deal with this issue, we simply recover uncommon clusters between the two pointers. For instance, to encode o_3 by using p', we first recover uncommon cluster such that $o_3 = c_1c_2c_4[p, 2]c_9c_{10}$. Note that $[p, 1] = c_4[p, 2]$. Since $p' = c_1c_2[p, 2]c_9c_{10}$, o_3 is encoded given p' such that $o_3 = p'c_4$.

Definition 5. *(Uncommon Clusters for rGpattern$^{\geq}$). Given a rGpattern$^{\geq}$, $p = c_1 \ldots c_n$ and two pointers refer to p, $[p, k]$ and $[p, l]$ with $k \leq l$. $uncom(p, k, l) = c_kc_{k+1} \ldots c_{l-1}$ is called an uncommon list of clusters between $[p, k]$ and $[p, l]$. Note that $[p, k] = c_kc_{k+1} \ldots c_{l-1}[p, l]$.*

Similarly, we also have $uncom(p, k, l)$ in the case p is a $rGpattern^{\leq}$. Until now, we are able to recover uncommon clusters between two pointers which refer to a pattern. Now, we start proving that given an object $o \in O_{DB}$ and a candidate $p' \in F$, if p' occurs in o then o can be encoded using p' even though they contain many pointers to other patterns. First, let us consider if p is a $rGpattern^{\geq}$ and p' is a closed swarm.

Lemma 1. *Given a rGpattern*$^{\geq}$*,* $p = c_1 \ldots c_n$*, an object* o *and a closed swarm* $p' \in F$*. In general, if* o *and* p' *refer to* p *then* $o = x_o[p,k]y_o$ *and* $p' = x_{p'}[p,l]y_{p'}$*. Note that* $x_o, y_o, x_{p'}$ *and* $y_{p'}$ *are lists of clusters. If* o *contributes to* p' *then:*

$$k \leq l \wedge o = x_o\ uncom(p,k,l)[p,l]\ y_o \tag{4}$$

Proof. After construction if $k > l$ then $\exists c_i \in \{c_l, \ldots, c_k\} (\subseteq p)$ s.t. $c_i \in p' \wedge c_i \notin o$. Therefore, o does not contribute to p' (Property 1). That suffers the assumption and thus we have $k \leq l$. Deal to the Defi ition 5, $[p,k] = uncom(p,k,l)[p,l]$. Consequently, we have $o = x_o\ uncom(p,k,l)[p,l]\ y_o$.

By applying Lemma 1, we have $o = x_o\ uncom(p,k,l)[p,l]\ y_o$ and $p' = x_{p'}[p,l]y_{p'}$. Then we can apply the regular encoding scheme to encode o given p'. let us assume that each object $o \in O_{p'}$ has a common list of pointers to other patterns as $\overrightarrow{(p',o)} = \{([p_1,l_1],[p_1,k_1]), \ldots, ([p_n,l_n],[p_n,k_n])\}$ where $\forall i \in [1,n], [p_i,l_i]$ is the pointer from p' to p_i and $[p_i,k_i]$ is the pointer from o to p_i. If we respectively apply Lemma 1 on each pointer in $\overrightarrow{(p',o)}$ then o can be encoded given p'. Similarly, we also have the other lemmas for other pattern types.

Data description length computation. Until now, we have defined an encoding scheme for movement patterns. The description length of the dictionary in Table 1 is calculated as $L(\mathcal{P}) = |p_1|+1+|p_2|+1+|p_3|+1+|\mathcal{P}| = 3+1+4+1+2+1+2 = 14$. Similarly, description length of o_2 is $L(o_2|\mathcal{P}) = 1 + |[p_1,1]| + |[p_3,0]| + 1 = 6$.

Note: for each pattern, we need to consider an extra memory cell of pattern type. Additionally, for any given dictionary $\mathcal{P}$ and the data O_{DB}, the cost of storing the timestamp for each cluster is always constant regardless the size of the dictionary.

4 Mining Compression Object Movement Patterns

In this section we will present the two greedy algorithms which have been designed to extract a set of *top-K* movement patterns that compress the data best.

4.1 Naive Greedy Approach

The greedy approach takes as input a database O_{DB}, a candidate set F and a parameter K. The result is the optimal dictionary which encodes O_{DB} best. Now, at each iteration of *NaiveCompo*, we select candidate p' which compresses the database best. Next, p' will be added into the dictionary $\mathcal{P}$ and then the database O_{DB} and F will be encoded given p'. The process is repeated until we obtain K patterns in the dictionary.

To select the best candidate, we generate a duplication of the database O^d_{DB} and for each candidate $p' \in F$, we compress O^d_{DB}. The candidate p' which returns the smallest data description length will be considered as the best candidate. Note that $p' = argmin_{p^* \in F}\big(L_{p^*}(O_{DB})\big)$. The NAIVECOMPO is presented in Algorithm 1.

4.2 Smart Greedy Approach

Algorithm 1. NaiveCompo

```
Input  : Database $O_{DB}$, set of patterns $F$, int $K$
Output: Compressing patterns $\mathcal{P}$
Input  : Database $O_{DB}$, set of patterns $F$, int $K$
Output: Compressing patterns $\mathcal{P}$
begin
    $\mathcal{P} \longleftarrow \emptyset$;
    while $|\mathcal{P}| < K$ do
        foreach $p \in F$ do
            $O^d_{DB} \longleftarrow O_{DB}$;
            $L^*(O^d_{DB}|p) \longleftarrow$
            $CompressionSize(O^d_{DB}, p)$;
        $p^* \longleftarrow \arg\min_p L^*(O^d_{DB}|p)$;
        $\mathcal{P} \longleftarrow p^*$; $F \longleftarrow F \setminus \{p^*\}$;
        Replace all instances of $p^*$ in $O_{DB}$ by its pointers;
        Replace all instances of $p^*$ in $F$ by its pointers;
    output $\mathcal{P}$;
CompressionSize($O^d_{DB}, p$)
begin
    $size \longleftarrow 0$;
    foreach $o \in O_{DB}$ do
        if $p.involved(o) = true$ then
            Replace instance of $p$ in $o$ by its pointers;
    foreach $o \in O_{DB}$ do
        $size \longleftarrow size + |o|$;
    $size \longleftarrow size + |p| + 1$;
    output $size$;
```

The disadvantage of naive greedy algorithm is that we need to compress the duplicated database O^d_{DB} for each pattern candidate at each iteration. However, we can avoid this computation by considering some useful properties as follows.

Given a pattern p', $\overline{O}_{p'}$ and $O_{p'}$ respectively are the set of objects that do not contribute to p' and the set of objects involving in p'. The compression gain which is the number of memory cells we earned when adding p' into dictionary can be defined as $gain(p', \mathcal{P}) = L_{\mathcal{P}}(O_{DB}) - L_{\mathcal{P} \cup p'}(O_{DB})$.

The fact is that we can compute the compression gain by scanning objects $o \in O_{p'}$ with p'. Each pattern type has its own compression gain computation function. Let us start presenting the process by proposing the property for a closed swarm p'.

Property 2. Given a dictionary $\mathcal{P}$, a closed swarm $p' \in F$. $gain(p', \mathcal{P})$ is computed as:

$$gain(p', \mathcal{P}) = |O_{p'}| \times |p'| - \Big(\sum_{o}^{O_{p'}} \sum_{i}^{\overrightarrow{(p', o)}} |l_i - k_i| + |p'| + |O_{p'}| + 2 \Big) \qquad (5)$$

Proof. After construction we have $L_{\mathcal{P} \cup p'}(O_{DB}) = L(\mathcal{P} \cup p') + L(O_{DB}|\mathcal{P} \cup p') = (L(\mathcal{P}) + |p'| + 2) + L(\overline{O}_{p'}|\mathcal{P}) + L(O_{p'}|\mathcal{P} \cup p')$. Note that $L(\overline{O}_{p'}|\mathcal{P}) = L(\overline{O}_{p'}|\mathcal{P} \cup p')$. Furthermore, $\forall o \in O_{p'} : L(o|\mathcal{P} \cup p') = L(o|\mathcal{P}) - |p'| + 1 + \sum_i^{\overrightarrow{(p', o)}} |l_i - k_i|$. Thus, $L(O_{p'}|\mathcal{P} \cup p') = \sum_{o \in O_{p'}} L(o|P \cup p') = L(O_{p'}|\mathcal{P}) - |O_{p'}| \times |p'| + \sum_o^{O_{p'}} \sum_i^{\overrightarrow{(p', o)}} |l_i - k_i| + |O_{p'}|$. Therefore, we have $L_{\mathcal{P} \cup p'}(O_{DB}) = L(\mathcal{P}) + L(\overline{O}_{p'}|\mathcal{P}) + L(O_{p'}|\mathcal{P}) - |O_{p'}| \times |p'| + \big(\sum_o^{O_{p'}} \sum_i^{\overrightarrow{(p', o)}} |l_i - k_i| + |p'| + |O_{p'}| + 2 \big)$. Note that $L(O_{DB}|\mathcal{P}) = L(\overline{O}_{p'}|\mathcal{P}) + L(O_{p'}|\mathcal{P})$. Consequently, we have $gain(p', \mathcal{P}) = |O_{p'}| \times |p'| - \big(\sum_o^{O_{p'}} \sum_i^{\overrightarrow{(p', o)}} |l_i - k_i| + |p'| + |O_{p'}| + 2 \big)$.

By applying Property 2, we can compute the compression gain when adding a new closed swarm p' into the dictionary $\mathcal{P}$. In the Equation 5, the compression $gain(p', \mathcal{P})$

depends on the size of p', $O(p')$ and the number of uncommon clusters that can be computed by scanning p' with objects $o \in O(p')$ without encoding O_{DB}. Due to the space limitation, we will not describe properties and proofs for the other pattern types (i.e. $rGpattern^{\geq}$, $rGpattern^{\leq}$) but they can be easily derived in a same way as Property 2.

To select the best candidate at each iteration, we need to chose the candidate which returns the best compression gain. SMARTCOMPO is presented in the Algorithm 2.

5 Experimental Results

Algorithm 2. SmartCompo

```
Input : Database $O_{DB}$, set of patterns $F$, int $K$
Output: Compressing patterns $\mathcal{P}$
begin
    $\mathcal{P} \longleftarrow \emptyset$;
    while $|\mathcal{P}| < K$ do
        foreach $p \in F$ do
            $L^*(O_{DB}|p) \longleftarrow Benefit(O_{DB}, p)$;
        $p^* \longleftarrow \arg\min_p L^*(O_{DB}|p)$;
        $\mathcal{P} \longleftarrow p^*$; $F \longleftarrow F \setminus \{p^*\}$;
        Replace all instances of $p^*$ in $O_{DB}$ by its pointers;
        Replace all instances of $p^*$ in $F$ by its pointers;
    output $\mathcal{P}$;
Benefit($O^d_{DB}, p$)
begin
    $b \longleftarrow 0$;
    foreach $o \in O_{DB}$ do
        if $p.involved(o) = true$ then
            $b \longleftarrow b + benefit(o, p)$;
    $b \longleftarrow b + |p| + 1$;
    output $b$;
```

A comprehensive performance study has been conducted on real-life datasets. All the algorithms are implemented in C++, and all the experiments are carried out on a 2.8GHz Intel Core i7 system with 4GB Memory. The system runs Ubuntu 11.10 and g++ 4.6.1.

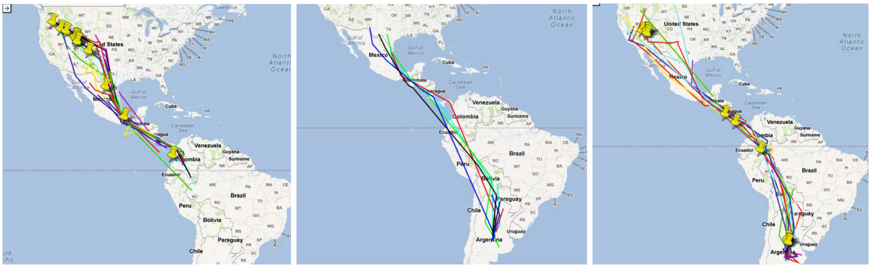
As in [10] [6], the two following datasets[1] have been used during experiments: Swainsoni dataset includes 43 objects evolving over 764 different timestamps. The dataset was generated from July 1995 to June 1998. Buffalo dataset concerns 165 buffaloes and the tracking time from year 2000 to year 2006. The original data has 26610 reported locations and 3001 timestamps. Similarly to [7] [10], we first use linear interpolation to fill in the missing data. Furthermore, DBScan [2] ($MinPts = 2; Eps = 0.001$) is applied to generate clusters at each timestamp. In the comparison, we compare the set of patterns produced by SmartCompo with the set of closed swarms extracted by *ObjectGrowth* [10] and the set of gradual trajectories extracted by *ClusterGrowth* [6].

Effectiveness. We compare the top-5 highest support closed swarms, the top-5 highest covered area gradual trajectory patterns and the top-5 compression patterns from Swainsoni dataset. Each color represents a Swainsoni trajectory involved in the pattern.

Top-5 closed swarms are very redundant since they only express that Swainsonies move together from North America to Argentina. Similarly, top-5 rGpatterns are also redundant. They express the same knowledge that is *"from 1996-10-01 to 1996-10-25, the more time passes, the more objects are following the trajectory* $\{Oregon\rangle\ Nevada\rangle\ Utah\rangle\ Arizona\rangle\ Mexico\rangle\ Colombia\}$*"*.

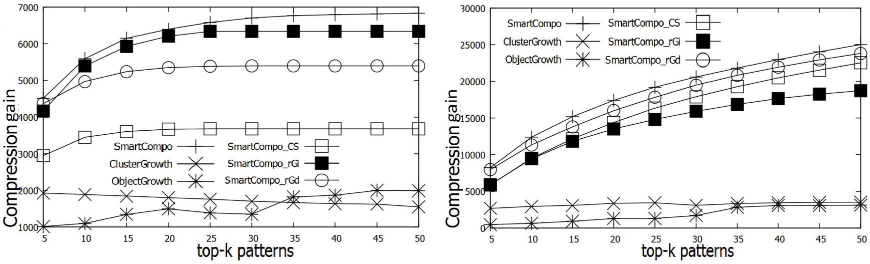
Figure 6 illustrates 3 patterns among 5 extracted ones by using SmartCompo. The $rGpattern^{\geq}$ expresses the same knowledge with the mentioned rGpattern in the top highest covered area. The closed swarm expresses new information that is *"after arriving South America, the Swainsonies tend to move together to Argentina even some*

[1] http://www.movebank.org



(a) $rGpattern \geq$ (b) Closed swarm (c) $rGpattern^{\leq}$

Fig. 6. Top-3 typical compression patterns



(a) Swainsoni dataset (b) Buffalo dataset

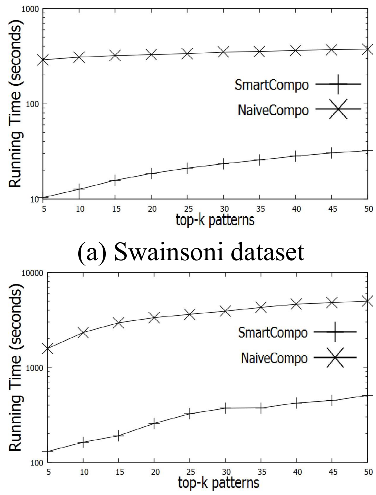
Fig. 7. Compressibility (higher is better) of different algorithms

of them can leave their group". Next, the $rGpattern^{\leq}$ shows that *"the Swainsonies return back together to North America from Argentina (i.e. 25 objects at Argentina) and they will step by step leave their group after arriving Guatemala (i.e. 20 objects at Guatemala) since they are only 2 objects at the last stop, i.e. Oregon State"*.

Compressibility. We measure the compressibility of the algorithms by using their *top-*K patterns as dictionaries for encoding the data. Since NaiveCompo and SmartCompo provides the same results, we only show the compression gain of SmartCompo.

Regarding to SmartCompo, the compression gain could be calculated as the sum of the compression gain returned after each greedy step with all kinds of patterns in F. For each individual pattern type, compression gain is calculated according to the greedy encoding scheme used for SmartCompo. They are respectively denoted as $SmartCompo\_CS$ (i.e. for closed swarms), $SmartCompo\_rGi$ (i.e. for $rGpattern^{\geq}$) and $SmartCompo\_rGd$ (i.e. for $rGpattern^{\leq}$). Additionally, to illustrate the difference between MDL-based approaches and standard support-based approaches, we also employ the set of *top-*K highest support closed swarms and *top-*K highest covered area gradual trajectories patterns.

Figure 7 shows the compression gain of different algorithms. We can consider that *top-*K highest support or covered area patterns cannot provide good compression gain since they are very redundant. Furthermore, if we only consider one pattern type, we cannot compress the data best since the compression gains of SmartCompo_CS, SmartCompo_rGi and SmartCompo_rGd are always lower than SmartCompo. This is because the pattern distribution in the data is complex and different patterns can cover different parts of the data. Thus, considering one kind of patterns results in losing interesting patterns and not good compression gain. By proposing overlapping allowed multi-pattern structure encoding scheme, we are able to extract more informative patterns.



(a) Swainsoni dataset

(b) Buffalo dataset

Fig. 8. Running time

One of the most interesting phenomena is that the Swainsonies and Buffaloes have quite different movement behavior. See Figure 7a, we can consider that $rGpattern^{\geq}$ is the most representative movement behavior of Swainsonies since they compress the data better than the two other ones. While closed swarm is not as representative as the other patterns. This is because it is very easy for Swainsonies which are birds to leave the group and congregate again at later timestamps. However, this movement behavior is not really true for Buffaloes. See Figure 7b, it clear that the compression gains of closed swarms, $rGpattern^{\geq}$ and $rGpattern^{\leq}$ have changed. The three kinds of patterns have more similar compression gain than the ones in Swainsonies. It means that Buffaloes are more closed to each other and they move in a dense group. Thus closed swarm is more representative compare to itself in Swainsoni dataset. Furthermore, the number of Buffaloes is very difficult to increase in a group and thus SmartCompo_rGi is lower than the two other ones.

Running Time. In our best knowledge, there are no previous work which address mining compression movement pattern issue. Thus, we only compare the two proposed approaches in order to highlight the differences between them. Running time of each algorithm is measured by repeating the experiment in compression gain experiment.

As expected, SmartCompo is much faster than NaiveCompo (i.e. Figure 8). By exploiting the properties, we can directly select the best candidate at each iteration. Consequently, the process efficiency is speed up.

6 Related Work

Mining informative patterns can be classified into 3 main lines: MDL-based approaches, statistical approaches based on hypothesis tests and information theoretic approaches.

The idea of using data compression for data mining was first proposed by R. Cilibrasi et al. [1] for data clustering problem. This idea was also explored by Keogh et al. [8], who propose to use compressibility as a measure of distance between two sequences. In the second research line, the significance of patterns is tested by using a standard statistical hypothesis assuming that the data follows the null hypothesis. If a pattern pass the test it is considered significant and interesting. For instance, A. Gionis et al. [3] use swap randomization to generate random transactional data from the original data.

A similar method is proposed for graph data by R. Milo et al. [11]. Another research direction looks for interesting sets of patterns that compress the given data most (i.e. MDL principle). Examples of this direction include the Krimp algorithm [14] and Slim algorithm [12] for itemset data and the algorithms for sequence data [9].

7 Conclusion

We have explored an MDL-based strategy to compress moving object data in order to: 1) select informative patterns, 2) combine different kinds of movement patterns with overlapping allowed. We supplied two algorithms NaiveCompo and SmartCompo. The latter one exploits smart properties to speed up the whole process obtaining the same results to the naive one. Evaluations on real-life datasets show that the proposed approaches are able to compress data better than considering just one kind of patterns.

References

1. Cilibrasi, R., Vitányi, P.M.B.: Clustering by compression. IEEE Transactions on Information Theory 51(4), 1523–1545 (2005)
2. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp. 226–231 (1996)
3. Gionis, A., Mannila, H., Mielikäinen, T., Tsaparas, P.: Assessing data mining results via swap randomization. TKDD 1(3) (2007)
4. Grunwald, P.: The minimum description length principle. The MIT Press (2007)
5. Gudmundsson, J., van Kreveld, M.: Computing longest duration flocks in trajectory data. In: ACM GIS 2006 (2006)
6. Hai, P.N., Ienco, D., Poncelet, P., Teisseire, M.: Ming time relaxed gradual moving object clusters. In: ACM SIGSPATIAL GIS (2012)
7. Jeung, H., Yiu, M.L., Zhou, X., Jensen, C.S., Shen, H.T.: Discovery of convoys in trajectory databases. Proc. VLDB Endow. 1(1), 1068–1080 (2008)
8. Keogh, E.J., Lonardi, S., Ratanamahatana, C.A., Wei, L., Lee, S.H., Handley, J.: Compression-based data mining of sequential data. DMKD 14(1), 99–129 (2007)
9. Lam, H.T., Moerchen, F., Fradkin, D., Calders, T.: Mining compressing sequential patterns. In: SDM, pp. 319–330 (2012)
10. Li, Z., Ding, B., Han, J., Kays, R.: Swarm: mining relaxed temporal moving object clusters. Proc. VLDB Endow. 3(1-2), 723–734 (2010)
11. Milo, R., Shen-Orr, S., Itzkovits, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: Simple building blocks of complex networks. Science 298(5594) (2002)
12. Smets, K., Vreeken, J.: Slim: Directly mining descriptive patterns. In: SDM (2012)
13. Tang, L.A., Zheng, Y., Yuan, J., Han, J., Leung, A., Hung, C.-C., Peng, W.-C.: On discovery of traveling companions from streaming trajectories. In: ICDE, pp. 186–197 (2012)
14. Vreeken, J., Leeuwen, M., Siebes, A.: Krimp: Mining itemsets that compress. Data Min. Knowl. Discov. 23(1), 169–214 (2011)
15. Wang, Y., Lim, E.P., Hwang, S.Y.: Efficient mining of group patterns from user movement data. Data Knowl. Eng. 57(3), 240–282 (2006)
16. Zheng, K., Zheng, Y., Yuan, J., Shang, S.: On discovery of gathering patterns from trajectories. In: ICDE (2013)

NARGES: Prediction Model for Informed Routing in a Communications Network

Hooman Homayounfard[1], Paul J. Kennedy[1], and Robin Braun[2]

[1] Centre for Quantum Computation and Intelligent Systems, School of Software, Faculty of Engineering and Information Technology, University of Technology, Sydney. PO Box 123 Broadway NSW 2007, Australia

[2] Centre for Real-Time Information Networks, Faculty of Engineering and Information Technology, University of Technology, Sydney. PO Box 123 Broadway NSW 2007, Australia

{Hooman.Homayounfard,Paul.Kennedy,Robin.Braun}@uts.edu.au

Abstract. There is a dependency between packet-loss and the delay and jitter time-series derived from a telecommunication link. Multimedia applications such as Voice over IP (VoIP) are sensitive to loss and packet recovery is not a merely efficient solution with the increasing number of Internet users. Predicting packet-loss from network dynamics of past transmissions is crucial to inform the next generation of routers in making smart decisions. This paper proposes a hybrid data mining model for routing management in a communications network, called NARGES. The proposed model is designed and implemented for predicting packet-loss based on the forecasted delays and jitters. The model consists of two parts: a historical symbolic time-series approximation module, called HDAX, and a Multilayer Perceptron (MLP). It is validated with heterogeneous quality of service (QoS) datasets, namely delay, jitter and packet-loss time-series. The results show improved precision and quality of prediction compared to autoregressive moving average, ARMA.

Keywords: Time Series Data Mining, Communications Network, Packet-Loss Prediction, Time Series Approximation, Heterogeneous Data Sources.

1 Introduction

Rapid increases in the number of Internet users and services have prompted researchers within academia and industry to contemplate smart ways of supporting applications with the required Quality of Service (QoS). Service availability is a crucial part of QoS and the network infrastructure must be designed so as to provide high availability to meet QoS. The target of 99.999% availability permits five minutes of downtime per year [1].

There are certain QoS parameters including packet delay, jitter and loss, which may be used as decision factors for online routing management. Although considerable efforts have been placed on the Internet to assure QoS within autonomous

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 327–338, 2013.

systems, the dominant best-effort communications policy does not provide sufficient guarantee without abrupt change in the protocols [2] for the Internet. Estimation and forecasting of the end-to-end delay, jitter and packet-loss are essential tasks in network routing management for detecting anomalies.

A considerable amount of research has been done to forecast time-series with "numerical" methods. However, while dealing with large online datasets, these methods are time consuming and may not be efficient for real-time application such as multimedia streaming. Zadeh [3] suggests a transition from "computing with numbers" to the manipulation of the "human perceptions." Consequently, research projects [4,5] have started focusing on approximation of time-series with non-numerical methods. In this way, the time cost for trivial forecasting accuracy in the numerical methods may be avoided.

By prioritising Internet traffic more efficiently, QoS functions can address performance issues related to emerging Internet applications such as real–time voice and video streaming. An effective routing mechanism and its management are crucial to satisfactorily support diverse services in such networks. Routing tables, as the maps in packet delivery throughout the network, are dynamic and get updated by network state–based events. Typical network events include node failure, link failure and congestion. However, a major issue with current routing mechanisms is that they generate routing tables that do not reflect the real–time state of the network and ignore factors like local congestion.

Packet-loss in the Internet mainly occurs due to congestion in the links [6]. Real–time multimedia applications are sensitive to packet-loss, and packet retransmission is not an acceptable solution with these sorts of application. Predicting packet-loss with a certainty from network dynamics of past transmissions is crucial knowledge to inform the next generation of smart routers with better decision factors. We propose a data mining model for classification of links that have a high probability of packet-loss. The model is originally intended to contribute to making informed decisions within smart edge routers where the quality of transmissions should be controlled and is primarily determined by the level of packet-loss.

The current work extends our previous work [7]. The delay and jitter provided by a historical symbolic delay approximation model, called HDAX, is used within the proposed data mining model to predict the average number of packets lost in a link. The experiments with HDAX show better accuracy in forecasting the delay and jitter time–series as well as a reduction in the time cost of the forecasting method. To make the forecasting faster, we changed the perception–based function to a deterministic mapping function to avoid the time-cost of fuzzification and defuzzification.

We propose an informed decision-making model for routing management in a communications network, called NARGES as shown in the Fig. 1. The basic idea of our proposed model is to predict packet-loss in a network node by approximating the trends and values of the delay according to observed patterns. As shown in Figure 1, the model estimates the current trend and value assigned to the node based on the most two recent trends and values. The approximated current values of delay and jitter are then fed into a multilayer perceptron for predicting the packet-loss.

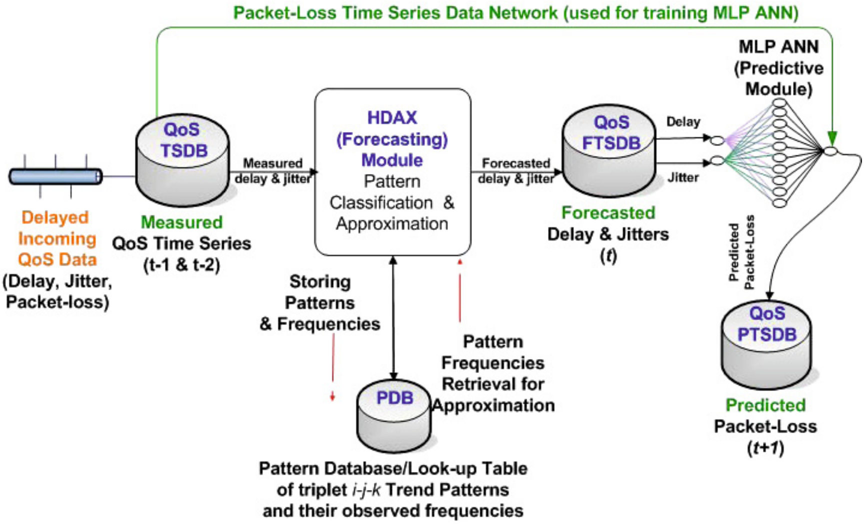

Packet-Loss Time Series Data Network (used for training MLP ANN)
MLP ANN (Predictive Module)
QoS TSDB
Measured delay & jitter
HDAX (Forecasting) Module
Pattern Classification & Approximation
Forecasted delay & jitter
QoS FTSDB
Delay
Jitter
Delayed Incoming QoS Data
(Delay, Jitter, Packet-loss)
Measured QoS Time Series (t-1 & t-2)
Forecasted Delay & Jitters (t)
Predicted Packet-Loss
Storing Patterns & Frequencies
Pattern Frequencies Retrieval for Approximation
PDB
QoS PTSDB
Predicted Packet-Loss (t+1)
Pattern Database/Look-up Table of triplet i-j-k Trend Patterns and their observed frequencies

Fig. 1. Schema of NARGES Prediction Model

To evaluate the proposed loss prediction model, we used heterogeneous data. The data is categorised into ten categories in terms of the end-to-end network path and the network queuing policy. Accordingly, we ran a series of independent experiments with these different categories of datasets. The accuracy of the results, the speed of the algorithms and the cross-correlation of the forecasts of HDAX and predictions of the proposed data mining model are compared to the results from autoregressive moving average (ARMA) model.

The rest of this paper is organised as follows. Section 2 describes related work in predicting performance traces. In section 3 we describe the forecasting and predictive module of the proposed model. In section 4 the evaluation procedure is explained and later in section 5 we present results of our experiments on applying our model. Section 6 concludes the paper.

2 Related Work

This research addresses the issue of defining a prediction model based on a symbolic time-series forecasting model. The model uses historical frequencies of observed patterns in adjacent time-stamped windows within a time–series.

Originally, the research project defined to produce an "informed" [8] data mining model to be used in a smart network router for online routing management. We used a Hybrid model framework as suggested in [9]. In [10] and [11], they present ideas for the definition of a perception based function.

Packet delay and jitter show a temporal dependency with the packet-loss in the Internet [12,13]. Consequently, the research projects have studied the delay, jitter, packet-loss and other performance traces in the network so as to predict network anomalies. If packet n is lost, packet $n + 1$ is likely to be lost. This can lead the network to a "bursty" packet-loss in a real–time communications network and may degrade the QoS and the effectiveness of recovery mechanisms.

A quantitative study of delay and loss correlation patterns from off-line analysis of measurement data from the Internet has been done by [14], although it did not consider real–time prediction of packet-loss from the jitter data of online communications [6].

The Rocha-Mier et al. model [9] suggests the measurement and modelling of sequences of network variables based on data network statistics. They have created a useful network scenario using OPNET Modeller. Although real network data variables could be derived from the data logs by the use of intelligent agents or manually by system administrators, there may be violation in accessing data throughout the Internet. Therefore, they adopted the modeller to study various levels of the network traffic load as well as types of services and applications.

The motivation of our work is to take a perception–based approach inspired by [10] embedded in a machine learning framework to predict network variables similarly to [9]. We validate our work using a combination of historical network traffic data and simulated data. Specifically our experiments used traces from network traffic archives generated by Napoli University "Federico II" [15] to test the impact of various network congestion levels, from "quiet" to a network with "bursty" packet-loss, on the proposed model. Simulated offline traces generated by OPNET Modeller, were used to test the impact of different queuing policies on the proposed model in longer experiments. This is a standard approach in the networking domain.

3 Proposed Algorithm

The NARGES model, as presented in Figure 1, is a hybrid model that predicts the packet-loss at time $t + 1$ based on the approximated values of delay and jitter at time t. The delay and jitter forecasts at time t, are approximations of the current values of the time-series forecasted by HDAX model according to the historical trends and values of the corresponding time-series in the previous "two" periods, i.e. $t - 1$ and $t - 2$. The following sections describe the two modules: HDAX and a multilayer perceptron, as the two constitutive parts of NARGES model.

3.1 Forecasting Module: HDAX

In this section, we briefly describe our approach to forecasting time-series values from previously observed patterns of delay and jitter. We use a mapping function for the definition of the patterns for time-series approximation, which is different from what we presented in our previous work [7].

The HDAX algorithm is defined based on the model definition suggested by Tresp [16].

$$y_t = f(y_{t-1}, y_{t-2}, \dots, y_{t-N}) + \epsilon_t \tag{1}$$

where f is either known or approximated sufficiently well by a function approximator and ϵ_t is the zero-mean noise with probability density $P_\epsilon(\epsilon)$, which represents dynamics that are not modelled. Let y_t be the value of the discrete time-series at time t and Y_t the trend of two consecutive values at times $t-1$ and t. We make the underlying HDAX model of the time series with order $N = 3$ as

$$y_t = f(y_{t-1}, y_{t-2}, y_{t-3}) + \epsilon_t \tag{2}$$

We represent possible future trends of the QoS time series y_t of delay and jitter values at time intervals $t - 1$ and $t - 2$ with categorical terms from the

$$Alphabet = \langle SI, I, P, D, SD, OUT \rangle \tag{3}$$

where these symbols are defined in Table 1. The basic trends in Table 1 are defined with linguistic variables based on a deterministic "mapping function." Within the mapping function, each of the categorical terms maps an interval on the domain of real numbers to a linguistic representative.

Table 1. The scale of time-series trends used for mapping numerical values to the trends. Note that we use y_t to denote the time series value at time t and Y_t to denote the difference of the two consecutive values at time $t-1$ and t. For simplicity, the linear scale in our experiment also has six linguistic grades (defined in Eq 3) each of which is a categorical term (assigned to the case number of zero to five respectively).

Case	Id	Description	$Y_t = y_t - y_{t-1}$
0	P	Plain	$Y_t = 0$
1	I	Increase	$0 < Y_t \leq \frac{max}{2}$
2	SI	Sharply Increase	$\frac{max}{2} < Y_t \leq max$
3	D	Decrease	$-\frac{max}{2} \leq Y_t < 0$
4	SD	Sharply Decrease	-$max \leq Y_t < -\frac{max}{2}$
5	OUT	Outlier	$\lvert Y_t \rvert > max$

We define "*previous-current-next*" patterns (also called the $i-j-k$ patterns) with a combination of three consecutive trends, at times $t-2$, $t-1$ and t as shown in Figure 2. Our approach consists of two phases: training and simulation. The max in Table 1 is the maximum value of a time-series, shown later with the dashed line in Fig. 2.

HDAX Training. Figure 2 shows a "sliding window" that moves over the training data and makes patterns consisting of three consecutive trends, i, j and k, at times $t-2$, $t-1$ and t (previous, current and next) together with

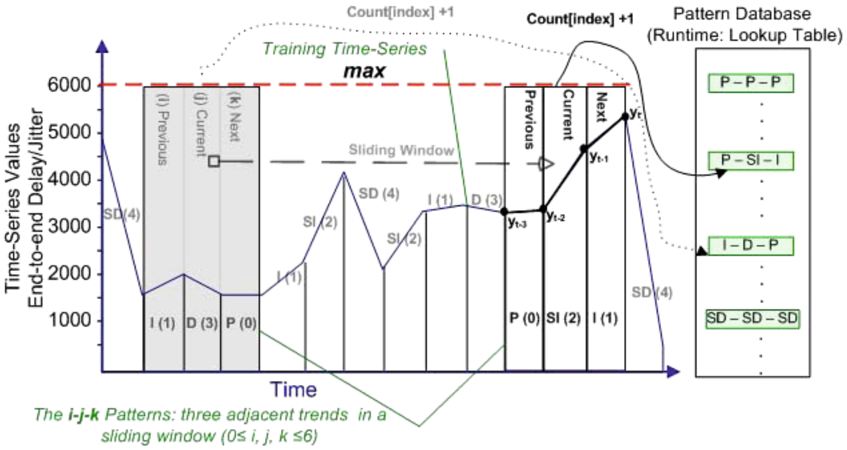


Fig. 2. The training phase uses a time series dataset values to recognise $i{-}j{-}k$ patterns and train the look–up table that maps each of these patterns to a respective frequency. The table is then used for forecasting the k trend at time $t+1$ in the simulation phase.

their frequency in the look–up table. The look–up table is then used in the simulation phase to approximate the next trend, at time t, and the associated delay/jitter value from the current and previously observed trend patterns, where i, j and k are the respective indices for previous, current and next trends ($0 \leq i, j, k \leq 5$) and $F(i, j, k)$ is their respective frequency. From the look–up table, the probability of $i - j - k$ patterns is estimated as

$$\bar{P}(i, j, k) = \frac{F(i, j, k)}{N_k}, \quad N_k = \sum_{k=0}^{5} F(i, j, k). \tag{4}$$

where i, j and k are the indices for the respective patterns at times $t-1$, $t-2$ and t, and N_k is the number of total observations for all $i - j - k$ patterns.

HDAX Simulation. Based on the most frequently observed patterns in the last two consecutive trends at time $t-1$ and $t-2$, the HDAX algorithm uses the estimated conditional probability to approximate the trend at time t. The Y_t is a trend value at time t in the time-series of trends, defined based on the last two trends seen at times $t-1$ and $t-2$. Formally speaking, we estimate the $P_k(i, j)$ as follows

$$\bar{P}_k(i, j) = \bar{P}(Y_t = k | Y_{t-1} = i, Y_{t-2} = j) + P_\epsilon(\epsilon) \tag{5}$$

where i and j are the indices of the observed trends at times $t-1$ and $t-2$, respectively. The trend at time t, with the index k, may take six possible values from the alphabets in Eq. 3. Based on this, the look-up table is used to forecast the next trend and value of the time-series based on the trend with highest frequency in this table:

$$i = t - 1, j = t - 2 : \hat{Y_t} = \arg\max_k(\bar{P}_k(i,j)) \tag{6}$$

With the trend k is estimated, the $\widehat{y_t}$ can be approximated based on time step-size between y_{t-1} and y_t and the slope of the line at y_{t-1}.

3.2 Predictive Module: Multi-layer Perceptron

The predictive module calculates the average packet-loss. As described above, the outputs of HDAX are approximations of the values of the network traces at time t. They are used within a multilayer perceptron (MLP) to get better precision for real-time packet-loss prediction at time $t + 1$.

The MLP is a feed-forward network with back-propagation learning rule and one hidden layer. As shown in Fig. 1, the MLP has two input layer nodes and forecasts delay and jitter. It is designed and implemented using the MATLAB neural network toolbox.

Optimum parameter values for number of hidden layer neurons, learning rate and momentum was defined empirically by using a training data-set and choosing the parameter values with the highest test accuracy. The network was tested with between 1 and 100 neurons in the hidden layer. Based on these experiments, ten hidden layer neuron were chosen.

4 Experimental Evaluation

The section describes experiments for evaluating the accuracy, speed and quality of the output of our model and ARMA. We implemented ARMA algorithms based on [17]. The results of HDAX and NARGES are compared with ARMA results. In the experiments, the accuracy of algorithms was compared using normalised root mean square error (NRMSE). Performance was compared using the run time. Prediction quality was evaluated with the normalised correlation coefficient with MATLAB's cross-correlation function.

4.1 Datasets

As mentioned above, the data consists of three QoS traces of delay, jitter and packet-loss. For each experiment three sets of data are considered: the original data generated by D-ITG or OPNET, the output of HDAX (or NARGES) and the output of ARMA. Each dataset is divided into training and test datasets (25% and 75% respectively).

The results were categorised according to datasets used for the experiments in two ways: (i) according to the end-to-end path and (ii) according to the queuing policy used. Forty-six datasets were used for computing results. Each dataset includes time-stamped traces of delay, jitter and loss values. There are

Table 2. Characteristics of the end-to-end paths for the data obtained from D-ITG (Dataset Categories: 1-7) and queuing policies for the data obtained from OPNET Modeler (Dataset Categories: 8-10)

Dataset Category	Access Networks (e2eP)	Operating Systems	User Device
1	ADSL-to-Ethernet	Linux-to-Linux	PC-to-Workstation
2	GPRS-to-Ethernet	Windows XP-to-Linux	Laptop-to-Workstation
3	UMTS-to-Ethernet	Windows XP-to-Linux	Laptop-to-Workstation
4	Ethernet-to-ADSL	Linux-to-Linux	Workstation-to-PC
5	Ethernet-to-GPRS	Linux-to-Windows XP	Workstation-to-Laptop
6	Ethernet-to-UMTS	Windows XP-to-Windows XP	Workstation-to-Laptop
7	Ethernet-to-WLAN	Linux-to-Windows XP	Workstation-to-Laptop
	Queuing Policy	Description	QoS Enabled
8	WFQ	Weighted Fair Queuing	Yes
9	FIFO	First in First out	No
10	PQ	Priority Queuing	Yes

36 datasets from D-ITG with an average 3000 values in each of delay, jitter and packet-loss time-series. There are also 10 datasets generated by OPNET, 9 with 48000 values and one with 1,000,000 values.

The data generated by D-ITG are gathered in two ways: an archive from the University of Napoli [15] and data probed over a University network test-bed. Samples were obtained by sending probe packets with a packet rate of 100 packets per seconds. The measurement unit of the delay and jitter is either milliseconds or nanoseconds while for packet-loss the value represents the average number of lost packets in the window of the times. The network test-bed was formed by two nodes on the University LAN: a laptop with 1.70 GHz processor, IntelPro 2200BG network connection and a node on a HPC Linux Cluster running Red Hat Enterprise Linux 6 (64bit) with processor rating between 3.06-3.46GHz and a gigabit inter-node connection. An end-to-end path (e2eP) definition is considered for the datasets obtained from D-ITG as shown in Table 2.

We used OPNET datasets to study the effect of different packet transmission policies, longer experiments and network congestion states on the performance of the implemented data mining model. The selection of service discipline in the routers (FIFO, WFQ and PQ) can affect VoIP applications and link congestion. Thus, the performance of the prediction model is evaluated using these datasets. Simulations ran on a laptop with 1.70 GHz processor with IntelPro 2200BG network connection and OPNET Modeler 15.0.A.

5 Results

Three methods were used analysis: accuracy, speed and correlation analysis for the proposed models versus ARMA. Friedman's test [18] is used to rank algorithms and to test the hypothesis of the similarity of the algorithms based on Holm's test. The $p{-}value$ is used to reject or accept the above “null hypothesis.”

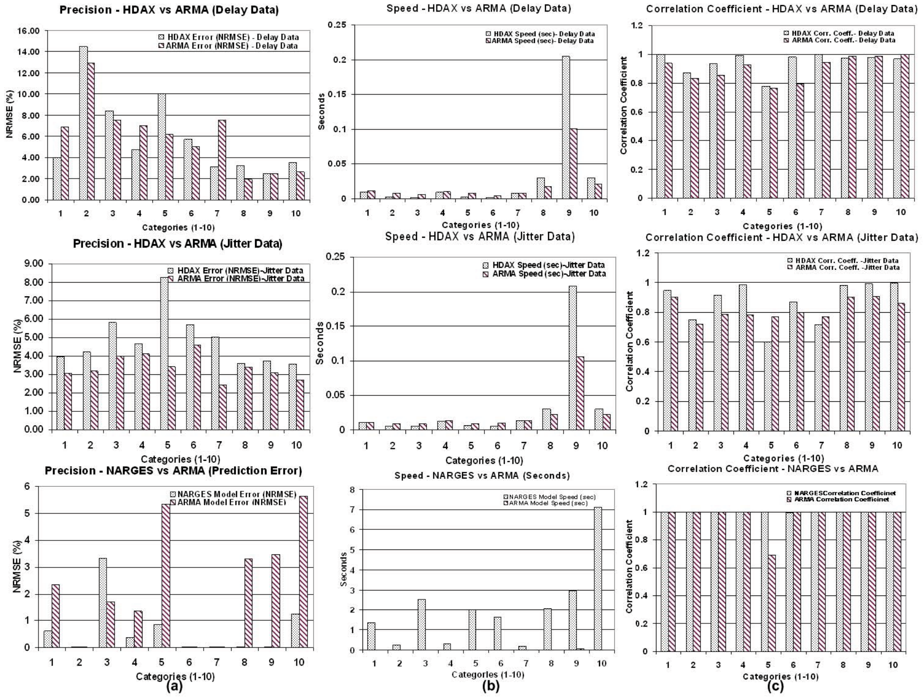


Fig. 3. Error (NRMSE) of HDAX and NARGES vs ARMA together with speed of algorithm and cross-correlation coefficients are shown in the column (a) to (c), respectively. The first and second rows are the HDAX results and the last row shows the whole model (NARGES) results. In the twin bar charts, the left "gray" bars shows HDAX (in the first two rows) and NARGES (in the last row) while the right bar filled with wide downward diagonal pattern denotes ARMA outcomes.

5.1 Model Performance

The accuracy and speed of NARGES model and its HDAX subcomponent was examined by comparing the NRMSE between the model outputs and the original traces. Figure 3, shows the results of HDAX for delay and jitter traces as well as the NARGES results over packet-loss traces in a top down order.

The average precision of the forecasted delay and jitter time-series of HDAX and ARMA as well as the NARGES precision for predicted packet-loss are shown within column (a) in Figure 3. Column (b) shows a comparison between the speed of the models with ARMA. In terms of similarity measurement between the original datasets and the output by HDAX, NARGES and ARMA, the maximum similarity of normalised cross-correlations was used for correlation analysis between the time-series. Column (c) in Figure 3 shows the respective correlation coefficient between the output of the models and the original data.

The predicted average packet-loss for NARGES was compared to ARMA. As the Fig. 3 shows, NARGES predicts more precisely than ARMA with OPNET

Table 3. Holm / Hochberg Table for $\alpha = 0.05$ and $z = (R_0 - R_i)/SE$. Note that in testing the algorithms for accuracy (ACC), speed (SPD) and cross-correlation (CCF) over Delay (D), Jitter (J) and Packet-Loss (L) models printed in **bold** are statistically significantly better.

	D-ITG Datasets			OPNET Datasets		
Data/Test	algorithm	z	p	algorithm	z	p
D/ACC	**HDAX**	3.0000	0.0027	**HDAX**	3.1628	0.0016
D/SPD	**HDAX**	1.9999	0.0455	**ARMA**	3.1623	0.0016
D/CCF	**ARMA**	5.3333	9.6×10^{-8}	**HDAX**	2.5298	0.0114
J/ACC	HDAX	1.0000	0.3173	HDAX	1.4×10^{-15}	0.9999
J/SPD	HDAX	1.3333	0.1824	**HDAX**	3.1623	0.0016
J/CCF	ARMA	0.9999	0.3173	**ARMA**	3.1623	0.0016
L/ACC	NARGES	0.4999	0.6171	**NARGES**	3.1623	0.0015
L/SPD	**ARMA**	6.0000	2.0×10^{-9}	**ARMA**	3.1623	0.0016
L/CCF	NARGES	2.7×10^{-15}	0.9999	**NARGES**	3.1623	0.0016

datasets but is slower. This is because NARGES has more modules and processes more data than ARMA to predict the final packet-loss values. The training time of the MLP module accounts for the longer time taken to run our model.

5.2 Model Ranking

Friedman test was run and the stored statistic used to calculate the Holm's test $p-value$, which is a decision factor for rejecting or accepting the null hypothesis. It also calculates the average ranking of the algorithms used in each test.

Friedman's test is a non-parametric equivalent to the parametric repeated measures ANOVA test. It computes the ranking of the measured outputs for an algorithm with other algorithms, assigning the best of them the ranking 1 and the worst the ranking k. According to the null hypothesis, it is supposed that the results of the algorithms are equivalent and the rankings are also similar.

A similarity test between the precision, performance (speed) and the correlation of the output of the models with the original test data is performed via nonparametric Holm tests. The tests were conducted over the results collected from the runs of HDAX, ARMA and NARGES models with the three traces of delay, jitter and loss within each datasets. We have done this to rank the algorithms and test the similarity hypothesis between HDAX and NARGES models and ARMA. The Holm tests are testing the similarity of algorithms accuracy, speed and correlation coefficient for delay, jitter and packet-loss time-series.

In Table 3, the algorithm shown in each row works significantly better than ARMA if their corresponding average ranks differ by at least the critical difference, which are the ones with $p-values \leq 0.05$. In Table 3, the algorithm name showed in each row are taken as the better when the null hypothesis is rejected.

The Holm's tests shows that HDAX ranking is better than ARMA for the precision of the results and the speed of the algorithms whereas ARMA ranking is better than HDAX for cross-correlation between the forecasted and original time-series. Moreover, according to the $p - values$, HDAX forecasts significantly better and faster than ARMA for delay traces while ARMA has more correlated outputs in comparison to the original delay data in the 36 runs for the D-ITG datasets in section 4.1. Two algorithms, HDAX and ARMA, perform the same in forecasting jitter values because Holm's tests accept all null hypothesis. For the whole model outputs accuracy and quality of the predictions, NARGES predicts the packet-loss the same as ARMA does, although NARGES ranks higher. In terms of the speed of the models, the $p - value$ of the ARMA speed is less than the significance level ($\alpha = 0.05$).

The Holm's test for OPNET datasets, explained in section 4.1, shows that HDAX forecasts significantly more accurate than ARMA over the delay datasets and has a better quality in terms of the cross-correlation between its forecasts and the original data. HDAX is faster over jitter datasets while ARMA forecasts faster over delay datasets. Unlike the tests with D-ITG datasets, the NARGES outputs with OPNET datasets show significantly better precision and quality of predictions. It means that our model shows better precision and prediction compared to ARMA in longer experiments. In terms of the speed of the models, the $p - value$ of the ARMA speed is less than the significance level ($\alpha = 0.05$).

Currently network routers must send information between routers to inform about the peer status. The results in this paper demonstrate, in a simulated setting at least, that a data mining agent can predict the peer status to reduce or perhaps eliminate the unnecessary network data transmission overhead and the time required for sending and receiving repeated packets.

6 Conclusions

This paper presented a packet-loss prediction model based on our earlier work [7]. We designed and implemented a hybrid data mining model, NARGES, which is using the forecasted current values of delay and jitter, to predict the future packet-loss rate assigned to a network node. We used a non-numerical approach to predict packet-loss in multimedia streams by observing the delay and jitter time-series. The Model is validated with heterogeneous QoS traces and the results show that the quality and the precision of the proposed model is significantly better than AMRA. However, NARGES was slower than ARMA because it has to process more inputs. As can be seen from the competing speed of HDAX module, it is the training time of the MLP module that degrades the speed of the model. In Table 3, the significant difference between the $p - value$ of the Holm's tests for the D–ITG and the longer OPNET datasets implies that our model can work faster in a real–time network experiment.

References

1. Torell, W.: Network-critical physical infrastructure: Optimizing business value. In: Twenty-Seventh International Telecommunications Conference, INTELEC 2005, pp. 119–124 (September 2005)

2. Floyd, S., Allman, M.: Comments on the usefulness of simple best-effort traffic. Request for Comments 5290, IETF (July 2008)
3. Zadeh, L.A.: From computing with numbers to computing with words from manipulation of measurements to manipulation of perceptions. Annals of the New York Academy of Sciences 929(1), 221–252 (2001)
4. Keogh, E., Lin, J., Fu, A.: Hot SAX: Efficiently finding the most unusual time series subsequence. In: ICDM 2005, Houston, pp. 27–30. IEEE (2005)
5. Batyrshin, I.Z., Sheremetov, L.B.: Perception–based approach to time series data mining. Applied Soft Computing Journal 8(3), 1211–1221 (2008)
6. Roychoudhuri, L., Al-Shaer, E.: Real–time packet loss prediction based on end–to–end delay variation. IEEE Trans. Network Service Manager 2(1) (2005)
7. Homayounfard, H., Kennedy, P.J.: HDAX: Historical symbolic modelling of delay time series in a communications network. In: Kennedy, P.J., Ong, K., Christen, P. (eds.) AusDM 2009. CRPIT, vol. 101, pp. 129–138. ACS, Melbourne (2009)
8. Debenham, J., Simoff, S., Leaney, J., Mirchandani, V.: Smart communications network management through a synthesis of distributed intelligence and information. In: Artificial Intelligence in Theory and Practice II, pp. 415–419 (2008)
9. Rocha-Mier, L.E., Sheremetov, L., Batyrshin, I.: Intelligent agents for real time data mining in telecommunications networks. In: Gorodetsky, V., Zhang, C., Skormin, V.A., Cao, L. (eds.) AIS-ADM 2007. LNCS (LNAI), vol. 4476, pp. 138–152. Springer, Heidelberg (2007)
10. Miloucheva, I., Hofmann, U., Gutiérrez, P.A.A.: Spatio-temporal QoS pattern analysis in large scale internet environment. In: Ventre, G., Canonico, R. (eds.) MIPS 2003. LNCS, vol. 2899, pp. 282–293. Springer, Heidelberg (2003)
11. Batyrshin, I., Panova, A.: On granular description of dependencies. In: Proc. 9th Zittau Fuzzy Colloquium, Zittau, Germany, pp. 1–8 (2001)
12. Jiang, W., Schulzrinne, H.: Modeling of packet loss and delay and their effects on real-time multimedia service quality. ACM Network and Operating Systems Support for Digital Audio and Video (2000)
13. Markopoulou, A., Tobagi, F., Karam, M.: Loss and delay measurements of internet backbones. Computer Communications 29(10), 1590–1604 (2006)
14. Moon, S., Kurose, J., Towsley, D.: Packet audio playout delay adjustment: performance bounds and algorithms. Multimedia Systems 6(1), 17–28 (1998)
15. Botta, A., Pescapé, A., Ventre, G.: Quality of service statistics over heterogeneous networks: Analysis and applications. European Journal of Operational Research 191(3), 1075–1088 (2008)
16. Tresp, V., Hofmann, R.: Nonlinear time-series prediction with missing and noisy data. Neural Computation 10(3), 731–747 (1998)
17. Chatfield, C.: Time-series forecasting. Chapman and Hall (2001)
18. García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences 180(10), 2044–2064 (2010)

Mining Usage Traces of Mobile Apps for Dynamic Preference Prediction

Zhung-Xun Liao[1], Wen-Chih Peng[1], and Philip S. Yu[2]

[1] Department of Computer Science and Information Engineering,
National Chiao Tung University, HsinChu, Taiwan, ROC
[2] Department of Computer Science,
University of Illinois at Chicago, Chicago, IL, USA

Abstract. Due to a huge amount of mobile applications (abbreviated as Apps), for Apps providers, the usage preferences of Apps are important in recommending Apps, downloading Apps and promoting Apps. We predict and quantize users' dynamic preferences by exploring their usage traces of Apps. To address the dynamic preference prediction problem, we propose Mode-based Prediction (abbreviated as MBP) and Reference-based Prediction (abbreviated as RBP) algorithms. Both MBP and RBP consist of two phases: the trend detection phase and the change estimation phase. In the trend detection phase, both algorithms determine whether the preference of an App is increasing or decreasing. Then, in the change estimation phase, the amount of preference change is calculated. In particular, MBP adopts users' current usage mode (active or inactive), and then estimates the amount of change via our proposed utility model. On the other hand, RBP calculates an expected number of usage as a reference, and then builds a probabilistic model to estimate the change of preference by comparing the real usage and the reference. We conduct comprehensive experiments using two App usage traces and one music listening log, the Last.fm dataset, to validate our proposed algorithms. The experimental results show that both MBP and RBP outperform the usage-based method that is based solely on the number of usages.

Keywords: Dynamic Preference Prediction, Mobile Applications, Apps.

1 Introduction

As mobile devices become more and more popular, a tremendous amount of mobile applications (abbreviated as Apps) are designed for varied functions and purposes. Users can download and execute Apps in their mobile devices to satisfy their needs and affinities. For App providers, to understand users' preferences is quite important to recommend new Apps, and to decide their marketing strategies for selling Apps [1–3]. Although users can rate the Apps they have experienced, only a small percentage of users rate their Apps. For example, the famous App, Angry Birds, only received 4% ratings from 1.3 million of downloads [4]. Besides, users may not be willing to consistently rate the Apps when they change their preferences. On the other hand, although [5] states an Apps

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 339–353, 2013.

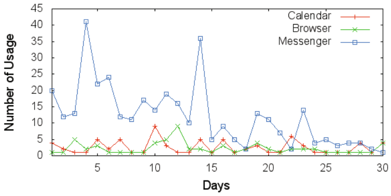


Fig. 1. The number of usages of different mobile applications

recommendation problem, it does not show the dynamic preferences of users. By contrast, through the dynamic preferences, we can not only recommend Apps but investigate more tasks on Apps.

In this paper, we aim to predict users' dynamic preferences of each App and further quantize the preferences to real numbers such that we can compare the preferences among different users. As users repeatedly invoke these Apps, their preferences are dynamic over time based on what they have experienced. Here, we claim that a user's dynamic preference is related to the usage trace (i.e., series of usage counts). For example, Fig. 1 shows three usage traces of Calender, Browser, and Messenger, for a certain user. As can be seen in Fig. 1, the number of usages on "Messenger" apparently drops down after 14 days (two weeks). Therefore, we can infer that the user decline his/her preference on "Messenger" in 14 days by either implicit or explicit reasons.

Nevertheless, usage counts of Apps are not directly related to the preferences of Apps. For example, in Fig. 1, although the usage count of Messenger is higher than the other two Apps, the preference of Messenger is not necessarily higher than the other two Apps. Probably, Messenger is a communication tool, which is designed to be used frequently. As for Calendar, users will not frequently check their calendar all the time. In our experimental results, we also show that the usage-based algorithm cannot predict anything, where its accuracy is often close to zero. To correctly predict preferences of Apps from the usage traces, we propose two methods, Mode-based Prediction (abbreviated as MBP) and Reference-based Prediction (abbreviated as RBP). Both methods utilize different strategies to avoid the impact of the inherent magnitude bias of the the usage counts. MBP adopts only the usage mode of Apps: active mode for using the App while inactive mode for not using the App. RBP refers to the previous usage counts of each App as a reference history and thus, the usage count becomes a relative value of the reference history.

Both MBP and RBP consist of two phases: the trend detection phase and the change estimation phase. The first phase determines whether the preference is decreasing or increasing. The second phase estimates the absolute value on the preference change. For MBP, we increase the preferences of those Apps which are used at current time unit, but decreases the preferences of others. Then, we propose a utility model: when a user uses more Apps at the same time unit, each App would receive less preference increment. According to the utility model, we can calculate the increment and decrement of each App. For RBP, it calculates an expected number of usage for each App at current time unit by solving an optimization problem where the expected number of usage can keep the trend of preference change staying static. If the actual number of usage is larger (smaller, respectively) than the expectation, the preference will increase (decrease, respectively). Then, RBP uses a probabilistic model to estimate the change of preferences.

The contributions of this study are:

1. We explore usage traces of Apps for dynamically predicting the perferences of Apps.
2. We analyze the characteristics of Apps, and propose two algorithms, MBP and RBP, to predict preferences of Apps.
3. In the MBP method, we derive the dynamic preferences according to only the usage mode and propose a utility model to calculate the change of users' preferences.
4. In the RBP method, by solving an optimization problem, the expected number of usage is derived as a reference, and a probabilistic model is constructed to estimate the users' preferences.
5. We conduct a comprehensive performance evaluation. The experimental results show that the predicted dynamic preferences of both MBP and RBP can better reflect users' behavior

2 Related Work

To the best of our knowledge, this paper is the first work discussing dynamic preferences prediction problem. Although there are many research works discussing the problem of predicting users' preference, they only focused on a static environment. In a static circumstance, such as renting movies and purchasing books, users generally only act on them once and the preference remain static. Therefore, they can use the existing user preferences to predict the unknown preferences through the attributes of items [6]. The attributes could be the metadata, such as artist, genre, etc., or the ratings the item already had. Although [6] focused on predicting the ratings of musics, they still treated the music ratings as static preferences. This is because their focus is on purchasing songs or CDs, not on the preference to listening to a song from a user collection at a particular moment. Only the authors in [7–9] recognized the temporal dynamics of users' preferences. Nevertheless, [7] still need to obtain at least a portion of static ratings as training data. [8, 9] only consider the evolution of users' behavior, instead of quantize their preferences. For predicting preferences of Apps, users can use Apps repeatedly; therefore, their preference changes over time, and even be impacted by new Apps [10]. Consequently, the traditional preference prediction methods cannot be adopted for the dynamic preference problem, because 1) the traditional methods all need to obtain at least a portion of static preferences as training data, and 2) the static preferences are out-of-date when we perform the prediction in a dynamic environment.

3 Preliminary

In this section, we first describe the symbols used in this paper. Explicitly, we use r_{min} and r_{max} to represent the minimum and maximum value of users' preference. Thus, the preference at time unit t is a real number $r_{ij}^{(t)} \in [r_{min}, r_{max}]$, which represents the preference of user i on App j. To facilitate the presentation of this paper, U is the set of users and I is the set of Apps. A dynamic preference matrix is used to represent the preferences of Apps at a certain time unit. Here, we divide time space into time units, and use l, an application dependent parameter, to represent the length of a time unit. The formal definition is below:

Definition 1. ***(Dynamic Preference Matrix)*** *A dynamic preference matrix at time unit t, $R^{(t)}$, is a $|U| \times |I|$ matrix, where $r_{ij}^{(t)} \in [r_{min}, r_{max}]$, for each $r_{ij}^{(t)}$.*

A usage count matrix constructed from users' traces is defined in Definition 2

Definition 2. ***(Usage Count Matrix)*** *A usage count matrix at time unit t, $C^{(t)}$, is a $|U| \times |I|$ matrix, where each element $c_{ij}^{(t)}$ represents how many times user i used App j at time unit t.*

We use a change matrix to record the preference change of each user-App pair. When the value is positive (negative, respectively), the preference is increasing (decreasing, respectively). Definition 3 shows the detail of change matrix. In this paper, the change matrix is derived from both usage count matrix and dynamic preference matrix.

Definition 3. ***(Change Matrix)*** *A change matrix at time unit t, denoted as $\Delta^{(t)}$, is a $|U| \times |I|$ matrix, where the value of each element $\delta_{ij}^{(t)}$ is in either [0,$r_{max} - r_{ij}^{(t-1)}$] for positive value, or in [$r_{ij}^{(t-1)} - r_{min}$,0] for negative value.*

We claim that the preference of an App would not change dramatically. Even when users do not use an App for a long time, the preference of it would decay smoothly over time. Therefore, we derive users' preferences according to the previous preferences and current usage behavior as described in Definition 4.

Definition 4. ***(Dynamic Preferences Prediction Problem)*** *Let $R^{(t-1)}$ be the dynamic preference matrix at time unit $t-1$, and $C^{(t)}$ be the usage count matrix at time unit t, the dynamic preference prediction problem is 1) calculating the change matrix, $\Delta^{(t)}$, and 2) deriving $R^{(t)}$ according to Eq. 1.*

$$R^{(t)} = R^{(t-1)} + \Delta^{(t)} \tag{1}$$

For example, suppose we have two users and three Apps, and the system parameters are $r_{min} = 0$ and $r_{max} = 5$. Let $R^{(t-1)} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \end{bmatrix}$ be the dynamic preference matrix derived at time unit $t-1$, and $C^{(t)} = \begin{bmatrix} 100 & 2 & 30 \\ 2 & 300 & 40 \end{bmatrix}$ be the usage count matrix. First, we calculate the change matrix according to $C^{(t)}$ and $R^{(t-1)}$, such that, in this example, the values of the change matrix are related to the usage counts and will be in the defined range to avoid the values in $R^{(t)}$ being out of range. Assume that we can obtain $\mathit{\Delta}^{(t)} = \begin{bmatrix} 4 & -1 & 1 \\ -2 & 2 & 1 \end{bmatrix}$. Then, the new dynamic preference could be derived as $R^{(t)} = \begin{bmatrix} 5 & 1 & 4 \\ 0 & 5 & 5 \end{bmatrix}$.

4 Dynamic Preference Prediction

As described in Definition 4, to obtain the dynamic preference matrix, $R^{(t)}$, we need to know the change matrix, $\mathit{\Delta}^{(t)}$, in advance. Here, we use δ^t_{ij} to represent the elements in $\mathit{\Delta}^{(t)}$. Empirically, we can calculate δ^t_{ij} by Eq. 2 which consists of two parts: 1) $m \in \{0, 1\}$ which indicates whether δ^t_{ij} is positive ($m = 0$) or negative ($m = 1$), and 2) $v^{(t)}_{ij} > 0$ which is the absolute value of δ^t_{ij}. Through this equation, we can calculate the change matrix, $\mathit{\Delta}^{(t)}$, by finding a proper pair of m and v^t_{ij} for each $\delta^{(t)}_{ij}$.

$$\delta^{(t)}_{ij} = (-1)^m \times v^{(t)}_{ij} \tag{2}$$

In this paper, we design a two-phase framework: the trend detection phase for the value of m and the change estimation phase to calculate $v^{(t)}_{ij}$. In order to smooth the preference change, the value of $v^{(t)}_{ij}$ depends on not only the current usage count, $c^{(t)}_{ij}$ but the previous preference, $r^{(t-1)}_{ij}$. In addition, when the preference is increasing (respectively, decreasing), the value of $v^{(t)}_{ij}$ is in the range of $[0, r_{max} - r^{(t-1)}_{ij}]$ (respectively, $[0, r^{(t-1)}_{ij} - r_{min}]$). Thus, we can formulate $v^{(t)}_{ij}$ as in Eq. 3, where $u^{(t)}_{ij}$ is a utility parameter determined by user's preference change. Explicitly, when $u^{(t)}_{ij}$ is larger (i.e. user's preference change is large), $v^{(t)}_{ij}$ would be larger.

In order to address the challenge related to the number of usages of Apps, we propose two algorithms based on different points of view. The first one is Mode-based Prediction (MBP) which takes into account of the binary usage mode of active and inactive. The second one is called Reference-based Prediction (RBP) which adopts the previous usage counts as a reference history to examine the $\mathit{\Delta}^{(t)}$ matrix.

$$v^{(t)}_{ij} = \begin{cases} (r^{(t-1)}_{ij} - r_{min}) \times u^{(t)}_{ij} & , m = 1 \\ (r_{max} - r^{(t-1)}_{ij}) \times u^{(t)}_{ij} & , m = 0 \end{cases} \tag{3}$$

4.1 Mode-Based Prediction (MBP)

The Mode-based Prediction (MBP) ignores the magnitude of usage counts by only considering two usage mode: one is active mode for using the App and another is inactive mode for not using the App. Then, a utility model is proposed to measure the usage change of a user, and the $\Delta^{(t)}$ matrix could be estimated through this model.

The trend detection phase. In this phase, we decide the value of m in Eq. 2. If user i executed App j at time unit t (i.e. $c_{ij}^{(t)} > 0$), we would set m as 0 (increase the preference). By contrast, if $c_{ij}^{(t)} = 0$, we set m to 1 (decrease the preference).

$$\sum_{k \in P} u_{ik}^{(t)} - \sum_{k \in N} u_{ik}^{(t)} = 0 \tag{4}$$

$$u_{ij}^{(t)} = \begin{cases} \frac{1}{|P|} & , c_{ij}^{(t)} > 0 \\ \frac{1}{|N|} & , c_{ij}^{(t)} = 0 \end{cases} \tag{5}$$

$$v_{ij}^{(t)} = \begin{cases} \frac{r_{max} - r_{ij}^{(t-1)}}{|P|} & , c_{ij}^{(t)} > 0 \\ \frac{r_{ij}^{(t-1)} - r_{min}}{|N|} & , c_{ij}^{(t)} = 0 \end{cases} \tag{6}$$

The change estimation phase. The second phase is to estimate the absolute value of the preference change. In other words, we need to derive the value of $v_{ij}^{(t)}$ according to utility parameter, $u_{ij}^{(t)}$. Since we only have the information of usage mode of each App, we propose a utility model to derive the utility parameter based only on the usage mode. Intuitively, when a user spends more time on some Apps, (s)he should spend less time on others. Thus, we claim that the overall usage change among Apps should be equal to 0. Eq. 4 formulates the utility model, where P (respectively, N) is the set of Apps with active (respectively, inactive) mode. Suppose that the importance of each App is the same, the utility parameter is derived by Eq. 5. As a result, we can obtain $\delta_{ij}^{(t)}$ from $u_{ij}^{(t)}$, as shown in Eq. 6.

4.2 Reference-Based Prediction (RBP)

Although MBP successfully avoids the magnitude of usage counts by adopting the usage mode and the utility model, ignoring the magnitude of the usage counts makes the estimated preferences not be able to reflect users' actual preferences. For example, in Fig. 2, the preference of Messenger predicted by MBP becomes higher and higher over time, since MBP increases the user's preference once the user invokes the Apps. However, we believe that the curve representing the preference of Messenger should be like the Ideal one. To obtain the ideal result, we propose a Reference-based Prediction (RBP) algorithm which compares the usage counts within an App instead of with other Apps.

RBP uses the previous usage counts of each App as a reference history, and derives a reference value from the reference history. In this paper, the size of reference history is decided by a tunable parameter, h, which means how many

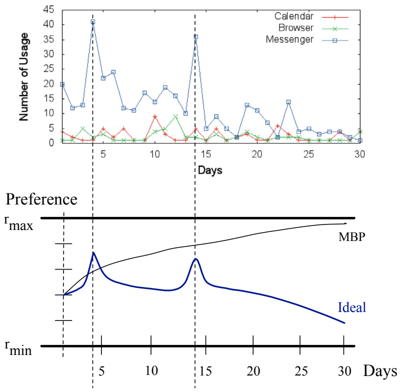


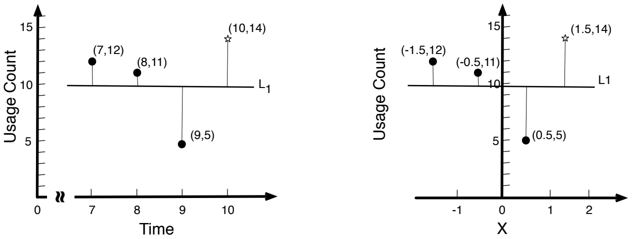
Fig. 2. The preferences derived by MBP comparing with the Ideal preferences

historical data points are included into the reference history. The concept is that only when the actual usage count of an App is higher than the reference value, its preference is increasing. Similarly, the preference decreases only when the number of usage is less than the reference value.

$$slope = \frac{(h+1)\sum_{k=t-h}^{t} k \times c_{ij}^{(k)} - \sum_k k \sum_k c_{ij}^{(k)}}{(h+1)\sum_k k^2 - (\sum_k k)^2} = 0 \tag{7}$$

$$(h+1)\sum_{k=t-h}^{t} k \times c_{ij}^{(k)} - \sum_k k \sum_k c_{ij}^{(k)} = 0 \tag{8}$$

The trend detection phase. In this phase, we decide whether the value of $\delta_{ij}^{(t)}$ is negative or positive. Here, we use the previous usage counts as the reference history and derive an expected number of usage as the reference value from the reference history. We adopt the linear regression to model the trend of reference history, and thus, the expected number of usage count should make the slope of the regression line be zero. Since the slope of a regression line represents the trend of the data points, the expected number of usage count which makes the regression line stay horizontal means that it makes the preference stay static. Then, if the actual number of usage is larger (smaller, respectively) than the expected number of usage, the preference is considered as increasing (decreasing, respectively). We use Fig. 3(a) to illustrate the concept of obtaining the expected number of usage by linear regression model. In Fig. 3(a), the three black points are the reference history (i.e. $h = 3$) and the reference value is the expected usage at time unit 10 (marked as a star point) which makes the regression line, L_1, be horizontal. Therefore, the goal of this phase is to find the value of star point by satisfying Eq. 7 which can be simplified into Eq. 8.



(a) Deriving the star point from the three black points. (b) Shifting the data points in Fig. 3(a).

Fig. 3. Estimate the expected usage count (marked as a star point)

Since we only consider the slope of the regression line, we can shift the regression line left such that $\sum_{k=1}^{h+1} x_k = 0$, where x_k represents the shifted position in x-axis of the k-th point of reference history and thus, x_{h+1} is the shifted x-axis of the star point. As shown in Fig. 3(b), we can shift the regression line to the positions of x-axis as $< -1.5, -0.5, 0.5, 1.5 >$. Eq. 9 shows how to calculate the shifted x-axis positions. Now, we can simplify Eq. 8 into Eq. 10, where the index of time units of $c_{ij}^{(k)}$ is also shifted to $(k + t - h - 1)$ for $k = 1, 2, \ldots, h + 1$.

$$x_k = \frac{2k - (h + 2)}{2} \tag{9}$$

$$(h + 1) \sum_{k=1}^{h+1} x_k \times c_{ij}^{(k+t-h-1)} = 0 \tag{10}$$

Therefore, we can extract $c_{ij}^{(t)}$ from Eq. 10, and it is the expected number of usage $EXP(c_{ij}^{(t)})$, which could be derived from Eq. 11. For example, the value of the star point in Fig. 3(a) is $EXP(c_{ij}^{(10)}) = [(3+2)(12+11+5) - 2(1 \times 12 + 2 \times 11 + 3 \times 5)]/3 = 42/3 = 14$.

$$\begin{aligned} EXP(c_{ij}^{(t)}) &= -\frac{\sum_{k=1}^{h} x_k \times c_{ij}^{(k+t-h-1)}}{x_{h+1}} \\ &= -\frac{\frac{2(1)-(h+2)}{2} c_{ij}^{(t-h)} + \ldots + \frac{2(h)-h-2}{2} c_{ij}^{(t-1)}}{\frac{2(h+1)-(h+2)}{2}} \\ &= \frac{(h+2) \sum_{k=1}^{h} c_{ij}^{(k+t-h-1)} - 2 \sum_{k} k \times c_{ij}^{(k+t-h-1)}}{h} \end{aligned} \tag{11}$$

The change estimation phase. As we have $EXP(c_{ij}^{(t)})$ to be the reference value, we need to formulate the utility parameter, $u_{ij}^{(t)}$, by calculating the distance between $c_{ij}^{(t)}$ and $EXP(c_{ij}^{(t)})$, denoted as $dist(EXP(c_{ij}^{(t)}), c_{ij}^{(t)})$. When $c_{ij}^{(t)}$ is far from $EXP(c_{ij}^{(t)})$, it means that the user is considered more likely to change his/her preference. Since we need a distance measure between 0 and 1, directly subtracting $EXP(c_{ij}^{(t)})$ from $c_{ij}^{(t)}$ or the other way around will not work. We devise the following distance measure. Here, $dist(EXP(c_{ij}^{(t)}), c_{ij}^{(t)})$ is estimated by evaluating how many possible cases are between $EXP(c_{ij}^{t})$ and $c_{ij}^{(t)}$. Therefore, when the preference is increasing ($m = 0$), we use $p(EXP(c_{ij}^{(t)}) \leq x \leq c_{ij}^{t})$ representing the probability of obtaining a number of usage in-between $EXP(c_{ij}^{(t)})$ and $c_{ij}^{(t)}$, where x is a random variable. On the other hand, when the preference is decreasing ($m = 1$), the distance between $EXP(c_{ij}^{(t)})$ and $c_{ij}^{(t)}$ is formulated as $p(c_{ij}^{(t)} \leq x \leq EXP(c_{ij}^{(t)}))$. In this paper, we approximate the probability, $p(c_{ij}^{(t)})$, of using an App j by $c_{ij}^{(t)}$ times in a given time duration l (a parameter for the length of each time unit) by assuming a Poisson distribution shown in Eq. 12, where $\lambda = EXP(c_{ij}^{(t)})$. Now, the utility parameter, $u_{ij}^{(t)}$, could be formulated as in Eq. 13 and the absolute amount of preference change, $v_{ij}^{(t)}$, as in Eq. 14. We also list algorithm 1 to describe the flow of RBP in detail. In the first iteration, we set $r_{ij}^{(t)}$ to an initial preference, r_{init}, which is a tunable parameter. The preference will stay the same when the actual number of usage equals to the expected number of usage.

$$p(c_{ij}^{(t)}) = \frac{\lambda^{c_{ij}^{(t)}} \times e^{-\lambda}}{c_{ij}^{(t)}!} \tag{12}$$

$$u_{ij}^{(t)} = dist(\lambda, c_{ij}^{(t)}) = \begin{cases} p(\lambda \leq x \leq c_{ij}^{(t)}) = \sum\limits_{k=\lambda}^{c_{ij}^{(t)}} p(k) \ , c_{ij}^{(t)} > \lambda \\ p(c_{ij}^{(t)} \leq x \leq \lambda) = \sum\limits_{k=c_{ij}^{(t)}}^{\lambda} p(k) \ , c_{ij}^{(t)} < \lambda \end{cases} \tag{13}$$

$$v_{ij}^{t} = \begin{cases} (r_{max} - r_{ij}^{(t-1)}) \times \sum\limits_{k=\lambda}^{c_{ij}^{(t)}} p(k) \ , c_{ij}^{(t)} > \lambda \\ (r_{ij}^{(t-1)} - r_{min}) \times \sum\limits_{k=c_{ij}^{(t)}}^{\lambda} p(k) \ , c_{ij}^{(t)} < \lambda \end{cases} \tag{14}$$

5 Experimental Results

To evaluate the accuracy of the derived dynamic preferences, we examine the accuracy by testing the performance of using those derived preferences to make

Algorithm 1. Algorithm of Reference-based Prediction

```
Input: Input: $R^{(t-1)}$, $C^{(t)}$
Output: Output: $R^{(t)}$
foreach $c_{ij}^{(t)}$ do
    Let $r_{ij}^{(t)} \leftarrow r_{init}$
end
foreach initialled $r_{ij}^{(t-1)}$ do
    $EXP(c_{ij}^{(t)}) \leftarrow \frac{(h+2)\sum_{k=1}^{h} C-2\sum_{k} k\times c^{(k)}}{h}$
    if $c_{ij}^{(t)} > EXP(c_{ij}^{(t)})$ then
        $m \leftarrow 0$ $P \leftarrow P \cup App_j$
    else
        $m \leftarrow 1$ $N \leftarrow N \cup App_j$
    end
    $\lambda \leftarrow EXP(c_{ij}^{(t)})$
    if $App_j \in P$ then
        $v_{ij}^{(t)} \leftarrow (r_{max} - r_{ij}^{(t-1)}) \times \sum_{k=c_{ij}^{(t)}}^{\lambda} \frac{\lambda^{c_{ij}^{(t)}} \times e^{-\lambda}}{c_{ij}^{(t)}!}$
    else
        $v_{ij}^{(t)} \leftarrow (r_{ij}^{(t-1)} - r_{min}) \times \sum_{k=\lambda}^{c_{ij}^{(t)}} \frac{\lambda^{c_{ij}^{(t)}} \times e^{-\lambda}}{c_{ij}^{(t)}!}$
    end
    $\delta_{ij}^{(t)} \leftarrow (-1)^{m} \times v_{ij}^{(t)}$
end
return $R^{(t)} \leftarrow R^{(t-1)} + \Delta^{(t)}$
```

recommendation. We adopt the All-But-One evaluation methods [11] which, for each user, we iteratively skip one App from a user's preference list, and then make recommendation for this user. If the skipped App is recommended, we treat it as a hit. The hit ratio of user u at time unit t is calculated by Eq. 15, where k is the number of recommended Apps, $I(\cdot)$ is an indicator function defined in Eq. 16, $App_k(u,t)$ is the top-k Apps with highest preference score for user u at time unit t, and $R_k(u,t)$ is the list of k recommended Apps for user u at time unit t. The length of one time unit, l, is 1 day for both App traces, and 7 days for the Last.fm dataset. Eventually, the overall accuracy is the average hit ratio of every user at every time unit, which is shown in Eq. 17.

$$HitRatio_k(u,t) = \frac{\sum_{i \in App_k(u,t)} I_{R_k(u,t)}(i)}{|App_k(u,t)|} \tag{15}$$

$$I_{R_k(u,t)}(i) \begin{cases} 1 \ , i \in R_k(u,t) \\ 0 \ , i \notin R_k(u,t) \end{cases} \tag{16}$$

$$Accuracy_k = OverallHitRatio_k = \frac{\sum_u \sum_t HitRatio_k(u,t)}{|U| \times |T|} \tag{17}$$

5.1 Environment

The range of users' preferences is set to [0,5]. The adopted recommendation algorithm is Collaborative Filtering (CF) provided by Apache project, Mahout, with its similarity function as Pearson correlation function.

Dataset description. We have three real-world datasets: two are App usage traces and one is music listening log from Last.fm [12]. For the two traces of App usage, one is a smaller trace which consists of 30 users and 226 Apps, while the other one has 80 users and 650 Apps. Through the two different scales of datasets, we can ensure whether our methods are scalable or not. For the music listening dataset, we have a relatively huge amount of users in the Last.fm 1K-users dataset. The music listening dataset consists of 1000 users and 48,361 music albums which is a very sparse dataset we have to deal with. The total time duration for two App traces is half a year and for the music listening log is one and half years.

$$\frac{r_{ij}^t - r_{min}}{r_{max} - r_{min}} = \frac{c_{ij}^t - \min\limits_k c_{ik}^t}{\max\limits_k c_{ik}^t - \min\limits_k c_{ik}^t} \tag{18}$$

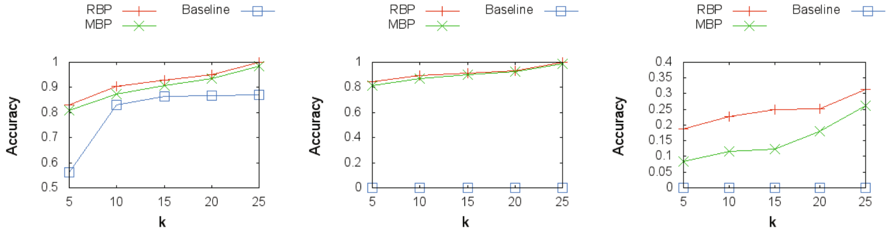
$$r_{ij}^t = \frac{(c_{ij}^t - \min\limits_k c_{ik}^t) \times (r_{max} - r_{min})}{\max\limits_k c_{ik}^t - \min\limits_k c_{ik}^t} + r_{min} \tag{19}$$

Compared methods. To compare the accuracy of our proposed algorithms, we adopt a usage-based method as the baseline. The usage-based method calculates the users' preferences only by the usage count. The item with largest number of usage will be assigned the preference of r_{max}, while the one with smallest number of usage will be assigned the preference of r_{min}. Besides, the preferences of other items are calculated by an interpolation method shown in Eq. 18.

5.2 Performance Evaluation

In this study, we evaluate the accuracy under various number of recommended Apps, k, and different length of a time unit, l over two proposed algorithms and one baseline method. Then, we focus on the proposed Reference-based Prediction (RBP) algorithm to see the accuracy when changing the parameter h which is used to control how many historical data are used.

Accuracy changed by k. Since k would affect the hit ratio, we calculate the hit ratio by different k from 5 to 25. However, although larger k could derive a better performance on hit ratio, fewer recommended items is more meaningful for users. Figs. 4(a) and 4(b) show the results of two App traces under different numbers of recommended items, k. Obviously, the accuracy increases as k grows up. Specifically, when $k = 5$, both RBP and MBP can achieve the accuracy of more than 80%. we note that in Fig. 4(b), the baseline remains close to zero



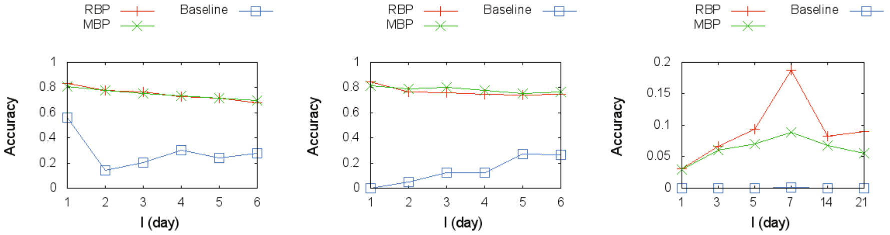
(a) App-small dataset. (b) App-large dataset. (c) Last.fm dataset.

Fig. 4. Accuracy evaluation with different k

even for $k = 25$, while in Fig. 4(a), the baseline achieves relatively low accuracy compared with RBP for $k = 5$. This is because the App-large dataset consists of more Apps and makes the dataset become sparser than App-small. Fig 4(c) depicts the results of Last.fm dataset. Since the music listening dataset is much sparser than the App traces, the performance on accuracy is not as good as the accuracy of the App traces. However, RBP is always the best method, while the baseline is close to zero. Here, for the two App traces, the length of one time unit is one day and the size of reference history, h, of RBP is set to 4 time units; for music listening dataset, the length of time unit is 1 week and the parameter h of RBP is 6 time units.

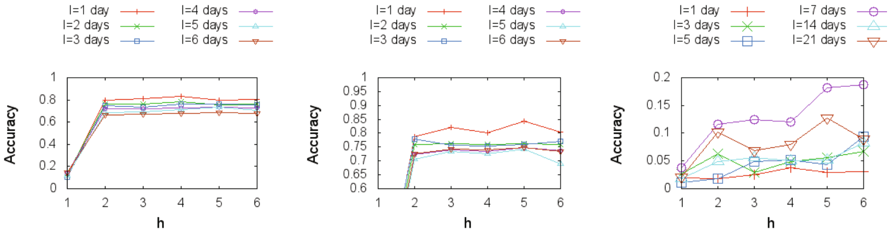
Impact of parameter *l*. Here, we evaluate the accuracy change of various length of a time unit. As can be seen in Figs. 5(a) and 5(b), both RBP and MBP slightly decrease their accuracy when the amount of training data increases. The best length of a time unit is one day which matches the human behavior. By contrast, the baseline method increases the accuracy when the number of training data becomes larger. Because the baseline method does not consider the temporal information, more training data could provide more information to overcome this drawback. However, when $d > 6$, the accuracy of baseline method also declines. On the other hand, as shown in Fig. 5(c), the best length of a time unit for Last.fm dataset is 7 days (one week) since music listening behavior is sparse and users may repeat the songs they listened in one week. Here, the reference history parameter, h, is set to 6 time units.

Impact of parameter *h* for RBP. Since the amount of reference history is a critical parameter for RBP algorithm, we evaluate the accuracy of recommendations under various reference histories. Figs. 6(a) and 6(b) depict the results of the App-small and App-large traces, and they reach the best accuracy on $h = 4$ and $h = 5$ respectively. Furthermore, the results of $h \geq 2$ are much better than the result of $h = 1$, because when $h = 1$, the number of reference points is too few to reflect the trend of users' usage. In addition, Fig. 6(c) shows the results of the Last.fm dataset, and the best accuracy falls on $h = 6$ and $l = 7$. Because the Last.fm is a sparse dataset, RBP algorithm needs more reference points and



(a) App-small dataset. (b) App-large dataset. (c) Last.fm dataset.

Fig. 5. Accuracy evaluation with the length of a time unit varied



(a) App-small dataset. (b) App-large dataset. (c) Last.fm dataset.

Fig. 6. Accuracy evaluation with the size of reference history varied

training data to achieve a better performance. Empirically, the setting of h does highly depend on different applications. In this paper, we suggest choosing a proper h larger than 4, since the regression line constructed in the first phase of RBP is more meaningful to reflect the trend of users' usage.

6 Conclusion

We proposed a novel dynamic preference prediction problem which is to dynamically quantize a user's preferences on Apps they have used from their usage traces. Two effective algorithms are designed to solve this problem. One is named Mode-base Prediction (MBP) which adopts a user's binary usage mode (active and inactive) and a proposed utility model to predict the preference value on an App. The other one is named Reference-base Prediction (RBP) which discovers a reference value by solving an optimization problem in a linear regression model and constructs a probabilistic model to check if the current behavior satisfies the reference model. RBP estimates the users' preferences by measuring the difference between actual usage and the derived reference value. In the experiments section, we evaluate the derived dynamic preferences by applying Collaborative Filtering. When the derived preferences can provide more accurate recommendation, the preferences are considered closer to users' actual affinities. As the experimental results show, the derived preferences of both MBP and RBP are effective. In addition, the RBP method can reach the accuracy of more than 80% for App traces. We suggest that the proposed dynamic preferences are valuable for many applications, such as providing recommendation of mobile applications, predicting and analysing users behavior, and make marketing decision.

Acknowledgement. Wen-Chih Peng was supported in part by the National Science Council, Project No.100-2218-E-009-016-MY3 and 100-2218-E-009-013-MY3, by Taiwan MoE ATU Program, by HTC and by Delta.

References

1. Dong, Y., Ke, Q., Rao, J., Wang, B., Wu, B.: Random walk based resource allocation: Predicting and recommending links in cross-operator mobile communication networks. In: 2011 IEEE 11th International Conference on Data Mining Workshops, ICDMW, Vancouver, BC, Canada, December 11, pp. 358–365 (2011)
2. Lymberopoulos, D., Zhao, P., König, A.C., Berberich, K., Liu, J.: Location-aware click prediction in mobile local search. In: Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24-28, pp. 413–422 (2011)
3. Guy, I., Zwerdling, N., Ronen, I., Carmel, D., Uziel, E.: Social media recommendation based on people and tags. In: Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland, July 19-23, pp. 194–201 (2010)
4. Yan, B., Chen, G.: Appjoy: personalized mobile application discovery. In: Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys 2011, Bethesda, MD, USA, June 28-July 1, pp. 113–126 (2011)
5. Shi, K., Ali, K.: Getjar mobile application recommendations with very sparse datasets. In: The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2012, Beijing, China, August 12-16, pp. 204–212 (2012)
6. Dror, G., Koenigstein, N., Koren, Y., Weimer, M.: The yahoo! music dataset and kdd-cup 2011. In: KDD-Cup Workshop (2011)
7. Koren, Y.: Collaborative filtering with temporal dynamics. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28-July 1, pp. 447–456 (2009)
8. Leskovec, J., McGlohon, M., Faloutsos, C., Glance, N.S., Hurst, M.: Patterns of cascading behavior in large blog graphs. In: Proceedings of the Seventh SIAM International Conference on Data Mining, Minneapolis, Minnesota, USA, April 26-28 (2007)
9. Fei, H., Jiang, R., Yang, Y., Luo, B., Huan, J.: Content based social behavior prediction: a multi-task learning approach. In: Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24-28, pp. 995–1000 (2011)
10. Lathia, N., Hailes, S., Capra, L., Amatriain, X.: Temporal diversity in recommender systems. In: Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland, July 19-23, pp. 210–217 (2010)

11. Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., Sun, J.: Temporal recommendation on graphs via long- and short-term preference fusion. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, pp. 723–732 (2010)
12. Celma, O.: Music Recommendation and Discovery in the Long Tail. Springer (2010)

Leveraging Hybrid Citation Context for Impact Summarization

Po Hu[1,2], Yujing Guo[1], Donghong Ji[1,*], and Jiacong He[3]

[1] Computer School, Wuhan University, China
[2] Computer School, Central China Normal University, China
[3] International School of Software, Wuhan University, China
phu@mail.ccnu.edu.cn,
{yujingguo.ximo,hejiacongtheone}@gmail.com,
donghong_ji2000@yahoo.com.cn

Abstract. Impact summarization aims to highlight the influential aspects of a cited paper by selecting a few representative citation sentences into t he summary. Most existing work considers only the citation sentence information while the hybrid citation context associated with each citation sentence has been ignored. This paper proposes a context-aware approach. In the approach, different kinds of relationships among papers and authors are leveraged to jointly infer the impact of hybrid citation context, which is further integrated in a sentence language smoothing model to measure citation sentence relationships more effectively. The experimental results show that the proposed approach can achieve significantly better results than several baselines.

Keywords: impact summarization, hybrid citation context, bibliographic network relationships.

1 Introduction

With the rapid evolution of scientific research, the volume of literature keeps on expanding fast. However, the explosive growth of the publications makes it r ather difficult to identify the influential aspects of papers quickly and effectively.

The abstract part of a s cientific paper may help researchers quickly understand the main content of the paper, but it o nly presents what the authors think to be the important contribution but not necessarily the actual impact of the paper. Actually, the impact of a paper sh ould be j udged by the consent of research community instead of the author himself. Moreover, the impact of a paper may dynamically change due to the progress of research. For example, a paper published before may be no longer the state of the art, but the research problem it addressed or the method it proposed will still attract peer attention.

Therefore, we argue that only the abstract part rep resenting the author's point of view is not enough, and how other papers cite and describe the target paper needs to

[*] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 354–365, 2013.

be comprehensively investigated to generate an impact summary, which can not only help researchers digest the results of research better, but also facilitate other literature mining applications such as research trend prediction, and survey generation, etc.

Actually, given a scientific paper, different citation sentences often focus on different aspects of that paper and all the citation sentences will provide a rich information source to summarize its impact [1]. Although some research has been done based on citation sentences, to the best of our knowledge, simultaneous consideration of the impact from hybrid citation context associated with each citation sentence has not been investigated. Therefore, we propose a novel approach by incorporating the impact of hybrid citation context into the summarization process. In the proposed approach, three kinds of relationships among papers and authors are first leveraged to jointly infer the impact of hybrid citation context. Next, the hybrid citation context and its impact are integrated in a sentence language smoothing model to measure citation sentence relationships more effectively. Lastly, a unified graph ranking algorithm is adopted to evaluate the significance of each citation sentence by taking advantage of the relationships between citation sentences.

The remainder of this paper is organized as follows. Section 2 reviews related work. The proposed approach is presented in Section 3. We then report the experimental results in Section 4. Finally, we present our conclusion and future work in Section 5.

2 Related Work

Automatic creation of scientific summaries has been studied for many years [2-4], but most previous work considers only the local features of the scientific paper, while other contextual information has been mostly ignored.

Recently, researchers have begun to make use of contextual information to aid news and webpage summarization [5-10]. Likewise, in order to summarize a paper, differentiating and utilizing citations from context have received increasing interests. Nakov et al. used sentences surrounding citations to create training and testing data for scientific paper summarization [11]. Nanba and Okumura classified different citation sentences into three categories and explored how to use them to aid survey generation [12]. Schwartz and Hearst utilized the citation sentences to summarize the key concepts and entities in bioscience texts [13]. Teufel et al. adopted rhetorical status analysis to reveal the scientific attribution of a paper, in which each citation sentence is labeled as one of Own, Other, Background, Textual, Aim, Basis, and Contrast [14]. Kan et al. used annotated bibliographies to cover certain summarization aspects [15]. Elkiss et al. performed a large-scale study on the PubMed repository and confirmed the importance of citation sentences in understanding what a paper contributes [16]. They also concluded that the citation sentences contain more focused information that generally does not appear in the abstract part of a paper.

Recently, Mei and Zhai proposed a language model based approach to impact summarization [17]. Qazvinian and Radev presented two different methods for the task [18] [19]. One utilized all the citation sentences of a paper to construct a similarity graph first, and then applied network analysis technique to cluster graph nodes and produce an impact summary. Another method first extracted a number of key phrases from the citation sentences, and then used these phrases to build the impact summary. How to produce more readable summaries based on citation sentences have also been investigated in [20]

As far as we know, none of the previous studies has investigated the impact from hybrid citation context (i.e., the combination of citation paper context and citation author context), and has used the citation context in the same way as we did. In this study, we propose a context-aware approach to simultaneously consider the impact from hybrid citation context and what is more, we further incorporate the hybrid citation context and its impact into a sentence language smoothing model to measure the citation sentence relationships beyond sentence level.

3 Impact Summarization Based on Hybrid Citation Context

Our approach incorporates hybrid citation context into the impact summarization, which consists of three steps: inferring the impact of hybrid citation context, estimation of citation sentence language model, and impact summary generation.

3.1 Inferring the Impact of Hybrid Citation Context

In the study, the sentence containing an explicit reference to the target paper and describing the work being cited is called a citation sentence. All the other citing papers and citing authors associated with the citation sentences are called hybrid citation context. The citation sentences occurring in the papers with higher topical relevance will contribute more than those in less important papers. The citation sentences written by the authors with better authority expertise will contribute more than those written by less professional authors. Therefore, to summarize the impact of a particular paper, the impact of hybrid citation context (i.e., the topical relevance of citing papers and the authority expertise of citing authors) should be inferred first.

Our approach operates over a bibliographic network G. $G=(V, E)=(V_P \cup V_A, E_P \cup E_A \cup E_{PA})$. G connects three subgraphs G_P, G_A, and G_{PA}. $G_P=(V_P, E_P)$ is a directed graph representing the citation relationships between papers. $V_P=\{p_i \mid p_i \in V_P\}$ denotes a collection of $|V_P|$ papers and E_P is the set of citation links between them. $G_A=(V_A, E_A)$ is an undirected graph representing the co-authorship relationships between authors. $V_A=\{a_i \mid a_i \in V_A\}$ is the set of authors with size $|V_A|$, and E_A is the set of co-authorship links between them. $G_{PA}=(V_P \cup V_A, E_{PA})$ is a bipartite graph that ties G_P and G_A and represents authorship associations between papers and authors.

Let $R_P(p_i)$ and $R_A(a_i)$ denote the topical relevance of paper p_i and the authority expertise of author a_i respectively. A query q can be constructed by extracting the title and keywords from the target paper, and then the initial scores $R_P{}^0(p_i)$ can be calculated by $R_P{}^0(p_i) = p(q \mid \theta_{p_i}) = \prod_{t \in q} p(t \mid \theta_{p_i})^{n(t,q)}$. Where $p(t \mid \theta_{p_i})$ is the maximum likelihood estimation of the term t in the paper p_i, and n(t, q) is the number of times that term t occurs in query q. Since an author a_i can be represented by the set of papers authored by a_i, the initial score $R_A{}^0(a_i)$ for author a_i can be calculated similarly.

Inspired by [22] and based on the assumption that similar papers will have similar relevance for a given query, we refine the topical relevance of citing papers by making use of the paper citation graph G_P and the initial topical relevance of papers.

Firstly, the adjacency matrix $W_P \in \Re^{|V_P| \times |V_P|}$ for the graph G_P is constructed. If a paper p_i cites another paper p_j ($i \neq j$), then we set the corresponding element $w_{p_{ij}}$ in W_P as 1, otherwise set it as 0. Then W_P is further normalized as a random walk transition matrix S_P by $S_P = \frac{D_P^{-1/2} W_P D_P^{-1/2} + D_P^{-1/2} W_P^T D_P^{-1/2}}{2}$. Where D_P is the diagonal matrix with (i,i)-element equal to the sum of the i-th row of W_P.

Next, inspired by [21], a regularization framework is developed by regularizing the smoothness of relevance over the graph and the cost function associated with it is defined as follows:

$$\Omega_P = \frac{1}{2}\sum_{i,j=1}^{|V_P|} s_{p_{ij}} \left\| \frac{R_P(p_i)}{\sqrt{d_{P_{ii}}}} - \frac{R_P(p_j)}{\sqrt{d_{P_{jj}}}} \right\|^2 + \frac{1}{2}\sum_{i=1}^{|V_P|} \left\| R_P(p_i) - R_P^0(p_i) \right\|^2 \qquad (1)$$

Where the first tem defines the global consistency of the refined relevance over the graph, while the second term defines the constraint to fit the initial relevance. By minimizing Ω_P, the topical relevance of papers can be refined.

Similarly, based on the assumption that if two authors co-authored many papers related to a given query, then their authority expertise in the queried field will be similar, we can refine the authority expertise of citing authors by making use of the author co-authorship graph G_A and the initial authority expertise of authors.

Firstly, the adjacency matrix $W_A \in \Re^{|V_A| \times |V_A|}$ for the graph G_A is constructed. If an author a_i coauthored with another author a_j ($i \neq j$), then we set the corresponding element $w_{a_{ij}}$ in W_A as the number of papers that they collaborated, otherwise set it as 0. Then W_A is further normalized by $S_A = D_A^{-1/2} W_A D_A^{-1/2}$. Where D_A is the diagonal matrix with (i,i)-element equal to the sum of the i-th row of W_A.

Next, a regularization framework is developed by regularizing the smoothness of expertise over the graph and the cost function associated with it is defined as follows [21]. By minimizing Ω_A, the authority expertise of authors can be refined.

$$\Omega_A = \frac{1}{2}\sum_{i,j=1}^{|V_A|} s_{a_{ij}} \left\| \frac{R_A(a_i)}{\sqrt{d_{A_{ii}}}} - \frac{R_A(a_j)}{\sqrt{d_{A_{jj}}}} \right\|^2 + \frac{1}{2}\sum_{i=1}^{|V_A|} \left\| R_A(a_i) - R_A^0(a_i) \right\|^2 \qquad (2)$$

In addition to Ω_P and Ω_A, another cost function Ω_{PA} is also presented based on the graph G_{PA} by considering the authorship relations between papers and authors.

$$\Omega_{PA} = \frac{1}{2}\sum_{i=1}^{|V_P|}\sum_{j=1}^{|V_A|} s_{pa_{ij}} \left\| \frac{R_P(p_i)}{\sqrt{d_{P_{ii}}}} - \frac{R_A(a_j)}{\sqrt{d_{A_{jj}}}} \right\|^2 + \frac{1}{2}\sum_{j=1}^{|V_A|}\sum_{i=1}^{|V_P|} s_{ap_{ji}} \left\| \frac{R_A(a_j)}{\sqrt{d_{A_{jj}}}} - \frac{R_P(p_i)}{\sqrt{d_{P_{ii}}}} \right\|^2 \qquad (3)$$

The intuition behind Ω_{PA} is that the authority expertise of an author is consistent with that of the relevant papers he published.

To define the cost function Ω_{PA}, the adjacency matrix $W_{PA} \in \Re^{|V_P| \times |V_A|}$ for the graph G_{PA} is constructed. If a paper p_i is written by an author a_j, then we set the corresponding element $w_{pa_{ij}}$ in W_{PA} as 1, otherwise set it as 0. Then W_{PA} is further normalized as S_{PA} such as the sum of each row of the matrix equal to one.

Next, a hybrid cost function Ω that combines Ω_P, Ω_A, and Ω_{PA} is developed in a unified regularization framework.

$$\Omega = \frac{1}{2}(\Omega_P + \Omega_A) + \frac{1}{2}\Omega_{PA} \tag{4}$$

We can minimize the hybrid cost function Ω using the standard conjugate gradient method, and a closed-form optimal solution can be derived. However, for a large-scale dataset, an iterative–form computation strategy would be more effective. So in the study, we calculate the optimal solutions R_P^* and R_A^* by adopting the equivalent iterative computation strategy, which details are omitted due to space limit, and you can find it in [21].

Finally, the converged solutions R_P^* and R_A^* correspond to the topical relevance of citing papers and the authority expertise of citing authors respectively.

3.2 Estimation of Citation Sentence Language Model

After inferring the impact of hybrid citation context, the next step is to make use of the contextual information to evaluate the relationships between citation sentences.

From the language model perspective, it can be assumed that a citation sentence s is generated from a sentence language model θ_s and Dirichlet prior smoothing [23] is often adopted to estimate θ_s as follows.

$$p(w \mid \theta_s) = \frac{c(w,s) + \mu_s * p(w \mid B)}{|s| + \mu_s} \tag{5}$$

Where |s| is the length of s, c(w, s) is the count of term w in s, p(w|B) is usually estimated by $\frac{c(w,B)}{\sum_{w' \in W} c(w',B)}$. Here B is the whole background paper set and μ_s is the sentence smoothing parameter which is set as 1000 as in [17].

In this study, we propose a citation sentence language smoothing model inspired by [6] to estimate $p(w \mid \theta_s)$ by using hybrid citation context as background, which can be defined as follows.

$$p(w \mid \theta_s) = \alpha * p(w \mid s) + \beta * \mathrm{R_P}(p_s) * p(w \mid p_s) + \gamma * \sum_i \mathrm{R_A}(a_{i_s}) * p(w \mid a_{i_s}) \tag{6}$$

Where α, β, and γ belong to [0, 1] and $\alpha+\beta+\gamma=1$. p_s is the citing paper that citation sentence s belongs to and $\mathrm{R_P}(p_s)$ denotes the topical relevance of paper p_s. a_{i_s} is the i-th citing author of the citing paper p_s. $\mathrm{R_A}(a_{i_s})$ denotes the authority expertise of author a_{i_s}. $p(w|a_{i_s})$ is estimated by the papers authored by a_{i_s}.

Based on the estimated citation sentence language model, the distance $Dis_{AvgKL}(s_i,s_j)$ between two citation sentences s_i and s_j can be measured by the average KL divergence as follows.

$$Dis_{AvgKL}(s_i,s_j)=\frac{Dis_{KL}(s_j\,||\,s_i)+Dis_{KL}(s_i\,||\,s_j)}{2} \tag{7}$$

$$Dis_{KL}(s_j\,||\,s_i)=\sum_{w\in W}p(w|\theta_{s_j})\log\frac{p(w|\theta_{s_j})}{p(w|\theta_{s_i})} \tag{8}$$

$$Dis_{KL}(s_i\,||\,s_j)=\sum_{w\in W}p(w|\theta_{s_i})\log\frac{p(w|\theta_{s_i})}{p(w|\theta_{s_j})} \tag{9}$$

Where W is the set of terms in our vocabulary and w is a term in W. And the similarity $\mathrm{Sim}(s_i, s_j)$ between two citation sentences s_i and s_j can then be inferred by the following formula.

$$\mathrm{Sim}(s_i,\ s_j)=\frac{1}{1+e^{Dis_{AvgKL}(s_i,s_j)}} \tag{10}$$

3.3 Impact Summary Generation

In this step, all the citation sentences are to be evaluated by the significance and a few sentences with highest significant scores will be selected into the impact summary.

In most of the methods for impact summarization, all citation sentences are treated uniformly. However, different citation sentences from different citation contexts should be treated differently, since the citation sentences from a more important context should receive higher significant score. Therefore, it is more reasonable to assign unequal weights to different citation sentences in accordance with the impact of different citation contexts which they belong to.

Given a set of citation sentences S for a target paper, let $G_S=(V_S, E_S)$ be an undirected graph to reflect the relationships between citation sentences in S. Here V_S is the set of citation sentences. E_S is the set of edges and each edge $e_{s_{ij}}$ is associated with the similarity $\mathrm{Sim}(s_i, s_j)$ between sentences s_i and s_j ($i\neq j$), which is calculated by formula 10. Two sentences are connected if their similarity is larger than 0 and we let $\mathrm{Sim}(s_i, s_i)=0$ to avoid self transition. We use the affinity matrix $\mathrm{M_S}$ to describe G_S. Then $\mathrm{M_S}$ is normalized to $\widetilde{\mathrm{M_S}}$ by making the sum of each row equal to 1.

Based on $\widetilde{M_S}$, the significant score $\text{SigScore}(s_i)$ for citation sentence s_i can be deduced from those sentences linked with it, which can be formulated in a recursive form as follows:

$$\text{SigScore}(s_i)=\delta * \sum_{all\ j \neq i} \text{SigScore}(s_j) * \widetilde{M_S}_{ji} + \frac{1-\delta}{|V_S|} \tag{11}$$

Where δ is the damping factor usually set to 0.85, as in the PageRank algorithm. For implementation, the initial significant scores of all citation sentences are set to 1. Usually the convergence of the iteration algorithm is achieved when the difference between the scores computed at two successive iterations for any citation sentences fall below a given threshold (0.0001 in this study).

After evaluating the significance of each citation sentence, we select a few representative sentences with highest significant scores to generate the impact summary.

Recall that in the proposed approach, we incorporate diverse relationships on G_P, G_A, and G_{PA} into a unified regularization framework to infer the impact of hybrid citation context, and then rank citation sentences on G_S by leveraging both the impact of hybrid citation context and the relationships between citation sentences, which can be intuitively represented by Figure 1.

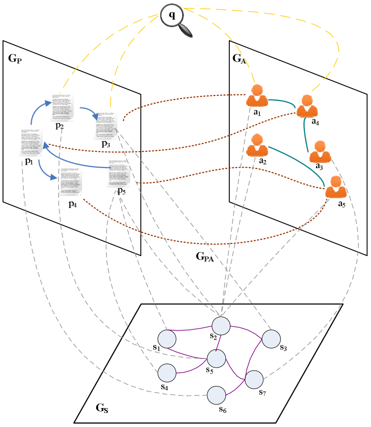


Fig. 1. The intuitive representation of the proposed approach

4 Experiments and Evaluation

4.1 Data Collection

We evaluate the proposed approach on the dataset[1], which contains 25 highly cited papers from computational linguistics domain. Each paper has a set of manually

[1] `http://www-personal.umich.edu/~vahed/resources/single.tar.gz`

selected terms representing the most important impacts of that paper and shared by multiple evaluators who has read all the citation sentences of that paper.

Considering that hybrid citation context may improve the performance of impact summarization, we extend the dataset by adding a number of papers with similar topic and related authors from the ACL Anthology Network[2], which is a large collection of more than 18,000 papers from computational linguistics domain. Table 1 shows general statistics about the extended dataset.

Table 1. The general statistics about the extended dataset

Papers	7921
Authors	1475
Citation links between papers	38542
Co-authorship links between authors	14176
Authorship links between papers and authors	13951

We deem that a good impact summary should cover more important impacts of the target paper. If an impact fact occurs in more citation sentences, it should be regarded as more important and be assigned higher weight. Under the condition, the citation sentence including more impact facts with higher weight will become a good candidate for impact summary. Accordingly, we construct a reference summary for each of the 25 highly cited papers by making use of the manually selected impact terms. We pick citation sentences that cover new and highly weighted impact terms into the reference summary until the defined summary length is reached. By this way, we expect a good system generated summary to be closer to the reference summary.

4.2 Evaluation Metrics

In the study, the ROUGE toolkit [24] is adopted, which was officially adopted by DUC for automatic summarization evaluation. ROUGE metrics measure a summary's content quality by counting overlapping units such as n-gram, word sequences, and word pairs between the automatically generated summary and the reference summary. The higher the ROUGE scores, the similar the two summaries are.

A few recall-oriented ROUGE metrics have been employed including ROUGE-1, ROUGE-2, and ROUGE-SU4, etc. Among the different ROUGE scores, ROUGE-1 has been shown to agree with human judgment most [24]. Therefore, we only report ROUGE-1 in the following experiments since other metrics gives very similar results.

4.3 Experimental Results

We compared our proposed approach with several baselines as follows. All the approaches for comparison are required to extract a few representative citation sentences into the impact summary for each of the 25 highly cited papers. The main difference

[2] http://clair.eecs.umich.edu/aan_site2/index.php

between our approach and other baselines is that we leverage the hybrid citation context associated with each citation sentence while other baselines do not.

Random: In this baseline, the sentences are selected randomly from the set of citation sentences and added to the impact summary.

OTS [25]**:** It integrates shallow NLP techniques with statistical word frequency analysis to rank and select citation sentences.

LexRank [26]**:** It runs on the set of citation sentences by first constructing a citation sentence affinity graph, and then extracting a few informative citation sentences based on eigenvector centrality.

C-LexRank [18]**:** This is another state-of-the-art impact summarizer in which the citation sentences are firstly clustered, and then the sentences within each cluster are ranked via LexRank algorithm.

We show the evaluation results of different methods in Tables 2, and the highest ROUGE-1 scores are shown in bold type.

Table 2. The evaluation results of different methods

Method	ROUGE-1
Our Approach	**0.39507**
C-LexRank	0.37837
LexRank	0.36021
OTS	0.34404
Random	0.32966

In the experiments, the best result of our approach is achieved when the weight adjusting parameters in the formula 6 are set as follows: α=0.4, β=0.3, and γ=0.3. These parameters give different weights to the citation sentence, the citation paper context, and the citation author context respectively.

Seen from Table 2, our proposed approach using the hybrid citation context achieves the best performance compared to that of the baseline approaches (i.e. C-LexRank, LexRank, OTS, and Random), which demonstrates that both citation paper context and citation author context are critical for improving the performance of impact summarization.

C-LexRank and LexRank perform better than those of OTS and Random. This is mainly because both C-LexRank and LexRank make use of the inter-relationships between citation sentences to rank them globally, while OTS only depends on the local features.

C-LexRank outperforms LexRank in our experiments, which indicates the use of appropriate cluster-level information is an improvement over the use of citation sentences alone.

Note that all these baselines generate the impact summary based only on the citation sentences or sentence clusters, regardless of the impact from hybrid citation context. Our proposed approach shows significantly better performance on ROUGE scores, and the result difference between our approach and other baselines is significant at the 95%

statistical confidence level. These observations again demonstrate the effectiveness of our approach by exploiting hybrid citation context to aid impact summarization.

In the following, we will explore the effect of different parameters in our approach. The key parameters we want to investigated are α , β , and γ .

Figure 2 to 4 demonstrate the influence of these parameters in the proposed approach when we tune a parameter from 0 to 1 with the step length 0.1 and vary the other two for the best performance to achieve.

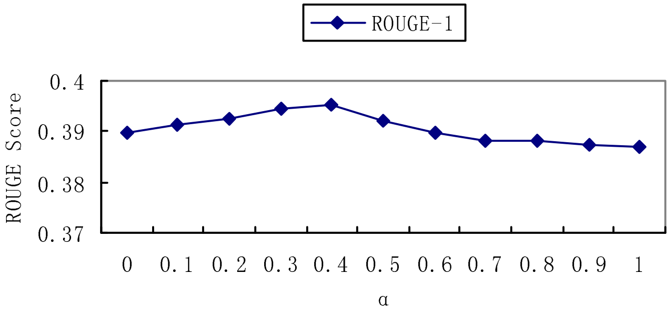


Fig. 2. ROUGE-1 score of the proposed approach vs. α

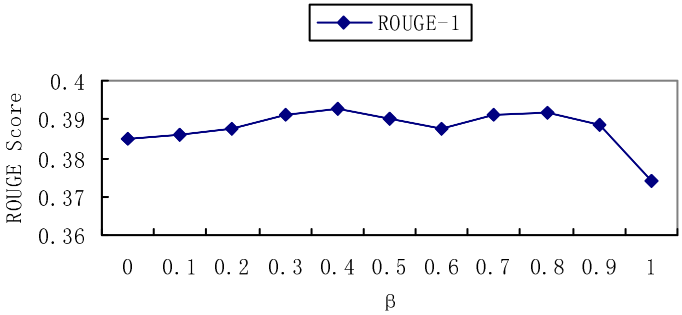


Fig. 3. ROUGE-1 score of the proposed approach vs. β

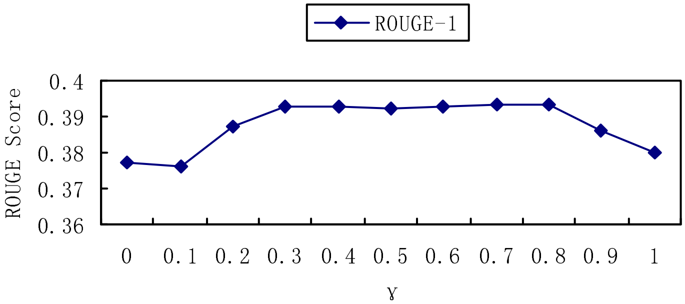


Fig. 4. ROUGE-1 score of the proposed approach vs. γ

From Figure 2 to 4, it can be found that the citation sentence information controlled by the parameter α is relatively stable and have little i mpact on the performance. Both citation paper context and citation author context can help improve the performance, but excessive dependence on any one of them will impair the performance to a certain extent.

5 Conclusion and Future Work

This paper pr oposes a con text-aware approach to impact summarization. In the proposed approach, different kinds of relationships among papers and authors are leveraged to jointly infer the impact of hybrid citation context, which is further integrated in a sentence language smoothing model to m easure citation sentence relationships more effectively.

In future work, it would be interesting to investigate the performance of the proposed approach on larger bibliographic datasets such as DBLP, ArnetMiner, etc. Besides, we will explore machine learning based methods to determine the parameters of our approach in an adaptive way.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No. 61133012, 61173062, 61070082) and the Major Projects of Chinese National Social Science Foundation (No.11&ZD189).

References

1. Amjad, A.J., Radev, D.R.: Reference Scope Identification in Citing Sentences. In: 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2012), pp. 80–90 (2012)
2. Paice, C.D., Jones, P.A.: The Identification of Important Concepts in Highly Structured Technical Papers. In: 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1993), pp. 69–78 (1993)
3. Saggion, H., Lapalme, G.: Generating Indicative-Informative Summaries with SumUM. Computational Linguistics 28(4), 497–526 (2002)
4. Teufel, S., Moens, M.: Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status. Computational Linguistics 28(4), 409–445 (2002)
5. Wan, X.J., Yang, J.W.: Single Document Summarization with Document Expansion. In: 22nd National Conference on Artificial Intelligence (AAAI 2007), pp. 931–936 (2007)
6. Yan, R., Yuan, Z., Wan, X., Zhang, Y., Li, X.: Hierarchical Graph Summarization: Leveraging Hybrid Information through Visible and Invisible Linkage. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part II. LNCS (LNAI), vol. 7302, pp. 97–108. Springer, Heidelberg (2012)
7. Sun, J.T., Shen, D., Zeng, H.J., Yang, Q., Lu, Y.C., Chen, Z.: Web-Page Summarization Using Clickthrough Data. In: 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005), pp. 194–201 (2005)
8. Hu, P., Sun, C., Wu, L.F., Ji, D.H., Teng, C.: Social Summarization via Automatically Discovered Social Context. In: 5th International Joint Conference on Natural Language Processing (IJCNLP 2011), pp. 483–490 (2011)

9. Hu, M.S., Sun, A.X., Lim, E.P.: Comments-Oriented Document Summarization: Understanding Documents with Users' Feedback. In: 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008), pp. 291–298 (2008)
10. Yang, Z., Cai, K.K., Tang, J., Zhang, L., Su, Z., Li, J.Z.: Social Context Summarization. In: 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011), pp. 255–264 (2011)
11. Nakov, P., Schwartz, A., Hearst, M.: Citances: Citation Sentences for Semantic Analysis of Bioscience Text. In: ACM SIGIR 2004 Workshop on Search and Discovery in Bioinformatics (2004)
12. Nanba, H., Okumura, M.: Towards Multi-Paper Summarization Using Reference Information. In: 16th International Joint Conference on Artificial Intelligence (IJCAI 1999), pp. 926–931 (1999)
13. Schwartz, A.S., Hearst, M.: Summarizing Key Concepts Using Citation Sentences. In: The Workshop on Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis (BioNLP 2006), pp. 134–135 (2006)
14. Teufel, S.: Argumentative Zoning for Improved Citation Indexing. Computing Attitude and Affect in Text: Theory and Applications, pp. 159–170 (2005)
15. Kan, M.Y., Klavans, J.L., McKeown, K.R.: Using the Annotated Bibliography as a Resource for Indicative Summarization. In: 3rd International Conference on Language Resources and Evaluation, LREC 2002 (2002)
16. Elkiss, A., Shen, S.W., Fader, A., Erkan, G., States, D., Radev, D.: Blind Men and Elephants: What Do Citation Summaries Tell Us about a Research Article. Journal of the American Society for Information Science and Technology 59(1), 51–62 (2008)
17. Mei, Q.Z., Zhai, C.X.: Generating Impact-Based Summaries for Scientific Literature. In: 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technology (ACL 2008), pp. 816–824 (2008)
18. Qazvinian, V., Radev, D.R.: Scientific Paper Summarization Using Citation Summary Networks. In: 22nd International Conference on Computational Linguistics (COLING 2008), pp. 689–696 (2008)
19. Qazvinian, V., Radev, D.R., Ozgur, A.: Citation Summarization through Keyphrase Extraction. In: 23rd International Conference on Computational Linguistics (COLING 2010), pp. 895–903 (2010)
20. Amjad, A.J., Radev, D.R.: Coherent Citation-Based Summarization of Scientific Paper. In: 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011), pp. 500–509 (2011)
21. Deng, H.B., Lyu, M.R., King, I.: A Generalized Co-HITS Algorithm and Its Application to Bipartite Graphs. In: 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009), pp. 239–248 (2009)
22. Deng, H.B., Han, J.W., Lyu, M.R., King, I.: Modeling and Exploiting Heterogeneous Bibliographic Networks for Expertise Ranking. In: 12th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2012), pp. 71–80 (2012)
23. Zhai, C.X., Lafferty, J.: A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. In: 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001), pp. 334–342 (2001)
24. Lin, C.Y., Hovy, E.: Automatic Evaluation of Summaries Using N-Gram Cooccurrence Statistics. In: 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 71–78 (2003)
25. Nadav, R.: The Open Text Summarizer, `http://libots.sourceforge.net/`
26. Erkan, G., Radev, D.R.: LexPageRank: Prestige in Multi-Document Text Summarization. In: 2004 Conference on Empirical Methods in Natural Language Processing, EMNLP 2004 (2004)

Optimal Allocation of High Dimensional Assets through Canonical Vines

Wei Wei[1], Jinyan Li[1], Longbing Cao[1], Jingguang Sun[2],
Chunming Liu[1], and Mu Li[1]

[1] Advanced Analytics Institute, FEIT, University of Technology Sydney, Australia
[2] Liaoning Technical University, China
{Wei.Wei-3,chunming.Liu}@student.uts.edu.au,
{Jinyan.Li,LongBing.Cao}@uts.edu.au, sunjinguang88@163.com,
muli60@gmail.com

Abstract. The widely used mean-variance criteria is actually not the optimal solution for asset allocation as the joint distribution of asset returns are distributed in asymmetric ways rather than in the assumed normal distribution. It is a computationally challenging task to model the asymmetries and skewness of joint distributions of returns in high dimensional space due to their own complicated structural complexities. This paper proposes to use a new form of canonical vine to produce the complex joint distribution of asset returns. Then, we use the utility function of Constant Relative Risk Aversion to determine the optimal allocation of the assets. The importance of using the asymmetries information is assessed by comparing the performance of a portfolio based on the mean-variance criteria and that of a portfolio based on the new canonical vine. The results show that the investors using the forecasts of these asymmetries can make better portfolio decisions than those who ignore the asymmetries information.

Keywords: Canonical Vine, Mean Variance Criterion, Financial Return.

1 Introduction

Financial asset returns follow non-normal distributions—asymmetries and skewness very often exist in the distribution of financial asset returns such as in stock returns [2] and [3]. These asymmetries facts violate the traditional distribution assumption on financial asset returns, making the traditional mean variance analysis [10] unreasonable. Some previous studies [12], [13] and [15] had attempted to compare the expected utility obtained from the mean-variance criterion with the approximated utility obtained from the benchmark portfolios (those equally divided portfolios). It was found that the mean variance criterion had poor performance on analyzing the skew and asymmetric portfolios. The mean variance criterion was good only at the portfolio that consists of riskless assets as riskless assets are driven by an normal distribution in line with the mean variance criterion assumption.

Arrow [4] laid down a theoretical foundation for the importance of using distributional asymmetries. He suggests that a desirable property of utility functions (including the Constant Relative Risk Aversion utility function) is the

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 366–377, 2013.

non-increasing absolute risk aversion. It means that under the non-increasing absolute risk aversion, investors may have a preference for positively skewed portfolios. Asymmetries in the dependence structure have direct impact on the skewness of the portfolio return. Therefore, while making the portfolio decision favorable to risky assets, it is also essential to consider the existence of and the impact between asymmetries and skewness. In the past, some studies, such as [12] and [13], had proposed models to construct the dependence structure with only two financial assets. That is far away from the need of investors. Investors and trading agents generally purchase tens of risky assets, rather than two assets in order to reduce aggressive risk. Therefore, it is demanded to develop a model that can resolve difficulties in the high dimensional asset allocation.

There are three challenges in the high dimensional asset allocation. First, as discussed above, the correlations between financial assets are asymmetric, rather than normal. With high dimensional input, the dependence structure becomes extremely complex, it is is difficult to capture and model all of these correlations between assets. Second, it is important to obtain the joint probability density function. However, the high dimensional joint distribution function has a big number of parameters which are increased exponentially as the data dimensions increase. Third, each individual asset has its own characteristics, such as volatility clustering and fat tail. It is a challenging task to combine these characteristics into the dependence structure.

To fulfill this need, we propose to use a canonical vine based dependence model for an optimization of the high dimensional asset allocation. The new model can capture asymmetric and skew correlations in the dependence structure, and can optimize the dependent structure. To address the high dimensional issue, we employ the idea of partial correlation to construct the canonical vine in our dependence model. It can capture the most important correlations in the dependence structure, and can reduce the complexity of the dependence structure remarkably to make invertors understand comprehensively. In addition, we also employ the ARMA-Garch model for the estimate of marginal distribution to capture volatility clustering and fat tail in financial assets.

The main contribution made by this paper is the partial correlation based canonical vine. The canonical vine is optimized to be suitable to high dimensional data input as it can remarkably reduce the number of nodes and parameters and simplify the canonical vine structure. Suppose there are 50 variables, the normal canonical vine will generate 1225 nodes. However, in our partial correlation based canonical vine, the number of nodes is only one tenth of that (around 267 nodes). In addition, our method can test hypotheses in parallel such as: (1) whether these asymmetries are predictable out of sample; and (2) whether we can make better portfolio decisions by using the forecasts of these asymmetries. If the answer to any of these questions is 'yes', then the asymmetries are very important for the high dimensional asset allocation. Finding models to fit in-sample data very well without considering the asymmetries and skewness does not imply that it will result in a better out-of -sample portfolio decisions. In the paper, we first build the model which can capture all important asymmetries and skewness in the dependence structure, then we compare it with those models that do not consider the correlations and/or asymmetric dependence structure between financial assets.

The rest of the paper is organized as follows. Section 2 presents a short introduction to copula theories which is closely related to canonical vine. Section 3 describes the problem of optimal assets allocation in portfolio, including how to construct the optimal canonical vine and marginal distribution. Section 4 discusses how to evaluate the performance of our model with equally divided allocation and mean-variance criterion. In Section 5, we apply the optimal canonical vine to capture the dependence structures of two portfolios, and evaluate the performance of our model in comparison to the performance by the mean variance criterion and equally divided allocation method. Section 6 concludes the paper with a summary.

2 Related Work

Copula is a useful tool to model the non-normal distribution. It can capture the complicated correlation between variables, including linear or non-linear. According to Sklar's theorem [14], a copula function is defined to connect univariate functions to form a multivariate distribution function. The definition of a copula function is given by:

$$F(x_1, ..., x_n) = C(F_1(x_1), F_2(x_2), ..., F_n(x_n)) \tag{1}$$

where, $x = [x_1, x_2, ..., x_n]$ is a random variable vector, F is a joint distribution and $F_1, F_2, ..., F_n$ are the marginal distributions of the corresponding variables respectively. It shows that all multivariate distribution functions, and copula function can be used in conjunction with univariate distributions to construct multivariate distribution functions. The differential of Equation (1) is:

$$f(x_1, ..., x_n) = \prod_{i=1}^{n} f_i(x_i) \cdot c(F_1(x_1), F_2(x_2), ..., F_n(x_n)) \tag{2}$$

where, c is the density copula function and $f_i(x_i)$ is the density function of marginal distributions. Equation (2) shows that the joint density function includes two parts. One is the description of the marginal behavior of individual factor, which is marginal distributions. The other is the description of their dependence structure, which is copula function. It implies an important property that copula function can separate dependence structure from marginal distribution function. We can model the individual variables using whichever marginal distributions provide the best fit and then model the dependence structure by using the copula function. The useful property can help understand the complex dependence structure, and describing the complex dependence structure on a quantile scale. The deep instruction regarding with copula can be found in [11].

Patton [12] builds an asset allocation with copula. The portfolio is composed by two assets, evaluating the asset allocation with the investor's utility function. Jondeau and Rockinger [7] use the Taylor series to calculate the expected utility.

An obvious advantage of the method is that it remains operational even if a large number of assets are involved. Sun et al. [15] proposed a copula arma-garch model to predict the co-movements of six German equity market indices at high frequency. It was found that the copula arma-garch model is able to capture multi-dimensional co-movements among the indices.

3 The Portfolio Optimization Problem and Our New Method

3.1 CRRA Optimization Function

Suppose that hypothetical investors follow the class of Constant Relative Risk Aversion (CRRA) utility functions:

$$U(\gamma) = \begin{cases} (1-\gamma)^{-1} \cdot (P_0 R_{port})^{1-\gamma}, & \text{if} \gamma \neq 1 \\ log(P_0 R_{port}), & \text{if} \gamma = 1 \end{cases} \tag{3}$$

where γ is the risk aversion parameter, P_0 is the initial wealth and R_{port} is the portfolio return. In this paper, the value of risk aversion parameter is considered at four different levels, including $\gamma = 2, 5, 7$ and 10, as suggested by [5]. We use CRRA utility function to calculate the expectation return of the hypothetical investors as its prominence in the finance literature. If the results are obtained by using the CRRA utility function, then the methods or algorithms are used as a conservative estimate of the other possible results or gains by using other more sensitive utility functions.

The next step is to built a portfolio of returns. Our work is focused on the portfolio return with high dimensional assets, defined as

$$P_0 R_{port} = P_0 \cdot (1 + \sum_{i=1}^{n} \omega_i X_t) \tag{4}$$

where $X_t = x_{i,t}$ is the asset return at time t, and ω is the proportion of wealth for each asset i. Generally, the initial wealth P_0 is set to zero as it does not affect the choice of weights. Suppose the joint distribution is F_t, with the associated marginal distribution $F_{1,t}, ..., F_{n,t}$, and copula C_t. We develop the density forecasts of the joint distribution $F_{1,t+1}, ..., F_{n,t+1}$ and the copula function C_{t+1}. Then, we use the forecast function to calculate the optimal weights ω^*_{t+1} for the portfolio. The optimal weights, ω^*_{t+1}, are found by maximizing the expected CRRA utility function:

$$\begin{aligned} \omega^*_{t+1} &= \arg\max_{\omega \in W} E_{F_{t+1}}[U(1 + \sum_{i=1}^{n} \omega_{i,t+1} X_{t+1})] \\ &= \arg\max_{\omega \in W} \int_{x_1} \int_{x_2} ... \int_{x_n} U(1 + \sum_{i=1}^{n} \omega_i x_i) \cdot f_{t+1}(x_1, x_2, ..., x_n) \cdot dx_1 \cdots dx_n \\ &= \arg\max_{\omega \in W} \int_{x_1} \int_{x_2} ... \int_{x_n} U(1 + \sum_{i=1}^{n} \omega_{i,t+1} x_{i,t+1}) \cdot f_{1,t+1}(x_1) \cdots f_{n,t+1}(x_n) \\ &\quad \cdot c_{t+1}(F_{1,t+1}(x_1), F_{2,t+1}(x_2), ..., F_{n,t+1}(x_n)) \cdot dx_1 \cdots dx_n \end{aligned} \tag{5}$$

where $W_{t+1} = \{(\omega_{1,t+1}, ..., \omega_{n,t+1}) \in [0,1]^n : \sum_{i=1}^{n} \omega_i \leq 1\}$ for the short sales constrained investors. The investors will estimate the model of conditional distribution of returns by using maximum likelihood estimation, and then optimize the portfolio weights via the predicted distribution of return. For the integral function, we use Monte Carlo replications to estimate the value of integral. For the optimal portfolio weights, we employ the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) algorithm to obtain the optimal weights.

3.2 Partial Correlation Based Canonical Vine: Our New Ideas

The key step in Equation 5 is to form the joint density function $f_{t+1}(x_1, ..., x_n)$ at time t+1. Equation (2) shows that the joint density function can be divided into two parts: the copula function c_{t+1} and the marginal distributions $f_{1,t+1}(x_1) \cdots f_{n,t+1}(x_n)$. In this Section, we discuss how to produce a copula density function. The construction of marginal distributions is explained in Section 3.3.

One way to build high-dimensional copula density function is to use canonical vine to build dependence structure as proposed by [1]. The basic scheme for modeling high-dimensional dependence structure with canonical vine is to decompose multivariate density functions into many conditional pair copulas. These pair copulas are bivariate copulas in one time. The model based on canonical vine transforms one high dimensional dependence structure into multiple two-dimensional structures. However, one important issue of canonical vine is that if the variables are large in number, the canonical vine will become quit complex. In addition, the nodes of canonical vine will increase exponentially as the variables increase. Therefore, we propose a new partial correlation based canonical vine to model high dimensional dependence structure.

The principle for the new canonical vine construction is to capture the most important correlation in the dependence structure, meanwhile to decrease the number of nodes, and to reduce the complexity of dependence structure. That is, the new canonical vine can capture the most important correlation, and ignore the weak correlations. Following this principle, we develop a new algorithm to construct and optimize the canonical vine by using partial correlation. First, we construct the canonical vine by using partial correlation rather than by using bivariate conditional copula function. Then, we optimize the partial correlation based canonical vine by setting the absolute small value of partial correlation to zero to decrease number of nodes and reduce the complexity of canonical vine. The optimal partial correlation based canonical vine can be mapped into the canonical vine based on bivariate conditional copula, provided that pair copulas are from the elliptical copula family [8]. The algorithm to construct the optimal canonical vine is presented in Algorithm 1.

We take an example to explain how to construct and optimize the canonical vine. Suppose there are one comprehensive index and 6 stocks which denoted by M, A, B, C, D, E and F. The canonical vine will consist of 6 trees and 21 nodes in both the canonical vine structure based on partial correlation and conditional copula. All the trees and nodes are shown in Figure 1. Each node

Algorithm 1. Canonical Vine Construction and Optimization

```
Require: observations of $n$ input variables
Calculate all values of partial correlation, and then allocate the smallest absolute value of partial correlation to the node in $T_{n-1}$ ($T_{n-1}$ is the bottom tree).
for $k = 1, ..., n-2$ do
    If $T_i > T_k$, find an appropriate root variables in $T_i$ which can minimize the function $\sum |\rho_{c:d}|$, where $T_i$ indicates the $i$th tree and $T_k$ is broken level tree;
    If $T_i \leq T_k$ , find an appropriate root variables in $T_i$ which can minimize the function of $\sum log(1-\rho^2_{c;d})$;
end for
There will be $(n-2)-1$ canonical vines as $k = 1, ..., n-2$. Calculate the function $-log(D)$ of all of the canonical vines based on partial correlation, and choose the maximum value of the function as the 'best' canonical vine. ($D$ is calculated in Equation 7);
For the 'best' canonical vine, the small absolute values of partial correlation, which are less than significance value $\tau$, are set to zero;
The optimal canonical vine based on conditional copula is corresponding to the canonical vine based on partial correlation;
return The optimal canonical vine dependence structure.
```

can be allocated to one bivariate copula or one partial correlation. The first step is to build the partial correlation based canonical vine. The partial correlation of all variables can be obtained by using the following Equation:

$$\rho_{12;3,\ldots,n} = \frac{\rho_{12;3,\ldots,n-1} - \rho_{1n;3,\ldots,n-1} \cdot \rho_{2n;3,\ldots,n-1}}{\sqrt{1-\rho^2_{1n;3,\ldots,n-1}} \cdot \sqrt{1-\rho^2_{2n;3,\ldots,n-1}}} \tag{6}$$

For these 7 variables, there are totally 21 partial correlations and 6 trees. T_i is defined as the ith tree in the paper. The smallest absolute value of these partial correlations is allocated to the root node in T_6 (the sixth tree shown in Figure 1) as the T_6 only has one node. Suppose the selected partial correlation in the T_6 is $\rho_{E,F;M,A,B,C,D}$. The variables in T_6 are variables E and F. The sets $c_6 = \{E, F\}$ and $d_6 = \{M, A, B, E, F\}$ are called conditioned set and conditioning set respectively. The next step is to chose the root variable in T_5. In T_5, there are two nodes which can be allocated as two partial correlations. The root variable of T_5 should be from d_6. If the selected root variable of T_5 is D, then it can generate two new conditioned sets: $c_5 = \{D, E\}$ and $c'_5 = \{D, F\}$. The corresponding conditioning set for c_5 and c'_5 is $d_5 = \{M, A, B, C\}$. The partial correlations allocated to the two nodes are $\rho_{D,E;M,A,B,C}$ and $\rho_{D,F;M,A,B,C}$. If we choose C as the root variable of T_5, the two new conditioned sets are: $c_5 = \{C, E\}$ and $c'_5 = \{C, F\}$. The corresponding conditioning set for c_5 and c'_5 is $d_5 = \{M, A, B, D\}$. The partial correlation allocated to the two nodes will be $\rho_{C,E;M,A,B,D}$ and $\rho_{C,F;M,A,B,D}$. If the selected root variable of T_5 is M, A or B , the processes of generating conditioned and conditioning set are similar to those root variable as C or D. However, we need to identify which variable is the most appropriate root variable in T_5.

We proposed a method to identify the appropriate root variable, which is called tree broken method. In the paper, we define that k is a tree-broken level. For trees beyond the kth tree ($T_i > T_k$), the appropriate root variable must minimize the value of function $\sum |\rho_{c:d}|$. For trees within the kth tree ($T_i \leq T_k$), the appropriate root variables must minimize the value of function $\sum log(1 - \rho^2_{c;d})$. For example, if $k = 3$, for T_1, T_2 and T_3 (the first, second and third trees), the appropriate root variables for these trees must minimize the value of function $\sum |\rho_{c:d}|$. For T_4, T_5 and T_6 (the fourth, fifth and sixth trees), the appropriate root variables for these trees must minimize the value of function $\sum log(1 - \rho^2_{c;d})$. The parameter k can be chosen from $1, 2, 3, 4, 5$. Therefore, there should totally have 5 canonical vines. Then, the 'Best' canonical vine should maximize the value of function $-log(D)$, where D is the determinant of canonical vine which is calculated by using:

$$D = \prod_{\{i,j\}} (1 - \rho^2_{i,j;d(i,j)}) \tag{7}$$

where $d(i, j)$ is the conditioning set excluding variable i, j. The corresponding conditioned set is i, j. The small absolute values of partial correlation in the 'Best' canonical vine, which is less than the significance value τ, is set to zero. Finally, the optimal canonical vine structure based on partial correlation is built. Since the canonical vine based on conditional vine has a similar structure, we can construct the optimal canonical vine based conditional copula by using the structure based on partial correlation.

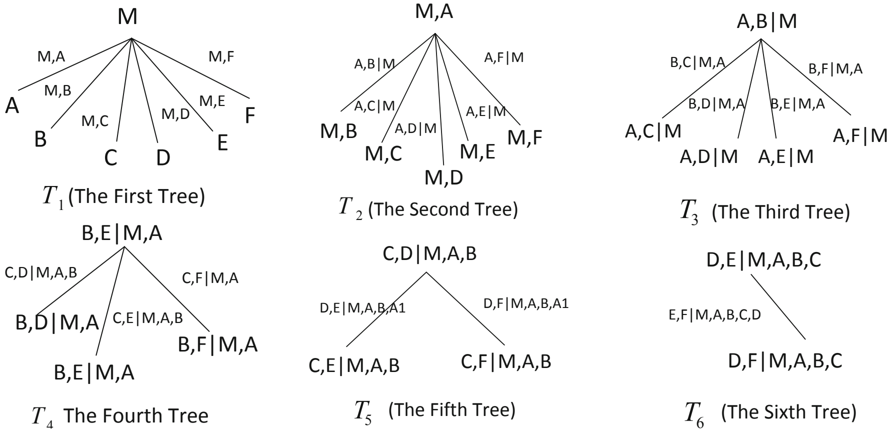


Fig. 1. Canonical Vine Trees

3.3 Marginal Models Specification

The second step of constructing joint probability density function is to build the marginal distribution for each asset return. In the paper, we choose the AR(1)-Garch(1,1) as the marginal distribution. Hansen and Lunde [6] provided evidence that it is difficult to find a volatility model which outperforms the Garch(1,1) model. We estimate the AR(1)-Garch(1,1) with skewed student t innovations. The reason is that the skewed student t innovations can capture and model the characteristics of financial asset return, such as volatility clustering and fat tail.

3.4 Parameter Estimation

We use a two-step procedure to estimate the canonical vine copula and marginal distributions. Taking a log of both sides in Equation 2, we can obtain:

$$\log f(x_1, ..., x_n) = \sum_{i=1}^{n} \log f_i + \log c[F_1(x_1), ..., F_n(x_n)] \tag{8}$$

The joint log-likelihood is equal to the sum of the marginal log-likelihoods and the canonical vine copula log-likelihood. Parameters can be estimated separately by optimizing the marginal log-likelihood and canonical vine copula log-likelihood in two steps.

3.5 Evaluation

In the paper, we use the final amount of the portfolio, the utility obtained by daily returns, to compare the assets allocation with different portfolio decisions. The final amount is the amount dollars obtained at the end of entire out of sample period (testing period). To compare the utility, we use opportunity cost, also called management fee or forecast premium, which is the amount that investor would pay to switch from the the equally divided portfolio to analyzed allocation. The benchmark of assets allocation is by equally divided portfolio, which means the weights of all assets are equal. We compare the performance of canonical vine with mean-variance criterion. Suppose that r_{port} is the optimal portfolio return obtained by canonical vine or mean-variance, and r^*_{port} is the return obtained from the equally divided portfolio. In other word, the opportunity cost is actually return which is added to the return obtained from equally divided portfolio, to make sure the investor be indifferent to the returns obtained from the analyzed model. Then, the opportunity cost Δ can be defined as:

$$U(1 + r_{port} + \Delta) = U(1 + r^*_{port}) \tag{9}$$

Equation (9) can be resolved via the Taylor approximation with CRRA utility function in [7].

4 Asymmetry Analysis with Case Study

4.1 Data and Model Specification

We used two financial asset portfolios in the evaluation of the performance of the newly proposed model. One portfolio composes a comprehensive index $S\&P$ 500, and 50 stocks from 10 industries. The other portfolio consists of a comprehensive index $Stoxx50$ Euro, five national leading stock indices, and 44 stocks. All the data are downloaded from Yahoo Finance (http://finance.yahoo.com). The data in the both portfolios span 1200 trading days from 01/10/2004 to 31/07/2009. In the data pre-processing step, these returns and indices are calculated by taking the log difference of the prices on every two consecutive trading prices.

As described in Section 3.3, AR(1)-Garch(1,1) is considered as the marginal distribution to capture the skewness. The Ljung Box Q test was used for checking the existence of residual autocorrelation for all of the stocks and indices. If the marginal distribution of any stock or index fails the Q test, we increased the value of p in the AR(p)-Garch(1,1) model until all pass the Q test. The results of the Ljung Box Q test are not listed in the paper due to page limit.

For the canonical vine, we choose the 'best' canonical vine that maximizes the value of function $-log(D)$. The root variables in T_1 are the comprehensive indices $S\&P500$ and $Stoxx50E$. This is reasonable as the comprehensive index has much stronger correlation than other stock prices very often. In the optimization of the canonical vine, we considered to use different values of τ (significance values), and compared with the 'unprune' canonical vine. The comparison between canonical vines are based on function $-log(D)$, where D is the determinant as mentioned in Section 3.2. The function is used to calculate the determinant of the partial correlation based canonical vine, and it can be then used for comparing the similarity of vine structures [9]. In the paper, all the nodes in the canonical vine are assigned to bivariate t copulas.

Table 2 shows all of the partial correlations in the 'unprune' canonical vine. All of these partial correlations are used with absolute value as our study is focused on the extent of correlations rather than positive/negative correlations. It can be seen that $Stoxx50E$ has a value larger than $S\&P500$ at all levels. It indicates that the stocks in the portfolio of $Stoxx50E$ has much stronger correlations than those in the portfolio of $S\&P500$. This is understandable as the portfolio of $S\&P500$ is built by using 500 stocks from 10 industries. The stocks in the portfolio $S\&P500$ is strictly selected from the least correlations the each other. The result shown in Table 1 implies a similar conclusion. Table 1 shows the determinant and number of nodes under various values of τ. The value of function $-log(D)$ means the strength of canonical vine. It is observed that the portfolio of $S\&P500$ has more number of nodes in 'unprune' canonical vine than the portfolio of $Stoxx50E$ as $S\&P500$ has 51 variables, compared with 50 variables in $Stoxx50E$. However, the portfolio of $Stoxx50E$ has more nodes than the portfolio of $S\&P500$ in each corresponding level of τ. It suggests that $Stoxx50E$ has stronger correlations. Under the case of the optimal canonical vine ($\tau = 0.1$), for the both portfolios, the optimal canonical vine has less number of nodes and parameters, namely only one tenth of those of the 'unprune' canonical vine. It means that the one tenth nodes can contribute the majority of the dependence structure. Other nodes contribute a little in the dependence structure.

4.2 Experiment Results and Analysis

The performance of our model was evaluated by measuring the opportunity cost. A moving window of 1200 observations, approximately 5 years of daily returns from 01/10/2004 to 31/07/2009, was used to construct the model. The test period was from 01/08/2010 to 01/03/2012 with 730 observations of daily returns. We evaluated the performance of our model with the two portfolios: the European stock markets $Stoxx50E$ and United Stated stock markets $S\$P500$. All the portfolio decisions are re-balanced at the end of every month, and no cost is assumed for the re-balancing. We considered to compare the performance of our model with the mean variance criterion and the equally divided allocation to understand whether our model is useful.

Table 3 shows the results related to the opportunity costs and the final amounts for the two portfolios $S\$P500$ and $Stoxx50E$. Table 3 provides strong evidence that our canonical vine based model is the best at all levels γ for both portfolios. In detail, for the portfolio $S\&P500$, we compare the two canonical vines with $\tau = 0.1$ and 0.05. There is no obvious difference between these two canonical vines, indicating that the two canonical vines implement a similar forecasting of the samples. However, the number of parameters in the canonical vine (τ=0.1) is only half of those in the canonical vine ($\tau = 0.05$). The canonical vine($\tau = 0.1$) is sufficient to model the dependence structure.

The mean variance criterion has a poor performance as the opportunity cost is negative at all of the levels γ. It indicates that if investors conduct assets allocation on a basis of mean variance analysis, the final profit would be less than those on the basis of equally divided allocation. Therefore, the mean variance criterion is not useful. Considering the good performances of canonical vine model, the mean variance criterion, which is caused by the normal distributions, cannot catch the features of asymmetry and skewness of these stocks and indices. For the portfolio $Stoxx50E$, the mean variance criterion is not useful either. The performance by the mean variance criterion in $Stoxx50E$ is worse than those in $S\&P500$. The final amount is even less than one, suggesting that investors will obtain loss if they allocate the assets under the mean variance criterion. The trouble to the mean variance criterion is that $Stoxx50E$ has stronger asymmetry and skewness than those in $S\&P500$. However, the two canonical vines in $Stoxx50E$ perform better than those in $S\&P500$ as the opportunity cost is large at all level γ.

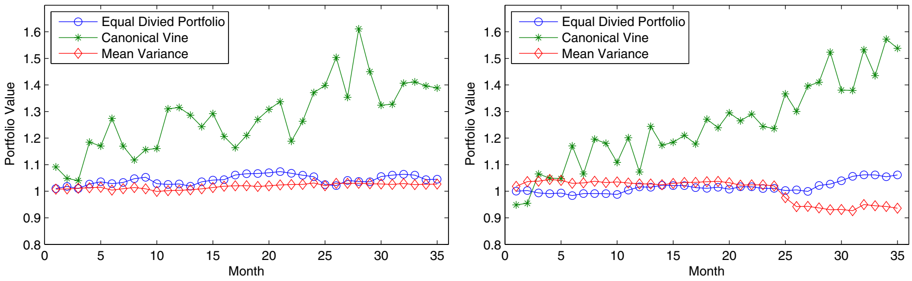


Fig. 2. Portfolio values over 35 months for the canonical vine ($\tau = 0.1$), mean variance, and equally divided allocation, $\gamma = 2$. The left one is for $S\&P500$, and the right one is for $Stoxx50E$.

Table 1. Determinants and Numbers of Nodes for Portfolio $S\&P500$ and $Stoxx50E$

τ	unprune	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10	0.20
$Stoxx50E$												
$-log(D)$	28.36	28.35	28.31	28.24	28.09	27.87	27.66	27.39	27.15	26.92	26.46	22.74
No. nodes	1225	1059	875	758	638	526	456	392	349	317	267	85
$S\&P500$												
$-log(D)$	13.22	13.21	13.16	13.05	12.91	12.70	12.41	12.09	11.81	11.45	11.16	9.44
No. nodes	1275	1044	847	678	560	457	362	285	235	186	153	62

Table 2. Partial Correlations in Canonical Vine for Portfolios $S\&P500$ and $Stoxx50E$

	Min	Max	Mean	25%Quantile	50%Quantile	75%Quantile
$S\&P500$	0.0002	0.5281	0.0572	0.0144	0.0340	0.0641
$Stoxx50E$	0.0004	0.8702	0.0787	0.0171	0.0422	0.0929

Table 3. Opportunity Costs and Final Amounts for $S\&P500$ and $Stoxx50E$

	$S\&P500$				$Stoxx50E$			
Relative Risk Aversion (γ)	2	5	7	10	2	5	7	10
Opportunity Cost								
Canonical Vine ($\tau = 0.1$)	1.33%	3.21%	5.64%	7.32%	1.91%	4.19%	6.14%	10.01%
Canonical Vine ($\tau = 0.05$)	1.31%	3.18%	5.56%	7.21	1.89%	4.17%	6.12%	9.95%
Mean Variance Criterion	-2.36%	-3.62%	-1.24%	-1.19%	-2.55%	-6.52%	-3.20%	-1.40%
Final Amount								
Canonical Vine ($\tau = 0.1$)	1.39	1.41	1.46	1.50	1.53	1.58	1.62	1.64
Canonical Vine ($\tau = 0.05$)	1.40	1.42	1.47	1.50	1.54	1.60	1.62	1.65
Mean Variance Criterion	1.02	1.00	1.04	1.04	0.93	0.80	0.91	0.99
Equally Divided Allocation	1.05	1.05	1.05	1.05	1.06	1.06	1.06	1.06

Figure 2 shows the portfolio values obtained from our canonical vine, the mean variance criterion and equally divided allocations at the end of each month. The trend obtained from our canonical vine in $Stoxx50E$ shows stronger increasing trends and less volatility than those in $S\&P500$. It indicates that the canonical vine has a better performance in $Stoxx50E$. For the trend obtained from the mean variance criterion, there is no obviously different from the trends from the equally divided allocation in $S\&P500$. However, the trend of mean variance is worse than those of the equally divided allocation. We can find that the mean variance criterion performs a bit better in $S\&P500$ than in$Stoxx50E$.

Overall, we can see that: (i) The new canonical vine has a better performance in high dimensional assets portfolios of strong asymmetry and skewness; (ii) The mean variance criterion does not have a good performance in high dimensional assets portfolio that has asymmetry and skewness; and (iii) Compared with the equally divided allocation, our new canonical vine has a better performance. However, the performance by the mean variance criterion has no obvious difference, or even worse if the high dimensional assets portfolio has strong asymmetry and skewness.

5 Conclusion

The paper proposed a canonical vine based model to optimize the asset allocations of high dimensional assets. To address the touch computational issues caused by high dimensional assets, we employed the partial correlation technique to reduce the complexity of the dependence structure to make invertors understand the model easily. Our experimental results and analysis have shown that the canonical vine model has a better performance for portfolios of strong asymmetry and skewness in comparison to the mean variance criterion, a current widely used method. As a future work, we will extend partial correlation based canonical vine to work with more copula families.

Acknowledgments. This work is sponsored in part by Australian Research Council Discovery Grant (DP1096218, DP130102691) and ARC Linkage Grant (LP100200774).

References

1. Aas, K., Berg, D.: Models for construction of multivariate dependence–a comparison study. The European Journal of Finance 15(7-8), 639–659 (2009)
2. Ang, A., Bekaert, G.: International asset allocation with regime shifts. Review of Financial Studies 15(4), 1137–1187 (2002)
3. Ang, A., Chen, J.: Asymmetric correlations of equity portfolios. Journal of Financial Economics 63(3), 443–494 (2002)
4. Arrow, K.: Essays in the theory of risk-bearing, pp. 90–120. Markham Publishing, Chicago (1971)
5. Campbell, R., Koedijk, K., Kofman, P.: Increased correlation in bear markets. Financial Analysts Journal, 87–94 (2002)
6. Hansen, B.E.: Autoregressive conditional density estimation. International Economic Review 35(3), 705–730 (1994)
7. Jondeau, E., Rockinger, M.: Optimal portfolio allocation under higher moments. European Financial Management 12(1), 29–55 (2006)
8. Kurowicka, D., Cooke, R.: Uncertainty analysis with high dimensional dependence modelling. John Wiley & Sons (2006)
9. Kurowicka, D., Cooke, R., Callies, U.: Vines inference. Brazilian Journal of Probability and Statistics 20, 103–120 (2006)
10. Markowitz, H.: Portfolio selection. The Journal of Finance 7(1), 77–91 (2012)
11. Nelsen, R.: An introduction to copulas. Springer (2006)
12. Patton, A.: On the out-of-sample importance of skewness and asymmetric dependence for asset allocation. Journal of Financial Econometrics 2(1), 130–168 (2004)
13. Riccetti, L.: A copula–garch model for macro asset allocation of a portfolio with commodities. Empirical Economics, 1–22 (2012)
14. Sklar, A.: Fonctions de répartition à n dimensions et leurs marges. Publ. Inst. Statist. Univ. Paris 8(1) (1959)
15. Sun, W., Rachev, S., Stoyanov, S., Fabozzi, F.: Multivariate skewed student's t copula in the analysis of nonlinear and asymmetric dependence in the german equity market. Studies in Nonlinear Dynamics & Econometrics 12(2), 42–76 (2008)

Inducing Context Gazetteers from Encyclopedic Databases for Named Entity Recognition

Han-Cheol Cho[1], Naoaki Okazaki[2,3], and Kentaro Inui[2]

[1] Suda Lab., Graduate School of Information Science and Technology,
the University of Tokyo, Tokyo, Japan

[2] Inui and Okazaki Lab., Graduate School of Information Science,
Tohoku University, Sendai, Japan

[3] Japan Science and Technology Agency (JST)

hccho@is.s.u-tokyo.ac.jp

{okazaki,inui}@ecei.tohoku.ac.jp

Abstract. Named entity recognition (NER) is a fundamental task for mining valuable information from unstructured and semi-structured texts. State-of-the-art NER models mostly employ a supervised machine learning approach that heavily depends on local contexts. However, results of recent research have demonstrated that non-local contexts at the sentence or document level can help advance the improvement of recognition performance. As described in this paper, we propose the use of a context gazetteer, the list of contexts with which entity names can co-occur, as new non-local context information. We build a context gazetteer from an encyclopedic database because manually annotated data are often too few to extract rich and sophisticated context patterns. In addition, dependency path is used as sentence level non-local context to capture more syntactically related contexts to entity mentions than linear context in traditional NER. In the discussion of experimentation used for this study, we build a context gazetteer of gene names and apply it for a biomedical NER task. High confidence context patterns appear in various forms. Some are similar to a predicate–argument structure whereas some are in unexpected forms. The experiment results show that the proposed model using both entity and context gazetteers improves both precision and recall over a strong baseline model, and therefore the usefulness of the context gazetteer.

1 Introduction

Named entity recognition (NER) is a task that recognizes the mentions of entities of interest. Entity types vary depending on the target domains. In the general domain, for example, the names of people, locations and organizations are most common entity types [5,25], whereas the names of genes and gene products are in the biomedical domain [12,22]. In fact, NER has been regarded as a fundamental sub-task in many natural language processing (NLP) applications such as information extraction, question and answering, and machine translation.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 378–389, 2013.

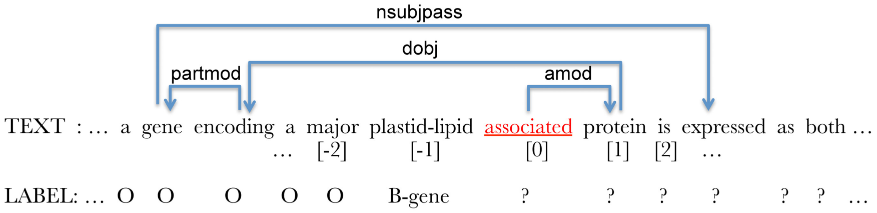

nsubjpass
dobj
partmod
amod
TEXT : … a gene encoding a major plastid-lipid associated protein is expressed as both …
… [-2] [-1] [0] [1] [2] …
LABEL: … O O O O O B-gene ? ? ? ? ? ? …

Fig. 1. Example of local and non-local context window. The local context window [-2,2] is shown under the text, whereas the non-local context window is shown with directed arrows. "plastid-lipid associated protein" is the name of a gene where the first word is labeled with the *B-gene* meaning that this is the beginning of a gene name. The dependency label *amod* stands for adjectival modifier, *dobj* for direct object, *partmod* for participial modifier and *nsubjpass* for the passive nominal subject.

NER has been tackled in various ways from rule-based to statistical approaches. However, most state-of-the-art NER models formalize it as a sequence labeling task and employ supervised machine learning approaches such as Conditional Random Fields (CRF) and Support Vector Machines (SVMs). To achieve high-performance, a supervised machine learning approach requires a set of features that are well designed to distinguish mentions of entities from others. Commonly used features are local features obtained from a small and linear window (local context hereinafter). For example, presuming that we shall determine the label of the underlined word "associated" in Fig. 1, then the neighboring and current words such as "major", "plastid-lipid", "associated", "protein" and "is" within the local context [-2,2] are useful as word uni-gram features (the relative position of each word is shown under the word). These local features contribute to production of strong baseline models [2,8,20]. However, recent studies [4,13] have demonstrated that incorporating features from non-local context can help further improve the recognition performance. In Fig. 1, for instance, direct and indirect head-words of the word "associated" such as "protein", "encoding", "gene", and "expressed" can be useful non-local features.

As described in this paper, we propose to use a context gazetteer, which is a list of contexts that co-occur with entity names, for incorporating new sentence level non-local features into NER model. A context gazetteer consists of dependency paths of variable lengths to capture more syntactically meaningful contexts than traditional local contexts. Confidence values are assigned to these contexts to reflect how they are likely to appear with entity names. We build a context gazetteer from a huge amount of highly precise and automatically labeled data using an encyclopedic database because manually annotated data are often too few to extract rich and sophisticated context patterns. Therefore, a context gazetteer is expected to help recognize unknown entity names that do not appear in training data, in addition to out-of-vocabulary (OOV) entity names that are not registered in entity gazetteers. In experiment, we build a context gazetteer of gene names and apply it for a biomedical named entity recognition task. It is particularly interesting that top-ranked entries in the created context gazetteer have various forms. As expected, there are many predicate–argument structure style contexts using domain specific verbal (and nominal) predicates such as “express”, “inhibit” and “promote.” Moreover, abbreviation, apposition, and conjunction dependencies are frequently included as a part of high confidence context patterns. These contexts can be interpreted as fragments of domain knowledge that appear in stereotypical syntactic structures in texts. The context gazetteer boosted both the precision from 89.06 to 89.32 and the recall from 82.78 to 83.46. As a consequence, the overall F1-score is improved from 85.81 to 86.29.

The remainder of this paper is organized as follows. In Sec. 2, we explain work related to our research. Section 3 describes the proposed method for creating a context gazetteer. In the next section, we build a context gazetteer of gene names from the EntrezGene database [16], and apply it to the BioCreative 2 gene name recognition task [22]. The usefulness of a context gazetteer is demonstrated experimentally. Representative output results are analyzed. We show what kinds of context patterns are mined and how they affect a proposed model using the context gazetteer. Section 5 summarizes the contributions of this work, and explains the future work for generalizing learned contexts.

2 Related Work

This section presents a summary of three types of related studies of sentence level non-local features, gazetteer induction and weakly supervised learning.

Sentence level non-local features usually depend on a deep parsing technique. For example, a previous work [7] used the Stanford dependency parser [17] to exploit features such as the head and governor of the noun phrases in a biomedical NER task. A more recent work [23] evaluated the effect of seven different parsers in feature generation for finding base noun phrases including gene names. However, they extract contexts only from training data, whereas we use a large amount of automatically annotated data. As a result, our approach is likely to provide richer and more sophisticated context patterns than their methods.

Gazetteers are invaluable resources for NER tasks, especially for dealing with unknown words that do not appear in training data. They might have the same semantic categories to target entity classes [9], or related classes that are often more fine-grained sub-classes of the target entity classes [20,26]. Word clusters are also useful resources for NER similar to gazetteers. In a related study [18], the Brown clustering algorithm [3] were applied to NER successfully. A more recent work [11] used the dependency relations between verbs and multiword nouns for clustering multiword expressions. However, to the best of our knowledge, all of the related work that we have surveyed produce entity gazetteers (clusters).

The most similar concept to the contexts in this research can be found in the studies related to weakly supervised learning approach. For instance, a bootstrapping method [21] extracts context patterns from unlabeled data using a small set of seed words (entity mentions in case of NER) for a target class. In turn, it extracts new entity mentions using the extracted context patterns, and repeats this process. However, the quality of context patterns (and also entity mentions) degrades as iteration goes on because it inevitably suffers from semantic drift. In contrast, our method induces a large number of highly precise contexts without a repetitive process by exploiting an encyclopedic database. This approach have become more realistic lately because of many publicly available resources such as Wikipedia[1] and domain-specific databases.

3 Building a Context Gazetteer

A context gazetteer is a confidence assigned list of dependency paths (hereinafter, contexts) of variable length that can co-occur with target entity names. Figure 2 portrays an exemplary context of length 3. It is a high confidence context in the context gazetteer of gene names that will be used in the experiment section. It means that a word X surrounded by the context consisting of the head word *expression*, a dependent *cells* and a grand-dependent *cancer* with the corresponding dependency relations *prep_of*, *prep_in* and *nn* is likely to be an entity word, which is a part of a target entity name. This context can help to recognize the headword of an underlined gene name in a sentence, “The *expression* of FasL in gastric *cancer cells* and of Fas in apoptotic TIL was also detected in vivo.”

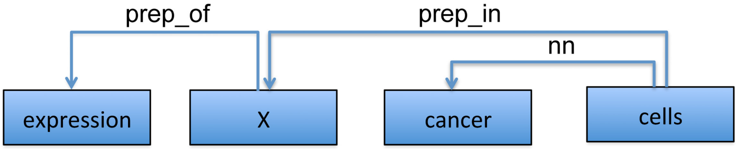


Fig. 2. Example context of the length 3. X is a slot for an entity word. (*pref_of* stands for prepositional modifier of, *pref_in* for prepositional modifier in and *nn* for noun compound modifier.)

A useful context gazetteer should have rich and sophisticated contexts that are specific to target semantic classes. For the first requirement, we extract contexts from a large amount of automatically labeled data rather than a few manually annotated data. To satisfy the second requirement, confidence values are assigned to the extracted contexts. Figure 3 is the flowchart for the context gazetteer generation. Each step is explained in detail in the following.

Step 1. An encyclopedic database consists of domain specific entity names and their descriptions. For each entity name, we label every mention of it in the description using exact string matching. The primary reason for using an encyclopedic database rather than the list of target entity names and some free texts is to

[1] http://www.wikipedia.org/

remove the ambiguity of the semantic categories of target entity names appearing in free texts [29]. For example, presuming that we are going to generate labeled data with the names of people using some free text (e.g. newspapers) and a list of the names of people automatically, the process would invariably create very noisy data because human names are often used as the names of companies (e.g., Hewlett-Packard and Ford Motor Company), diseases (e.g. Alzheimer disease), places (e.g., Washington, D.C and St. Paul, Minnesota), and so on.

Step 2. The labeled texts are then parsed. The dependency paths (contexts) involving entity words are extracted. Because of the excessive number of possible contexts, we applied two constraints to context generation. First, the contexts that have no content words (nouns, verbs and adjectives) except an entity word are removed because these contexts are often too general to be effective contexts. Second, we limit the maximum length of contexts depending on the data size.

Step 3. For each context, an entity word is anonymized. Then, contexts can be normalized to increase the coverage of a context gazetteer. For example, stems (or lemmas) are useful instead of words. After normalization, we remove duplicated contexts and keep them unique.

Step 4. Contexts are often ambiguous even if they frequently appear with target entity names. We solve this problem by assigning confidence to each context. Presuming that data D is annotated automatically with the mentions of T different entity types[2], then, the confidence (conditional probability) of an entity type t given a context c is defined as in

$$confidence(t|c) = p(t|c) = \frac{C(t,c)}{C(c)} = \frac{\sum_{e_t \in D} C(e_t, c)}{C(c)}, \qquad (1)$$

where e_t is an entity word of the semantic type $t \in T$ in the data D. The estimated confidence is pessimistic, meaning that they are usually lower than they should be because automatically annotated data have high precision but low recall.

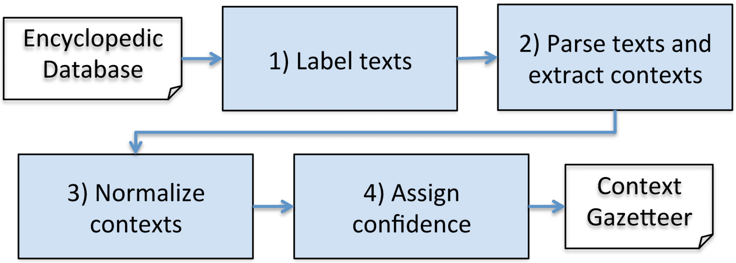


Fig. 3. Building a context gazetteer from an encyclopedic database

[2] The set T includes non-entity type O too.

4 Evaluation

In this section, we create a context gazetteer of gene names from the EntrezGene database [16], and apply it to the BioCreative 2 gene name recognition task [22]. We analyze the effect of the context gazetteer by comparing the NER models with and without the context gazetteer.

4.1 Data

Context Gazetteer. For gazetteer generation, we use the gene names (including synonyms) and the human curated reference information in the EntrezGene. At the first step in Fig. 3, 358,049 abstracts are extracted from the MEDLINE database[3] using reference information. Each abstract is labeled using the gene names referenced in the abstract. The labeled gene names are highly precise because explicit references exist between the gene names and the abstracts.

Second, the labeled texts are parsed using the Stanford POS tagger [27] and dependency parser [17] included in the CoreNLP tool[4]. Then, we extracted the dependency paths (contexts) that involve entity words. Contexts that have no content words aside from entity words are filtered out. The maximum length is set to 5 experimentally.

Third, the entity words of the contexts are anonymized. In the biomedical domain, many entity names include symbols and numbers. For domain-specific normalization, continuous numbers and symbols of the words in the contexts are converted into a representative number (0) and symbol (under-bar), respectively. Lastly, confidence values are assigned to each context using Eq. 1. Contexts appearing less than 10 times are removed in this process because the estimated confidence might be unreliable.

Several extracted contexts having high confidence are presented in Table 1. At the beginning of this study, we expected to obtain contexts similar to predicate-argument structure (PAS) and domain specific relations. For example, the second context in this table indicates that X is likely to be a gene if it appears in a relation with *C-jun* as in "... interaction between X and C-Jun". The fourth and seventh contexts are in the form of PAS using nominal and verbal predicates respectively. However, we also found unexpected but interesting contexts too. First, many contexts capture factual knowledge. The first and fifth contexts are the simplest ones meaning that X is likely to be a gene if it is a *globin* or a *repressor*. The sixth context means X is likely to be a gene if it acts as a *mediator*. Second, some contexts represent procedural information. The third context, for instance, indicates that there is a screening process for analyzing mutations of a gene. Lastly, the eighth context, seemingly uninformative at first glance, means that discovering the function of a gene is a common task as in "The exact function of IP-30 is not yet known, but it may play a role ..."

[3] MEDLINE is the U.S. National Library of Medicine's (NLM) premier bibliographic database.

[4] `http://nlp.stanford.edu/software/corenlp.shtml`

Table 1. Examples of high confidence extracted context patterns. Conf. stands for confidence. (X is a place-holder, *nsubj* is nominal subject, *conj_and* is conjunction and, *nn* is noun compound modifier, *amod* is adjectival modifier, *dobj* is direct object, and *nsubjpass* is passive nominal subject.)

Conf.	Pattern
1.0	nsubj(globin, X)
1.0	prep_between(interaction, X), conj_and(X, C-Jun)
1.0	prep_for(screened, mutations), prep_of(mutations, gene), nn(gene, X)
0.91	prep_of(secretion, X), amod(X, inhibitory)
0.81	nsubj(repressor, X)
0.78	prep_as(X, mediator)
0.65	dobj(express, X)
0.55	nsubjpass(known, function), prep_of(function, X)

Entity Gazetteer. We use four entity gazetteers compiled from the EntrezGene, Universal Protein Resource (UniProt) [6], Unified Medical Language System (UMLS) [1] and the Open Biological and Biomedical Ontologies (OBO)[5]. For improving the coverage of these gazetteers, continuous numbers and symbols of the entity names are normalized into a representative number and symbol (0 for numbers and under-bar for symbols), and all alphabet characters are lowercased. This process also applies to the input texts.

For the entity gazetteers compiled from the EntrezGene and the UniProt, we use the single semantic categories: gene and protein. However, the UMLS and the OBO gazetteers have multiple categories, some of which are related to gene names such as peptides and amino acids, but many of which are different biomedical entity categories. During NER system development, we found that not only gene-related categories but also other categories are beneficial for increasing performance.

GENETAG corpus. The BioCreative 2 gene mention recognition task uses the GENETAG corpus [24] comprising 20,000 sentences, of which 15,000 sentences were used for training and 5,000 sentences were used for testing.

We processed raw texts to obtain additional syntactic information for use in feature generation. Raw texts consisting of sentences are split into tokens using a fine-grained tokenization scheme that uses whitespace and non-alphanumeric characters as token boundary markers. When a string is tokenized at non-alphanumeric character, this character also becomes a single character token (e.g., “p53-activated” to “p53”, “-” and “activated”). Next, the tokenized text is fed to the GENIA tagger [28] for lemmatization, POS-tagging, and chunking. For each entity gazetteer, the sequences of tokens that appear in the gazetteer are tagged using the BIO labels (e.g., “EntityGaz_B-EntrezGene”, “EntityGaz_B-UniProt”, etc.). Lastly, for the EntrezGene context gazetteer, the tokens

[5] `http://www.obofoundry.org/`

Table 2. Features used for experiments. Ortho. stands for orthographical features, E. gaz. for entity gazetteer and C. gaz. for context gazetteer.

Class	Description
Token	$\{w_{t-2}, .., w_{t+2}\} \wedge y_t$, $\{w_{t-2,t-1}, .., w_{t+1,t+2}\} \wedge y_t$, $\{\bar{w}_{t-2}, .., \bar{w}_{t+2}\} \wedge y_t$ $\{\bar{w}_{t-2,t-1}, .., \bar{w}_{t+1,t+2}\} \wedge y_t$,
Lemma	$\{l_{t-2}, .., l_{t+2}\} \wedge y_t$, $\{l_{t-2,t-1}, .., l_{t+1,t+2}\} \wedge y_t$, $\{\bar{l}_{t-2}, .., \bar{l}_{t+2}\} \wedge y_t$, $\{\bar{l}_{t-2,t-1}, .., \bar{l}_{t+1,t+2}\} \wedge y_t$
POS	$\{p_{t-2}, .., p_{t+2}\} \wedge y_t$, $\{p_{t-2,t-1}, .., p_{t+1,t+2}\} \wedge y_t$,
Lemma & POS	$\{l_{t-2}p_{t-2}, .., l_{t+2}p_{t+2}\} \wedge y_t$, $\{l_{t-2,t-1}p_{t-2,t-1}, .., l_{t+1,t+2}p_{t+1,t+2}\} \wedge y_t$
Chunk	$\{c_t, w_{t\_last}, \bar{w}_{t\_last}, the_{lhs}\} \wedge y_t$
Character	Character 2,3,4-grams of w_t
Ortho.	All capitalized, all numbers, contain Greek letters, ... (Refer to [15] for the detailed explanation)
E. gaz.	$\{ge_{t-2}, .., ge_{t+2}\} \wedge y_t$, $\{ge_{t-2,t-1}, .., ge_{t+1,t+2}\} \wedge y_t$, $\{ge_{t-2}l_{t-2}, .., ge_{t+2}l_{t+2}\} \wedge y_t$, $\{ge_{t-2,t-1}l_{t-2,t-1}, .., ge_{t+1,t+2}l_{t+1,t+2}\} \wedge y_t$,
C. gaz.	$\{gc_t \wedge y_t\}$

surrounded by the contexts of the gazetteer are tagged with context gazetteer class label. The confidence of a context is quantized at every 0.1 step. For example, if a token is surrounded by two contexts with the confidence 0.31 and 0.56, then we assign two labels to the token, "ContextGaz_EntrezGene_3" and "ContextGaz_EntrezGene_6", where the confidence is rounded up.

4.2 Machine Learning and Features

For machine learning, we use the CRFsuite [19], which implements first-order linear-chain Conditional Random Fields [14]. The regularization parameter (C) is optimized using the first 90% of the original training data as training data and the rest, 10% as the development data. Fifteen C values (0.03125, 0.0625, 0.125, 0.25, 0.5, 0.75, 1, 2, 3, 4, 5, 6, 8, 10, and 16) are tested. The best performing one is chosen.

A set of features used in the experiment is presented in Table 2, and the symbols are explained in Table 3.

4.3 Experiment Results

Table 4 shows an experiment result obtained using various combinations of the four entity gazetteers and the context gazetteer. The numbers in a pair of parentheses show improvement from the model 0 (baseline model) using no gazetteers.

When the context gazetteer is used in combination with the entity gazetteer(s), both precision and recall increase, as shown in models 3 and 5. Considering that precision and recall are tradeoff measures, the experiment result demonstrates the usefulness of the context gazetteer. In addition, the context gazetteer improves recall notably. This is an important merit because NER models usually

Table 3. Symbols used for features (see Table 2)

Symbol	Description
w_t	A *t-th* word
$\bar{w}_t$	A normalized *t-th* word where successive numbers and symbols are converted into a single zero and under-bar
l_t	A *t-th* lemma
$\bar{l}_t$	A normalized *t-th* lemma
p_t	A *t-th* POS-tag
c_t	The chunk type of w_t
w_{t_last}	The last word of a current chunk
$\bar{w}_{t\_last}$	The normalized last word of a current chunk
the_{lhs}	True if 'the' exists from the beginning of a current chunk to w_{t-1}
ge_t	Entity gazetteer label for the *t-th* word
gc_t	Context gazetteer label for the *t-th* word

Table 4. Performance evaluation using entity and context gazetteers. ALL means the gazetteers compiled from the EntrezGene, UniProt, UMLS, and OBO databases.

Model #	Entity Gaz.	Context Gaz.	Precision	Recall	F1-score
0	None	None	87.99 (+0.00)	81.71 (+0.00)	84.73 (+0.00)
1	None	EntrezGene	88.06 (+0.07)	81.42 (-0.29)	84.61 (-0.12)
2	EntrezGene	None	88.54 (+0.55)	82.17 (+0.46)	85.24 (+0.51)
3	EntrezGene	EntrezGene	88.66 (+0.67)	82.99 (+1.28)	85.73 (+1.00)
4	ALL	None	89.06 (+1.07)	82.78 (+1.07)	85.81 (+1.08)
5	ALL	EntrezGene	89.32 (+1.33)	83.46 (+1.75)	86.29 (+1.56)

exhibit high precision but low recall [10] because of the asymmetric data where one class label, O, dominates all other classes.

Surprisingly, when only the context gazetteer is used, the overall performance drops slightly. We suspect that some relation exists between entity gazetteers and context gazetteers but further investigation is necessary to reveal it.

4.4 Result Analysis

We manually compared about 20% of the output of models 4 and 5 to see how the context gazetteer features affect the tagging results.

There are 32 gene names correctly recognized by model 5 but not by model 4. In all of these cases, one or more context gazetteer features are triggered. The following list shows several examples in which model 5 recognized the under-barred gene names and model 4 recognized the italicized gene names.

- One major transcript encodes MEQ, a *339-amino-acid bZIP protein* which is homologous to the Jun/Fos family of transcription factors.
- The association of I-92 with *p92*, *p84*, *p75*, *p73*, *p69*, and *p57* was completely reversible after treatment with the detergent deoxycholate (DOC).

- The exact function of IP-30 is not yet known, but it may play a role in *gamma-interferon* mediated immune reactions.

Two context gazetteer features are triggered for the gene name “MEQ”, “dobj(encode, X)”, and “appos(X, protein).” The second feature is a strong evidence of X being a gene name because a word X is in apposition with the word protein. In the second example, “I-92” has a feature “prep_of(association, X), prep_with(X, p0)” meaning that X is likely to be a part of gene name if it is associated with the gene name “p0” where 0 is a normalized number. Contexts of these kinds are the fragments of domain specific knowledge and usually have high confidence (0.5 for this context). In the last example, the gene name “IP-30” has a context gazetteer feature “prep_of(function, X)” and a more specific one “nsubjpass(known, function), prep_of(function, X)” with confidence 0.44 and 0.54. These contexts can be interpreted as domain-specific expressions where figuring out the function of a gene is a much more important task than others (54% vs. the rest).

However, 15 gene names are recognized by model 4, but not by model 5. Context gazetteer features are not triggered for 3 cases. Because we use the words (not stems or lemmas) in the contexts, the coverage might be not sufficiently high. For the other 12 cases, context gazetteer features are fired, but these gene names are not recognized. We are currently investigating the causes of these cases.

5 Conclusion and Future Work

As described in this paper, we proposed the use of a context gazetteer as a new non-local feature for NER. We also described how to induce a rich and sophisticated context gazetteer from automatically annotated data using an encyclopedic database. Compared to the feature aggregation methods [4,13,20], the proposed method can be easily applied to streaming data such as tweets and pre-processed data with sentence selection where recognizing document (or discourse) boundaries is difficult. The proposed method is applied to a biomedical NER task. Its usefulness is demonstrated in addition to entity gazetteers.

However, we also uncovered difficulties. First, for this research, we used words and their dependencies as contexts. However, these contexts sometimes include uninformative words in the middle of contexts. If it is possible to generalize the contexts by replacing these unimportant words with POS-tags or wildcards, then the coverage of the context gazetteer can be enhanced. Second, gene names (or parts of them) often appear as a part of contexts. Although these contexts often have very high confidence, they may not be general patterns. They can be more useful if they were replaced by some general gene name wildcards.

Acknowledgments. This research was partly supported by JST, PRESTO. This research was partly supported by JSPS KAKENHI Grant Numbers 23240018 and 23700159.

References

1. Bodenreider, O.: The unified medical language system (umls): integrating biomedical terminology. Nucleic Acids Research 32(suppl. 1), D267–D270 (2004)
2. Borthwick, A., Sterling, J., Agichtein, E., Grishman, R.: Nyu: Description of the mene named entity system as used in muc-7. In: Proceedings of the Seventh Message Understanding Conference, MUC-7 (1998)
3. Brown, P.F., de Souza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Journal of Computational Linguistics 18(4), 467–479 (1992)
4. Chieu, H.L., Ng, H.T.: Named entity recognition with a maximum entropy approach. In: Proceedings of the Seventh CoNLL at HLT-NAACL 2003, vol. 4, pp. 160–163 (2003)
5. Chinchor, N.A.: Overview of MUC-7/MET-2. In: Proceedings of the Seventh Message Understanding Conference (MUC7) (April 1998)
6. Consortium, T.U.: Reorganizing the protein space at the universal protein resource (uniprot). Nucleic Acids Research 40(D1), D71–D75 (2012)
7. Finkel, J., Dingare, S., Nguyen, H., Nissim, M., Manning, C., Sinclair, G.: Exploiting context for biomedical entity recognition: from syntax to the web. In: Proceedings of the International Joint Workshop on NLPBA, pp. 88–91 (2004)
8. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting on ACL, pp. 363–370 (2005)
9. Florian, R., Ittycheriah, A., Jing, H., Zhang, T.: Named entity recognition through classifier combination. In: Proceedings of the Seventh CoNLL at HLT-NAACL 2003, vol. 4, pp. 168–171 (2003)
10. Kambhatla, N.: Minority vote: at-least-n voting improves recall for extracting relations. In: Proceedings of COLING-ACL, pp. 460–466 (2006)
11. Kazama, J., Torisawa, K.: Inducing Gazetteers for Named Entity Recognition by Large-Scale Clustering of Dependency Relations. In: Proceedings of ACL-HLT, pp. 407–415 (2008)
12. Kim, J.D., Pyysalo, S., Ohta, T., Bossy, R., Nguyen, N., Tsujii, J.: Overview of bionlp shared task 2011. In: Proceedings of the BioNLP Shared Task 2011 Workshop, pp. 1–6 (2011)
13. Krishnan, V., Manning, C.D.: An effective two-stage model for exploiting non-local dependencies in named entity recognition. In: Proceedings of COLING-ACL, pp. 1121–1128 (2006)
14. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289 (2001)
15. Lee, K.J., Hwang, Y.S., Kim, S., Rim, H.C.: Biomedical named entity recognition using two-phase model based on svms. Journal of Biomedical Informatics 37(6), 436–447 (2004)
16. Maglott, D., Ostell, J., Pruitt, K.D., Tatusova, T.: Entrez gene: Gene-centered information at ncbi. Nucleic Acids Research 33(suppl. 1), D54–D58 (2005)
17. Marneffe, M.C.D., MacCartney, B., Manning, C.D.: Generating typed dependency parses from phrase structure parses. In: Proceedings of LREC 2006 (2006)
18. Miller, S., Guinness, J., Zamanian, A.: Name tagging with word clusters and discriminative training. In: Susan Dumais, D.M., Roukos, S. (eds.) Proceedings of HLT-NAACL, May 2-May 7, pp. 337–342 (2004)

19. Okazaki, N.: Crfsuite: A fast implementation of conditional random fields, crfs (2007), `http://www.chokkan.org/software/crfsuite/`
20. Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: Proceedings of the Thirteenth Conference on CoNLL, pp. 147–155 (2009)
21. Riloff, E., Shepherd, J.: A corpus-based approach for building semantic lexicons. In: Proceedings of the Second Conference on EMNLP, pp. 117–124 (1997)
22. Smith, L., Tanabe, L., Ando, R., Kuo, C.J., Chung, I.F., Hsu, C.N., Lin, Y.S., Klinger, R., Friedrich, C., Ganchev, K., Torii, M., Liu, H., Haddow, B., Struble, C., Povinelli, R., Vlachos, A., Baumgartner, W., Hunter, L., Carpenter, B., Tsai, R., Dai, H.J., Liu, F., Chen, Y., Sun, C., Katrenko, S., Adriaans, P., Blaschke, C., Torres, R., Neves, M., Nakov, P., Divoli, A., Mana-Lopez, M., Mata, J., Wilbur, W.J.: Overview of biocreative ii gene mention recognition. Genome Biology 9(suppl. 2), S2 (2008)
23. Smith, L.H., Wilbur, W.J.: Value of parsing as feature generation for gene mention recognition. Journal of Biomedical Informatics 42(5), 895–904 (2009)
24. Tanabe, L., Xie, N., Thom, L., Matten, W., Wilbur, W.J.: Genetag: a tagged corpus for gene/protein named entity recognition. BMC Bioinformatics 6(suppl. 1), S3 (2005)
25. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the conll-2003 shared task: language-independent named entity recognition. In: Proceedings of the Seventh CoNLL at HLT-NAACL 2003, vol. 4, pp. 142–147 (2003)
26. Torisawa, K.: Exploiting wikipedia as external knowledge for named entity recognition. In: Proceedings of the Joint Conference on EMNLP-CoNLL, pp. 798–707 (2007)
27. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of the 2003 Conference of the HLT-NAACL, vol. 1, pp. 173–180 (2003)
28. Tsuruoka, Y., Tsujii, J.: Bidirectional inference with the easiest-first strategy for tagging sequence data. In: Proceedings of the Conference on HLT-EMNLP, pp. 467–474 (2005)
29. Usami, Y., Cho, H.C., Okazaki, N., Tsujii, J.: Automatic acquisition of huge training data for bio-medical named entity recognition. In: Proceedings of BioNLP 2011 Workshop, pp. 65–73 (2011)

An Optimization Method for Proportionally Diversifying Search Results

Lin Wu, Yang Wang, John Shepherd, and Xiang Zhao

School of Computer Science and Engineering
The University of New South Wales, Sydney, Australia
{flinw,wangy,jas,xzhaog}@cse.unsw.edu.au

Abstract. The problem of diversifying search results has attracted much attention, since diverse results can provide non-redundant information and cover multiple query-related topics. However, existing approaches typically assign equal importance to each topic. In this paper, we propose a novel method for diversification: proportionally diversifying search results. Specifically, we study the problem of returning a top-k ranked list where the number of candidates in each topic is proportional to the popularity degree of that topic with respect to the query. We obtain such a top-k proportionally diverse list by maximizing our proposed objective function and we prove that this is an NP-hard problem. We further propose a greedy heuristic to efficiently obtain a good approximate solution. To evaluate the effectiveness of our model, we also propose a novel metric based on the concept of proportionality. Extensive experimental evaluations over our proposed metric as well as standard measures demonstrate the effectiveness and efficiency of our method.

Keywords: Diversity, Optimization, Proportions.

1 Introduction

Diversification models for search results [14,3,7,23,20,22] have attracted much attention since they can effectively identify possible aspects of the query and return documents for each aspect. In many cases, this is more useful than conventional search methods which focus on finding the top-k most relevant documents, often favouring (near) duplicates in the top positions of the ranked list at the expense of topic diversity. Although methods for finding a diverse search result list have been well studied, they primarily address the problem from the perspective of *minimizing redundancy*, and promoting lists that contain documents covering multiple topics. One limitation of these approaches is that they treat each document equally while overlooking the fact that some topics are more popular than others; this can result in too much prominence being given to topics that are unlikely to be interesting to a majority of searchers. Ideally, the number of documents from each topic should reflect the popularity degree of that topic. Consider the case of recommending a set of 10 musical documents in a music recommendation system where two topics are considered (e.g., *rock*

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 390–401, 2013.

and *classical*) with 90% popularity voting for the topic *rock* and 10% for the topic *classical*. For most users, it would be more useful to return a list which included mainly results related to *rock* with less results for *classical* (e.g. 9 rock, 1 classical). Existing approaches to diversification would return roughly equal numbers of results for each topic (i.e. 5 rock, 5 classical), which is less than ideal.

Motivated by this, we aim to better solve the problem of diversification by considering it from a different viewpoint: *proportionally diversifying search results*. Specifically, we study the problem of diversifying the top-k representative search results by respecting the overall popularity degree for each topic; we acheive this by determining the number of representative documents on each topic proportional to the topic's popularity by maximizing a novel objective function. Since the computation of this objective function is NP-hard, the final proportionally representative results are obtained by using an effective greedy heuristic to approximately maximize the objective function.

We evaluate both our method and state-of-the-art approaches by conducting comparison experiments over standard metrics [7,8,6] for diversity based on redundancy penalization, and our proposed metric, which considers proportional diversification.

Our principal contributions are as follows.

- We present a novel method for diversification: proportionally diversifying search results. Specifically, a novel objective function is proposed to obtain the top-k diverse list by considering the popularity degree over each topic.
- We show that finding the optimal diversified top-k results by our objective function is NP-hard. To address that, an efficient greedy heuristic is proposed with good approximation ratio.
- A novel metric for diversity is proposed to verify our technique from the perspective of proportion. To demonstrate the efficiency and effectiveness of our approach, extensive experiments are conducted on a real-world database, which are evaluated by standard metrics and our proposed metric.

The rest of the paper is organized as follows: Related work is briefly reviewed in Section 2. We formulate the problem into a combinatorial optimization problem and show its potential to find a proportionally diverse ranking list in Section 3. We present the objective function and near-optimal algorithm in Section 4. Then, in Section 5, we report the experimental studies. Section 6 provides the conclusion.

2 Related Work

There has been rising interest in incorporating diversity into search results to meet the diverse requirements of users by both covering a large range of topics as well as preserving relevance. Standard diversification techniques [3,2,21,17,16] attempt to form a diverse ranked list by repeatedly selecting documents that are different to the selected items. One of the typical techniques is "Maximal

Marginal Relevance" (MMR) proposed by Carbonell *et al.*[3], where each candidate is iteratively selected by the MMR objective function until a given condition is met. MMR was extensively studied by Rafiei *et al.*[17], who aimed to find the best strategy for ordering the results such that many users would find their relevant documents in the top few slots by formulating a weight vector as facets of the query were discovered.

Other than the work discussed above, there are many recent works studying result diversification [22,20,1,14,19]. For instance, in [22], the authors proposed a random-walk-based approach to encourage diversity by assigning absorbing states to the items that have been selected, which effectively "drags down" the importance of similar unranked states. In a similar vein, a model based on a reinforced random-walk is proposed in [14] to automatically balance the relevance and diversity in the top returned answers. Tong *et al.*[20], propose to address diversity from an optimization viewpoint which considers relevance and diversity optimally. Although the experimental results in [20] show improved performance in terms of diversity, it is still less than ideal in applications where the awareness of proportional popularity is desirable. The work most relevant to our own is proposed in [10], where an election-based method is proposed to address the problem of diversifying searched results proportionally. The method is divided into two phases. First, it diversifies the topics of all candidates by an election-based strategy, and then the final ranked list is yielded by selecting an appropriate number of candidates for each topic. However, this method can lead to some documents in popular topics being irrelevant to the query due to the separation of topic selection and candidate selection. It aims at diversifying the topics of all candidate documents rather than the candidate documents in essence. In contrast, our technique unifies the above phases, and effectively obtains a diverse top-k ranked list taking into account both the popularity degree of each topic and the relevance of each document to the query.

In this paper, we propose a novel objective function where the final top-k proportionally diversifying search results are obtained by achieving the optimal set of the function. To the best of our knowledge, our work is the first to obtain an effective solution for proportionally diversifying search results in an optimizing environment.

3 Problem Formulation

In this section, we formulate a description of the problem of *proportionally diversifying search result* as follows. Let $Q = \{w_1, \ldots, w_n\}$ $(n \geq 1)$ be a set of keywords comprising a query, let $T = \{t_1, \ldots, t_m\}$ be the set of all m topics in the result of Q, and let $\mathcal{U}$ denote the set of all documents. We denote p_i to be the popularity degree of topic $t_i \in T (1 \leq i \leq m)$ in $\mathcal{U}$.

Definition 1. *The ranked list R is a* ***proportional representation*** *of $\mathcal{U}$ iff the number of documents in R within topic $t_i \in T(1 \leq i \leq m)$ is proportional to* *its popularity degree* p_i. *Suppose* $N(i)$ *is the number of candidates from* t_i *in* R, *then we have*

$$\frac{N(i)}{\sum_{j=1}^{m} N(j)} \approx \frac{p_i}{\sum_{j=1}^{m}(p_j)} \tag{1}$$

We normally present the top-k elements of R as the result for query Q; the proportion of documents for each topic in the query result should roughly follow the the popularity degree for that topic Note that Eq.1 shows that the number of candidates for each topic in the final ranked list is not required to exactly match the proportion of the popularity degree for that topic. This is because the relevance between query and each candidate could degenerate if we strictly adhere to the precise proportions (this is demonstrated in section 5).

Example 1. Consider a document collection $\mathcal{U}$ *where we assume that 80% of the documents in* $\mathcal{U}$ *about the "Apple" computer company and 20% are about the fruit "apple". In this case,* $\mathcal{U}$ *is associated with two topics. Let* $R = [R_1, R_2]$ *where* R_1 *denotes the set of documents about "Apple" (the company) and* R_2 *is the set of documents about "apple" (the fruit). In a top-10 ranked result list for the query "apple", we would expect roughly 8 results from* R_1 *and 2 results from* R_2*.*

Challenges. There are two challenges to be solved in our framework. The first challenge is how to design an objective set function where the optimal or near-optimal set can best describe the proportionally diverse ranked list, which is proportionally representative of the document set. The second challenge is developing an effectiveness measure; that is, given a proportionally ranking list, how to quantify its goodness. To solve the above two challenges, we propose a novel objective set function as well as a metric, both of which are shown in section 4.

4 Proportionally Diversifying Search Results

In this section, we first introduce the preliminaries and then describe our novel objective set function to obtain the top-k ranked list proportionally to the popularity degree of each topic.

4.1 Preliminaries

As our diversification algorithm is developed based on the availability of pair-wise similarities between documents, we adopt the personalized PageRank technique to compute the values [11]. Suppose there are n documents in the database and q_i is ith query. We represent q_i by a $n \times 1$ vector $\mathbf{q}_i$ such that $\mathbf{q}_i(i) = 1$ and $\mathbf{q}_i(j) = 0$ $(j \neq i)$. The pair-wise similarities from each document d_j (for $1 \leq j \leq n$) to the query d_i (i.e., q_i) can be precalculated by Eq.(2) below and are denoted by a $n \times 1$ vector $\mathbf{r}_i$.

$$\mathbf{r}_i = c\mathbf{P}'\mathbf{r}_i + (1-c)\mathbf{q}_i \tag{2}$$

$\mathbf{P}$ is the row-normalized adjacency matrix (i.e. $\sum_{j=1}^{n} P(i,j) = 1$) of the similarities, $\mathbf{P}'$ is the transpose of $\mathbf{P}$, c is a damping factor, and $\mathbf{r}_i(j)$ is the similarity of j to i. Note that $\mathbf{r}_i(j)$ is not necessarily equal to $\mathbf{r}_j(i)$. For each pre-computed $n \times 1$ vector $\mathbf{r}_i$, we use $\|\mathbf{r}_i\|$ to denote the sum of all elements in $\mathbf{r}_i$ except $\mathbf{r}_i(i)$; that is, $\|\mathbf{r}_i\| = \sum_{j=1,j\neq i}^{n} \mathbf{r}_i(j)$.

4.2 Objective Function

Given a set R of k documents, we propose to measure the quality of R, regarding the relevance to a given query q_{i_0} and the proportional diversity based on the topic popularity, as follows.

$$g(R) = \sum_{i\in R}(1 - \frac{w_i}{\|\mathbf{r}_i\|} \sum_{j\in R, j\neq i} \mathbf{r}_i(j))\mathbf{r}_{i_0}(i) \tag{3}$$

where $\mathbf{r}_{i_0}(i)$ is a relevance score; the more relevant each individual document d_i is to the query, the higher the value of $g(R)$. Nevertheless, the inclusion of d_i in R is penalized against the pair-wise similarities ($\mathbf{r}_i(j)$) from document d_i to other documents d_j in R; that is, subtract $\frac{w_i}{\|\mathbf{r}_i\|}\sum_{j\in R,j\neq i}\mathbf{r}_i(j)$ $(0 \leq w_i \leq 1)$ where $\mathbf{r}_i(j)$ is large when d_i and d_j have the same topic, which will further reduce the value of $g(R)$. Thus, $g(R)$ is expected to capture simultaneously the high closeness and the great diversity by maximizing its value while confirming the proportionality. Thereby, we aim to efficiently retrieve a set R of k documents such that $g(R)$ is maximized.

The question here is how to proportionally diversify top-k results? We argue that it is implemented by w_i, which indicates the importance of discounting the pair-wise similarity to include d_i in R. Herein, w_i determines the topic to be selected; we call this the *topic coefficient.* In fact, the proportion for the number of documents in each topic is guaranteed by automatically updating the topic coefficient w_i, which manages the possibility of declining d_i provided that many items belonging to the same topic as d_i have already been included. Specifically, we define w_i as

$$w_i = e^{1-\frac{z_i}{u_i+1}} \tag{4}$$

where z_i denotes how many documents that belong to the same topic as d_i have been included in R and u_i is the number of documents with the topic t_i. We assume that z_i is always less then u_i in our setting. It is natural to observe that the larger w_i is, the heavier penalty on document d_i, and it becomes more difficult for d_i to be selected.

We now prove that the problem of maximizing $g(R)$ is NP-hard even when all $w_i = 1$, which is a special case of this optimization problem. To deal with this, we then propose a near-optimal algorithm with a performance guarantee (i.e., accuracy guarantee and time-complexity) regarding a general $g(R)$.

Theorem 1. *The problem of retrieving a set R of k documents to maximize $g(R)$ is NP-hard with respect to k even if all $w_i = 1$ in Eq. (3).*

Proof. We convert the decision problem of maximum clique to a special case of the decision problem of the optimization problem in Theorem 1.

Decision Problem of Maximum Clique (MC)

INSTANCE: Given a graph G with n vertices, and an integer $k \leq \frac{n}{2}$.
QUESTION: Is there a complete subgraph of G with size k?

It is well known that the maximum clique problem is NP-hard [13]; thus, its decision problem, above, is NP-complete regarding k.

Proof of Theorem 1. For each instance (i.e. G and k) in MC, we construct $\mathbf{r}$ as follows. Suppose that the query has label 0 and each vertex $v_i \in G$ corresponds to a document i. For each vertex $v_i \in G$ $(1 \leq i \leq n)$, we assign $\mathbf{r}_0(i) = \frac{1}{n}$ and $\mathbf{r}_i(0) = 0$ (note $\mathbf{r}$ is not always symmetric); clearly, $\mathbf{r}_0(0)$ should be 1. Then, for each edge $(v_i, v_j) \in G$, we assign $\mathbf{r}_i(j) = \mathbf{r}_j(i) = 0$, and for each pair of vertices v_i and v_j which are not connected by an edge in G, we assign $\mathbf{r}_i(j) = \mathbf{r}_j(i) = \frac{1}{n^2}$. Then, based on a preliminary calculation, it can be immediately verified that $g(R) \geq 1$ with $|R| = k$ if and only if the following two conditions hold:

1. R contains the query with label 0; and
2. R contains a complete subgraph, with $(k-1)$ vertices, of G. Note that the $(k-1)$ vertices correspond to the $k-1$ documents.

4.3 Efficient Near-Optimal Algorithm

Theorem 1 shows that retrieving a set of k documents to maximize $g(R)$ is NP-hard. The function $g(R)$ is *submodular* and [15] states that a greedy algorithm to maximize a submodular function has the approximation ratio $(\frac{e-1}{e})$. Our algorithm (see Algorithm 1) is a greedy algorithm, for which we increase the value of topic coefficient w_j if some documents belonging to the same topic t_j have already been included in R. The set R resists document d_j with higher value of w_j while it prefers document $d_h (1 \leq h \leq n)$ if its topic coefficient is lower. Furthermore, when the maximum number of a topic is reached (e.g. $z_i = u_i$), the corresponding topic coefficient is set to be a prohibitive value Ω. Setting $w_i = \Omega$ in Eq. (3) ensures that we reject any further documents which have topic t_i. A suitable value of Ω was determined via our empirical studies.

According to the *Proposition 4.3* in [15], the greedy algorithm of diversification has the following accuracy guarantee.

Theorem 2. *The greedy algorithm achieves an approximation ratio of* $(\frac{e-1}{e})$ *for the submodular function of diversification with proportionality constraint.*

Proof. Omitted for brevity. Refer to [15] for details.

4.4 Proposed Metric

Observing that most existing metrics measure diversity by explicitly penalizing redundancy over each returned document while maintaining relevance, we propose a novel metric that considers the proportionality on the diversified search

Algorithm 1. Diversification by Popularity-based Proportionality.

```
Input: $\mathbf{r}_i$ (for $1 \leq i \leq n$); $k$; query $i_0$.
Output: A list $R$ of $k$ documents.
set initial $R$ as $i_0$;
set both of initial $w_i$ and $z_i$ (for $1 \leq i \leq n$) as 0;
set initial $u_i$ (for $1 \leq i \leq n$) as ceil($\frac{1}{k}$);
for looper=1:k do
    choose the document $d_j$ such that $g(R)$ is maximized;
    if $z_j \leq u_j$ then
        add $d_j$ into $R$;
        $z_j = z_j + 1$;
        $w_i = e^{1-\frac{z_i}{u_i+1}}$;
    else
        $w_j = \Omega$;
        discard $d_j$;
Return $R$
```

results by extending the metric in [10]. The metric proposed in [10] considers the following principles: First, each document need not belong to just one aspect of the query; that is, a document might be related to multiple aspects. Second, selecting a document that is related to some topics which already have enough relevant documents should be evaluated better than a non-relevant document. In other words, non-relevant documents should not be evaluated as highly as over-selection. However, the metric in [10] ignores the importance of rank positions of documents. Therefore, another critical property should be added: non-relevant documents appearing at earlier rank positions should be evaluated worse than relevant documents in later positions.

Considering the above three principles and least square index (LSq) [12], which is a standard metric for measuring *dis-proportionality*, we formulate our metric as Eq.(5) for penalizing the dis-proportionality for each rank position $L(1 \leq L \leq k)$:

$$DP@L = \sum_{t_i} c_i \| \frac{u_i - v_i}{v_i} \|^2 + \frac{1}{L} \cdot Y^2 \tag{5}$$

where u_i indicates the number of documents relevant to topic t_i, v_i is the number of documents that are actually found for t_i, Y denotes the number of non-relevant documents at positions $1..L$. The coefficient c_i on topic t_i is defined as follows:

$$c_i = \begin{cases} 1, u_i \geq v_i; \\ 0, \text{otherwise.} \end{cases} \tag{6}$$

We now briefly discuss how our metric satisfies the aforementioned three principles. Our metric addresses the first principle associated with metric design by penalizing a list for under-selecting ($v_i \leq u_i$) on some topics but not for over-selecting ($v_i \geq u_i$) on it. At the same time, non-relevant aspects are penalized

($Y \geq 0$) while over-selecting is not, which meets the second principle. Finally, the third principle of rank positions is implemented by considering the positions that are occupied by the non-relevant documents in the top-k diverse ranked list. To make the metric comparable across queries, we normalize the proportionality measure as follows:

$$PM@L = 1 - \frac{DP@L}{\sum_{t_i} u_i^2 + L} \tag{7}$$

In the end, the proportionality diversification metric for a ranked list R can be computed as follows:

$$PM(R) = \frac{1}{R} \sum_{L=1}^{|R|} PM@L \tag{8}$$

5 Experimental Evaluations

In this section, we conduct extensive experiments to evaluate the effectiveness and efficiency of our algorithms. The setting of experiment is introduced in section 5.1, followed by the study of parameter learning in section 5.2. Then elaborate evaluations are presented in section 5.3.

5.1 Experimental Setup

Baseline Diversity Models. We implemented the model described above, along with four other diversity models as baselines. The first diversity model is MMR [3], which has been widely considered standard in diversity literature. Another model, xQuAD [18], uses a probabilistic framework which determines how well each document satisfies each topic and outperforms many others in the task of diversitfication. The third model, proposed by Dang *et al.*[10], is referred to as Election in our experiment, and uses an election-based approach to address the problem of search result diversification. Finally, we implemented the approach of Dragon, which captures relevance and diversity in an optimized way [20].

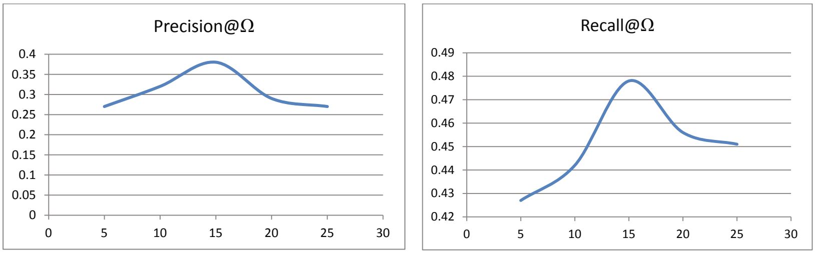
Query and Topic Collection. There are 50 queries in our query set, which come from the diversity task of the TREC 2009 Web Track [5]. To obtain the relevant documents for each query, we adopt the query-likelihood framework to conduct the relevance search [9]. The evaluation is conducted on the ClueWeb09 Category B retrieval collection [1], which contains 50 million webpages. As our approach and xQuAD require the availability of query topics and their popularity, we utilize the sub-topics provided by TREC as aspects for each query. Since the popularity of each topic is not available in TREC data, we follow the model in [18] by adopting suggestions provided by a search engine as topic representation.

[1] `http://boston.lti.cs.cmu.edu/Data/clueweb09`

Evaluation Metrics. We evaluate our approach and baseline models in terms of the proportionality metric proposed in Section 4. Considering that the proportion metric is specialized towards our model, we also report performance using several standard metrics including α-NDCG[7], ERR [4] and NRBP [8].

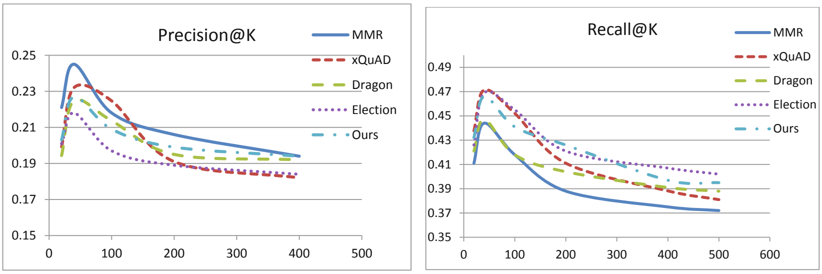
5.2 Parameter Learning

Parameter learning aims to determine "optimal" values for Ω and k. The Ω measure is specific to our approach, and we evaluate precision and recall using Ω values ranging from 5 to 25. Fig.1 shows that our model achieves the best results when Ω has the value of 15. The k measure applies to all algorithms, and we need to ensure that we do not choose a k value that is biassed towards any particular approach. Fig.2 shows that all approaches perform best with a value of k around 40. Thereby, we conduct the diversification search with a ranked list of 40 documents.



(a) (b)

Fig. 1. Parameter learning on Ω

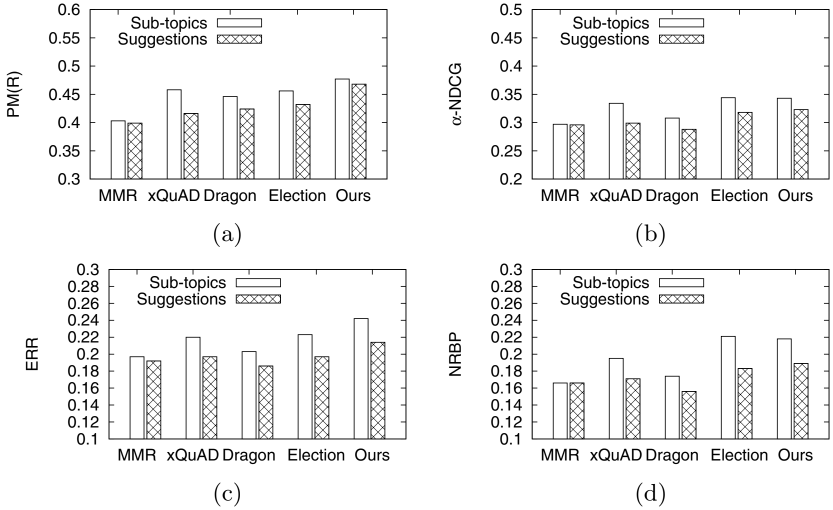


(a) (b)

Fig. 2. Parameter learning on k

5.3 Performance Evaluations

Metrics and Proportionality Evaluations. We first compared our proposed technique to MMR, xQuAD, Dragon and Election using our proportional metric $PM(R)$ for a list of R. From Fig.3 (a), we can see that our technique outperforms the other four, which demonstrates the effectiveness of our method at preserving proportionality. Secondly, we conducted comparisons in terms of three standard metrics from the diversity literature: α-NDCG, ERR and NRBP. The results are reported in Fig.3 (b) to (d), from which we can observe the similar result as in the previous example with proportional metric. Specifically, MMR is the least effective because of its ignorance of query topics. On the other hand, our method outperforms greatly over all the other method on almost all metrics. Note that these measures are computed using top 20 documents retrieved by each model, which is consistent with the standard TREC evaluations [5].



(a)

(b)

(c)

(d)

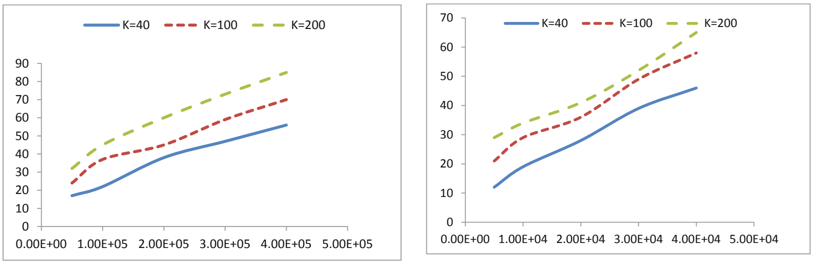
Fig. 3. Performance of diversity models in standard measures and our proposed metric

As all algorithms rank the top-k retrieved documents according to different principles, we examine the performance in both precision and sub-topic recall. We summarize the results in Table 1, which suggest that documents returned by MMR are more relevant and Election covers more topics than others. However, in terms of suggestions, i.e., the representation of popularity on retrieved documents, our model achieves better performance than the other four.

Scalability. Fig.4 gives our evaluation on the scalability of our algorithm (using synthetic data). The number of edges are fixed when we evaluate the scalability with respect to the number of nodes and vice versa. Fig.4 shows that our model

Table 1. Precision and recall for top-40 results

	Precision@40					Recall@40				
	MMR	xQuAD	Dragon	Election	Ours	MMR	xQuAD	Dragon	Election	Ours
Sub-topics	**0.2231**	0.1907	0.1775	0.2107	0.1902	0.4673	0.4724	0.4550	**0.4820**	0.4644
Suggestions	0.1801	0.1891	0.1609	0.1576	**0.2133**	0.4341	0.4410	0.4122	0.3978	**0.4522**



(a) Time w.r.t. number of nodes (b) Time w.r.t. number of edges

Fig. 4. Scalability of our model

increases linearly with respect to nodes and edges, which demonstrates that it can be applied to large-sized databases.

6 Conclusion

In this paper, we present a novel technique to address the problem of proportionally diversifying search results. A novel objective function is proposed to obtain a top-k ranked list by maximizing the value of the function. We prove that obtaining the optimal maximal value with respect to the proposed objective function is NP-hard, and resolve this by proposing an efficient greedy heuristic. We also propose a metric ($PM(R)$) to measure how effectively a diversification algorithm captures proportionality. Our experimental studies, evaluated on both standard metrics and our proposed metric, validate that our algorithm is not only able to effectively balance the relevance and diversity of search results, but is also capable of keeping approximate proportionality of the top-k search results according to the popularity degree of the various topics.

References

1. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying search results. In: WSDM (2009)
2. Bahmani, B., Chowdhury, A., Goel, A.: Divdb: A system for diversifying query results. In: PVLDB, pp. 1395–1398 (2011)
3. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: SIGIR, pp. 335–336 (1998)

4. Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: CIKM (2009)
5. Clarke, C., Craswell, N., Soboroff, I.: Overview of the trec 2009 web track. In: TREC (2009)
6. Clarke, C., Craswell, N., Soboroff, I., Ashkan, A.: A comparative analysis of cascade measures for novelty and diversity. In: WSDM (2011)
7. Clarke, C., Kolla, M., Cormack, G., Vechtomova, O., Ashkan, A., Buttcher, S., MacKinnon, I.: Novelty and diversity in information retreival evaluation. In: SIGIR (2008)
8. Clarke, C.L.A., Kolla, M., Vechtomova, O.: An effectiveness measure for ambiguous and underspecified queries. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) ICTIR 2009. LNCS, vol. 5766, pp. 188–199. Springer, Heidelberg (2009)
9. Croft, W., Metzler, D., Strohman, T.: Search Engines: Information Retrieval in Practice (2009)
10. Dang, V., Croft, W.B.: Diversity by proportionality: an election-based approach to search result diversification. In: SIGIR (2012)
11. Fogaras, D., Rácz, B., Csalogány, K., Sarlós, T.: Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. Internet Mathematics 2(3), 333–358 (2005)
12. Gallagher, M.: Proportionality, disproportionality and electoral systems. Electoral Studies 10(1), 33–51 (1991)
13. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness (1979)
14. Mei, Q., Guo, J., Radev, D.: Divrank: the interplay of prestige and diversity in information networks. In: ACM SIGKDD, pp. 1009–1018 (2010)
15. Nemhauser, G., Wolsey, L., Fisher, M.: An analysis of approximations for maximizing submodular set functions. Mathematical Programming 14, 265–294 (1978)
16. Radlinski, F., Dumais, S.: Improving personalized web search using result diversification. In: SIGIR (2006)
17. Rafiei, D., Bharat, K., Shukia, A.: Diversifying web search using result diversification. In: WWW (2010)
18. Santos, R., Macdonald, C., Ounis, I.: Exploiting query reformulations for web search result diversification. In: WWW (2010)
19. Slivkins, A., Radlinski, F., Gollapudi, S.: Learning optimally diverse rankings over large document collections. In: ICML (2010)
20. Tong, H., He, J., Wen, Z., Konuru, R., Lin, C.-Y.: Diversified ranking on large graphs: An optimization viewpoint. In: ACM SIGKDD, pp. 1028–1036 (2011)
21. Zhai, C., Cohen, W.W., Lafferty, J.: Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In: ACM SIGIR, pp. 10–17 (2003)
22. Zhu, X., Goldberg, A.B., Gael, J.V., Andrzejewski, D.: Improving diversity in ranking using absorbing random walks. In: HLT-NAACL, pp. 97–104 (2007)
23. Ziegler, C.-N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: WWW, pp. 22–32 (2005)

Joint Naïve Bayes and LDA for Unsupervised Sentiment Analysis

Yong Zhang[1,2], Dong-Hong Ji[1], Ying Su[3], and Hongmiao Wu[4]

[1] Computer School, Wuhan University, Wuhan, P.R. China
[2] Department of Computer Science, Huazhong Normal University, Wuhan, P.R. China
[3] Department of Computer Science, Wuchang Branch, Huazhong University of Science and Technology, Wuhan, P.R. China
[4] School of Foreign Languages and Literature, Wuhan University, P.R. China
ychang.cn@gmail.com, donghong_ji2000@yahoo.com.cn,
{suying929,hongmiao23}@163.com

Abstract. In this paper we proposed a hierarchical generative model based on Naïve Bayes and LDA for unsupervised sentiment analysis at sentence level and document level of granularity simultaneously. In particular, our model called NB-LDA assumes that each sentence instead of word has a latent sentiment label, and then the sentiment label generates a series of features for the sentence independently in the Naïve Bayes manner. The idea of NB assumption at sentence level makes it possible that we can use advanced NLP technologies such as dependency parsing to improve the performance for unsupervised sentiment analysis. Experiment results show that the proposed NB-LDA can obtain significantly improved results for sentiment analysis comparing to other approaches.

Keywords: Sentiment Analysis, Latent Dirichlet Allocation, Naïve Bayes, Opinion Mining.

1 Introduction

In recent years sentiment analysis or opinion mining aiming to uncover the attitude of users from the content has drawn much attention in the NLP. But most methods usually rely heavily on an annotated corpus for training the sentiment classifier. So the sentiment corpora are considered as the most valuable resources for sentiment analysis. To circumvent laborious annotation efforts, developing an unsupervised method for sentiment analysis is one of the goals of this paper.

Previous methods have tackled the problem at different levels of granularity, from document level, sentence level, phrase-level, as well as the speaker level in debates. Intuitively sentiment analysis at different level of granularity can benefit from each other [11, 15]. Though some studies have been performed to analyze the sentiment of document level and sentence level simultaneously, their approaches are usually based on structure model and try to exploit the structure information in document, and usually are supervised or semi-supervised [11, 16, 17]. So another goal of this paper is

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 402–413, 2013.

to develop a unsupervised model that jointly classifies sentiment at sentence level and document level of granularity.

For fine-grained sentiment analysis such as for sentence, many problems must be tackled. Negation is the main problem, especially involving long-distance dependencies such as the negation of the proposition or the n egation of the subject (e.g., no one thinks that it's good). In addition, contextual polarity may also be influenced by modality, word sense, the syntactic role of a word in the sentence and diminishers such as little [18]. To solve such problems, it is n ecessary to employ advanced NLP technologies such as dependency parsing [25]. So the third goal of this paper is to incorporate advanced NLP technologies with statistical model for sentiment analysis.

In this study, we propose an unsupervised model to incorporate the sentiment analysis at the document level and sentence level. The model is a novel and unified hierarchical generative model combining Naïve Bayes and Latent Dirichlet Allocation (LDA), which we refer to as NB-LDA. But unlike LDA, our model does not assume documents are "bags of words". Rather it a ssumes that each sentence has a late nt sentiment label, the latent sentiment label is drawn from the distribution of sentiment over document, and the words or features in the sentence are generated by the latent sentiment label in a Naïve Bayes manner. Thus the Naïve Bayes Assumption in our model makes it ea sy to integrate rich features and advanced NLP technique results into it to achieve better performance. We show that this model naturally fits the task of sentiment analysis, and experimental results show the effectiveness of the proposed model.

The rest of this paper is organized as follows: Section 2 introduces related work. The proposed NB-LDA model is described in detail in Section 3. Section 4 shows the experiments setup, and experiment results are des cribed in section 5. L astly we conclude this paper and discuss the future work.

2 Related Work

Previous work on sentiment analysis has covered a wide range of tasks, including polarity classifcation [15], opinion extraction [5], and opinion source assignment [20]. And these systems have tackled the problem at different levels of granularity. Usually for short of fine-grained corpus and lack of information, the fine-grained sentiment analysis is much harder than at d ocument level. In order to recognize the contextual polarity in phrase-level, Wilson et al. [18] have compiled a lexicon of over 8000 subjective clues and used a s et of features based on dependency tree of the sentence to disambiguates the polarity of the polar expressions. But most of their approaches relay on the availability of fine-grained annotations.

Recently topic modeling has been an area of active research [3, 7, 14]. Some models extended LDA have been proposed for sentiment analysis [10, 12, 19]. These approaches jointly learn topic and sentiment to improve predictions at document level. But these approaches have a shortcoming that they take the documents as "bag of words" and each word has a latent sentiment variable. Mukherjee and Liu [14] proposed two latent variable models (TME model and ME-TME model) to simultaneously model and extract topics (aspects) and various comment expressions for online review. But only n-grams (up to 4-grams) were modelled as a commenting expression.

There are, however, obvious advantages for sentiment analysis at both document level and sentence level. Pang and Lee [15] performed minimal cuts in a sentence graph to select subjective sentences to improve the performance of document sentiment classification. But these methods try to only improve document sentiment classification via selecting useful sentences.

Some structured models have previously been used for sentiment analysis [11, 16, 17]. McDonald et al. [11] presented a structured graphical model for fine-to-coarse sentiment analysis, and adopted a sequence classification with a constrained version of Viterbi for inference of the model. Täckström and McDonald [16] used latent variable structured prediction models for fine-grained sentiment analysis in the common situation where only coarse-grained supervision is available. Similarly Täckström and McDonald [17] derive two structured conditional models, which combine coarse-grained supervision with fine-grained supervision for sentence-level sentiment analysis. But these approaches are supervised or semi-supervised.

Another very relevant model is ASUM [8], as shown in Fig. 1(a), but there are some key differences between our model and ASUM. Firstly our model can use rich feature to improve sentiment analysis. For example the sequence of the negation and polarity words can be considered in our work. But ASUM can't do so. Secondly, our model is a unified generative model. Like LDA, Naive Bayes is a generative model, and they can be naturally integrated into a unified model. Notably NB can be supervised or unsupervised while other classification algorithms such as MaxEnt, SVM are discriminative only for supervised classification and difficult to integrate into LDA. Thirdly, our model focuses on sentiment classification. However ASUM is proposed mainly to discover pairs of senti-aspects.

To our knowledge, some similar models have been proposed by taking the best of Naïve Bayes and LDA models, such as Latent Dirichlet conditional Naïve Bayes (LD-CNB) [1] and Bayesian Naive Bayes[1]. But these models are different from ours. LD-CNB assumes a Dirichlet prior with parameter α from which the mixing weights θ is sampled. Further, for an observed feature f_j , a component z_j is first sampled from θ, and x_j is sampled from the corresponding component distribution. The LD-CNB process of generating is different from ours. Also there has been some work where arbitrary features were included into the LDA model, such as Dirichlet Multinomial Regression [13], MaxEnt-LDA [20] and etc. But the purposes and methods of these works are different from ours.

[1] http://lingpipe-blog.com/2009/10/02/bayesian-naive-bayes-aka-dirichlet-multinomial-classifiers/

3 NB-LDA Model

3.1 Motivation

Table 1. Distribution of sentence labels (columns) in documents by their labels (rows) in fine-grained sentiment dataset [16]

	Pos.	Neg.	Neut.
Pos.	0.53	0.08	0.39
Neg.	0.05	0.62	0.33
Neut.	0.14	0.35	0.51

For the distribution of sentence level sentiment in each document sentiment category, Täckström and McDonald [16] have shown that the sentence level sentiment is aligned with the document sentiment, as shown in Table 1. That is, in positive documents, most of sentences are positive, and in negative documents, most of sentences are negative, and in neutral documents, most of sentences are neutral Obviously this distribution can be exploited in sentiment analysis. As we know, topic models like LDA use co-occurrence information to group similar words into a single topic. Intuitively sentence level sentiment co-occurrence in documents may be mined to achieve better performance like topic model. So we proposed a new model, NB-LDA, which can employ rich features and exploit the distribution of sentence level sentiment in each document sentiment category to improve sentiment analysis. Here Naïve Bayes is to identify the sentence sentiment polarity based on a set of features in the local context, and LDA is to exploit the sentence level sentiment co-occurrence in documents globally.

In addition, to improve the performance of unsupervised sentiment analysis, our approach employs dependency parsing and a variety of features to identify the contextual polarity of the sentence inspired by [18].

3.2 NB-LDA Model

The NB-LDA model belongs to a family of generative models for text where a document contains a fixed number of sentences, each sentence expresses a kind of sentiment polarity represented by a latent variable z, and words in the sentence are viewed as features conditionally independent given sentiment label z. As a generative classification algorithm, Naive Bayes is used to identify sentiment polarity of the sentence via a set of features. Here we assume that a sentiment polarity (positive, negative or neutral) is expressed in a sentence rather than in a word, as done in previous studies [16, 17]. As to a document, we assume that it is "bag of sentences" and is generated by a mixture distribution θ of T sentiments, which is sampled from a prior Dirichlet distribution $Dir(\alpha)$.

This generative process can be expressed more formally by defining some of variables in the model. Assume we have T sentiment labels, for example positive, negative and neutral, which play similar roles as topics. Let D be the number of documents, F be the number of features, S_d be the number of sentence in document d, S_{di} be the i-th sentence in document d, and F_{di} be the number of features in sentence S_{di}. We can parameterize the multinomial distribution over sentiment labels for each document using a matrix Θ of size $D * T$, where each row θ_d stand for the probabilities of sentiment labels in document d. Similarly the matrix Φ of size $T * F$ denotes the distribution over features associated with each sentiment label, where φ_t stand for the probabilities of generating features from sentiment label t. These matrix distributions are assumed to be generated from symmetric Dirichlet priors with hyper parameters α and β respectively.

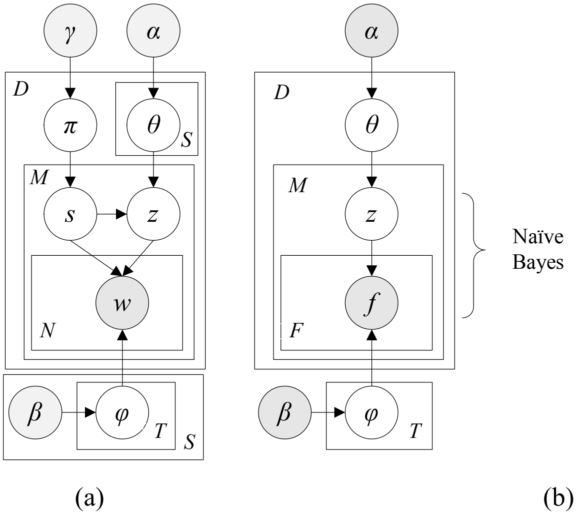


Fig. 1. (a) Graphical model of ASUM. (b) Graphical model of NB-LDA

The formal definition of NB-LDA model is the following:

- For each document d, sample θ_d~$Dir(\alpha)$
- For each label t, sample φ_t ~$Dir(\beta)$
- For each sentence i in S_d sentences of document d
  - Choose a sentiment label z_{di}~ $Multi$ (θ_d)
  - For each feature j in F_{di} features of sentence s_{di}
  - Choose a feature value f_{dij}~$Multi(\varphi_{z_{di}})$.

In above generation process, the most difference from LDA is that NB-LDA chooses a latent variable z for each sentence in document d and then generates the feature values of the sentence independently with a Diric hlet prior. The graphical model corresponding to this process is shown in Figure 1(b).

Although previous studies have shown that topic and sentiment are dependent, the dependencies are usually limited in the range of one sentence. In NB-LDA, we use the NB and a set of features produced by NLP technologies such as dependency parsing to solve the local dependencies.

3.3 Inference with NB-LDA

A variety of algorithms have been used to estimate the parameters of topic models [3, 6]. In this paper, we will use Gibbs sampling [6], as it provides a simple method for obtaining parameter estimates under Dirichlet priors and allows combination of estimates from several local maxima of the posterior distribution.

Under this generative process, the joint probability of the *z* assignments and the features *f* can be factored into the following terms:

$$p(z, f) = \prod_{d=1}^{D} \prod_{i=1}^{S_d} p(z_{di}) \prod_{f=1}^{F_{di}} p(f \mid z_{di}) \tag{1}$$

By applying Gibbs sampling, we construct a Markov chain that converges to the posterior distribution on *z* like [6]. The transition between successive states of the Markov chain results from repeatedly drawing *z* from its distribution conditioned on all other variables, summing out *θ* and *φ* using standard Dirichlet integrals:

$$p(z_{d,i} = k \mid s_{d,i}, z_{d,-i}, s_{d,-i}) \propto \frac{C_{d,k}^{DT} + \alpha}{\sum_{t=1}^{T} C_{d,t}^{DT} + T\alpha} \cdot \prod_{j \in F_{d,i}} \frac{C_{j,k}^{FT} + \beta}{\sum_{f=1}^{F} C_{f,k}^{FT} + F\beta} \tag{2}$$

Here $C_{d,k}^{DT}$ is the number of times that sentences in the document *d* were assigned to sentiment label *k*, not including the current sentence. $C_{j,k}^{FT}$ is the total number of times that sentences containing feature *j* were assigned to sentiment label *k*, not including the current sentence.

For any sample from this Markov chain, being an assignment of every sentence to a sentiment label, we can estimate *θ* and *φ* using

$$\theta_{d,k} \propto \frac{C_{d,k}^{DT} + \alpha}{\sum_{t}^{T} C_{d,t}^{DT} + T\alpha} \tag{3}$$

$$\varphi_{k,j} \propto \frac{C_{j,k}^{FT} + \beta}{\sum_{f}^{F} C_{f,k}^{FT} + F\beta} \tag{4}$$

where $\varphi_{k,j}$ is the probability of using feature *j* in sentiment label *k*, and $\theta_{d,k}$ is the probability of sentiment label *k* in document *d*.

4 Experiments Setup

In this section, we introduce the data sets, prior-polarity subjectivity lexicon and feature space. In our experiments, the hyper parameters α and β are fixed at 0.3 and 0.01 respectively [6].

4.1 Corpus, Prior-Polarity Subjectivity Lexicon and Features

We used 2 corpora to evaluate our model. The first one is the Movie Review Data[2]. The second is Fine-grained Sentiment Dataset [16]. For pre-processing, we used Stanford's Suite of NLP Tools[3]. Here some tokens were removed according to the stop words list and POS, and "Fine-grained Sentiment Dataset" was adjusted slightly in order to adapt to Stanford's Suite of NLP Tools.

In the experiments, we used a public subjectivity lexicon as prior-polarity knowledge [18]. We compared the word tags to clues in the lexicon. If matched, the word token was marked with the corresponding "prior polarity". And the sentence was marked with the majority polarity voted by all matched words while taking negation into account. The prior information of sentence was only utilized during the initialization. The initialization starts by checking the "prior polarity" each sentence in corpus. If the sentence has a "prior polarity", the corresponding sentiment label is assigned to it. Otherwise, a sentiment label is randomly sampled.

Table 2. Features used in NB-LDA

Word Features
word lemma
word prior polarity: positive, negative, both, neutral
reliability class: strongsubj or weaksubj
Polarity Features
Negated: binary
negated subject: binary
modifies polarity: positive, negative, neutral, both, notmod
modified by polarity: positive, negative, neutral, both, notmod
conj polarity: positive, negative, neutral, both, notmod
Sentence Features
cardinal number in sentence: binary
pronoun in sentence: binary
modal in sentence (other than will): binary
Structure Features
Sentence position: first, mid, last

Table 2 lists the features in the experiments. Most of the features are used in [18]. Here we used word lemma as features instead of word tokens. Besides we added

[2] http://www.cs.cornell.edu/People/pabo/movie-review-data/

[3] http://nlp.stanford.edu/software/corenlp.shtml

sentence position as structure features. Note there are some differences between Stanford typed dependencies and that used in [18], and we handled them.

4.2 Baselines

In this work, we adopted four baselines to evaluate our model.

lemma-prior: In this baseline, we only use lemma as features, but didn't use any prior knowledge. In the initialization of Gibbs, all the latent variables of sentences are randomly chosen.

lemma+prior: In this baseline, we only use lemma as features, and use the prior knowledge. In the experiments, we only compared the tokens or lemmas morphologically with the words in subjectivity lexicon, and ignored the POS.

ASUM+prior: ASUM is so similar to our model. So we compared ASUM model as baseline, where we use the same prior-polarity subjectivity lexicon instead of seed words used in [8] for prior knowledge.

NB-ASUM: In the baseline, we assumed that ASUM can generate feature values just like our model. Besides the setting of ASUM+prior, we use all features described in Table 2.

5 Experiment Results

5.1 Results for Movie Review Data

In Movie Review Data, the reviews are only labeled as positive or negative at document level. So in the experiments, T is set to 2 since we only consider 2 sentiment labels: positive and negative. The document sentiment is classified based on $\theta_{d,k}$, the probability of sentiment label given document. A document d is classified as a positive sentiment document if its probability of positive sentiment label given document $\theta_{d,pos}$, is greater than its probability of negative sentiment label given document $\theta_{d,neg}$, and vice versa. As shown as Table 3, classifcation accuracies were averaged from 10 runs with 1000 Gibbs sampling iterations.

As can be observed from Table 3, the performance of "lemma-prior" is mediocre when no prior information was incorporated. A significant improvement, with 10.7%, is observed after incorporating prior information. It is also noted that NB-LDA with all features achieved 3.9% improvement over "lemma+prior", implying that the richer features can benefit sentiment analysis. So discovering more effective features is one of the future works.

When compared to the recently proposed unsupervised approach based on a spectral clustering algorithm [4], NB-LDA achieved better performance with about 1% overall improvement. Nevertheless, the approach proposed in [4] requires users to specify which dimensions (defined by the eigenvectors in spectral clustering) are most closely related to sentiment by inspecting a set of features derived from the reviews for each dimension, and clustering is performed again on the data to derive the final results. In our model studied here, no human judgment is required. Another recently proposed non-negative matrix tri-factorization approach in [9] also employed lexical prior knowledge for semi-supervised sentiment classification. However, when incorporating 10% of labeled documents for training, the non-negative matrix tri-factorization approach performed much worse than NB-LDA, with only around 60% accuracy achieved. Even with 35% labeled documents, it still performs worse than NB-LDA. It is worth noting that no labeled documents were used in the NB-LDA results reported here.

Table 3. Average results from 10 runs in terms of accuracy for Movie Review. Above double line: results from our model. Below double line: results from other literatures.

	Pos.	Neg.	Total
lemma -prior	55.0	59.5	57.25
lemma +prior	72.7	63.2	67.95
ASUM+prior	72.1	63.3	67.7
NB-ASUM	74.5	67.4	70.95
NB-LDA	75.3	68.4	71.85
Lin et al. [10]	74.1	66.7	70.4
Dasgupta and Ng [4]			70.9
Li et al.[9] with 10% doc. Label			60
Li et al.[9] with 35% doc. Label			about 69

Another notable result is that the prior information has different effects on positive documents and negative ones. Without prior information, the positive accuracy is less than the negative accuracy. While with prior information, the positive accuracies are better than the negative ones significantly. Manually analyzing the results reveals that there are more words matching the positive clues than negative clues in the corpus, which causes the imbalance of distribution of positive prior and negative prior during the initialization. Actually we found the similar problem in the experiments for fine-grained dataset reported in the next subsection, and in the results of [10, 16, 17].

5.2 Results for Fine-Grained Dataset

There are three sentiment categories at document level in this dataset, but there are five sentiment categories at sentence level: POS, NEG, NEU, MIX, and NR. Like [16], we considered the MIX and NR categories as belonging to the NEU category. So in the experiments, T is set to 3. The sentence sentiment polarity is classified by the latent sentiment label z after sampling.

Table 4 shows the results for our model in terms of sentence and document accuracy as well as F1-scores for each sentence sentiment category. In Table 4, the results above double line come from our experiments, and that below double line are results presented in [16] and [17] for the same dataset.

In terms of sentence accuracy, from these results it is clear that the NB-LDA model significantly outperform VoteFlip with quite a wide margin. Comparing to SaD and

DaS, the results from NB-LDA are also very competitive. However our results are still lower about 7% ~ 12% than HCRF and Interpolated in the last three rows in terms of both sentence and document accuracy. Actually the results below double line are evaluated via 294 reviews, but with 143,580 labeled reviews as supervision. Notably NB-LDA does not use any labeled data.

In our results, "lemma-prior" is mediocre. But with the prior information, "lemma+prior" performs better about 6% for sentence accuracy and about 8% for document accuracy than "lemma-prior". The NB-LDA with all features and prior information achieved much better performance than "lemma-prior" and "lemma+prior". The results suggest that both proper features and prior information are important for improving sentiment analysis. In fact, NB-LDA model can easily integrate rich features and advanced NLP technique into it to achieve better performance.

Table 4. Average results from ten runs for fine-grained dataset. Above double line: results from our model. Below double line: VoteFlip, SaD, DaS and HCRF results without observed document label in [16], and Interpolated best result in [17].

	Sentence				Document Acc
	Total Acc	Pos. F1	Neg. F1	Neut. F1	
lemma-prior	36.0	25.7	38. 6	38.8	40.5
lemma+prior	41.9	46.4	49.5	31.6	48.6
ASUM+prior	41.4	45.9	49.0	31.2	47.3
NB-ASUM	45.7	52.2	48.5	36.9	52.7
NB-LDA	46.8	53.4	49.6	38.0	54.4
VoteFlip	41.5	45.7	48.9	28.0	-
SaD	47.6	52.9	48.4	42.8	-
DaS	47.5	52.1	54.3	36.0	66.6
HCRF(soft)	53.9	57.3	58.5	47.8	65.6
HCRF(hard)	54.4	57.8	58.8	48.5	64.6
Interpolated	59.1	-	-	-	-

5.3 NB-LDA vs ASUM

Similar to our NB-LDA, ASUM is also a generative model for sentiment analysis, which is proposed mainly to discover pairs of senti-aspects. In this work, our model only focuses on sentiment classification. But ASUM incorporates aspect and sentiment together to model sentiments toward different aspects [8]. The differences between them have been discussed in motivation. Here we compared the performances between the two models, and analyzed the differences, although NB-ASUM should obtain the same results with NB-LDA in theory.

In fact as shown in Table 3, the “lemma+prior” performs better than ASUM+prior slightly. It seems that only using lemma as feature, ASUM can obtain comparable result with our model. While using all features, the total accuracy of NB-LDA is higher about 1% than that of NB-ASUM. In fine-grained dataset as shown in Table 4, we can see the similar results, where the "lemma+prior" can achieve better performances than ASUM+prior, and NB-LDA is better than NB-ASUM.

From the experimental results, we can see that NB-LDA is consistently better than ASUM. As we know, the generative process of ASUM is more complex than NB-LDA. In NB-ASUM we need compute the probabilities for each feature value conditioned on topic variable and sentiment variable. It means that in NB-ASUM, the distributions would be s parser than in NB-LDA. The sparsity might block the propagation of sentiment information during iterations, and could not conduct good results.

6 Conclusions and Future Work

In this paper we proposed a hierarchical generative model based on Naïve Bayes and Latent Dirichlet Allocation for unsupervised sentiment analysis at sentence level and document level simultaneously, only using a public subjectivity lexicon. The idea of NB assumption at sentence level makes it possible that we can use advanced NLP technologies such as dependency parsing to improve the performance of sentiment analysis. The experiments show that our model obtained better performance than VoteFlip, a rule-based approach and ASUM. However for now our model hardly reach the competitive performance to the supervised or semi-supervised approaches.

Since unsupervised approaches hardly obtain comparable performance to supervised ones. A simple extended model could be designed based on supervised LDA [2] and Naïve Bayes. Another extension of our model is to capture sentimental structures within the documents simultaneously, inspired by HTMM [7]. The third future work is to empirically investigate the effects of more features on more datasets. Finally we may split sentences not only by punctuations but also by conjunctions, since one sentence may contain multiple sentiment clauses.

Acknowledgments. The work was funded by the National Natural Science Foundation of China (No. 61133012, 61173062, 61070082) and the Major Projects of Chinese National Social Science Foundation (No.11&ZD189). Dong-Hong Ji is the corresponding author.

References

1. Banerjee, A., Shan, H.: Latent Dirichlet conditional Naive-Bayes models. In: The IEEE International Conference on Data Mining (ICDM), pp. 421–426 (2007)
2. Blei, D., McAuliffe, J.: Supervised topic models. In: Proceeding of Neural Information Processing Systems, NIPS (2007)
3. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of Machine Learning Research 3(5), 993–1022 (2003)
4. Dasgupta, S., Ng, V.: Topic-wise, Sentiment-wise, or Otherwise? Identifying the Hidden Dimension for Unsupervised Text Classification. In: Proceedings of EMNLP (2009)

5. Ding, X., Liu, B., Zhang, L.: Entity discovery and assignment for opinion mining applications. In: Proceedings of KDD 2009, pp. 1125–1134 (2009)
6. Griffiths, T., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences 101(90001), 5228–5235 (2004)
7. Gruber, A., Rosen-Zvi, M., Weiss, Y.: Hidden Topic Markov Models. In: Artificial Intelligence and Statistics (AISTATS), San Juan, Puerto Rico (2007)
8. Jo, Y., Oh, A.H.: Aspect and sentiment unification model for online review analysis. In: Proceedings of WSDM 2011, pp. 815–824 (2011)
9. Li, T., Zhang, Y., Sindhwani, V.: A nonnegative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In: Proceedings of ACL-IJCNLP 2009, pp. 244–252 (2009)
10. Lin, C., He, Y.: Joint sentiment/topic model for sentiment analysis. In: Proceeding of the Conference on Information and Knowledge Management, CIKM (2009)
11. McDonald, R., Hannan, K., Neylon, T., Wells, M., Reynar, J.: Structured models for fine-to-coarse sentiment analysis. In: Proceedings of ACL (2007)
12. Mei, Q., Ling, X., Wondra, M., Su, H., Zhai, C.X.: Topic sentiment mixture: modeling facets and opinions in weblogs. In: Proceedings of WWW (2007)
13. Mimno, D.M., McCallum, A.: Topic Models Conditioned on Arbitrary Features with Dirichlet-multinomial Regression. In: Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, pp. 411–418 (2008)
14. Mukherjee, A.: Liu, Bing: Modeling Review Comments. In: Proceedings of ACL 2012, Jeju, Republic of Korea, July 8-14 (2012)
15. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of ACL (2004)
16. Täckström, O., McDonald, R.: Discovering fine-grained sentiment with latent variable structured prediction models. In: Clough, P., Foley, C., Gurrin, C., Jones, G.J.F., Kraaij, W., Lee, H., Mudoch, V. (eds.) ECIR 2011. LNCS, vol. 6611, pp. 368–374. Springer, Heidelberg (2011)
17. Täckström, O., McDonald, R.: Semi-supervised Latent Variable Models for Sentence-level Sentiment Analysis. In: Proceedings of ACL (2011)
18. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of EMNLP (2005)
19. Zhang, Y., Ji, D.-H., Su, Y., Sun, C.: Sentiment Analysis for Online Reviews Using an Author-Review-Object Model. In: Salem, M.V.M., Shaalan, K., Oroumchian, F., Shakery, A., Khelalfa, H. (eds.) AIRS 2011. LNCS, vol. 7097, pp. 362–371. Springer, Heidelberg (2011)
20. Zhao, W.X., Jiang, J., Yan, H., Li, X.: Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In: Proceedings of EMNLP 2010, Stroudsburg, PA, USA, pp. 56–65 (2010)

An Unsupervised Learning Model to Perform Side Channel Attack

Jung-Wei Chou[1], Min-Huang Chu[1], Yi-Lin Tsai[1], Yun Jin[2], Chen-Mou Cheng[2], and Shou-De Lin[1]

[1] Department of Computer Science National Taiwan University
{r99922018,r96943077,r99922099,sdlin}@csie.ntu.edu.tw
[2] Department of Electric Engineering National Taiwan University
razzzer@gmail.com, ccheng@cc.ee.ntu.edu.tw

Abstract. This paper proposes a novel unsupervised learning approach for Power Analysis – a form of side channel attack in Cryptanalysis. Different from existing works that exploit supervised learning framework to solve this problem, our method does not require any labeled pairs, which contains information of the form {X,Y}={key, power-trace}, but is still capable of deciphering the secret key accurately. Besides proposing a regression-based, unsupervised approach for this purpose, we further propose an enhanced model through exploiting the dependency of key bits between different sub-processes during the encryption process to obtain accurate results in a more efficient way. Our experiment shows that the proposed method outperforms the state-of-the-art non-learning based decipherment methods significantly.

Keywords: Power analysis, side channel attack, unsupervised learning.

1 Introduction

In cryptography, side channel attack is a kind of attacking strategy taking advantage of the information gained from physical implementation of a cryptosystem to obtain the cryptographic keys of the device. One major advantage of side channel attack lies in its non-intrusive characteristic that allows the attacker to obtain side information that facilitates the deciphering of the key. It also enjoys a much lower computational complexity than cryptanalytic-theoretical attack, most of which is of super-polynomial complexity. For well-designed ciphers, side channel attack might be the only feasible way to recover the key to the device in practice. In this work we would like to introduce a machine-learning based attacking strategy for side channel attack.

We focus on a specific type of side channel attack called power analysis, but in general the proposed technique can be applied to several other kinds of side channel attacks such as electromagnetic attacks and acoustic cryptanalysis. Power analysis is a form of side channel attack, with which the power consumption of a cryptographic device (e.g. a smart card) is analyzed to obtain the cryptographic key of the device. As some cryptographic devices are implemented using semiconductor logic gates, and current flows across the silicon substrate when change is applied to or removed from

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 414–425, 2013.

the gate, it is not hard to imagine that through examining the power consumption of the device externally it is possible to determine what kind of operations (i.e. macro-characteristics) are executed on the device chip. One can then use such information to guess the secret key that corresponds to the hypothesized operations.

Le (2006) classified these techniques into two categories: attacks without reference devices and attacks using reference devices. With a reference device, it is possible to arrange different keys and plaintexts to feed into the device and record the output ciphertexts and power traces for further analysis. Without the reference device, while the outputs can still be measured, we have no idea what the inputs (i.e. keys) are. Hence, attacks using reference devices is like the supervised machine learning scenario, where the training data are labeled with known keys, and no (input, output) relation is provided for the other case. Therefore, attacking without reference devices is considered a much harder unsupervised learning problem.

In this paper, we propose an efficient, extensible unsupervised framework of power analysis based on machine learning techniques. We model the decipherment process as identifying a key that minimizes the training error of a given time stamp, which can be done unsupervised without using any labeled training data. Besides, the approach can be viewed as parameter estimation in abstraction, where the parameter domain contains all possible key candidates. To tackle sparse-training situation, we further propose a technique to exploit the dependency of multiple round functions in the encryption process. Finally we perform experiments on datasets obtained from the DPA Contest to show that the proposed method outperforms the competitors significantly.

The contributions of this paper are as follows. First, to the best of our knowledge, there has not yet been any work aiming at exploiting machine learning approaches to perform unsupervised side channel attack. Here we show that with careful design, simple machine learning techniques such as regression models can be exploited to tackle a cryptography problem. In this work, we hope to send an encouraging message to ML researchers on how the bridge between machine learning and cryptography can be established by demonstrating how the side-channel attack problem can be conducted from learning point of view.

2 Related Work

The concept of side channel cryptanalysis was first proposed by Kelsey (1998), which describes the use of side channel information such as current consumption leaked from imperfect implementation to facilitate breaking the cipher system. It is conceptually different from traditional cryptanalysis. That is, side channel cryptanalysis uses the correlation between plaintext and ciphertext to guess what happens inside a cryptosystem and further infer the key of the system. Side channel attack exploits the fact that most implementations of a cryptography system are not perfect and hence could inevitably leak some side channel information. The side channel information can be in the form of, for example, the electromagnetic (EM) gauged from CMOS device (Agrawal et al., 2002), or the electric current in standard block ciphers (Kocher et al., 1999) such as what has been used to attack many implementations of Data Encryption

Standard (DES) or Advanced Encryption Standard (AES). There are some other forms of power analysis, such as timing attack (Kocher, 1996), template attack (Chari et al., 2002), and acoustic attack (Backes et al., 2010).

Analyzing the snooped data is a non-intrusive attack of a cryptographic implementation, and power analysis is one of the most successful forms of such attack. The key reason to the success of power analysis lies in that the power consumption of a device generally possesses some correlation to the intermediate values that can be produced based on the cipher algorithm. In other words, maximum correlation can be obtained given correct hypothesis on the key. Below we will discuss some popular approaches based on this concept including DPA (Kocher et al., 1999), CPA (Brier et al., 2004), and BS-CPA (Komano et al., 2010).

Differential Power Analysis (DPA) is a type of attack that examines the power consumption signals through exploiting statistic measures to retrieve the correct key that has the maximum likelihood of producing the observed power consumptions. Similar to DPA, Correlation Power Analysis (CPA) is based on the linear relation between the real power consumption of the device and the intermediate values from the encipher model; it can be regarded as a form of multi-bit DPA. Messerges et al. (2002) demonstrate that CPA is just another form under DPA divided by a normalization factor. Built-In Determined Sub-Key Correlation Power Analysis (BS-CPA) is an enhancement of CPA that results in efficient trace usage. Whenever a sub-key is determined in each S -box, the BS-CPA can pass such information to assist other S-boxes to decrease the signal-to-noise ratio. In DPA Contest 2008, BS-CPA has been shown to be the most effective method. We will later compare our method with it.

In recent years, machine learning techniques have been playing an increasingly important role in attacking a cryptosystem. Hospodar (2011) proposed a supervised learning architecture to attack an AES system by side channel information. It regards the power consumption signal as an instance, divides the key bits into several binary labels and treats the problem as several binary classification tasks with Least Squares Support Vector Machine (LS-SVM) as the learner. The experiments show that LS-SVM is suitable for such purpose (Chari et al., 2002). A similar supervised approach is proposed by Lerman (2011). In practice, however, such labeled training examples are not available in most situations because it requires knowing the hidden key information in advance. Acknowledging such fact, we design an unsupervised learning approach that follows different assumptions than the previous work. In our case, only one single encrypted device with unknown key is needed..

3 Methodology

We start by interpreting the encoding process using Shannon's Noisy Channel Model. As shown in Figure 1, the inputs X to the channel contain a set of plaintext or known ciphertext (denoted as C={c_1...c_n}) and an unknown secret key (denoted as key), while the interaction of the inputs produces the observed outcome Y={y_1...y_n} that reflects a sequence of measured power consumption. Note that it is possible (and generally required for side channel attack) to use a variety of cipher-texts interacting with the same key to produce a set of observations. The noisy channel P(X|Y) can be considered as a black box that produces an output given an input. Given a fully observed Y and a partially observed X with P(X|Y) unobserved, the goal then becomes to recover the missing part of X (i.e. the secret key) using Y and C. The problem can then be mathematically formulated as argmax_x P(X|Y). We can first use Baye's rule to decompose argmax_xP(X|Y) into argmax_x[P(Y|X)*P(X)]. This essentially tells us that a proper X should not only possess a higher chance to produce the observation Y, but also has a higher chance to occur among other *X*'s. Here we assume no prior knowledge about X in a cryptography system, and consequently *P(X)* is uniformly distributed. In this problem, we are given a set of n instances as inputs X={x_k=(c_k,key), *where k=1...n*}, where c_k is a known cipher-text with the corresponding observation y_k, but the secret *key* is unknown. Assuming the deciphering processes are independent for each cipher-text, the problem argmax_xP(Y|X) can be transformed to

$$\text{argmax}_{key}[\text{P}(y_1|c_1, key)\ \text{P}(y_2|c_2, key) \ldots \text{P}(y_n|c_n, key)] \tag{1}$$

Then it becomes obvious that with a faithful estimation of P(y_k|c_k.key), one can eventually solve (1) by enumerating all possible keys. Here, a faithful estimation of P(y_k|c_k.key) implies that among all possible keys (key_1...key_m), only the correct key (denoted as key_c) shall obtain high P(y|c,key) value. Mathematically, a faithful estimation of P(y_k|c_k.key) should possess the following property:

$$\max_z[\text{P}(y_1|c_1, key_z)\ \text{P}(y_2|c_2, key_z)\ \ldots \text{P}(y_n|c_n, key_z), z\in[1,m], z\neq c] \ll \text{P}(y_1|c_1,key_c)\ \text{P}(y_2|c_2,key_c)\ \ldots\text{P}(y_n|c_n,key_c)$$

In other words, incorrect keys should possess much lower probability of producing y than the correct one. After taking log on both sides, we can obtain

$$\max_z[\text{logP}(y_1|c_1, key_z) + \text{logP}(y_2|c_2, key_z) + \ldots + \text{logP}(y_n|c_n, key_z), z\in[1,m], z\neq c] \ll \text{logP}(y_1|c_1, key_c) + \text{logP}(y_2|c_2, key_c)\ + \text{logP}(y_n|c_n, key_c)$$

The above equation is reasonable as the power signature reflects only the interaction between the correct key and the ciphertext. Therefore, we propose a three-stage framework to solve this problem:

1. Build m different learners ML_1... ML_m, each contains n instances (I_1,...I_n) that correspond to one single key:

$$\text{ML}_r = \{\text{I}_1 = (\text{c}_1, key_r, \text{y}_1), \text{I}_1 = (\text{c}_2, key_r, \text{y}_2), \ldots, \text{I}_n = (\text{c}_n, key_r, \text{y}_n)\}, \text{r} \in [1, \text{m}]$$

For each input instance *(c, key)*, we generate a set of features *f(c, key)* to train the learners ML_r. The generation of such features depends heavily on the backend cryptography algorithm. An example will be demonstrated in the experiment section.

2. We propose to model logP(y_k|c_k,key) using the inverse of training errors of the learners.

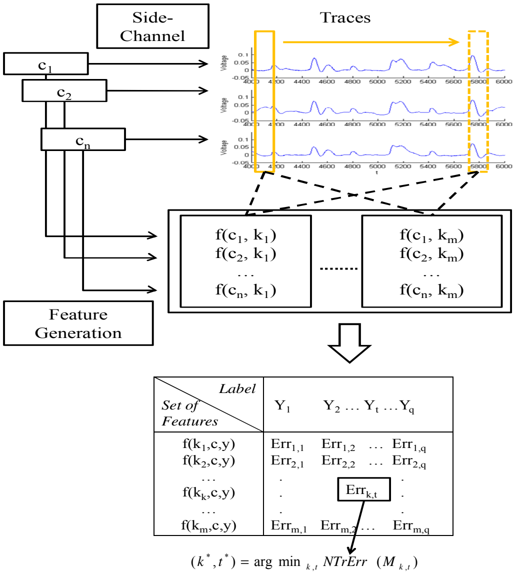


Fig. 1. The Framework

Acknowledging the fact that $P(y_k|c_k,key)$ represents how likely y_k results from an interaction between c_k and key, here we assume that the relationship between the input {c, key} and output *y* are learnable (i.e. low prediction errors) for correct key, and not learnable (high prediction errors) for incorrect keys. Therefore, the predictability of a machine learner has been used here to estimate the quality of a noisy channel. The channel corresponds to the right key should contain less noise and consequently be more learnable. Figure 1 illustrates the process. Next we assume the relation between trace sample *y* and features *x* generated from each pair of cipher-text and key candidate can be modeled simply as

$$y = w^T + n$$

where n can be assumed as Gaussian noise as in (Prouff et al., 2009), and its density function can be written as

$$P(n) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(y - w^T x)^2}{2\sigma^2}\right)$$

where σ is the standard deviation of the noise. As a result, the weight vector *w* for each timestamp and each possible key candidate can be derived by ordinary least square (OLS). Furthermore, coefficients of determination are used to normalize different scales at different time stamps.

Above we have explained how a supervised learning model can be exploited to solve such an unsupervised decipherment problem. However, for power analysis, there is still one more issue to be addressed. Although the outputs y for power analysis can be regarded as a sampled time sequence $\{y_1...y_q\}$, truth is that for most of the features generated, only very few signal in certain particular time stamps of y reveals apparent relationships with respect to (c, key). That is, usually only the right combination of key (denoted as key_c) and time stamp (denoted as t_c) possess higher learnability than other time stamp. Therefore, instead of building m different learners, we propose to create m*q different learners (q represents the number of samples for each time sequence) and argue that the one with the best predictability does reflect the correct key and time stamp.

3.1 Sub-Key Breaker

One major concern for such key enumeration approach lies in the fact that there are exponentially many keys to try. To conquer this problem, we follow a commonly used strategy of CPA (Brier et al., 2004) to divide the key into several sub-keys according to the permutation of inputs of substitution boxes (S-boxes). S-boxes are important components in block cipher design, which perform substitution and provide nonlinearity between ciphertexts and plaintexts. For instance, the length of key is 56 bits for DES; based on such a 56-bit key, each round a 48-bit round key is derived and distributed to 8 S-boxes (see section 4.1 for details). Generally, each sub-key has certain physical meaning, and we can extract features from it given some domain knowledge. Then we can apply the method proposed in 3.1 on each sub-key independently for better efficiency.

3.2 Dual-Round Approach for Multi-round Ciphers

So far we have introduced our approach to obtain the secret key from power traces. The quality of the results depends significantly on whether there are sufficient examples (or traces) to learn from. Without sufficient training examples, by chance some incorrect key might possess high P(y|c,key) and create false positives under our framework. One practical method to determine whether there is sufficient trace is to draw the "learning curve" that indicates whether the deciphered key becomes stable. In our experiments, we consider the deciphering process as completed if the results do not change after 100 additional traces are added.

The method mentioned in 3.1 might not be as effective if the number of traces is not enough to reveal the correct key (or to eliminate the false positives). Experiments show that in multiple-round ciphers, the encryption process that affects the power signal in each round does correspond to one particular time interval. Therefore it is possible to generate multiple training instances based on information from different rounds. Remember in Section 3.1 we have described how to break a longer key into sub-keys for analysis. In general these sub-keys are organized differently in different rounds. For examples, some bits might be grouped into the same sub-key in one round and broken into several different sub-keys in another round. For each round, some sub-keys can be deciphered easily (i.e. requiring fewer traces to converge to a steady result), while others require more training examples. Here we would like to further describe how one can fully exploit the side channel information from multiple-round ciphers given a limited number of traces. For Feistel ciphers with multiple rounds such as DES, if we are given some information of dependency between sub-keys in different rounds, a modified version of our method can exploit the relations between these sub-keys to improve the deciphering performance. The intuition behind our idea is as follows: if a correct sub-key in one round is identified (i.e. possesses lower training error than most of other candidates), then we can propagate such information to other learners and group the overlapped bit using the learned values. By doing such, the search space for other harder sub-keys is reduced, which alleviates the high demand for training traces and reduces the false positives.

An example is shown in Figure 2. It is known that some of the key bits such as b_0 and b_4 are used in both rounds, indicated by the connecting edges. Therefore, we can search for the key bits in these two rounds simultaneously. The difficulty of finding the correct key of the harder round (need more traces to break) can be reduced with the help of the easier one because of the increasing of signal-to-noise ratio when considering all cases at the same time. In dual-round approach, during training we first identify a set of S-boxes, called S-boxes set, whose bits overlapped with each other to some extent. Then within each S-boxes set, we train each of the S-boxes independently, but weighted-sum the errors of each to represent the quality of a particular assignment of bits. The weight is determined by the inverse of the "number of S-boxes" in each round containing in this particular S-boxes set. That is, if an S-box set contains one S-box in round 1 and two S-boxes in round 2, then the weight for the round 1 S-box is twice as much as that of the ones in round 2.

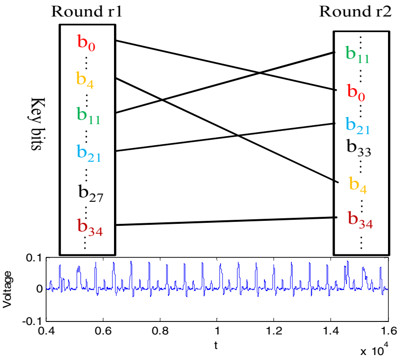


Fig. 2. Example of key dependency. The endpoints of each edge have the same key bit. In other words, the key bits used in both rounds are overlapped.

4 Experiments

We evaluate our models using the power consumption traces of the unprotected DES crypto-processor on the SecmatV1 SoC (System on Chip) in ASIC (Application-Specific Integrated Circuits), which is provided by DPA Contest 2008 and focuses on Differential Power Analysis or similar attacks. Here we select the first 1000 traces from secmatv1_2006_04_0809 for experimentation, with each trace containing 16 nominal DES runs. The raw signal of each trace looks like the one shown in Figure 3. To smooth the signal and eliminate apparent outliers, we take the average of ten original samples as one sample for every power consumption trace in our dataset. As the number of learners we need to create is proportional to the number of temporal samples, here we choose to use only 20 samples per trace for efficiency purpose. Experiments show that we can still achieve high-quality results based on only 20 learners.

This experiment tries to address two issues. First, we would like to know whether our method can accurately identify the correct key. Second, given that the correct key can be discovered, we want to compare our method with two popular non-learning based methods, CPA and BS-CPA, to see whether our method can find the correct key using fewer number of power traces (i.e. achieve the same quality using fewer data). For the second purpose, we gradually increase the training data size until the outputs become stable. Based on the rule from the DPA Contest, the attack is considered successful if the correct key appears and remains unchanged for 100 consecutive trace additions.

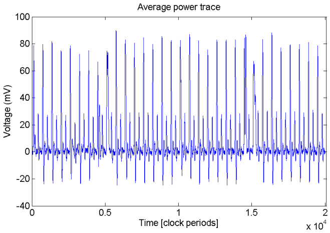


Fig. 3. A sample averaged power consumption trace, containing 20003 samples

4.1 Feature Generation

Here we first describe the encryption process of DES. In the encryption of DES, the plaintext is first permuted and divided into two 32-bit halves, L_0 and R_0. These two half-blocks are then processed by 16 identical stages called rounds. For each round, there is a 48-bit round key involved, which is a permutation of the original key and can be computed reversely. Hence, we can figure out 48 bits of the original key by revealing a round key, and then the other 8 bits can be found by exhaustive search. In every round, the right half is expanded to 48 bits and XORed with the round key. Then it is split into eight 6-bit blocks and fed to eight S-boxes, each of which has a 4-bit output and generates a 32-bit value that is then permuted again, XORed with the left half to become the new right half. For a typical implementation of the DES, the two half-blocks keep the same addresses in memory throughout the encryption. Therefore, after a round ends and before the next one begins, the register storing the right half must be replaced by a new value, which results in several bit flips and extra power consumption. By modeling the power consumption, we can derive the condition of the bit flips and eventually derive the round key. Because the eight S-boxes form a parallel structure, we can attack them one at a time. Each S-box is related to 6 bits of the round key and the four output bits are stored to some known addresses.

We take the first S-box of the first round as an example. The right half before the first round, R_0, is known and the four output bits of the target S-box can be computed by assuming a hypothetical 6-bit value as the relevant part of the round key. Therefore, we are able to find whether the bit flips for these four bits and can generate four features, each is 1 or 0, representing whether the bit flips or not.

Thus, the first feature we extract is to compute the hamming distance, which measures the corresponding bits flip between the old and new right half of register in the first or the last round of DES (Kocher et al., 1999). The second feature is to compute the hamming distance between the old and new left half of register in the first or the last round of DES (Almeida, 2008). The difference between left half and right half is that the left half inherits from previous right half directly and does not divide into 8 S-boxes. Therefore, when we attack different S-boxes in a round, each S-box has its hamming distance value. Since the output of each S-box has 4 bits, the first feature value of each S-box is between 0 and 4. On the other side, the hamming distance of left half register is always the same in a round because it inherits directly from the right half of previous round. Since there are 32 bits in the left half register, the second feature value of each round is between 0 and 32. In single-round approach, we extract those features in the last, or the 16th round, of DES. All traces and features are then normalized to zero mean and unit variance.

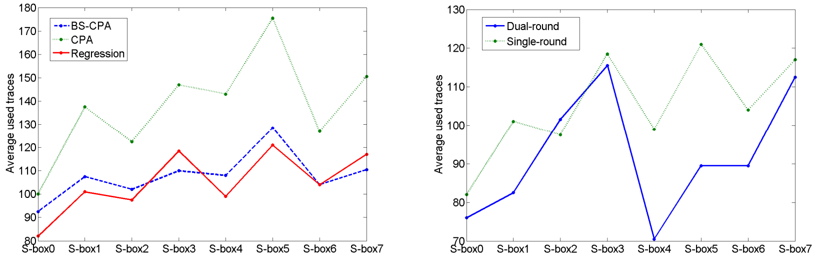


Fig. 4. On the left hand side is average traces used for each algorithm, S-box of BS-CPA, CPA, and learning-based regression methods. On the right side is the comparison of our regression-based method and the dual-round approach.

4.2 Experiment of Single-Round Approach

In order to compare the efficiency of our model, we use CPA and built-in determined sub-key CPA (BS-CPA) (Komano et al., 2010) as competitive algorithms against our model. For each competitive algorithm, we add 10 traces each time until a correct and stable key is found. As mentioned previously, we adopt the sub-key breaker to attack the sub-key (or S-box) one by one. Acknowledging the fact that the order of the traces to be added can affect the results as some training inputs are more representative, here we shuffle the order of traces each time and then average results from 20 different orders to obtain the average number of traces used for each algorithm. We depict the average traces of each method in the left hand side of Figure 4. In some cases such as S-box3 or S-box7, our method gets worse results. We believe that it is caused by the noise in the current traces or overfitting; however, our learning-based method performs better than others in most of the cases.

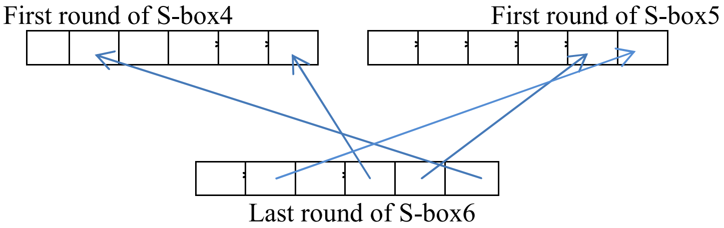


Fig. 5. Key dependency between the first round of S-box4, S-box5 and the last round of S-box6

4.3 Experiment of Dual-Round Approach

In real world scenario, sometimes there are only limited numbers of traces available. Therefore, we can resort to the dual-round approach that exploits extra multi-round information. For DES, we exploit the first and the last rounds. That is, we explore key dependency between the first and the last of nominal DES round. Once we obtain a possible key candidate in one round with high confidence, we can pass such information to the other round to reduce the search space of all possible key candidates. Before pursuing dual-round attack, we need to first observe the dependency of key bits, which can be derived from the encryption algorithm itself. For example, the bit0 and bit2 of the S-box6 in the last nominal DES round do not have corresponding key bits in the first round. The bit1, bit3, bit4 and bit5 of S-box6 in the last round have corresponding key bit positions 35, 29, 34 and 24 in the first nominal DES round respectively.

The single-round results in Figure 4 show that S-box5 is the most difficult sub-key to attack, as it requires the most traces on average. If we can use the knowledge of key dependency from another round, it is possible to reduce traces required to attack S-box5. Figure 5 shows the key dependency between the first round of S-box4, S-box5 and the last round of S-box6. We have realized that there are two overlapped keys between S-box5 in the first round and S-box6 in the last round, and another two overlapped bits between S-box6 in the last round with S-box4 in the first round. Therefore, it is possible to propagate the key bits learned in an easier S-box (e.g. S-box4) to the harder ones. We realize such idea by considering the bits in these three S-boxes altogether and use the weighted sum of the errors to evaluate the quality of certain assignment. Even though not all key bits has a corresponding mapping between the first and the last round, we still need to search all possible combinations of those non-overlapped bits. The higher the key dependency, the more likely we can use fewer traces or training examples to decipher the key.

4.4 Results of Dual Round Experiment

Here we compare the dual-round approach with the single-round approach. We focus on deciphering the S-boxes in the last nominal DES round using the dual-round attack technique because we can easily compare the results with our single-round approach. The right hand side of Fig. **4.** shows the results of regression-based single-round approach and dual-round approach. Table 1 shows the numbers of average traces required for each S-box. Not surprisingly, the dual-round approach has better performance than single-round approach in most situations. Such results demonstrate that dual-round approach can trim the search space to avoid the interference of some potential false positives because an incorrect key needs to perform well in both rounds to be selected as false positives, which is less likely to happen comparing with single-round approach.

Table 1. The experiment results of single-round approach and dual-round approach

Method	Single-round approach	Dual-round approach
Used traces	Avg.	Avg.
S-box0	82	76
S-box1	101	82.5
S-box2	97.5	101.5
S-box3	118.5	115.5
S-box4	99	70.5
S-box5	121	89.5
S-box6	104	89.5
S-box7	117	112.5

5 Conclusion

Side channel attacks play an important role in cryptography. Despite that in theory, cryptographers can design provably-secure cryptographic algorithms, these algorithms need to be implemented and carried out by computing hardware. The implementations can subject to side channel attacks no matter how secure the algorithms are in theory. This is why many industrial and governmental standards such as FIPS (Federal Information Processing Standard), CC (Common Criteria), and EMV (Europay, MasterCard, and VISA) require that compliant security products have various levels of countermeasure against side channel attacks. It is therefore crucial to understand how efficient such attacks can be with advanced techniques from, e.g., machine learning, as well as to gain some insights into how these attacks work in order to design more effective countermeasures. In this paper, we introduced a novel unsupervised, regression-based approach to perform side-channel attack. We further extend this approach to consider information from multiple rounds with promising results. We hope this paper can serve as an encouraging example to show how machine learning approaches can be carefully crafted to solve a well-known security problem.

Acknowledgement. This work was supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC101-2911-I-002-001, NSC101-2628-E-002-028-MY2 and NTU102R7501.

References

1. Kelsey, J., Schneier, B., Wagner, D., Hall, C.: Side channel cryptanalysis of product ciphers. In: Quisquater, J.-J., Deswarte, Y., Meadows, C., Gollmann, D. (eds.) ESORICS 1998. LNCS, vol. 1485, pp. 97–110. Springer, Heidelberg (1998)
2. Agrawal, D., Archambeault, B., Rao, J.R., Rohatgi, P.: The EM side–channel(s). In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 29–45. Springer, Heidelberg (2003)
3. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
4. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
5. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Examining Smart-Card Security under the Threat of Power Analysis Attacks. IEEE Transactions on Computer 51(5), 541–552 (2002)
6. Bévan, R., Knudsen, E.W.: Ways to Enhance Differential Power Analysis. In: Lee, P.J., Lim, C.H. (eds.) ICISC 2002. LNCS, vol. 2587, pp. 327–342. Springer, Heidelberg (2003)
7. Le, T.-H., Clédière, J., Canovas, C., Robisson, B., Servière, C., Lacoume, J.-L.: A proposition for Correlation Power Analysis enhancement. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 174–186. Springer, Heidelberg (2006)
8. Brier, E., Clavier, C., Olivier, F.: Correlation Power Analysis with a Leakage Model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004)
9. DPA contest (2008-2009), `http://www.dpacontest.org/home/`
10. Komano, Y., Shimizu, H., Kawamura, S.: BS-CPA: Built-in Determined Sub-key Correlation Power Analysis. In: Proceedings of IEICE Transactions (2010)
11. Lerman, L., Bontempi, G., Markowitch, O.: Side-channel attack - an approach based on machine learning. In: Second International Workshop on Constructive Side Channel Analysis and Secure Design, COSAED 2011 (2011)
12. Almeida, A.: A Simple Improvement of Classical Correlation Power Analysis Attack on DES, DPA contest (2008/2009)
13. Chari, S., Rao, J.R., Rohatgi, P.: Template Attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003)
14. Backes, M., Durmuth, M., Gerling, S., Pinkal, M., Sporleder, C.: Acoustic side-channel attacks on printers. In: USENIX, p. 20. USENIX Association, USA (2010)
15. Hospodar, G., Mulder, E.D., Gierlichs, B., Verbauwhede, I., Vandewalle, J.: Least Squares Support Vector Machines for Side-Channel Analysis. In: Second International Workshop on Constructive SideChannel Analysis and Secure Design (2011)
16. Prouff, E., Rivain, M., Bevan, R.: Statistical Analysis of Second Order Differential Power Analysis. IEEE Transactions on Computers 58(6), 799–811 (2009)

Decisive Supervised Learning

Eileen A. Ni, Da Kuang, and Charles X. Ling

Department of Computer Science
The University of Western Ontario
London, Ontario, Canada
{ani,dkuang,cling}@csd.uwo.ca

Abstract. Traditional active learning selects the most informative (e.g., the most uncertain) example and queries an oracle for the label. However, as more examples are learned in the process, even the most uncertain examples can become certain. In this case, would it be better to make predictions directly and take the consequence if the prediction is wrong, rather than asking the oracle for labels? In this paper, we propose a new learning paradigm. In contrast to the traditional active learning, the learner can obtain true labels not only by querying oracles but also by making predictions and taking the consequence. Under this paradigm, we further propose a novel algorithm named *Decisive Learner* which always chooses the most decisive action (either querying oracles or making predictions) in the learning process. Compared to other typical learners (indecisive learners, traditional active learners, conservative learners), we show empirically that our decisive learner makes fewer mistakes and incurs the smallest total costs in the learning process.

Keywords: active learning; decisive learner; decisive actions;

1 Introduction

In traditional active learning, the learner chooses the most informative example in the unlabeled pool to query the oracle for its label. By doing so, the learner is likely to achieve high accuracy by using few labeled examples. Many variants of active learning [10,13] have been proposed (see reviews in Section 2), among which uncertain sampling [12] is one of the most intuitive and commonly used strategies. In uncertain sampling, the learner selects the most uncertain example which is closest to the decision boundary. By halving the version space, uncertain sampling can make the learning process converge quickly.

Most previous works on active learning assume the true labels can only be obtained by querying oracles. However, in many real world applications, labels can be acquired (revealed) alternatively by making predictions directly and taking the consequence. For example, when a learning system sorts letters by using OCR (optical character recognition) devices of the post office, if the hand-written codes are ambiguous, or too difficult to recognize, they will be passed to the oracles (human) for labels. However, if the OCR can predict accurately the handwritten zip codes, the letter will be sorted and mailed to the recipient directly.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 426–437, 2013.

If the prediction is not correct, the letter will be returned and redelivered (cost or consequence of the wrong prediction).

In the above example, the acquired true labels can be given to the learner (OCR system) to further train the predictive model. We can see, besides querying oracles, making predictions provides another way to obtain the true labels in the learning process. In fact, this type of problems is ubiquitous in the real world. In order to effectively deal with the problems, we propose a new learning paradigm, where we consider two actions for acquiring true labels in the learning process. One action is to query oracles (e.g., human or experts). Certain cost has to be paid for each query. The other action is to make predictions (e.g., directly mailing the letter). After the prediction, the true label will be revealed. If the prediction is correct, no cost is paid; otherwise, misclassification cost has to be paid. To determine which action to choose for an example, we should compare the query cost and the misclassification cost. Since the correct action is unknown before each action is taken, we use *expected misclassification cost.*

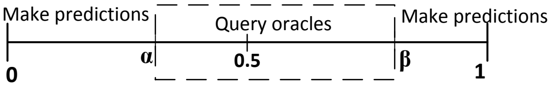


Fig. 1. Probability interval to query oracles. For the examples with the probability in $[\alpha, \beta]$, the learner will query oracles; otherwise, the learner will make predictions.

Based on the querying cost and the expected misclassification cost, we can easily derive a probability interval as $[\alpha, \beta]$ (see more details in Section 3.2) shown in Figure 1. For the examples with the probability falling in the interval, the learner will choose to learn them by querying an oracle for its label; otherwise, the learner will choose to make predictions on them directly. The interval is, in fact, the same as the rejection interval in the classifiers with reject option [6,2], where the learner can reject to make predictions on an example when it is not certain about it. However, unlike our paradigm, it does not make predictions during the learning process.

During the learning process of our new paradigm, if the posterior probability produced by the learner is accurate, then the action taken on each example will be always correct and optimal. Therefore, the action order in the learning process will not matter. When all unlabeled examples are learned, the learner can always learn the same predictive model. However, in reality, the posterior probability may not be very accurate particularly in the beginning of the learning. For those examples close to α or β, it is likely that wrong actions will be taken, which may consequently lead to a higher cost. We call those examples *indecisive examples*, and the actions taken on them *indecisive actions.* We call the examples far away from α and β *decisive examples*, and the actions taken on them *decisive actions.*

What is the optimal order of actions in the case where the probability is not perfectly accurate? To tackle the problem under the new paradigm, in this paper, we propose a novel learning algorithm, called *Decisive Learner* (*DL*).

DL always takes the most decisive actions and attempts to make as few mistakes on the actions as possible in the learning process. As more examples are learned and the learner becomes more reliable, the indecisive examples may become decisive, thus fewer wrong actions will be taken. We also empirically study the performance of DL on 10 real-world datasets, by comparing it with other typical learners under different cost settings. The experimental results demonstrate that DL has an overall best performance in terms of the total cost.

The main contributions of this work are in two folds.

1. We propose a new learning paradigm, where the learner can take two actions (querying oracles and making predictions). Different from traditional active learning, in addition to querying oracles, the learner can acquire the true labels of the examples by making predictions and taking the consequence.
2. Under the new paradigm, we propose a novel learning algorithm to find the optimal action sequence. As a result, the total cost (querying cost and misclassification cost) is minimized in the learning process.

The rest of the paper is organized as follows. In Section 2, we will review some related work and discuss the difference with our work. In Section 3, we will introduce some preliminary for our problem setting and the new concept of decisive action. Section 4 will talk about the proposed algorithm to select actions. In Section 5, we will experimentally compare our algorithm with other typical learning algorithms. Discussion and future work will be presented in the last section.

2 Related Work

Our paradigm is bridging between the traditional active learning and classification with rejection. It is also similar to two-oracle setting in active learning. In this section, we will discuss the similarities and differences with them.

In traditional active learning, learner tends to choose the most informative example in the unlabeled pool to query oracle for its label. Uncertainty sampling [10] is one of the most intuitive and commonly used active learning strategies. It selects the most uncertain example which is closest to the decision boundary, which is one option in our learner DL.

As we mentioned in Section 1, traditional active learning has only one option to obtain the true labels, which is to query oracles. Most of the previous works assume that there exists at least one oracle who can provide the labels of the examples. In [15], the assumption is that we have one perfect oracle who can correctly give all labels. [14,5] study the setting where the oracle is noisy and may mislabel the examples. [4,14] explore the case where multiple oracles or labelers may contribute to the quality of the labels. We can see in those works querying oracles (human experts or labelers) has been regarded as the only approach to retrieve the true labels.

In our paradigm, we can have two options (actions) to acquire the true labels. Besides querying oracles, the learner can make predictions directly and the true label will be revealed from the feedback (success or failure). Thus, we need to consider not only the cost to query oracles but also the cost of the wrong predictions. The best learning strategy might not be always selecting the most informative (or uncertain) examples as in the traditional active learning. Our goal is to explore the best learning sequence that minimizes the total cost under this new paradigm.

For the classification with rejection (also known as abstaining classifiers), the learner can reject to make predictions on the uncertain examples. The decision when to reject to make predictions also relies on the ratio between the misclassification cost and the reject cost. [7] studies how to effectively reduce the misclassification rate without considering the cost ratio. However, the existing works on classification with rejection only study how to minimize the cost given a predictive model, while in our paradigm we care more about the learning process that builds the predictive model with the minimal total cost.

In our paradigm, since making predictions can reveal the true labels, it can be regarded as another oracle. However, our paradigm is substantially different from the two-oracle setting [4] in active learning. In our paradigm, the "oracle" (the model for making predictions) is updated with each new labeled example, while the two oracles in the two-oracle setting are static.

3 New Learning Paradigm

In this section, we will first formally define our paradigm setting where the goal is to minimize the total cost during the learning process, and then introduce a new concept named *decisive action* in detail.

3.1 Paradigm Setting

Given a set of labeled data L, a set of unlabeled data U and a learner M learned from L, M is allowed to select examples from U, retrieve the labels from an oracle O and update its model iteratively. Given an example in U, its label can be obtained by taking one of the two actions. The first action is to query the oracle O for its label by paying the querying cost C_q. The second action is to make a prediction (positive or negative) on the example. The consequence of the prediction will reveal the true label. If the prediction is wrong, we have to pay the misclassification cost C_{FN} for false negative and C_{FP} for false positive predictions. For each example with a posterior probability produced by the learner, the action to take can be determined. The goal of this learning problem is to find a proper learning sequence for the examples from U, such that the total cost is minimized.

3.2 Choice of Actions

For an example x in U, how to choose the action depends on the costs of the two possible actions. Given the probability of being positive $p(1|x)$ predicted by the learner, the expected misclassification cost will be $P(1|x) \times C_{FN}$ if $0 \leq P(1|x) \leq 0.5$, or $(1 - P(1|x)) \times C_{FP}$ if $0.5 < P(1|x) \leq 1$, where C_{FN} and C_{FP} are the costs for false negative and false positive. If the expected misclassification cost is less than the cost of querying the oracle, we should make the prediction directly; otherwise, we should query the oracle for the label. Therefore, we can calculate the values of α and β in Figure 1 of Section 1: $\alpha = C_q/C_{FN}$ and $\beta = 1 - C_q/C_{FP}$. Without loss of generality, we transform Figure 1 into Figure 2 and look close into the region of [0.5, 1.0]. Instead of $P(1|x)$, the horizontal axis in Figure 2 changes to $P(d|x)$. Here, d is the prediction (0 or 1) made by the learner, which depends on the higher probability of $P(0|x)$ and $P(1|x)$. If $P(0|x) > 0.5$, then $d = 0$; otherwise, $d = 1$. In the following, we will introduce two concepts that are related to the choice of actions.

The first concept we will introduce is *action boundary*. In Figure 2, due to the similarity, we only look into the threshold $\beta = 1 - C_q/C_{FP}$ for $P(d|x)$, instead of two thresholds α and β in Figure 1. For examples with $0.5 \leq P(d|x) < \beta$, we should query the oracle; while for examples with $\beta \leq P(d|x) \leq 1$, we should make predictions. We call the threshold β *action boundary*. The position of β is not necessarily in the center of the axis, instead it relies on the cost of querying the oracle and the misclassification cost. If the oracle is too expensive to query, the value $1 - C_q/C_{FP}$ will be small, and thus β will be closer to 0.5. If the wrong prediction is costly, β will be closer to 1.

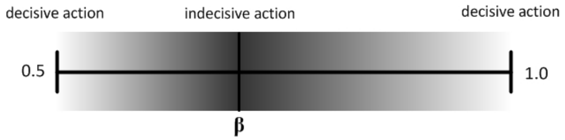


Fig. 2. Illustration for action boundary as well as decisive and indecisive action. The horizontal axis represents $P(d|x)$.

The correct choice of the actions depends on the accuracy of the posterior probability $P(d|x)$. Here, the accurate probability means that the probability is perfectly calibrated [16]. Different classifiers have different calibration behaviors. For example, the scores produced by *naive bayes*, *decision tree* and *SVM* are not calibrated, while *bagged decision tree* and *random forest* can produce calibrated probabilities [11,16]. In addition, the insufficiency of training data may also lead to the inaccuracy of the probability, particularly in the beginning of the learning process when the learner does not observe enough examples. If a classifier can produce perfectly calibrated probability $P(d|x)$, then the action taken on each example in U will always be the correct choice.

The second concept, in terms of the choice of actions, is *Indecisive* and *Decisive Actions*. The inaccuracy of the posterior probabilities $P(1|x)$ will easily lead to the wrong choice of the actions, particularly in the boundary area close to β as shown in Figure 2. For the boundary examples, the learner is not sure which action to take. Therefore, we call those boundary examples *indecisive examples*, and the actions taken on them *indecisive actions*. For the examples far away from the action boundary β (approaching 0.5 or 1), the learner is more certain about which action to take and the actions taken on them will be less likely to be mistaken. Hence, we call those examples *decisive examples*, and their actions *decisive actions*[1]. In fact, there is no clear threshold to distinguish decisive and indecisive actions. In Figure 2, we use the darkness to demonstrate the decisiveness of the actions. The darker the color of the area, the less decisive the actions taken in the area. We can see that the decisiveness of the actions gradually decreases from β to the two ends (0.5 or 1). In Figure 2, the decisiveness of actions looks similar to the uncertainty of examples, but they are different. The probabilities of uncertain examples are close to 0.5, while the probabilities of decisive actions varies within [0.5, 1] depending on C_q and C_{FP}.

Based on the decisiveness of action, in the next section, we will propose a new learning strategy to minimize the total cost.

4 Decisive Learner

The key issue to the learning problem defined in Section 3 is how to correctly determine the learning sequence on the examples in U such that we can minimize the total cost. In the following, we will propose a novel learning strategy that selects examples from the most decisive to the most indecisive.

4.1 Algorithm

We design a novel learner called *Decisive Learner* (*DL*). The basic idea of DL is that the decisive examples take precedence to be selected for learning since the actions taken on them are more likely to be correct, and the indecisive examples will be left for learning later. As we mentioned in Section 1, when more examples are observed by the learner, the model built will become more accurate, the indecisive examples may become decisive, and consequently actions will be less likely to be mistaken. There are two types of decisive examples: examples (close to 0.5) to query the oracle and examples (close to 1) to make predictions. Both of the two types examples are beneficial for the learner. The examples with probabilities close to 0.5 can help the learner achieve high accuracy with few examples. Direct prediction on the examples with probabilities close to 1 makes good use of the current learner and is likely to obtain the true labels without any cost. Moreover, those (certain) examples can make the learner more robust. Therefore, in our algorithm, we take advantage of both types of examples by alternating them.

Specifically, the algorithms of decisive learner can be decomposed into the following four steps.

[1] In this paper, for each selected example an action will be taken, thus we regard taking actions and selecting examples equivalently.

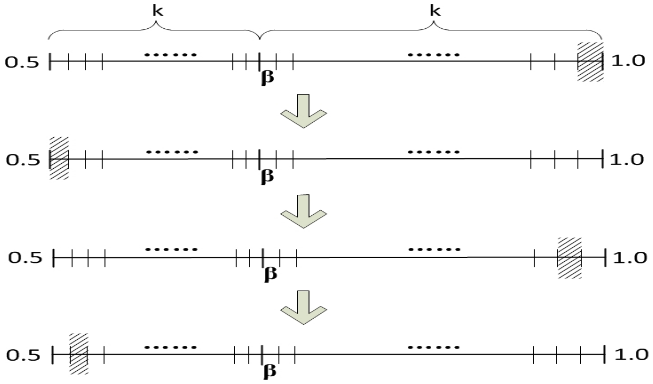


Fig. 3. Illustration of the learning process for the decisive learner (DL). DL learns the examples in the intervals (shadowed) alternately on the two sides of the action boundary β, gradually approaching β. The actions are taken from the most decisive to the most indecisive.

1. **Splitting Probability Interval**: Each of the probability ranges $[0.5, \beta)$ and $[\beta, 1.0]$ is split into k equal intervals. Each example in U falls in an interval according to the posterior probability $P(d|x)$ predicted by the current learner.
2. **Selecting the Starting Interval**: The learner chooses the most decisive interval which is furthest from the action boundary β to start the learning.
3. **Learning in an Interval**: The current learner checks if there is any example in U located in the current interval. If yes, we select the most decisive example in the current interval, acquire its label by taking the corresponding action and update the learner, and then repeat step 3; otherwise, we go to step 4.
4. **Alternating Interval**: If all examples in U are learned, we terminate the learning; otherwise, the learner selects an interval on the other side of β as the next interval, and then go back to step 3 to learn examples in it.

From the algorithm of DL, it is clear that the learner always learns the most decisive examples and attempts to make as few mistakes on the actions as possible. This feature ensures that DL achieves a good performance in terms of the total cost.

5 Empirical Studies

In this section, we will empirically study the performance of DL in terms of the total cost. We will compare it with other three typical learners.

5.1 Datasets and Cost Ratios

We will evaluate the performance of DL on 10 UCI datasets [1], including abalone, adult-census, anneal, credit-g, diabetes, nursery, sick spambase, splice and waveform, with the size ranging from 898 to 32561.

To be more comprehensive, our evaluation will be conducted under different cost ratios (the misclassification cost C_{FN} and C_{FP} over the oracle querying cost C_q). As different values of C_{FN} and C_{FP} will not affect the comparison results, we let $C_{FN} = C_{FP} = C_m$, and choose three cost ratios, $C_m/C_q = 2.5$, $C_m/C_q = 4$ and $C_m/C_q = 10$. Since $\beta = 1 - C_q/C_m$, the corresponding action boundary β are 0.6, 0.75 and 0.9. The reason we choose the minimum cost ratio as 2.5 is that we should make sure $C_m/C_q \geq 2$; otherwise, for any $P(d|x)$, $C_m \times (1 - P(d|x)) < 2 \times C_q \times 0.5 < C_q$, meaning that making predictions is always better than querying oracles regardless of $P(d|x)$ and thus we do not need to conduct our research.

5.2 Other Learners

In our experiments, DL will be compared with other three typical learners with different learning sequence under our learning paradigm. We will give a brief introduction of them in this subsection.

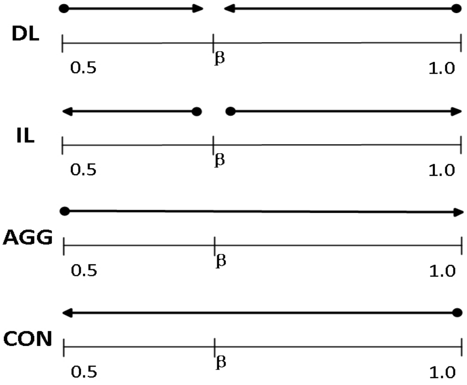


Fig. 4. Illustration of the learning process for four learners. DL is the decisive learner. IL is the indecisive learner, opposite of DL. AGG is the aggressive learner which is the traditional uncertain sampling. CON is the conservative learner which learns in the same sequence as self-directed learning.

1. **Indecisive Learner**: The first learner is named indecisive learner (*IL*), which takes the opposite learning sequence of DL (see Figure 4 for illustration). It always selects the most indecisive examples in the learning process. Specifically, it has the same interval splitting strategy as DL and also alternates the intervals on both sides of β. The only difference is that it starts from the closest interval to β where examples are indecisive. It can be expected that the learner will make many mistakes on actions, especially at the beginning of the learning process.
2. **Aggressive Learner**: Aggressive learners (AGG) are those that choose the most challenging example (i.e., the example that is least certain by the current learner), to learn in each iteration. It is the same as *uncertainty sampling*

[15] in the traditional active learning. It always gives preference to the example that is closest to the decision boundary and queries the oracle for its label.

3. **Conservative Learner**: In contrast to the aggressive learner, conservative learner (*CON*) always exploits the data it can predict well and tries to make as few mistakes on the predictions as possible. Thus, it prefers to learn the examples with the posterior probability $P(d|x)$ close to 1. The learning sequence of conservative learner is the same as self-directed learning [9].

Furthermore, for all the four learners (DL, IL, AGG and CON), we use *bagged decision tree* as the base classifier, due to its well-calibrated posterior probability as we mentioned in Section 3.2.

5.3 Experimental Setting

For each of the 10 UCI datasets, we randomly select 100 examples as the labeled set, and use it to train the initial classifier for each of the four learners. The rest of the examples belong to the unlabeled set. For the four learners mentioned in Section 5.2, we calculate the total cost (the misclassification cost and the cost of querying the oracle) spent in the entire learning process. The less the cost, the better the learner. We run the four learners on the 10 datasets under the three cost ratios (Section 5.1) for 10 times. Friedman test and Wilcoxon signed-rank test will be chosen to statistically test the difference of the total cost of the four learners. It should be noted that after learning all the unlabeled examples, the four learners should have the same predictive model, since the model is built on the same set of examples.

5.4 Statistic Testing Methods

The total cost in each repeat can be affected by the initial split of the dataset, thus the cost may have large variance in different repeats and even the data itself may not be normally distributed. In this case, Friedman test can be a reasonable choice for our statistic testing, since it uses the ranks of the data rather than their raw values to calculate the statistic. Friedman test has been widely used to test whether there is a statistically significant difference between a group of values [3,8]. If significant difference exists in the group, we still need a post-hoc test on different pairs of groups to report their statistical difference. Wilcoxon signed-rank is a commonly used post-hoc test following Friedman test [3,8], thus we will use it in our experiment.

5.5 Comparative Results

Table 1 demonstrates the average total costs of the four learners on the 10 UCI datasets, under three cost ratios. In order to evaluate the statistical difference, we also calculate the ranking of the four learners based on Wilcoxon signed-rank test. The ranking is calculated by the following steps. For each row in the table, we first sort the four learners by their average costs ascending. Then we compare the learner with the smallest mean to the one with the largest mean. If there is no significant difference, all the learners will be ranked as 1; otherwise, we continue to compare the smallest mean with the second largest mean, until all the learners have been compared or no significant difference is found. In the next round, we will compare the second smallest mean with the largest mean, and repeat the same step. The process iterates until all learners are ranked.

In Table 1, the rank of each learner is presented in the bracket next to the average total cost. The four rows in the bottom of Table 1 present the average rank of the four learners over all the 10 datasets under the cost ratio (2.5, 4 and 10) respectively, as well as the overall average rank over 30 rows in the table. We can see clearly that DL is top ranked in 27 out of the total 30 comparisons and has the lowest average rank 1.1 over the 30 rows. The average costs on all the 10 datasets are also presented in Table 1, and we can see that DL is much better than the other three learners. Both the rank and the average costs illustrate that DL has the overall best performance in terms of the total cost.

Theoretically, IL is supposed to have the poorest performance since it always selects the most indecisive actions and is expected to make many mistakes. However, we observe that it is not exactly the case in Table 1. Overall, the average rank (2.4) of IL is slightly better than that (2.5) of AGG. A closer look reveals that under the cost ratio 4 and 10, IL indeed has the lowest rank among the four learners. The slight superiority of IL to AGG is due to the fact that IL performs much better than AGG when the cost ratio is 2.5. Under the cost ratio 2.5, β (1-1/2.5=0.6) is relatively close to 0.5. IL starts learning from the examples with probability around 0.6 while AGG from the examples with probability close to 0.5. Although the examples selected by IL are not as informative as those selected by AGG, they are still useful for the learner. Furthermore, IL can even save more costs by directly making predictions on the examples to the right side of β, as the expected cost of making predictions on those examples is lower than the cost to query the oracle ($(1 - P(d|x)) \times C_m < C_q$, where $P(d|x) > \beta$).

From the average rank in Table 1, we can also observe that DL is more likely to have better performance when the cost ratio is high, as its average rank increases when cost ratio becomes higher. It means when the misclassification cost is much higher than the querying cost, it is safer and more desirable to use DL as the learner.

6 Conclusion

This paper proposed a new learning paradigm where the learner is able to take two possible actions (querying oracles and making predictions) to acquire true labels for new examples. The new paradigm is different from the traditional active learning where oracles are the only source to obtain true labels. Under the new learning paradigm, we proposed a novel learning algorithm named decisive learner (DL), which always selects the most decisive examples and makes as few mistakes on the actions as possible in the learning process. In the experiments,

Table 1. Statistical comparisons between the four learners in terms of the cost. Each cell shows the average cost and its rank (in the bracket) of a specific learner on a dataset under a cost ratio. The rank is calculated by a statistical test named Wilcoxon signed-rank. The cells in bold represent the winner(s) in the corresponding row.

	Cost ratio	DL	IL	AGG	CON
abalone	2.5	**889(1)**	955(2)	1031(3)	**891(1)**
	4	**1081(1)**	1313(3)	1238(2)	1341(3)
	10	**1461(1)**	1520(2)	**1410(1)**	1723(3)
adult-census	2.5	**4803(1)**	4904(2)	5511(3)	4813(1)
	4	**5893(1)**	7178(3)	6887(2)	7102(3)
	10	**8097(1)**	9228(3)	9121(2)	9199(2)
anneal	2.5	**66(1)**	75(2)	88(2)	**58(1)**
	4	**102(1)**	124(2)	124(2)	**101(1)**
	10	**130(1)**	210(3)	194(3)	167(2)
credit-g	2.5	**268(1)**	282(2)	308(3)	**271(1)**
	4	**331(1)**	397(3)	350(2)	396(3)
	10	**375(1)**	400(2)	380(2)	445(3)
diabetes	2.5	**195(1)**	204(2)	211(2)	**194(1)**
	4	**234(1)**	262(2)	**241(1)**	274(2)
	10	**290(1)**	**294(1)**	**285(1)**	**309(1)**
nursery	2.5	**273(1)**	**276(1)**	385(3)	285(2)
	4	**349(1)**	436(3)	497(4)	391(2)
	10	**610(1)**	879(3)	802(2)	764(2)
sick	2.5	**84(1)**	**93(1)**	**93(1)**	**83(1)**
	4	**122(1)**	143(2)	**132(1)**	137(2)
	10	**217(1)**	258(2)	254(2)	270(2)
spambase	2.5	416(3)	**364(1)**	628(4)	389(2)
	4	**554(1)**	704(3)	891(4)	623(2)
	10	**753(1)**	1436(3)	1415(2)	1335(2)
splice	2.5	391(2)	**322(1)**	625(3)	**333(1)**
	4	**557(1)**	673(2)	802(3)	**535(1)**
	10	**688(1)**	1094(2)	1077(2)	1055(3)
waveform	2.5	**640(1)**	**624(1)**	940(2)	**614(1)**
	4	**847(1)**	959(2)	1101(3)	971(2)
	10	**1039(1)**	1503(3)	1341(2)	1952(4)
average cost	overall	1058(1)	1237(2)	1278(3)	1234(2)
average rank	2.5	1.3	1.5	2.6	1.2
	4	1	2.5	2.4	2.1
	10	1	2.4	1.9	2.4
	overall	1.1	2.1	2.3	1.9

we demonstrated the outstanding performance of DL in reducing the total cost, compared to three other typical learning strategies under the same learning paradigm. The proposed learning algorithm (decisive learner) can be applied to various real-world applications, such as optical character recognition and online advertising.

References

1. Newman, D., Asuncion, A.: UCI machine learning repository (2007)
2. Bartlett, P., Wegkamp, M.: Classification with a reject option using a hinge loss. The Journal of Machine Learning Research 9, 1823–1840 (2008)
3. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30 (2006)
4. Donmez, P., Carbonell, J.G.: Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, pp. 619–628. ACM, New York (2008)
5. Du, J., Ling, C.X.: Active learning with human-like noisy oracle. In: Proceedings of the 2010 IEEE International Conference on Data Mining, ICDM 2010, pp. 797–802. IEEE Computer Society, Washington, DC (2010)
6. Du, J., Ni, E., Ling, C.: Adapting cost-sensitive learning for reject option. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, pp. 1865–1868. ACM (2010)
7. Fumera, G., Roli, F., Giacinto, G.: Reject option with multiple thresholds. Pattern Recognition 33(12), 2099–2101 (2000)
8. García, S., Herrera, F.: An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons. Journal of Machine Learning Research 9, 2677–2694 (2008)
9. Goldman, S.A., Sloan, R.H.: The power of self-directed learning. Mach. Learn. 14(3), 271–294 (1994)
10. Lewis, D., Gale, W.: A sequential algorithm for training text classifiers. In: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 3–12. Springer-Verlag New York, Inc. (1994)
11. Niculescu-Mizil, A., Caruana, R.: Predicting good probabilities with supervised learning. In: Proceedings of the 22nd International Conference on Machine Learning, ICML 2005, pp. 625–632. ACM, New York (2005)
12. Settles, B.: Active learning literature survey. Sciences New York 15(2) (2010)
13. Settles, B., Craven, M.: An analysis of active learning strategies for sequence labeling tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1070–1079. Association for Computational Linguistics (2008)
14. Sheng, V.S., Provost, F., Ipeirotis, P.G.: Get another label? improving data quality and data mining using multiple, noisy labelers. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2008, pp. 614–622. ACM, New York (2008)
15. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. 2, 45–66 (2002)
16. Zadrozny, B., Elkan, C.: Transforming classifier scores into accurate multiclass probability estimates. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 694–699. ACM (2002)

Learning Overlap Optimization for Domain Decomposition Methods

Steven Burrows[1], Jörg Frochte[2], Michael Völske[1],
Ana Belén Martínez Torres[2], and Benno Stein[1]

[1] Web Technology and Information Systems,
Bauhaus-Universität Weimar, 99421 Weimar, Germany
{steven.burrows,michael.voelske,benno.stein}@uni-weimar.de
[2] Electrical Engineering and Computer Science,
Bochum University of Applied Science, 42579 Bochum, Germany
{joerg.frochte,ana.martinez}@hs-bochum.de

Abstract. The finite element method is a numerical simulation technique for solving partial differential equations. Domain decomposition provides a means for parallelizing the expensive simulation with modern computing architecture. Choosing the sub-domains for domain decomposition is a non-trivial task, and in this paper we show how this can be addressed with machine learning. Our method starts with a baseline decomposition, from which we learn tailored sub-domain overlaps from localized neighborhoods. An evaluation of 527 partial differential equations shows that our learned solutions improve the baseline decomposition with high consistency and by a statistically significant margin.

Keywords: domain decomposition, numerical simulation.

1 Introduction

Numerical simulation in product development has become a standard. It is used in various applications such as semiconductor manufacturing [2], crash-test simulation [8], and fluidic system design [11]. Numerical simulation can also be supported by machine learning for the purpose of approximating solutions [14]. In this setting, a margin of error is tolerated in return for predictions from learned models that bypass on-the-fly simulation.

A key challenge in numerical simulation concerns the efficiency and stability of parameterized numerical simulation methods. These methods are often difficult to deploy for end-users. As a result, the most robust and parameter-*independent* methods (such as direct solvers for linear systems) are often preferred over the most efficient and parameter-*dependent* methods (such as domain decomposition and iterative solvers) in many engineering contexts. Our key idea is that the parameters in this latter group (for example, the overlaps of sub-domains when simulating partial differential equations) can be learned to converge the ease of use towards the parameter-independent methods. A consequential benefit is that the efficiency of the learned solutions can be increased.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 438–449, 2013.

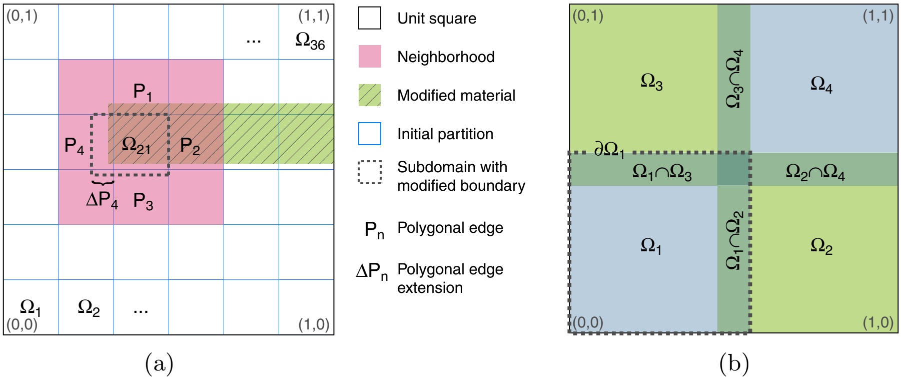


Fig. 1. (a) Our strategy for implementing a learnable domain decomposition problem. **(b)** A smaller problem showing overlaps (e.g., $\Omega_1 \cap \Omega_2$) and boundaries (e.g., $\partial\Omega_1$).

Parallelizing the simulation of models using domain decomposition in natural and engineering sciences is based on partial differential equations on a given domain. The approach requires the decomposition of a domain $\Omega \subset \mathbf{R}^m$, $m \in \{1, 2, 3\}$, into some number of sub-domains Ω_i, $i = 1..n \in \mathbf{N}$. For overlapping domain decomposition methods, the size of the sub-domain overlaps influence the stability and computational cost of the approach. Finding a near-optimal choice of parameters for optimizing the overlaps is an open problem. In many applications it is chosen with human intuition and experience using only a global setting. In this paper, we demonstrate that when using machine learning we can automate the choice of these parameters with local settings.

Our approach is to improve the efficiency of a default checkerboard sub-domain pattern such as the example shown in Figure 1a. Here, we consider the properties of the boundaries of each sub-domain as features. When dealing in two dimensions, each sub-domain Ω_i contains interesting relationships with the adjacent sub-domains $\{\Omega_i, i = 1..36 \mid \partial\Omega_{21} \cap \Omega_i \neq \emptyset\}$ that we may capture. This is represented in Figure 1a as the red *neighborhood* of nine sub-domains. In this example, a modified material setting passes through most of sub-domain Ω_{21}, and we should extend some of the sub-domain boundaries such that this material boundary is not too close to the initial partition. So for example, area boundary P_4 has been extended by some amount ΔP_4 so that the modified western sub-domain boundary is not too close to the material boundary.

For the purposes of machine learning, we are interested to learn the relative computational costs of the neighborhoods and then combine this knowledge to reduce the computational costs of the whole domain. To do this, the numerical simulation of the domain is computed for several variations of the neighborhood, and the relative improvement or degradation of the computational cost is recorded. So in the case of Figure 1a, we have 36 neighborhoods to consider including some interesting cases around the perimeter. The neighborhood features that we wish to capture include material regions that cross sub-domain boundaries or have close proximity to these boundaries. Therefore the mapping between these features and the computational cost can be cast as a regression problem, and we wish to learn solutions that reduce the cost.

Our contributions in this paper are summarized as follows: (1) We propose a machine learning approach for automating and optimizing the overlap construction for domain decomposition methods. (2) We create a unique problem set of interest to both novice and expert users. (3) We develop and apply a novel taxonomy of the feature space in our problem setting. (4) We devise a machine learning framework for overlap optimization including a training corpus and an appropriate performance measure.

The remainder of this paper is organized as follows. In Section 2 we provide the necessary background on partial differential equations and domain decomposition. In Section 3 we give the details of our methodology including data and the evaluation measure. In Section 4 we compare the results of our method to an expert human baseline. Finally in Section 5 we offer concluding remarks.

2 Solving PDEs Using Domain Decomposition

Numerical methods for solving any partial differential equation (PDE) [3] tend to involve a huge number of unknowns, especially in three dimensions. The power of a single computer is often no longer sufficient to solve the resulting equations. In this case, parallel computing with domain decomposition is one of the most successful strategies to solve these equations efficiently [10,12].

A PDE-based model consists of four components: the equations, the related parameter sets (or material properties), the domain these equations are solved on, and the boundary values given for the domain. The goal of domain decomposition is to split the domain into smaller sub-domains and iterate to coordinate the merging of the solution between these sub-domains.

Schwarz and other domain decomposition methods have overlapping and non-overlapping variants for solving boundary-value problems [10,12]. If a domain decomposition method is overlapping, then some portion of each problem is solved redundantly. Conversely if a domain decomposition method is non-overlapping, then the sub-domain boundaries are only touching. Most overlapping methods are categorized as additive or multiplicative, concerning the transfer data from one sub-domain to another. Additive methods have better properties concerning parallelization than multiplicative ones.

The overlapping additive Schwarz method, as utilized in this paper, is a simple and robust approach for applying domain decomposition. With sufficient overlap as demonstrated in Figure 1b, it can be applied to nearly every PDE. A disadvantage of this method is its slow convergence. For example, faster but more complicated non-overlapping methods such as FETI approaches [12] exist for some structural mechanic problems. These approaches can work without overlaps on the application domains, but they are less robust. They are not suitable, for example, for many fluid mechanic problems where overlapping Schwarz methods are also applicable. Nevertheless, all overlapping domain decomposition methods share the same basic demands and requirements that we want to solve in this paper, so our results are broadly applicable.

The parameters of domain decomposition are the positions of the introduced artificial boundaries and thus the size of the overlap. This means that the efficiency of domain decomposition is highly parameter-dependent since each new sub-domain creates artificial boundaries. An artificial boundary introduces errors arising from the domain decomposition procedure, which in turn leads to more iterations. The numerical effect of an artificial boundary diminishes when moving from the boundary to the inner part of a sub-domain. Thus, a bigger overlap leads to more information exchange between the sub-domains and therefore to artificial boundary conditions that are more closely related to the solution of the PDE, which leads to faster and more stable convergence. Beyond this, it is known that jumps in the material parameters influence convergence behavior if they occur next to the artificial boundaries.

In summary, a bigger overlap will decrease the number of iterations, however this will increase the size of the sub-domains and the computational overhead of the domain decomposition approach. This is a critical trade-off that influences the division of Ω into sub-domains. Beyond this trade-off, the number of computation units must be kept in mind when applying parallelization technology. For a computing cluster with m units, at least m sub-domains are desired, otherwise domain decomposition is not used to its fullest potential. However, we are less interested in obeying this technical constraint in this work, as our goal is to instead evaluate performance with a generalized strategy that is independent of specific computing infrastructure.

3 Overlap Optimization Learning Approach

In this section, we first describe our general problem specification, and the approaches for generating the associated data and feature sets. Following this, we provide the details of our evaluation measure and describe our learning objective together with the methodology that we deploy.

3.1 Problem Definition

As an example PDE for solving in this paper, we use Poisson's equation, which is a prototype of so-called elliptic PDEs of second order, with some Dirichlet boundary conditions:

$$-\,\varepsilon(x)\nabla^2 u = f(x) \text{ on } \Omega\ ;\ u = g(x) \text{ on } \partial\Omega \tag{1}$$

This equation has application in modeling stationary heat, and we use it as an example to motivate our work. Consider Ω as the geometry on which we want so solve the heat equation such as a bar, $f(x) \geq 0$ as heat sources, $\varepsilon(x)$ as the material property, and $g(x)$ as known temperatures on the boundary $\partial\Omega$ of the domain Ω. A direct connection of two materials in a model could represent an ε-jump, and a blending of materials could represent a smoother ε-transition. We note that Poisson's equation is a quite simple PDE, but there are additional applications in Newtonian gravity and electrostatics, which is why we use it in this paper. The results concerning machine learning and domain decomposition achieved on this example can easily be transferred to other problems with a similar behavior, such as stress modeling used in engineering science. However, there are some models arising in fluid dynamics, for example, that behave differently concerning domain decomposition. These are less stable and need more care concerning parameter fitting. Transferring our approach to these models might need more work, but there are bigger benefits to gain because it is harder for humans to fit the model parameters.

For solving PDEs, domain decomposition can be applied to numerical methods such as spectral methods [5] and finite volumes [13]. We concentrate on the finite element method (FEM) [3], which is a standard method in most engineering software solutions. This problem is applied on the unit square using finite elements with continuous piecewise linear base functions on a regular triangulation, thus in Equation 1 we have $\Omega = [0,1] \times [0,1]$. Apart from the unit-square restriction, we only use rectangular partitioning in order to constrain the initial problem space. Notice that for the unit square with a structured grid, the checkerboard pattern with equal-size squares is a common and good default choice for the sub-domains. In this setting, the sub-domains all contain the same number of unknowns, and they have a good ratio between area and boundary.

3.2 Approach for Generating Diffusion Specifications

Each diffusion specification, or set of material values within the unit square to solve Poisson's equation, is a unique problem. This can imply that each problem must be learned individually and that results cannot carry over between different diffusion specifications. To address this, we are interested in learning patterns from neighborhoods of sub-domains as per Figure 1a, so that the knowledge about common patterns can be reapplied to whole problems. In this respect, we develop a deterministic algorithm for producing a large dataset of diffusion specifications for the neighborhood patterns to be learned from.

Our dataset is based on the placement of shapes with different material values in the domain, whereas in every shape the material value is constant. The shapes are circles and squares of various sizes that fit within the unit square or are truncated at the boundary. The shapes are arranged in one of three patterns with up to four shapes each: The Nested pattern uses a nested arrangement of shapes where the first shape is the largest and all successive shapes are included within. The Isolated pattern is comprised of stand-alone shapes without overlap or connection. The Sequence pattern uses different shapes arranged from a start point in a specific straight-line direction that can be just in contact or overlapping. The patterns, shape positions, and shape sizes are selected with a pseudo-random number generator with a deterministic sequence and fixed seed to make the data reproducible. Note also that the last-defined material setting takes precedence in the case of overlap. In general, one can expect that a skilled human will often be able to do a better job than machine learning for simple two-dimensional problems such as the Isolated case. But for more complicated problems in two dimensions (such as Nested and Sequence problems) and certainly in three dimensions, which is a typical application area for domain decomposition, machine learning will be helpful for both novice and expert users.

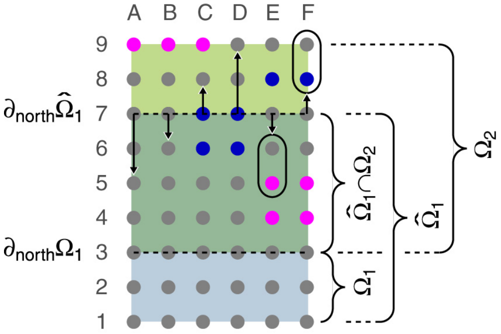
3.3 Approach for Generating Domain Specifications

Considering any checkerboard organization of sub-domains with uniform overlap, our goal is to improve on this baseline by learning from various permutations. Specifically, we can learn from permutations when the boundaries of the sub-domains are extended, retracted, or left alone in the north, east, south, and west directions. Since our goal is to learn from individual 9-region neighborhoods by modifying the boundaries of the central sub-domain, we can apply boundary modifications to each sub-domain in isolation. For example, when considering a uniform overlap of 0.4%, we can optionally modify the boundaries by ±0.2% to create three variations per sub-domain (0.2%, 0.4%, and 0.6%) when the boundaries are adjusted uniformly therefore creating $3 \times 16 = 48$ combinations for a 4×4 checkerboard. Other combinations are possible, such as adjusting boundaries in all individual combinations instead of uniformly, but we leave this for future work.

3.4 Extracting Features from Neighborhoods

In consideration of a 9-region neighborhood as per Figure 1a, we have developed features that capture interesting changes in the material setting ϵ around and within the overlapping region of a pair of sub-domains. Figure 2a provides an example for a modified sub-domain $\widehat{\Omega}_1$ and the northern overlapping region $\widehat{\Omega}_1 \cap \Omega_2$ with Ω_2. The example shows nine rows of unknowns surrounding the boundary $\partial_{north}\Omega_1$ and the modified boundary $\partial_{north}\widehat{\Omega}_1$. In this case the rows are of key interest but the columns are not, as they capture changes in the materials that are perpendicular to the overlapping regions, and extending or shrinking the overlapping region does not affect the measurement.

The features we are interested in come from a single-line or multi-line region immediately above or below the $\partial_{north}\widehat{\Omega}_1$ boundary as shown in Figure 2a. The number of lines in such a region is variable, and for now we simply consider two lines per region. For example, Figure 2a shows two lines for regions 'E' and 'F'. In this respect, we propose two feature sets called Fine and Coarse based on single-line and multi-line features respectively. We also propose a third feature set called Combined for Fine and Coarse together. For all three feature sets, we can capture various maximums, minimums, and differences between epsilon values within the regions of interest. Specifically, we capture (1) the maximum value in a region, (2) the minimum value in a region, (3) the maximum difference between values in a region, (4) the maximum difference between values in a region and the boundary, and (5) the minimum difference between values in a region and the boundary. Figure 2 provides a worked example where Fine is regions A–D (20 features), Coarse is regions E–F (10 features), and Combined


A
B
C
D
E
F
9
8
7
6
5
4
3
2
1
$\partial_{north}\widehat{\Omega}_1$
$\partial_{north}\Omega_1$
$\widehat{\Omega}_1 \cap \Omega_2$
$\widehat{\Omega}_1$
Ω_1
Ω_2

(a)

Region	Max Value in Region	Min Value in Region	Max Diff in Region	Max Diff to Boundary	Min Diff to Boundary
A	10 000	1 000	9 000	9 000	0
B	1 000	100	900	0	0
C	1 000	100	900	900	0
D	10 000	1 000	9 000	9 900	0
E	10 000	100	9 900	9 000	0
F	10 000	100	9 900	9 900	0

(b)

Fig. 2. (a) A fragment of a unit square for two overlapping sub-domains $\widehat{\Omega}_1$ and Ω_2. Notations 'A' to 'F' denote our features as the relationship between the rows referenced with arrows and the $\partial_{north}\widehat{\Omega}_1$ boundary. **(b)** Extracted values assuming the pink, gray, and blue unknowns take $\epsilon = 10\,000$, $\epsilon = 1\,000$, and $\epsilon = 100$ respectively.

is regions A–F (30 features). Then the full feature sets are realized when the eastern, southern, and western boundaries are processed.

3.5 The FPO Evaluation Measure

Developing a measure to evaluate the goodness of a sub-domain specification for any diffusion specification is non-trivial. Key variables such as the number of iterations and the amount of overlap have a complex relationship between one another, so these need to be carefully combined. Our approach should optimize the use of domain decomposition techniques that are designed to speed up simulation for parallel platforms. In a real-world application, a user is interested in optimizing the real-world time that a simulation needs, such as the chosen implementation, the computing network, and the use of cluster computing. Given this variability, we want to concentrate on the theoretical aspects of the algorithm together with a given abstract hardware scenario. Our goal is to minimize the number of floating point operations (FPO).

To justify this choice, first assume that we have a hardware architecture with s computation nodes. To use this architecture in an optimal way, let s also represent the number of sub-domains, n_i be the number of unknowns in a sub-domain, and l be the number of domain decomposition iterations. In a single iteration step, s linear equation systems have to be solved whereas the size of all equation systems is n_i. If one now assumes that a direct solver such as LU decomposition [9] is used, the first domain decomposition iteration of the matrix has the complexity of $O(n^3)$. For all remaining l iterations, one just has to solve one upper right and one lower left matrix of the complexity $O(n^2)$. Hence, FPO is defined as:

$$FPO \approx \sum_{i=1}^{s} \frac{n_i^3}{3} + l \cdot n_i^2 \ .$$

FPO is only meaningful when comparing solutions with the same number of sub-domains on the same hardware architecture.

3.6 Machine Learning Methodology

In order to learn from 9-region neighborhoods, we must simulate a large number of diffusion specifications with multiple sub-domain boundary settings in each neighborhood and derive the corresponding FPO scores. We do not simulate the neighborhoods of the unit square in isolation, because introduced additional artificial boundary conditions will influence the numerical behavior and so perturb the machine learning. Instead, the areas outside the neighborhood of interest are simply considered constant, and we are interested in relative changes to FPO when varying the sub-domain boundaries.

With a database of simulation results, we aim to predict the FPO scores for unseen neighborhoods with regression. Then we adopt the boundary recommendations that minimize FPO for each neighborhood and combine these to create a new solution. Using 527 diffusion specifications each having 48 domain decomposition permutations as per Section 3.3, the steps are as follows:

1. Training. For each diffusion file:
   (a) Perform feature extraction for all 48 permutations of the neighborhoods.
   (b) Compute FPO for all 48 permutations with simulation.
   (c) Record the mapping from the set of input features to FPO.
2. Testing. For each diffusion file:
   (a) Perform feature extraction for all 48 permutations of the neighborhoods.
   (b) Predict FPO for all 48 permutations using a regression model with the data from Step 1c.
   (c) Identify the minimum FPO value for each neighborhood.
3. Evaluation. For each diffusion file:
   (a) Create a new domain specification using the best results from Step 2c and compute FPO with simulation.
   (b) Compare the FPO score from Step 3a with that of the baseline.

The training and testing described above is performed with 10-fold cross-validation. A separate validation set of diffusion specifications was used during the development of our approach to avoid overfitting.

4 Analysis and Results

In this section we determine an expert human baseline, analyze our data, report improvements achieved by our method, and offer a forward plan with ideas to generate further improvements.

4.1 Baseline Overlap Decision

The baseline comparison for our methodology should be a human solution, since we are aiming to improve the FPO estimator from what humans can achieve. One solution is to apply a global overlap to all sub-domains. From our expertise of numerical simulation, the overlapping regions may consume up to around 5% of the available unknowns as a guideline. Guidelines are rarely given in the literature, but the work of Bjorstad and Hvidsten [1] provides one example based on 6%. An analysis of several checkerboard grid sizes and global overlap settings will guide the decision. The required data is given in Table 1 with our highlighted choice in bold. We adopt this choice because: (1) The 4 × 4 checkerboard gives a good mixture of center, boundary, and corner neighborhoods, (2) the 0.4% global overlap avoids extremes, and (3) it is compatible with the literature.

4.2 Data Analysis

We examined the data from all diffusion specifications from our validation and test sets to better understand the effectiveness of our feature sets. Partial results for 1 000 diffusion specifications are given in Table 2. The columns show the number of times the invalid (i.e., outside of unit square), no difference, default setting, and other measurements are observed. As shown, 25% of all values are invalid cases, which accounts for attempted measurements outside the unit square for a 4 × 4 grid — this effect diminishes for larger grids. We could give special consideration to boundary cases, but we leave this for future work for now, as many regression algorithms can handle missing values. We also see large numbers of no difference and default cases, however this redundancy is mitigated by the fact that our method uses vectors of measurements.

The other cases are where we can learn the most from. However, features (2) and (4) indicate that we do not have enough data for our problem setting, so we omit these. Also features (1) and (3) have the same number of measurements, and our analysis showed that these are correlated in almost all cases, so one should be omitted here too due to redundancy. In general, relative values (such as differences) are more interesting than absolute values (such as maximums and minimums), since the absolute values are dependent on the specific problem settings. With all the above considerations in mind, we only consider features (3) and (5), cutting the feature sets to 40% of those proposed in Section 3.4.

Table 1. Baseline choice considering experiment size, global overlap, and the literature

Global	Total overlap for various grid sizes (% of unknowns)							
overlap	1 × 1	2 × 2	3 × 3	**4 × 4**	5 × 5	6 × 6	7 × 7	8 × 8
minimum	0.00	0.40	0.80	1.19	1.59	1.99	2.38	2.77
0.2%	0.00	1.19	2.38	3.56	4.73	5.90	7.06	8.21
0.4%	0.00	1.99	3.95	**5.90**	7.82	9.73	11.62	13.48
0.6%	0.00	2.77	5.51	8.21	10.87	13.48	16.06	18.60
0.8%	0.00	3.56	7.06	10.49	13.85	17.16	20.40	23.57

Table 2. The feature extraction data demonstrates some redundancy in the feature sets. All values shown are for the "north" boundary and the "FINE A" region.

	Feature	Invalid	No Diff	Default	Other	Total
(1)	Max Value in Region	12 000	0	34 171	1 829	48 000
(2)	Min Value in Region	12 000	0	35 990	10	48 000
(3)	Max Diff in Region	12 000	34 171	0	1 829	48 000
(4)	Min Diff to Boundary	12 000	36 000	0	0	48 000
(5)	Max Diff to Boundary	12 000	35 361	0	639	48 000

4.3 Regression Algorithms and Feature Sets

We now compare the learned FPO scores with the baseline. We consider the three feature sets, COMBINED, FINE, and COARSE, and four regression algorithms, namely simple linear regression, nearest neighbor regression, decision tree regression, and support vector machine regression.[1] We found that only the nearest neighbor regression algorithm offered improvement. Since our interesting features are sparse (cf. Table 2), this indicates that we only have so many interesting near neighbors to learn from for each prediction, making nearest neighbor a good choice as the learning algorithm.

The median learned FPO scores for the nearest neighbor regression algorithm expressed as a fraction of the baseline are 0.9778 for COMBINED, 0.9791 for FINE, and 0.9830 for COARSE. This improvement is statistically significant in all cases ($p < 2.2 \times 10^{-16}$) when using a paired Student's t-test and the effect size is large (Cohen's $d = 0.85$ for COMBINED versus baseline, $d = 0.79$ for FINE versus baseline, and $d = 0.62$ for COARSE versus baseline). We also examined the differences between the features sets, but we found these of little interest as the effect sizes are small.

We must point out that our baseline is an expert human baseline, which we consider as the best baseline. In contrast, the novice baseline setting can be considered the minimum overlap comprising one line of unknowns. This baseline can lead to extreme behavior in many cases and simulation that does not converge in a reasonable amount of time. As a result we cannot compute this baseline, but we note that our approach does not exhibit the behavior of the novice baseline.

So far FPO is reduced to around 0.98 of the baseline and we have statistically significant improvements with large effect sizes. An explanation for this result is that our method provides a very consistent improvement for our test instances. For instance, Figures 3a and 3b show a shift in the score distributions leaving little overlap. Specifically, our method improves the baseline score for all instances except for 33 of 527 as shown in Figure 3c. We would like to further improve the cost saving to a 0.95 or 0.90 fraction to be completely satisfied, which we aim to achieve in future work with the following forward plan.

[1] Weka 3.7.7 [7]: weka.classifiers.functions.LinearRegression, weka.classifiers.lazy.IBk, weka.classifiers.trees.M5P, and weka.classifiers.functions.SMOreg respectively.

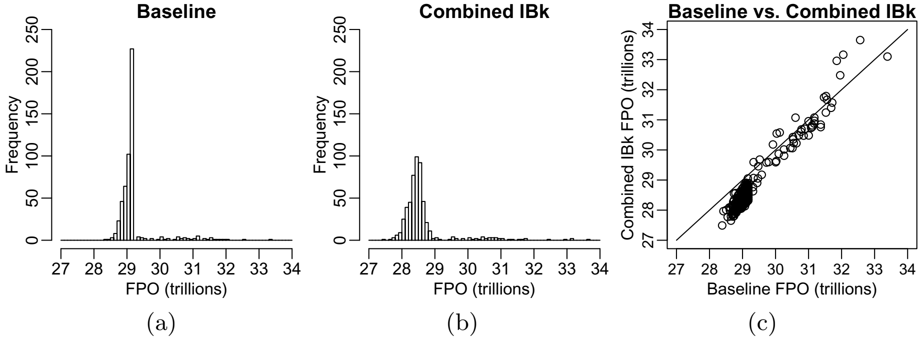


Fig. 3. The histogram shift and the scatterplot demonstrates consistent improvement

4.4 Forward Plan

The results presented above are our first for overlap optimization. An end goal is to consider three-dimensional problems later. First we wish to improve our approach for two-dimensional problems by implementing and experimenting with several extensions. For the first extension we wish to increase the checkerboard size for more fine-grained and precise learning. Second, we want to increase the training set size with additional diffusion specifications. Third, we wish to apply non-uniform boundary adjustments with sub-domains. Finally, we would like to drop the checkerboard constraint in favor of polygonal boundaries. We anticipate that these items each offer incremental improvements, and the final sum will be of most interest.

5 Conclusions

In this paper, we proposed a machine learning method for optimizing overlaps of domain decomposition problems. The key idea proposed was to learn properties of sub-domain neighborhoods, so that a complete solution can be assembled automatically and solved more efficiently with domain decomposition. To achieve this, our method introduced a novel feature set with the purpose of capturing interesting properties of sub-domain boundaries. When compared with an expert human baseline, our method offered a consistent and statistically significant improvement for the Poisson's equation. In addition, several avenues of future work have been identified that we expect will offer further improvements.

Finally, we emphasize that this work represents one part of a many-fold application of machine learning in a practical numerical simulation setting. Our earlier work in this field has demonstrated the behavioral learnability of bridge models [4] in an integrative structural design setting [6] for identifying robust solutions in civil engineering. Conversely, domain decomposition concerns parallelization for efficiency, so the bringing together of both dimensions provides potency for machine learning to have a high impact in numerical simulation.

References

1. Bjorstad, P., Hvidsten, A.: Iterative Methods for Substructured Elasticity Problems in Structural Analysis. In: Glowinski, R. (ed.) Proceedings of the First International Symposium on Domain Decomposition Methods for Partial Differential Equations, pp. 301–312. SIAM, Paris (1987)
2. Brady, T.F., Yellig, E.: Simulation Data Mining: A New Form of Computer Simulation Output. In: Kuhl, M.E., Steiger, N.M., Armstrong, F.B., Joines, J.A. (eds.) Proceedings of the Thirty-Seventh Winter Simulation Conference, pp. 285–289. ACM, Orlando (2005)
3. Brenner, S.C., Scott, L.R.: The Mathematical Theory of Finite Element Methods, 2nd edn. Springer, Berlin (2002)
4. Burrows, S., Stein, B., Frochte, J., Wiesner, D., Müller, K.: Simulation Data Mining for Supporting Bridge Design. In: Christen, P., Li, J., Ong, K.L., Stranieri, A., Vamplew, P. (eds.) Proceedings of the Ninth Australasian Data Mining Conference, pp. 163–170. ACM, Ballarat (2011)
5. Canuto, C.G., Hussaini, M.Y., Quarteroni, A., Zang, T.A.: Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics, 1st edn. Scientific computation. Springer, Heidelberg (2007)
6. Gerold, F., Beucke, K., Seible, F.: Integrative Structural Design. Journal of Computing in Civil Engineering 26(6), 720–726 (2012)
7. 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)
8. Mei, L., Thole, C.A.: Data Analysis for Parallel Car-Crash Simulation Results and Model Optimization. Simulation Modelling Practice and Theory 16(3), 329–337 (2008)
9. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: LU Decomposition and its Applications. In: Numerical Recipes in Fortran: The Art of Scientific Computing, 2nd edn., pp. 34–42. Cambridge University Press, Cambridge (1992)
10. Quarteroni, A., Valli, A.: Domain Decomposition Methods for Partial Differential Equations, 1st edn. Numerical Mathematics and Scientific Computation. Oxford Science Publications, New York City (1999)
11. Stein, B., Curatolo, D.: Selection of Numerical Methods in Specific Simulation Applications. In: del Pobil, A.P., Mira, J., Ali, M. (eds.) IEA/AIE 1998. LNCS, vol. 1416, pp. 918–927. Springer, Heidelberg (1998)
12. Toselli, A., Widlund, O.: Domain Decomposition Methods – Algorithms and Theory., 1st edn. Springer Series in Computational Mathematics, vol. 34. Springer, Heidelberg (2004)
13. Versteeg, H.K., Malalasekera, W.: An Introduction to Computational Fluid Dynamics: The Finite Volume Method, 2nd edn. Pearson Education, Essex (2007)
14. Yang, S.-H., Hu, B.-G.: Reformulated Parametric Learning based on Ordinary Differential Equations. In: Huang, D.-S., Li, K., Irwin, G.W. (eds.) ICIC 2006. LNCS (LNAI), vol. 4114, pp. 256–267. Springer, Heidelberg (2006)

CLUEKR : CLUstering Based Efficient kNN Regression

Harshit Dubey[1] and Vikram Pudi[2]

[1] International Institute of Information Technology, Hyderabad,
Andhra Pradesh, India 500032
harshit.dubeyug08@students.iiit.ac.in

[2] International Institute of Information Technology, Hyderabad,
Andhra Pradesh, India 500032
vikram@iiit.ac.in

Abstract. K-Nearest Neighbor based regression algorithm assigns a value to the query instance based on the values of its neighborhood instances. Although kNN has proved to be a ubiquitous classification/regression tool with good scalability but it suffers from some drawbacks. One of its biggest drawback is that, it is a lazy learner i.e. it uses all the training data at runtime. In this paper, we propose a novel, efficient and accurate, clustering based kNN regression algorithm CLUEKR having the advantage of low computational complexity. Instead of searching for nearest neighbors directly in the entire dataset, we first hierarchically cluster the data and then find the cluster in which the query point should lie. Our empirical experiments with several real world datasets show that our algorithm reduces the search space for kNN significantly and is yet accurate.

Keywords: Regression, Efficient, Accurate, K-Nearest Neighbor, Clusters, Hierarchy, Likelihood.

1 Introduction

The problem of regression is to estimate the value of a dependent variable based on the values of one or more independent variables, e.g., predicting price increase based on demand or money supply based on inflation rate etc. Regression analysis helps to understand how the typical value of the dependent variable changes when any one of the independent variables is varied, while the other independent variables are held fixed. Regression algorithms can be used for prediction (including forecasting of time-series data), inference, hypothesis-testing and modeling of causal relationships.

In statistics, Regression is considered as collection of statistical function-fitting techniques. These techniques are classified according to the form of the function being fit to the data. Regression analysis has been studied extensively in statistics, but there have been only a few studies from the data mining perspective. Majority of this study resulted into algorithms that fall under the following

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 450–458, 2013.

broad categories - Linear Regression [12], Nearest Neighbor Algorithms [5], Decision Trees [3], Support Vector Machines [4], Neural Networks [7] and Logistic Regression [8]. However most of these algorithms were originally developed for classification purpose, but have been later modified for regression.

In the recent past, a lot of research centered at nearest neighbor methodology has been performed. Instead of being computationally expensive kNN algorithm is very simple to understand, accurate, requires only a few parameters to be tuned and is robust with regard to the search space. Also kNN classifier can be updated at a very little cost as new training instances with known classes are presented. A strong point of kNN is that, for all data distributions, its probability of error is bounded above by twice the Bayes probability of error[10]. However one of the major drawbacks of kNN is that, it is a lazy learner i.e. it uses all the training data at runtime. However for majority of the datasets, we perform as accurate as kNN. In this paper, we propose a novel, efficient and accurate, clustering based kNN regression algorithm CLUEKR, which instead of searching for nearest neighbors directly in the entire dataset, first find the cluster in which the query point has maximum likelihood of occurrence. We first hierarchically cluster the data in the pre-processing step, then a recursive search starting from root node of the hierarchy is performed. For current search node in the hierarchy, we select a cluster among its child, in which the query point has maximum likelihood of occurrence and then a recursive search is applied to it. Finally we find the k nearest neighbors of query points in the obtained cluster and return the weighted mean of their response variable as result.

The organization of rest of the paper is as follows. In section 2, we throw light on related, and recent, work in the literature. Section 3 deals with problem formulation. We explain the modified algorithm in Section 4. In Section 5, experimental results are presented together with a thorough comparison with the state-of-the-art algorithms. Finally, in Section 6, conclusions are drawn.

2 Related Work

Traditional Statistical Approaches follow a methodology that requires the form of the curve to be specified in advance. This requires regression problems in each special application domain be studied and solved optimally for that domain. Another problem with these approaches is outlier (extreme cases) sensitivity. The most common statistical regression approach is linear regression which assumes the entire data to follow a linear relationship between the response and feature variables. But this assumption is unlikely to hold on variety of application.

Segmented or piecewise regression [11] is a method in regression analysis in which the independent variable is partitioned into intervals and a separate line segment is fit to each interval. It is is essentially a wedding of hierarchical clustering and standard regression theory. It can also be performed on multivariate data by partitioning the various independent variables. Segmented regression is useful when the independent variables, clustered into different groups, exhibit different relationships between the variables in these regions. The boundaries between the segments are breakpoints.

Most of the Regression approaches in data mining falls under these categories - Nearest Neighbor, Regression trees, Neural Networks, and Support Vector Machines. Regression trees are a variation of decision trees in which the predicted outcome is a real number. Neural Networks and SVM techniques are quite complex and an in-depth analysis of results obtained is not possible.

One of the oldest, accurate and simplest method for pattern classification and regression is K-Nearest-Neighbor (kNN) [5]. kNN algorithms have been identified as one of the top ten most influential data mining algorithms [14] for their ability of producing simple but powerful classifiers. It has been studied at length over the past few decades and is widely applied in many fields. Despite its simplicity, the kNN rule often yields competitive results. However one of the major drawbacks of kNN is that, it is a lazy learner i.e. it uses all the training data at the runtime.

A recent work on prototype reduction, called Weighted Distance Nearest Neighbor (WDNN) [9] is based on retaining the informative instances and learning their weights for classification. The algorithm assigns a non negative weight to each training instance tuple at the training phase and only the training instances with positive weight are retained (as the prototypes) in the test phase. However the algorithm is specifically designed for classification purpose and cannot be used for regression.

In another recent work [13], a Parameterless, Accurate, Generic, Efficient NN-Based Regression algorithm PAGER is proposed, which is based on the assumption that value of the dependent variable varies smoothly with the variation in values of independent variable. For each dimension in the search space, the authors construct a 1-dimensional predictor as a line passing through two closest neighbor of the query point. For each of the predictor obtained, they determine the mean error occurred if the predictor was used in prediction of the k-nearest neighbors. A weight inversely proportional to the mean error is assigned to each predictor. Finally a weighted sum of the value output by individual predictors for the query instance is assigned to it.

Saket and others propose a kNN based regression algorithm BINER : BINary search based Efficient Regression [2], which instead of directly predicting the value of response variable recursively narrows down the range in which the response variable lies. In the pre-processing step training data is sorted based on the value of response variable and is hierarchically structured. At each level in the hierarchy, data is divided into three parts, one containing elements from first to middle, other contains elements from middle to last and the third contains elements from middle of first portion to middle of second portion. The algorithm then finds the portion in which the query point has maximum likelihood to lie and finally kNN algorithm is applied to that portion.

Qi Yu and others in one of their recent work [15] proposed a methodology named Optimally Pruned K-Nearest Neighbors (OP-kNNs) which builds a one hidden-layer feedforward neural network using K-Nearest Neighbors as kernels to perform regression. The approach performed better compared to state-of-the-art methods while remaining fast.

3 Problem Formulation

In this section, we present the problem of regression and notation used to model the dataset.

The problem of regression is to estimate the value of a dependent variable (known as response variable) based on the values of one or more independent variables (known as feature variables). We model the tuple as $\{X, y\}$ where X is an ordered set of attribute values like $\{x_1, x_2, \ldots, x_d\}$ and y is the numeric variable to be predicted. Here x_i is the value of the i^{th} attribute and there are d attributes overall corresponding to a d-dimensional space.

Formally, the problem has the following inputs:

- An ordered set of feature variables Q i.e. $\{q_1, q_2, \ldots, q_d\}$
- A set of n tuples called the training dataset, D, = $\{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\}$.

The output is an estimated value of y for the given query Q. Mathematically, it can be represented as

$$y = f(X, D, parameters), \tag{1}$$

where *parameters* are the arguments which the function $f()$ takes. These are generally set by user and are learned by trial and error method.

4 The CLUEKR Algorithm

We describe our proposed algorithm in this section. Our algorithm proceeds in two steps :

- It first find the cluster in the hierarchy, in which the query point has maximum likelihood of occurrence.
- kNN is applied to points present in the cluster, and weighted mean of the k nearest neighbors of query point in the cluster is quoted as output.

Size of the obtained cluster is less compared to the size of entire dataset, in this way our algorithm reduce the search space for K-Nearest Neighbor algorithm. Now we explain in detail about the clustering phase (pre-processing step) first and later throw light on the actual algorithm.

4.1 Pre-processing

In the pre-processing step, data is hierarchically clustered and mean value for each of the cluster is calculated and stored. Each node in this hierarchy consist of clusters containing data point, which are recursively divided into three child clusters as we move down the hierarchy. We make use of the fact that similar instances have similar value of response variable (basic analogy on which kNN works), while selecting the cluster center. Each cluster node is sorted based on the value of response variable, then $n/4$ and $3n/4$ ranked instance are selected as the center for two of the child clusters. For the third cluster, mean of the other two cluster's center is taken as center. All the points present in the cluster node are divided into child cluster, based on to which child cluster center the point is closest.

We aim to divide each cluster nodes is such a fashion, that each child node contains half the data present in the parent node. However at each level we have added an extra child cluster to make the division of points smooth, this cluster also contain points belonging to other cluster that lies at its boundary with other cluster. This will help to properly classify the query instance that lie at the boundary of clusters. Boundary points to third cluster are defined as, points whose distance ratio from other cluster center to third cluster center is between 0.9 to 1.0.

We recursively divide the cluster nodes, till we get a cluster node which contain less that $2 * k$ points. The limiting size of $2 * k$ was chosen in order to keep a margin for selection of k nearest neighbors.

4.2 Actual Algorithm

Pseudo code for the actual algorithm is provided in Algo. 1. The recursive search starts from the root node and goes down the hierarchy. In order to find the cluster among the child nodes (for current node in the search) in which the query point has maximum likelihood of occurrence, distance of query point from mean value of all the child cluster of current node is calculated (lines 1-3). If the two closest distances comes out to be similar ($Confidance$ returns $false$) then we can't say with confidence, that which child cluster to pick and hence kNN algorithm is applied to the current cluster (line 5), else recursive search continues on the child cluster which is closest to the Query point (lines 7-8) i.e distance from mean of that cluster is least.

We say that two distances, d_i and d_j are similar if $min(d_i/d_j, d_j/d_i)$ is greater than 0.90. The value of 0.90 was selected by experimentations and it works well on most of the datasets as shown in the experimental section.

Algorithm 1. CLUEKR : Pseudo code

```
Input: $Query\ instance\ Q, Pointer\ to\ Root\ node\ in\ hierarchy\ R, Parameter\ k$
Output: $Value\ of\ Response\ Variable\ for\ Query\ instance\ Q$
1: for $each\ node\ j\ in\ childs\ of\ R$ do
2:     $dist[j] = distance(Q, mean(j))$
3: end for
4: if $Confidance(dist)$ then
5:     $ChildPtr = argmin(dist)$
6:     return $CLUEKR(Q, ChildPtr,\ k)$
7: else
8:     return k$NN(Q,\ Data\ of\ R,\ k)$
9: end if
```

4.3 Complexity Analysis

We perform complexity analysis for both pre-processing step and actual algorithm in this section.

- In the pre-processing step, hierarchy consisting of cluster node is constructed, for which each cluster node is divided into child clusters. Assuming that we have data points sorted based on the value of response variable in the cluster node to be split, we can select the centers for child cluster directly and then data can be redistributed among the clusters in $O(n_s)$, where n_s is the size of cluster node to be spitted. If the cluster node is transversed in sorted order of response variable to redistribute data in child cluster, then child cluster will also contain data, sorted based on response variable. So if the root node has sorted data based on response variable, all other nodes in the hierarchy will also follow the same ordering of data. As data is distributed from parent to child cluster, total amount of data present in child cluster is equal to data present in the parent cluster (ignoring the extra boundary points present in third cluster). Total time complexity of pre-processing step comes out to $O(n * log(n)) + O(h * n)$, where $n * log(n)$ is the cost involved in sorting root node and $h * n$ is the cost of splitting cluster nodes to construct a hierarchy of height h. This also involve the cost of calculating mean of each cluster node, as it can be calculated during the transversal of cluster node for distribution of data.
- At runtime our algorithm first performs a search for the cluster in the hierarchy,in which the query point has maximum likelihood of occurrence and then kNN is applied to the obtained cluster. Runtime complexity of our algorithm is $O(h + n_c)$, where h is the height of the hierarchy and n_c is the size of the cluster obtained. As n_c is supposed to be less compared to n, our algorithm is faster at runtime compared to kNN.

5 Experimental Study

5.1 Performance Model

In this section, we demonstrate our experimental settings. The experiments were obtained on a wide variety of real life datasets obtained from UCI data repository [1] and Weka Datasets [6].A short description of all the datasets used is provided in Table 1. We have compared our performance against the following approaches: K Nearest Neighbor, Isotonic , Linear Regression(Linear Reg.), Least Mean Square (LMS) algorithm, Radial Basis Function Network (RBF Network), Regression Tree (RepTree) and Decision Stump(Dec Stump). Most of the algorithms are available as part of the Weka toolkit. All the results have been obtained using 10-fold cross validation technique.

We have used two metrics for quantifying our results, namely, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). MAE is mean of the absolute errors (actual output - predicted output). RMSE is square root of mean of squared errors. We have used euclidean distance matrix to calculate the distances in our algorithm.

Table 1. Dataset Description

Dataset	#Instances	#Attributes	Dataset	#Instances	#Attributes
Autompg	392	8	Bank	8192	9
Bodyfat	252	15	Concrete	1030	9
Cpu	209	7	Forestfire	517	11
Flow	103	8	Housing	507	14
Space	3107	7	Slump	103	8
Synfriedman	500	6	Synfriedman1	500	6

Table 2. Comparison of results of CLUEKR with other standard approaches

Dataset	CLUEKR		*k*NN		Isotonic		Linear Reg.		LMS		RBF		Rep Tree		Dec Stump	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Autompg	**2.36**	**3.24**	**2.40**	**3.59**	3.16	4.3	2.56	3.4	**2.50**	**2.59**	3.90	5.07	**2.30**	**3.31**	4.2	5.18
Bank	**0.02**	**0.03**	**0.02**	**0.03**	0.03	0.46	**0.02**	**0.03**	0.03	0.05	0.04	0.06	**0.02**	**0.03**	0.04	0.05
Bodyfat	0.51	0.65	0.49	0.62	0.52	0.67	**0.43**	**0.56**	**0.45**	**0.58**	0.61	0.77	0.52	0.67	0.63	0.8
Concrete	7.20	9.92	**5.71**	**8.14**	10.81	13.45	8.30	10.45	9.52	17.53	13.38	16.56	**5.43**	**7.38**	11.54	14.46
Cpu	**21.45**	**70.76**	**18.79**	**74.37**	**23.12**	**50.95**	36.05	66.24	33.97	108.32	51.87	126.35	32.34	93.21	71.56	126.40
Flow	**11.27**	**14.78**	11.89	16.42	11.55	14.18	**10.99**	**13.26**	13.28	18.56	14.76	17.42	12.11	15.57	12.48	15.41
Forestfire	21.83	69.04	21.75	68.13	19.18	63.5	19.92	64.28	**12.88**	**64.91**	**18.86**	**63.86**	19.24	64.56	18.93	64.68
Housing	**2.96**	**4.63**	**2.97**	**4.63**	3.80	5.32	3.39	4.91	3.42	5.55	6.13	8.42	3.18	4.84	5.61	7.5
Slump	**5.56**	**7.75**	5.89	9.83	**5.86**	**7.52**	6.67	7.82	6.68	10.56	6.98	8.73	6.14	8.19	7.05	8.86
Space	**0.10**	**0.17**	**0.10**	**0.17**	0.12	0.16	0.11	0.16	0.11	0.15	0.14	0.19	**0.10**	**0.14**	0.13	0.18
Synf.	**2.10**	**2.64**	**1.96**	**2.45**	3.69	4.44	2.25	2.83	2.24	2.82	3.86	4.81	2.69	3.45	3.73	4.63
Synf. 1	**1.92**	**2.48**	**1.75**	**2.16**	3.59	4.32	2.76	4.65	2.45	4.71	3.92	4.91	2.57	3.24	3.69	4.4

Table 3. Comparison of dataset size with size of the cluster obtained

Dataset	Dataset Size	Cluster Size	%Ratio	Dataset	Dataset Size	Cluster Size	%Ratio
Autompg	392	76	20	Bank	8192	2975	36
Bodyfat	252	50	20	Concrete	1030	191	18
Cpu	209	53	25	Forestfire	517	156	30
Flow	103	35	33	Housing	507	105	20
Space	3107	262	9	Slump	103	34	33
Synf.	500	143	28	Synf. 1	500	136	27

5.2 Results and Discussion

Table 2 compares the results obtained for our algorithm with other existing state of art approaches, top two results are highlighted in bold. As can be seen from the results, for majority of the datasets, we perform as accurate as *k*NN, however for some of the datasets, our algorithm also outperform *k*NN. Also our algorithm more than often outperforms other existing state-of-the art algorithms. However for concrete dataset in which the response variable is highly non-linear function of its attributes, we do not perform as good as for other datasets. Table 3 compares average size of cluster obtained by our algorithm with original dataset size.Ratio column in the table shows the percentage ratio of the cluster size obtained by our algorithm to the original dataset size. It is clear from the data, that our algorithm reduces the search space for *k*NN significantly, however the degree of reduction varies depending on the type of dataset.

6 Conclusion

In this paper, we have proposed a novel clustering based k nearest neighbor regression algorithm which is efficient and accurate. Our work is based on reducing the search space for nearest neighbors for any given point. We hierarchically cluster the data in pre-processing step and then search is performed to find the cluster in the hierarchy, in which the query point has the maximum likelihood of occurrence. We have also evaluated our approach against the existing state-of-the-art regression algorithms. As shown in the experimental section, our approaches reduces the search space for *k*NN significantly and is yet accurate.

References

1. Newman, D., Asuncion, A.: UCI machine learning repository (2007)
2. Bharambe, S., Dubey, H., Pudi, V.: BINER: BINary Search Based Efficient Regression. In: Perner, P. (ed.) MLDM 2012. LNCS, vol. 7376, pp. 76–85. Springer, Heidelberg (2012)
3. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth and Brooks, Monterey (1984)
4. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20, 273–297 (1995), doi:10.1007/BF00994018
5. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley, New York (2001)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11, 10–18 (2009)
7. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice-Hall (1999)
8. Hosmer, D., Lemeshow, S.: Applied Logistic Regression. Wiley Series in Probability and Statistics: Texts and References Section. John Wiley & Sons (2000)
9. Jahromi, M.Z., Parvinnia, E., John, R.: A method of learning weighted similarity function to improve the performance of nearest neighbor. Inf. Sci. 179, 2964–2973 (2009)

10. Loizou, G., Maybank, S.J.: The nearest neighbor and the bayes error rates. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9(2), 254–262 (1987)
11. McZgee, V.E., Carleton, W.T.: Piecewise regression. Journal of the American Statistical Association 65(331), 1109–1124 (1970)
12. Montgomery, D., Peck, E., Vining, G.: Introduction to linear regression analysis. Wiley series in probability and statistics: Texts, references, and pocketbooks section. Wiley (2001)
13. Singh, H., Desai, A., Pudi, V.: PAGER: Parameterless, accurate, generic, efficient *k*NN-based regression. In: Bringas, P.G., Hameurlain, A., Quirchmayr, G. (eds.) DEXA 2010, Part II. LNCS, vol. 6262, pp. 168–176. Springer, Heidelberg (2010)
14. Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.-H., Steinbach, M., Hand, D.J., Steinberg, D.: Top 10 algorithms in data mining. Knowl. Inf. Syst. 14, 1–37 (2007)
15. Yu, Q., Miche, Y., Sorjamaa, A., Guillen, A., Lendasse, A., Séverin, E.: Op-knn: method and applications. Adv. Artif. Neu. Sys. 1, 1:1–1:6 (2010)

AREM: A Novel Associative Regression Model Based on EM Algorithm

Zhonghua Jiang and George Karypis

Department of Computer Science & Engineering,
Digital Technology Center, University of Minnesota,
Minneapolis, MN, 55455, U.S.A.
{zjiang,karypis}@cs.umn.edu

Abstract. In recent years, there have been increasing efforts in applying association rule mining to build Associative Classification (AC) models. However, the similar area that applies association rule mining to build Associative Regression (AR) models has not been well explored. In this work, we fill this gap by presenting a novel regression model based on association rules called AREM. AREM starts with finding a set of regression rules by applying the instance based pruning strategy, in which the best rules for each instance are discovered and combined. Then a probabilistic model is trained by applying the EM algorithm, in which the right hand side of the rules and their importance weights are updated. The extensive experimental evaluation shows that our model can perform better than both the previously proposed AR model and some of the state of the art regression models, including Boosted Regression Trees, SVR, CART and Cubist, with the Mean Squared Error (MSE) being used as the performance metric.

Keywords: association rule, regression rule, associative regression, probabilistic model, EM algorithm, instance based pruning.

1 Introduction

In recent years, there have been increasing efforts in applying association rule mining to build classification models [1] [2] [3] [4] [5], which have resulted in the area of Associative Classification (AC) modeling. Several studies [1] [2] [3] have provided empirical evidence that AC classifiers can outperform tree-based [6] and rule-induction based models [7] [8]. The good performance of the AC models can be attributed to the fact that by using a bottom-up approach to rule discovery (either via frequent itemset mining or instance-based rule mining) they can discover better rules than the traditional heuristic-driven top-down approaches.

Regression is a data mining task that is applicable to a wide-range of application domains. However, despite the success of association rule mining for classification, it has not been extensively applied to develop models for regression. We are only aware of the Regression Based on Association (RBA) method

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 459–470, 2013.

developed by Ozgur *et al.* [9] that uses association rule mining to derive a set of regression rules. Since regression models need to predict a continuous value, whereas the classification models need to predict a categorical value, the methods developed for AC modeling are in general not applicable for solving regression problems.

Motivated by the success of AC modeling, we study the problem of applying the association rule mining to build an Associative Regression (AR) model. We believe this is an important problem for the following two reasons: First, an AR model is built upon a set of regression rules, which in many cases, can be easily interpreted by domain experts and thus provide valuable insights. Second, the good performance of the well studied AC classifiers leads us to believe that the AR model may potentially perform better than the tree-based [10] [11] and rule-induction based [12] regression models.

We present an associative regression model utilizing expectation maximization [13], called AREM. An AR model consists of three major components: (i) the method used to identify the sets of itemsets that form the left hand sides of the rules, (ii) the method used to estimate the right hand sides of the rules, and (iii) the method used to compute a prediction. Drawing upon approaches used for developing AC models, AREM uses an instance-based approach to select a subset of frequent itemsets that are used to form the left hand side of the rules. However, unlike existing AC and AR models, it develops and utilizes a probabilistic model coupled with an EM-based optimization approach to determine the right hand side of the rules and also assign a weight to each rule that is used during prediction. The advantage of this probabilistic model is that it allows AREM to capture the interactions of the various rules and to learn the parameters that lead to more accurate predictions. Our experimental evaluation shows that AREM outperforms several state of the art regression models including RBA [9], Boosted Regression Trees [10], SVR [14], CART [11] and Cubist [12] on many data sets, with the Mean Square Error (MSE) being used as the performance metric.

The remainder of this paper is organized as follows. Section 2 introduces some notations and definitions. Section 3 presents the related work in this area. AREM is formally presented in Section 4. In Section 5, we explain the experimental design and results for model evaluation. And finally Section 6 concludes.

2 Notations and Definitions

The methods developed in this work apply to datasets whose instances are described by a set of features that are present. Such datasets occur naturally in market basket transactions (features represent the set of products purchased) or bag-of-word modeling of documents (features correspond to the set of words in the document). We will refer to these features as items. Note that other types of datasets can be converted to the above format via discretization techniques [15].

Let the data set $\mathcal{D} = \{(\tau_i, y_i) | i = 1, 2, ..., N\}$ be a set of N instances. The instance (with index) i is a tuple (τ_i, y_i), where τ_i is a set of items (or, an itemset), and y_i is the real-valued target variable. Given an itemset x, and an instance (τ_i, y_i), we say, x is contained in (τ_i, y_i), or, (τ_i, y_i) contains x, if $x \subseteq \tau_i$. The support of itemset x, is defined as the number of instances in $\mathcal{D}$ that contain x. The itemset x is frequent if its support is not less than s_0, where s_0 is the user specified parameter. For itemset x, we define its mean (μ_x) and standard deviation (σ_x) as computed from the set of target variables from instances in $\mathcal{D}$ that contain x.

A regression rule is of the form $r_x : x \rightarrow \alpha_x$. The rule's left hand side (LHS) x is an itemset.The rule's right hand side (RHS) α_x is the target value predicted by this rule. Each rule is also associated with a positive value w_x which is used as the weight when combining multiple rules together for making predictions. The rule r_x is frequent if its itemset x is frequent.

3 Related Work

To our best knowledge, the RBA [9] model is the only previous work on associative regression. It starts with mining the set of frequent itemsets which form the set of rules' LHS. For each frequent itemset x, RBA computes the rule's RHS as the mean of x. It also computes the standard deviation σ_x of x. These rules are then ranked by variance (i.e., σ_x^2) from small to large. The database sequential coverage is applied to prune rules which are ranked low. For making predictions, three weighting schemes for w_x are developed: (1) *equal*, where rules are equally weighted, (2) *supp*, where the rule r_x is weighted by the support of x, and (3) *inv-var*, where the rule's weight is inverse proportional to the variance σ_x^2.

Associative Classification (AC) [16] is an area that applies similar techniques, but the focus is on the Classification task. Among the many methods developed for AC modeling [1] [2] [3] [5], Harmony [4] is the model that employs a similar rule pruning strategy to AREM: it mines the highest confidence rules for each instance and combines them to the final rule set.

AR and AC models are descriptive in that they can be easily interpreted by end users. Tree based and rule induction based models are another two groups of descriptive models. The classification and regression tree (CART) [11] partitions the input space into smaller, rectangular regions, and assigns the average of the target variables as the predicted value to each region. Cubist [12] is a rule based algorithm and fits a linear regression model to each of the regions. Boosting [10] is a technique to build ensemble models by training each new model to emphasize the training instances that previous models misclassified. Boosted regression trees have shown to be arguably the best algorithms for web-ranking [17].

4 The AREM Model

The AREM model training consists of two major components. First, it discovers a set of frequent regression rules $r_x : x \rightarrow \mu_x$, where μ_x is the mean value of x in $\mathcal{D}$. We denote this set of rules by $\mathcal{R}$. Second, for each $r_x \in \mathcal{R}$, AREM updates its RHS to a new value α_x by learning a probabilistic model. The EM algorithm is applied for model learning where α_x is learned together with the rule's weight w_x.

For the rule discovery component (i.e., the first component above), AREM follows a two-step approach to find the rule set $\mathcal{R}$. First, it uses the FP Growth algorithm [18] to find all frequent itemsets x in $\mathcal{D}$. For each frequent itemset x, AREM generates the rule $r_x : x \rightarrow \mu_x$, where μ_x is the mean value of x in $\mathcal{D}$. Let $\mathcal{F}$ be this set of frequent rules. Second, for each training instance i, let $\mathcal{F}_i$ be the set of rules r_x from $\mathcal{F}$ such that $x \subseteq \tau_i$. AREM selects K rules from $\mathcal{F}_i$ to form the set $\mathcal{R}_i$. Finally, $\mathcal{R}$ is the union of these rules $\mathcal{R}_i$ over all training instances i in $\mathcal{D}$. Since $\mathcal{R}$ will in general contain fewer rules than $\mathcal{F}$, this step applies instance based approach to prune the initial set of frequent rules.

Using the set of updated rules $\mathcal{R}$ with the associated weights, AREM predicts the target variable of a test itemset τ as follows. First, it identifies the set of rules $\mathcal{R}_\tau = \{r_{x_1}, \ldots, r_{x_m}\} \subseteq \mathcal{R}$ whose LHS are subsets of τ (i.e., $(x_i \rightarrow \alpha_{x_i}) \in \mathcal{R}_\tau$ if $x_i \subseteq \tau$), then it eliminates from $\mathcal{R}_\tau$ all but the k rules that have the highest w_{x_i} values among them. This set of rules, denoted by $\mathcal{R}_\tau^k$, is then used to predict the target variable using the formula

$$\hat{y} = \frac{\sum_{r_{x_i} \in \mathcal{R}_\tau^k} w_{x_i} \alpha_{x_i}}{\sum_{r_{x_i} \in \mathcal{R}_\tau^k} w_{x_i}}, \tag{1}$$

which is nothing more than the average of the RHS of the k rules weighted by their corresponding w_{x_i} values. In the case when the test itemset τ is not covered by rules in $\mathcal{R}$, i.e., $|\mathcal{R}_\tau| = 0$, we simply predict $\hat{y}$ as the global mean of target variables in database $\mathcal{D}$.

AREM model requires the specification of four parameters: (i) the minimum support s_0, (ii) the number of rules K that are selected for each training instance, (iii) the number of EM steps M for rule parameter learning, and (iv) the number of rules k from $\mathcal{R}$ that are used for predicting the target variable. Even though the optimal values of these parameters need to be determined using a cross-validation framework, our experience has been that the performance of AREM remains consistently good for a wide range of these values.

In the rest of this section we describe the probabilistic model that we developed for estimating from $\mathcal{D}$ the α_x and w_x parameters of the rules in $\mathcal{R}$ and the method used to select for each training instance i the K rules from $\mathcal{F}_i$.

4.1 The Probabilistic Model

Let $\mathcal{X}$ be the set of itemsets of rules in $\mathcal{R}$ (i.e., $\mathcal{X} = \{x | r_x \in \mathcal{R}\}$). Consider an arbitrary training instance (τ, y). The goal of the probabilistic model is to specify the probability of target variable y given τ, i.e., $P[y|\tau]$. We want to relate this quantity to the set of itemsets in $\mathcal{X}$. To this end, we treat itemset x as a random variable that takes values in $\mathcal{X}$ and write $P[y|\tau]$ as

$$P[y|\tau] = \sum_x P[y, x|\tau] = \sum_x P[y|\tau, x] P[x|\tau],$$

where $P[y|\tau, x]$ is the probability of generating the target variable y given τ and x, which is generated from τ with probability $P[x|\tau]$. Our goal then becomes to specify $P[y|\tau, x]$ and $P[x|\tau]$ and relate them to α_x and w_x.

In order to specify $P[y|\tau, x]$, we first assume the conditional independence $P[y|\tau, x] = P[y|x]$. That is, we assume that once the itemset x is known, the probability of y is not dependent on τ, which simplifies our model so that the dependency of τ is fully captured in $P[x|\tau]$. Given that, we then model $P[y|x]$ as a Normal distribution whose mean is the RHS of the rule $x \rightarrow \alpha_x$ and standard deviation β_x. That is,

$$P[y|x] = \mathcal{N}(y|\alpha_x, \beta_x^2). \tag{2}$$

Next, we specify $P[x|\tau]$ by considering how AREM makes predictions. In order to simplify this discussion we ignore the fact that AREM picks the top k rules (i.e., it uses the set of rules in $\mathcal{R}_\tau^k$) and assume that it predicts the target value by using all the rules in $\mathcal{R}_\tau$. Specifically, Equation 1 now becomes

$$\hat{y} = \frac{\sum_{r_{x_i} \in \mathcal{R}_\tau} w_{x_i} \alpha_{x_i}}{\sum_{r_{x_i} \in \mathcal{R}_\tau} w_{x_i}} = \sum_x \alpha_x \frac{I_{x \subseteq \tau} w_x}{\sum_{x' \subseteq \tau} w_{x'}}, \tag{3}$$

where $I_{x \subseteq \tau}$ is the indicator function which takes value 1 (0) when $x \subseteq \tau$ is true (false).

From the probabilistic modeling point of view, we predict the target variable as the expected value of y given τ, that is,

$$\hat{y} = E[y|\tau] = \sum_x E[y|\tau, x] P[x|\tau]. \tag{4}$$

From Equation 2, we get $E[y|\tau, x] = \alpha_x$. To specify $P[x|\tau]$, we compare Equation 3 with 4, and get

$$P[x|\tau] = \frac{I_{x \subseteq \tau} w_x}{\sum_{x' \subseteq \tau} w_{x'}}. \tag{5}$$

To summarize, we have reached a two step model $P[y, x|\tau] = P[y|x]P[x|\tau]$. In the first step, a regression rule's LHS $x \in \mathcal{X}$ is generated based on τ with probability $P[x|\tau]$ given by Equation 5. In the second step, the target variable y is generated by x with probability $P[y|x]$ given by Equation 2.

4.2 EM Algorithm: Learning $\boldsymbol{\alpha_x}$, $\boldsymbol{\beta_x}$ and $\boldsymbol{w_x}$

Denote by $\boldsymbol{\theta} = \{\alpha_x, \beta_x, w_x | x \in \mathcal{X}\}$ the complete set of model parameters. The maximum likelihood estimation of $\boldsymbol{\theta}$ given the training data set is to maximize

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_i \log\left(P[y_i|\tau_i, \boldsymbol{\theta}]\right) = \sum_i \log\left(\sum_{x_i} P[y_i, x_i|\tau_i, \boldsymbol{\theta}]\right), \tag{6}$$

where we have introduced x_i to denote the itemset generated by our probabilistic model for instance i. The difficulty of this optimization problem comes from the summation inside the logarithmic function. This is due to the existence of the hidden variables x_i, which are not directly observable from the training data set. EM algorithm is the standard approach to solve this problem.

EM algorithm is an iterative optimization technique. In the following, we add a subscript t to all model parameters to denote the parameters used by EM algorithm at iteration t. For each iteration t, EM algorithm finds the updated set of parameters $\boldsymbol{\theta}_{t+1}$ given the current parameter estimations $\boldsymbol{\theta}_t$. This is accomplished by maximizing the function

$$\mathcal{Q}(\boldsymbol{\theta}_{t+1}, \boldsymbol{\theta}_t) = \sum_i \sum_{x_i} P[x_i|\tau_i, y_i, \boldsymbol{\theta}_t] \log(P[y_i, x_i|\tau_i, \boldsymbol{\theta}_{t+1}]). \tag{7}$$

This optimization problem is much easier than the original one for Equation 6, due to the fact that the logarithmic function is now inside the summation. The EM algorithm at iteration t is splitted into an E-step which computes $\pi_{i,x_i,t} = P[x_i|\tau_i, y_i, \boldsymbol{\theta}_t]$ and an M-step which optimizes $Q(\boldsymbol{\theta}_{t+1}, \boldsymbol{\theta}_t)$ given $\pi_{i,x_i,t}$. After each iteration, the log-likelihood function $\mathcal{L}$ is guaranteed to be increased, that is, $\mathcal{L}(\boldsymbol{\theta}_{t+1}) \geq \mathcal{L}(\boldsymbol{\theta}_t)$.

At iteration $t = 0$, we initialize the weight $w_{x,0}$ to one and $\alpha_{x,0}$, $\beta_{x,0}$ to the mean and standard deviation of x in $\mathcal{D}$. For the E-step, we first apply Bayes' Theorem so that

$$\pi_{i,x_i,t} = P[x_i|\tau_i, y_i, \boldsymbol{\theta}_t] = \frac{P[y_i|\tau_i, x_i, \boldsymbol{\theta}_t]P[x_i|\tau_i, \boldsymbol{\theta}_t]}{P[y_i|\tau_i, \boldsymbol{\theta}_t]} \propto P[y_i|\tau_i, x_i, \boldsymbol{\theta}_t]P[x_i|\tau_i, \boldsymbol{\theta}_t].$$

According to Equations 5 and 2, we have

$$P[y_i|\tau_i, x_i, \boldsymbol{\theta}_t]P[x_i|\tau_i, \boldsymbol{\theta}_t] \propto \mathcal{N}(y_i|\alpha_{x_i,t}, \beta^2_{x_i,t})w_{x_i,t}I_{x_i \subseteq \tau_i}.$$

Combining these two Equations, we get

$$\pi_{i,x_i,t} = \frac{\mathcal{N}(y_i|\alpha_{x_i,t}, \beta^2_{x_i,t})w_{x_i,t}I_{x_i \subseteq \tau_i}}{\sum_{x' \subseteq \tau_i} \mathcal{N}(y_i|\alpha_{x',t}, \beta^2_{x',t})w_{x',t}}. \tag{8}$$

For the M-step, we split $P[y_i, x_i|\tau_i, \boldsymbol{\theta}_{t+1}]$ as $P[y_i|x_i, \boldsymbol{\theta}_{t+1}]P[x_i|\tau_i, \boldsymbol{\theta}_{t+1}]$, so that $\mathcal{Q} = \mathcal{Q}_1 + \mathcal{Q}_2$, where $\mathcal{Q}_1$ contains only $\alpha_{x,t+1}$, $\beta_{x,t+1}$ and $\mathcal{Q}_2$ contains only $w_{x,t+1}$.

Next, we optimize $\mathcal{Q}_1$ which is given by

$$\mathcal{Q}_1 = \sum_i \sum_{x_i \subseteq \tau_i} \pi_{i,x_i,t} \log(P[y_i|x_i, \boldsymbol{\theta}_{t+1}]).$$

By changing the order of summation, we can write $\mathcal{Q}_1 = \sum_x \mathcal{Q}_x$, where

$$\mathcal{Q}_x = \sum_{i:x \subseteq \tau_i} \pi_{i,x,t} \log(P[y_i|x, \boldsymbol{\theta}_{t+1}]).$$

One can see that different itemsets are decoupled from each other, so we only need to solve $\mathcal{Q}_x$ for $\forall x \in \mathcal{X}$. Observe that $\mathcal{Q}_x$ is nothing but the weighted version of the log-likelihood function of model $P[y|x, \boldsymbol{\theta}_{t+1}] = \mathcal{N}(y|\alpha_{x,t+1}, \beta^2_{x,t+1})$, where the weights are given by $\pi_{i,x,t}$ for instance i. The solution is straightforward:

$$\alpha_{x,t+1} = \frac{\sum_{i:x\subseteq\tau_i} \pi_{i,x,t} y_i}{\sum_{i:x\subseteq\tau_i} \pi_{i,x,t}}, \tag{9}$$

and,

$$\beta^2_{x,t+1} = \frac{\sum_{i:x\subseteq\tau_i} \pi_{i,x,t}(y_i - \alpha_{x,t+1})^2}{\sum_{i:x\subseteq\tau_i} \pi_{i,x,t}}. \tag{10}$$

In Equations 9 and 10, the parameters α_x and β_x are the *weighted mean and standard deviation* where the weight of instance i at iteration t is given by $\pi_{i,x,t}$. This weighting mechanism can help to remove the outlier instance whose $\pi_{i,x,t}$ is small.

Now, we optimize $\mathcal{Q}_2$ which is given by

$$\mathcal{Q}_2 = \sum_i \sum_{x_i\subseteq\tau_i} \pi_{i,x_i,t} \log(P[x_i|\tau_i, \boldsymbol{\theta}_{t+1}]).$$

By plugging Equation 5 into $\mathcal{Q}_2$, and taking the derivative, we get

$$\frac{\partial\mathcal{Q}_2}{\partial w_{x,t+1}} = \sum_{i:x\subseteq\tau_i} (\frac{\pi_{i,x,t}}{w_{x,t+1}} - \frac{1}{\sum_{x'\subseteq\tau_i} w_{x',t+1}}).$$

One can see that different weights $w_{x,t+1}$ are coupled in the above equation. So the exact analytical solution becomes impossible. To ensure the simplicity and computational efficiency of our approach, we make an approximation here by replacing $t+1$ by t in the second term of RHS. Then by setting the derivative to zero, we get

$$\frac{w_{x,t+1}}{w_{x,t}} = \frac{\sum_{i:x\subseteq\tau_i} \pi_{i,x,t}}{\sum_{i:x\subseteq\tau_i} \frac{w_{x,t}}{\sum_{x'\subseteq\tau_i} w_{x',t}}}. \tag{11}$$

From Equations 9, 10 and 11, we see that $\pi_{i,x}$ plays the key role of relating parameters α_x and β_x to weights w_x, so that they can interact with each other and be optimized consistently.

Finally, we note that AREM introduces a parameter M which controls the number of EM-steps. After the EM algorithm is completed, the rule's RHS and weight are finalized to be $\alpha_{x,M}$ and $w_{x,M}$.

4.3 Instance Based Rule Mining

The instance based rule mining is applied in the rule discovery component of AREM discussed at the beginning of Section 4, which selects K rules from $\mathcal{F}_i$ to form $\mathcal{R}_i$ for each training instance i. For this, AREM first ranks rules in $\mathcal{F}_i$ by some "quality" metric, and then select the top K rules. The "quality" metric captures the quality of a rule from an instance's perspective. From our probabilistic model, $P[x|\tau_i, y_i]$ is the natural choice for the "quality" metric: a rule is

Table 1. Data Set Summary

Data Set	BestBuy		CitySearch		Yelp		Airline	Socmob	Pollen	Spacega
	dep	wf	dep	wf	dep	wf				
# of instances	10k	10k	10k	10k	10k	10k	10k	1156	3848	3107
# of items	1347	1010	1530	1080	2273	1662	676	44	17	24
density (%)[a]	1.29	1.52	1.36	1.95	1.35	1.94	1.63	11.36	23.53	25.00
# of trials[b]	20	20	20	20	20	20	20	200	50	60

[a] The "density" captures how sparse the data set is. It is the percentage of non-zero entries if the data is converted into the matrix format.

[b] Number of trials the data set is randomized and then splitted into 80% training set, 10% validation set and 10% testing set.

better if it has a higher probability of being generated by the instance. We use the initialized rule parameters $w_{x,0}$, $\alpha_{x,0}$ and $\beta_{x,0}$ for computing $P[x|\tau_i, y_i]$. From $P[x|\tau_i, y_i] \propto P[x, y_i|\tau_i] \propto \mathcal{N}(y_i|\alpha_{x,0}, \beta^2_{x,0})w_{x,0}$, We have that for the ranking's purpose $P[x|\tau_i, y_i]$ is equivalent to $\mathcal{N}(y_i|\alpha_{x,0}, \beta^2_{x,0})$, where $w_{x,0} = 1$ is dropped. Thus, AREM uses $\mathcal{N}(y_i|\alpha_{x,0}, \beta^2_{x,0})$ for rule ranking for each instance.

4.4 Comparing AREM with RBA

We summarize the main differences between AREM and RBA as follows. First, in determining a small set of itemsets to form the final rules' LHS, AREM applies an instance based approach, while RBA applies the database sequential coverage technique. Second, in determining the final rules' RHS, AREM learns them in the EM framework, while RBA simply uses the mean of the rules' itemsets. It turns out that, in AREM, the rule's RHS is the weighted mean, which is likely to be a better estimation than the unweighted mean used by RBA. Third, in determining the rule weights used for predictions, AREM learns them together with rules' RHS, while RBA pre-specifies methods for computing them. These pre-specified methods may be reasonable but they are not optimized. Finally, in determining top k rules used for making predictions, AREM selects rules with the highest weights, while RBA selects rules with the smallest variance. Our choice is consistent with our probabilistic model in that rules with higher chance of being generated (see Equation 5) are more important and should be selected.

5 Experimental Study

5.1 Data Sets

We evaluate the performance of AREM on 10 data sets summarized in Table 1. The first six data sets are randomly sampled from user reviews downloaded from three websites: "BestBuy" [19], "CitySearch" [20], and "Yelp" [21]. Each instance corresponds to the review of a product where the target variable to predict is the user's rating which ranges from one to five. The review text is parsed and a set of features, or items, is extracted. We constructed two types of features: "dep" and "wf". For "dep", the Stanford dependencies [22] between words in each sentence are extracted. Each dependency is a triplet containing the name of the relation, the governor and the dependent. For "wf", words in the review text are extracted. We remove the infrequent items whose relative supports (that is, the support divided by $|\mathcal{D}|$) are less than 0.5%. The "Airline" data set is downloaded from DataExpo09 competition [23]. The last three data sets are downloaded from CMU StatLib [24].

5.2 Models

For model comparison's purpose, we focus on descriptive models and select several state of the art tree-based and rule-based regression models. The support vector regression (SVR) [14] is an exception. It is included because it is one of the best known and standard models for regression.

SVR We use "libsvm" [25] for *SVR*, and use only the linear kernel. Model parameters tuned are: C and ϵ, where ϵ is the size of ϵ-insensitive tube, and C controls the model complexity.

CART$_k$ This group of models contain the Classification And Regression Tree (CART) [11] and the Boosted Regression Tree [10] where CART of fixed size is acting as the weak learners. So, $CART_k$ stands for CART being boosted k times [26]. We tuned three parameters for $CART_k$: *depth*, *leaf* and *lrate*, where *depth* is the maximum depth of the tree, *leaf* is the minimum number of leaf samples of the tree, and *lrate* is the learning rate of the gradient boosting method.

CUBIST$_k$ Cubist [12] is a rule based algorithm which has the option of building committee models. The number of members in the committee is captured in k. We tuned two binary parameters for $CUBIST_k$: *UB* (unbiased), and *CP* (composite). Parameter *UB* instructs CUBIST to make each rule approximately unbiased. Parameter *CP* instructs CUBIST to construct the composite model.

RBA$_k$ We implemented the RBA model following [9]. Here k is the number of top ranked rules used for prediction. We tuned two parameters for RBA_k: s_0 and *weight*, where s_0 is the minimum support threshold, and *weight* is the weighting scheme used for prediction, which can take three values *supp*, *inv-var* and *equal.*

AREM$_k$ Here, k is the number of top ranked rules used for prediction. We tuned three parameters for $AREM_k$: s_0, K and M, where s_0 is the minimum support threshold, K is the number of high quality rules for each training instance during pruning, and M is the number of EM steps during model training.

The parameter k in the above models (except SVR) can be uniformly interpreted as the number of rules used for making predictions. For our experimental study, we choose k to be 1, 5, 10, 15 and 20 for all four models. The rationale of choosing these values comes from the following: if k is too large, these models' strength of being interpretable essentially disappears; on the other hand, if k is too small, the performance may not be satisfactory. We choose the maximum k value to be 20 as a compromise from these two extreme case considerations.

5.3 Evaluation

We used the Mean Squared Error (MSE) between the actual and predicted target variable's values as the performance metric. For each (model, data) pair,

Table 2. Model Comparison: Average MSE

model\data	BestBuy		CitySearch		Yelp		Airline	Socmob	Pollen	Spacega
	dep	wf	dep	wf	dep	wf				
SVR	0.945	0.810	0.961	0.814	0.935	0.770	**0.643**	0.535	**0.469**	**0.480**
$CART_1$	1.014	0.875	1.131	0.974	1.118	0.924	**0.649**	0.440	0.487	**0.488**
$CART_5$	0.937	0.815	0.997	0.847	0.994	0.804	**0.640**	0.349	0.481	**0.480**
$CART_{10}$	0.921	0.799	0.962	0.827	0.962	0.782	**0.642**	0.349	0.482	**0.481**
$CART_{15}$	0.913	0.790	0.956	0.809	0.946	0.765	**0.640**	0.349	0.483	**0.482**
$CART_{20}$	0.909	0.787	0.949	0.814	0.939	**0.755**	**0.640**	0.341	0.483	**0.484**
$CUBIST_1$	1.043	0.880	1.210	0.990	1.130	0.959	0.658	0.363	0.501	0.490
$CUBIST_5$	1.070	0.937	1.213	0.966	1.129	0.949	0.663	0.367	0.500	0.494
$CUBIST_{10}$	1.074	0.943	1.216	0.973	1.138	0.946	0.664	0.370	0.499	0.492
$CUBIST_{15}$	1.080	0.947	1.218	0.976	1.138	0.944	0.664	0.369	0.499	0.493
$CUBIST_{20}$	1.081	0.951	1.221	0.985	1.137	0.944	0.664	0.369	0.499	0.493
RBA_1	1.111	1.004	1.200	1.141	1.156	1.023	0.730	0.533	0.507	0.530
RBA_5	0.969	0.898	1.044	0.928	1.026	0.930	0.682	0.562	0.496	0.496
RBA_{10}	0.964	0.878	1.041	0.894	1.019	0.915	0.685	0.594	0.497	0.496
RBA_{15}	0.962	0.872	1.040	0.893	1.015	0.904	0.685	0.603	0.497	0.497
RBA_{20}	0.964	0.872	1.038	0.890	1.013	0.903	0.685	0.603	0.497	0.497
$AREM_1$	1.248	1.235	1.354	1.248	1.311	1.241	0.754	0.421	0.581	0.628
$AREM_5$	0.875	0.763	0.908	0.844	0.953	0.799	0.670	**0.307**	0.499	0.529
$AREM_{10}$	**0.862**	**0.751**	**0.896**	0.784	**0.920**	**0.753**	0.657	**0.299**	0.483	0.507
$AREM_{15}$	**0.864**	**0.753**	**0.894**	**0.773**	**0.921**	**0.748**	0.652	**0.299**	0.481	0.490
$AREM_{20}$	**0.865**	**0.758**	**0.899**	**0.770**	**0.926**	**0.749**	**0.646**	**0.300**	0.481	**0.483**

we first identified a set of parameter configurations that was likely to achieve the best performance. The model was then trained on the training set and MSE was calculated on the validation set for each of the parameter configurations. Then we selected the parameter configuration that gives the best MSE on the validation set, and computed the corresponding MSE on the testing set. This process is repeated for the number of trials shown in Table 1. Finally, we reported the average MSE on all testing trials.

For a given data set, in order to compare model m_1 to model m_2, we take into account the distribution of the MSE values computed on multiple testing trials for each model. Let μ_1, σ_1, n_1 (μ_2, σ_2, n_2) be the mean, standard deviation and the number of observations of the set of MSE values for model m_1 (m_2), respectively. We introduce $\mu_{m_1-m_2} = \mu_2 - \mu_1$ and $\sigma_{m_1-m_2} = \sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}$. The quantity $\mu_{m_1-m_2}/\sigma_{m_1-m_2}$ is used in statistical testing [27] for the comparison of two population means. Under the null hypothesis that two population means are the same, $\mu_{m_1-m_2}/\sigma_{m_1-m_2}$ can be assumed to have the Normal distribution $\mathcal{N}(0, 1)$. So the more deviated from zero this quantity is, the more likely that two models are performing differently.

5.4 Experimental Results

The average MSE for the discussed set of models on the various data sets are shown in the Table 2, where the best results have been highlighted. Table 3 shows the quantity $\mu_{m_1-m_2}/\sigma_{m_1-m_2}$ for comparing $AREM_k$ to the rest of the models. Note that $CART_1$ is the standard CART model, in contrast to $CART_k$ which stands for the boosted regression tree. For easy comparison, we derive the win-tie-loss from Table 3 and present them in Table 4.

Table 3. Compare $AREM_k$ To Other Models: $\mu_{m_1-m_2}/\sigma_{m_1-m_2}$

Model\Data	BestBuy		CitySearch		Yelp		Airline	Socmob	Pollen	Spacega
	dep	wf	dep	wf	dep	wf				
$CART_k$	2.86	2.71	4.26	2.65	1.73	0.56	-0.61	2.29	0.09	-0.12
SVR	4.65	4.16	4.97	2.95	1.26	1.80	-0.35	10.51	-2.14	-0.11
RBA_k	4.98	8.15	10.18	8.11	8.64	12.17	3.15	9.68	2.74	0.52
$CART_1$	7.89	8.36	16.78	11.48	16.50	13.84	0.23	6.77	1.15	0.23
$CUBIST_k$	8.04	7.50	20.25	11.49	16.43	13.76	1.03	3.49	3.15	0.31

Table 4. Compare $AREM_k$ To Other Models: win-tie-loss

comparing criteria[a]\model	$CART_k$	SVR_k	RBA_k	$CART_1$	$CUBIST_k$
$\lvert\mu_{m_1-m_2}\rvert \geq \sigma_{m_1-m_2}$	6-4-0	7-2-1	9-1-0	8-2-0	9-1-0
$\lvert\mu_{m_1-m_2}\rvert \geq 2\sigma_{m_1-m_2}$	5-5-0	5-4-1	9-1-0	7-3-0	8-2-0
$\lvert\mu_{m_1-m_2}\rvert \geq 3\sigma_{m_1-m_2}$	1-9-0	4-6-0	8-2-0	7-3-0	8-2-0

[a] It is a tie if $|\mu_{m_1-m_2}| < n\sigma_{m_1-m_2}$. Otherwise, it is a win or loss depending on the sign of $\mu_{m_1-m_2}$.

Tables 3 and 4 show that AREM is performing better than all competing methods on most of the data sets. For almost all cases, AREM is either better or at least as good as the competing method (with the only exception on "Pollen" when compared to *SVR*). It is also interesting to observe that AREM performs almost uniformly well on the review data sets, but not as uniform on the rest of the data sets. Given that the review data sets have much larger number of items (see Table 1), we think this is an indication that AREM is more suitable for high-dimensional and sparse data sets. Finally, from Table 2, we can see how different k values affect the AREM's performance. When $k = 1$, the performance is not satisfactory. This is not surprising because our probabilistic model is optimized for large number of rules. However, as k becomes sufficiently large (15 or 20), the performance improves considerably and remains quite stable.

6 Conclusions

We have proposed a novel regression model based on association rules called AREM. AREM applies the instance based rule mining approach to discover a set of high quality rules. Then the rules' RHS and importance weights are learned consistently within the EM framework. Experiments based on 10 in house and public datasets show our model can perform better than RBA [9], Boosted Regression Trees [10], SVR [14], CART [11] and Cubist [12].

Acknowledgment. This work was supported in part by NSF (IOS-0820730, IIS-0905220, OCI-1048018, CNS-1162405, and IIS-1247632) and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

References

1. Li, W., Han, J., Pei, J.: Cmar: Accurate and efficient classification based on multiple class-association rules. In: ICDM, pp. 369–376 (2001)
2. Yin, X., Han, J.: Cpar: Classification based on predictive association rules. In: SDM (2003)
3. Thabtah, F.A., Cowling, P., Peng, Y.: MMAC: A New Multi-class, Multi-label Associative Classification Approach. In: Proceedings of the 4th IEEE International Conference on Data Mining, ICDM 2004, pp. 217–224 (2004)
4. Wang, J., Karypis, G.: Harmony: Efficiently mining the best rules for classification. In: Proc. of SDM, pp. 205–216 (2005)
5. Cheng, H., Yan, X., Han, J., Yu, P.S.: Direct discriminative pattern mining for effective classification. In: ICDE, pp. 169–178 (2008)
6. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
7. Quinlan, J.R., Cameron-Jones, R.M.: Foil: A midterm report. In: Brazdil, P.B. (ed.) ECML 1993. LNCS, vol. 667, pp. 3–20. Springer, Heidelberg (1993)
8. Cohen, W.W.: Fast effective rule induction. In: ICML, pp. 115–123 (1995)
9. Ozgur, A., Tan, P.N., Kumar, V.: Rba: An integrated framework for regression based on association rules. In: SDM (2004)
10. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. Annals of Statistics 29, 1189–1232 (2000)
11. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth (1984)
12. Quinlan, J.R.: Cubist, `http://www.rulequest.com`
13. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society, Series B 39(1), 1–38 (1977)
14. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Technical report, Statistics and Computing (2003)
15. Kotsiantis, S., Kanellopoulos, D.: Discretization techniques: A recent survey (2006)
16. Thabtah, F.A.: A review of associative classification mining. Knowledge Eng. Review 22(1), 37–65 (2007)
17. Chapelle, O., Chang, Y.: Yahoo! learning to rank challenge overview. Journal of Machine Learning Research - Proceedings Track 14, 1–24 (2011)
18. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: SIGMOD Conference, pp. 1–12 (2000)
19. BestBuy, `http://www.bestbuy.com`
20. CitySearch, `http://www.citysearch.com`
21. Yelp, `http://www.yelp.com`
22. Marneffe de, M.C., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation, CrossParser 2008, pp. 1–8 (2008)
23. Airline, `http://stat-computing.org/dataexpo/2009`
24. StatLib, `http://lib.stat.cmu.edu/datasets`
25. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011)
26. Pedregosa, F., Varoquaux, G., Gramfort, A., et al.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
27. NIST-handbook: Two-sample t-test for equal means, `http://www.itl.nist.gov/div898/handbook/eda/section3/eda353.htm`

One-Class Transfer Learning with Uncertain Data

Bo Liu[1,⋆], Philip S. Yu[2,3], Yanshan Xiao[4], and Zhifeng Hao[4]

[1] Department of Automation, Guangdong University of Technology
csbliu@gmail.com
[2] Department of Computer Science, University of Illinois at Chicago
[3] Computer Science Department, King Abdulaziz University
psyu@uic.edu
[4] Department of Computer Science, Guangdong University of Technology
xiaoyanshan@gmail.com, mazfhao@scut.edu.cn

Abstract. One-class learning aims at constructing a distinctive classifier based on the labeled one class data. However, it is a challenge for the existing one-class learning methods to transfer knowledge from a source task to a target task for uncertain data. To address this challenge, this paper proposes a novel approach, called uncertain one-class transfer learning with SVM (UOCT-SVM), which first formulates the uncertain data and transfer learning into one-class SVM as an optimization problem and then proposes an iterative framework to build an accurate classifier for the target task. Our proposed method explicitly addresses the problem of one-class transfer learning with uncertain data. Extensive experiments has found our proposed method can mitigate the effect of uncertain data on the decision boundary and transfer knowledge to help build an accurate classifier for the target task, compared with state-of-the-art one-class learning methods.

Keywords: Transfer Learning, Data of Uncertainty, One-class Learning.

1 Introduction

One-class learning has been proposed to handle the case where only one class of data is labeled in the training phase [19,17]. In this case, the labeled class of data is called *target class*, while all other samples not in this class are called the *non-target class*. In some real-world applications, such as anomaly detection [5,21], it is easy to obtain one class of normal data, whereas collecting and labeling abnormal instances may be expensive or impossible. To date, one-class learning has been found in a large variety of applications from anomaly detection [5], automatic image annotation [11], to sensor data drift detection [18].

The previous one-class learning can be classified into two broad categories: (1) the methods for one-class learning with unlabeled data [13,12,25,4,24], in which they first extracts negative examples from the unlabeled data, and then constructs a binary classifier based on the labeled target class and the extracted

[⋆] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 471–483, 2013.

negative class. For example, the method in [25] first uses a 1-DNF technique to extract negative documents and utilize SVM to iteratively build a binary classifier. (2) the method for one-class learning without unlabeled data [17,14], in which one-class SVM first maps the target data into a feature space and then constructs a hyper-plane to separate the target class and the origin of the feature space. The learned classifier is then utilized to classify a test sample into target class or non-target class.

Despite much progress on the one-class learning, most of the previous work considers the one-class learning as a single learning task. However, in many real-world applications, we expect to reduce the labeling effort of a new task (referred to as target task) by transferring knowledge from the related task (source task), which is called transfer learning [15]. For example, we may have plenty of user's previously labeled documents, which indicate the users' interest; as time goes on, user's interest may gradually drift; however, we may not have too much user's currently labeled documents, since labeling plenty of documents timely may be impossible for the user. Therefore, we expect the user's previously labeled documents can transfer knowledge to help build an one-class classifier for the target task. Another important observation is that, collected data in many real-world applications is uncertain in nature [2]. This is because data collection methodologies are only able to capture a certain level of information, making the extracted data incomplete or inaccurate [2]. For example, in environmental monitoring applications, sensor networks typically generate a large amount of uncertain data because of instrument errors, limited accuracy or noise-prone wireless transmission [2]. Therefore, it is necessary to develop the one-class transfer learning method for uncertain data, and build an accurate classifier by transferring knowledge from the source task to the target task for prediction.

This paper addresses the problem of one-class transfer learning with uncertain data. To build an one-class transfer learning classifier for uncertain data, we have two challenges. The first one is to formulate data of uncertainty and transfer learning into the one-class learning. The second is to solve the formulated optimization to build an one-class classifier for the target task. To handle the above challenge, we propose a novel approach, called uncertain one-class transfer learning with SVM (UOCT-SVM), which incorporates data uncertainty and knowledge transfer into one-class SVM and provides an efficient framework to build an one-class classifier for the target task. The contribution of our work can be summarized as follows.

1. We incorporate the transfer learning and uncertain data into the one-class SVM such that the transferred knowledge can benefit the one-class classifier for the target task. To handle uncertain data, we introduce the bound score into the learning to relocate the uncertain data and refine the decision boundary.
2. We propose the usage of an iterative framework to mitigate the effect of noise on the one-class classifier and transfer knowledge from the source task to the target task. To the best of our knowledge, this is the first work to

explicitly handle data uncertainty and knowledge transfer in the one-class learning.

3. We conduct extensive experiments to evaluate the performance of our UOCT-SVM method. The results show that our UOCT-SVM can mitigate the effect of noise on the decision boundary and transfer knowledge to help build an accurate classifier for the target task compared with state-of-the-art one-class learning methods.

Section 2 discusses the related work. Section 3 introduces the preliminaries. Section 4 presents our proposed approach. Section 5 reports experimental results. Section 6 concludes the paper and future work.

2 Related Work

In this section, we briefly review previous work related to our study.

2.1 Mining Uncertain Data

In data collection, some records in the data might be degraded due to noise, precision of equipment, and are considered uncertain in their representation [2]. We briefly review the previous work on uncertain data as follows.

For the clustering and classification methods with uncertain data, they develop on the clustering and classification methods. FOPTICS [9] introduces a fuzzy distance function to measure the similarity between uncertain data on top of the hierarchical density-based clustering algorithm. The method in [8] studies the problem of clustering uncertain objects whose locations are described by probability density functions to cluster uncertain data. In addition, binary SVM is extended to handle uncertain data [7] to provide a geometric algorithm.

2.2 Transfer Learning

In transfer learning [15], the knowledge is expected to transfer from a source task into the learning of target task such that the transferred knowledge can benefit the learned classifier for the target task. We briefly review some of them as follows.

The work in [10] assumes the distribution of target and source tasks fit the Gaussian process. However, it assumes the distribution of the data to be specified as a priori, which makes them inapplicable to many real-world applications. Other algorithms such as [16] assume that some instances or features can be used as a bridge for knowledge transfer.

Multi-task learning [20] is closely related to transfer learning. In multi-task learning, several tasks are learned simultaneously. In contrast to multi-task learning, transfer learning focuses on transferring knowledge from the source task to the target task, rather than ensuring the performance of each task.

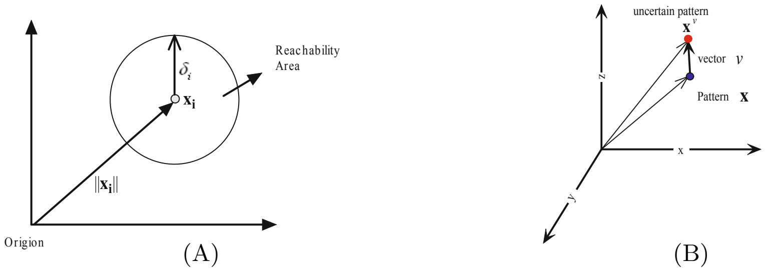


Fig. 1. (A): Illustration of reachability area of instance $\mathbf{x}_i$. (B): Illustration of the method used to add the noise to a data example: $\mathbf{x}$ is an original data example, $\mathbf{v}$ is a noise vector, $\mathbf{x}^\mathbf{v}$ is the new data example with added noise. Here we have $\mathbf{x}^\mathbf{v} = \mathbf{x} + \mathbf{v}$. .

3 Preliminary

3.1 One-Class SVM

Suppose the training target class is $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{|S|}\}$, where $\mathbf{x}_i \in R^n$. In one-class SVM, input data is mapped from the input space into a feature space and the inner product of two vectors $\phi(\mathbf{x})$ and $\phi(\mathbf{x}_i)$ can be calculated by a kernel function $K(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i)$. One-class SVM aims to determine a hyperplane to separate the target class and the origin of the space:

$$\begin{aligned} & min\ \tfrac{1}{2} \parallel \mathbf{w} \parallel^2 - \rho + C \textstyle\sum_{i=1}^{|S|} \xi_i \\ & s.t. \qquad \mathbf{w} \cdot \mathbf{x}_i \geq \rho - \xi_i \\ & \qquad\quad \xi_i \geq 0, \quad i = 1, 2, \ldots, |S|, \end{aligned} \tag{1}$$

where $\mathbf{w}$ is vector, parameter C is used to tradeoff the sphere volume and the errors. After solve problem (1) and obtain $\mathbf{w}$ and $\rho = \mathbf{w} \cdot \phi(\mathbf{x})$. For a test sample $\mathbf{x}_t$, if $\mathbf{w} \cdot \phi(\mathbf{x}_t) > \rho$, it is classified into the target class; otherwise, it belongs to the non-target class.

In this paper, we extend the standard one-class SVM for one-class transfer learning with data of uncertainty.

3.2 Uncertain Model

For the labeled target class, we assume each input data $\mathbf{x}_i$ is subject to an additive noise vector $\triangle \mathbf{x}_i$. In this case, the original uncorrupted input $\mathbf{x}_i^s$ is denoted $\mathbf{x}_i^s = \mathbf{x}_i + \triangle \mathbf{x}_i$. We can assume $\triangle \mathbf{x}_i$ follows a given distribution. The method of bounded and ellipsoidal uncertainties has been investigated in [6,14]. In this situation, we consider a simple bound score for each instance such that $\parallel \triangle \mathbf{x}_i \parallel \leq \delta_i$.

We then let $\mathbf{x}_i + \triangle \mathbf{x}_i$ ($\parallel \triangle \mathbf{x}_i \parallel \leq \delta_i$) denote the *reachability area* of instance $\mathbf{x}_i$ as illustrated in Figure 1. (A). We then have

$$\parallel \mathbf{x}_i^s \parallel = \parallel \mathbf{x}_i + \triangle \mathbf{x}_i \parallel \leq \parallel \mathbf{x}_i \parallel + \parallel \triangle \mathbf{x}_i \parallel \leq \parallel \mathbf{x}_i \parallel + \delta_i \tag{2}$$

In this way, $\mathbf{x}_i^s$ falls in the reachability area of $\mathbf{x}_i$. By using the bound score for each input sample, we can convert the uncertain one-class transfer learning into standard one-class learning with constraints.

4 One-Class Transfer Learning on Uncertain Data

In this section, we put forward our one-class transfer learning to handle uncertain data. Suppose we have two tasks, that is, to train one-class classifier on S_s for source task and on S_t for target task. Let

$$w_1 = w_o + v_1 \quad and \quad w_2 = w_o + v_2, \tag{3}$$

where w_1 and w_2 are parameters of the one-class SVM for source and target tasks, respectively. w_o is a common parameter while v_1 and v_2 are specific parameters. Here, w_o can be considered as a bridge to transfer knowledge from source task to the target task [20]. By assuming $\rho_1 = w_1 \cdot \mathbf{x}$ and $\rho_2 = w_2 \cdot \mathbf{x}$ to be two hyperplanes for S_s and S_t respectively, w_1 and w_2 can be denoted as $w_t = w_0 + v_t, t = 1, 2$ and the extended version of one-class transfer learning for uncertain data can be written as follows.

$$\begin{aligned} min \quad & \tfrac{1}{2}\|w_o\|^2 + \textstyle\sum_{t=1}^{2} C_t\|v_t\|^2 - \rho_1 - \rho_2 + C(\sum_{\mathbf{x}_i \in S_s} \xi_i + \sum_{\mathbf{x}_j \in S_t} \xi_j) \\ s.t. \quad & (w_o + v_1) \cdot (\mathbf{x}_i + \triangle\mathbf{x}_i) \geq \rho_1 - \xi_i, \qquad \mathbf{x}_i \in S_s \\ & (w_o + v_2) \cdot (\mathbf{x}_j + \triangle\mathbf{x}_j) \geq \rho_2 - \xi_j, \quad \xi_j \geq 0 \quad \mathbf{x}_j \in S_t \\ & \xi_i \geq 0, \quad \xi_j \geq 0, \quad \|\triangle\mathbf{x}_i\| \leq \delta_i, \quad \|\triangle\mathbf{x}_j\| \leq \delta_j. \end{aligned} \tag{4}$$

For the above optimization, we then have:

1 $\mathbf{x}_i + \triangle\mathbf{x}_i$ and $\mathbf{x}_j + \triangle\mathbf{x}_j$ denote the original vectors which are affected by $\triangle\mathbf{x}_i$ and $\triangle\mathbf{x}_j$. Thus, one-class transfer learning classifier can be less sensitive to the sample corrupted by noise since we can always determine a choice of $\triangle\mathbf{x}_i$ to render $\mathbf{x}_i + \triangle\mathbf{x}_i$ to refine the one-class transfer decision boundary. $\|\triangle\mathbf{x}_i\| \leq \delta_i$ and $\|\triangle\mathbf{x}_j\| \leq \delta_j$ restrict the range of the uncertain information by a bound score, which has been utilized in previous work [14].

2 We utilize common parameter w_o as a bridge to transfer knowledge from source task to target task. Parameters C_1 and C_2 control the preference of the two tasks. If $C_1 > C_2$, task 1 is preferred to task 2; otherwise, task 2 is preferred to task 1. Parameters ξ_i and ξ_j are defined as measures of error.

4.1 Solution to Uncertain One-Class Transfer Learning Classifier

As the above optimization problem (4) is far more complicated than the standard one-class SVM, we will use an iterative approach to calculate ρ_1, ρ_2, $\triangle\mathbf{x}_i$ and $\triangle\mathbf{x}_j$ such that we can obtain the one-class transfer learning classifier for uncertain data. The iterative steps can be summarized as follows. (a): fix each $\triangle\mathbf{x}_i$ and $\triangle\mathbf{x}_j$ to solve the problem (4) to obtain $\overline{\rho}_1$ and $\overline{\rho}_2$; (b): fix the obtained $\overline{\rho}_1$ and $\overline{\rho}_2$ to calculate $\triangle\overline{\mathbf{x}}_i$ and $\triangle\overline{\mathbf{x}}_j$ iteratively. We detail the alternating two steps as follows. (We omit the detailed derivation of the Theorems in this section due to space limitation)

Calculation of Classifier by Fixing $\triangle\ \overline{\mathbf{x}}_i$ and $\triangle\ \overline{\mathbf{x}}_j$. First of all, we fix each $\triangle\ \overline{\mathbf{x}}_i$ and $\triangle\ \overline{\mathbf{x}}_j$ as small values such that $\| \triangle\overline{\mathbf{x}}_i \| < \delta_i$, $\| \triangle\overline{\mathbf{x}}_j \| < \delta_j$ [1]. Based on this, the constrains $\| \triangle\overline{\mathbf{x}}_i \| < \delta_i$, $\| \triangle\overline{\mathbf{x}}_j \| < \delta_j$ in problem (4) won't have effect on the solution. Then, problem (4) is equivalent to

$$\begin{aligned} min \quad & \tfrac{1}{2}\|w_o\|^2 + \textstyle\sum_{t=1}^{2} C_t\|v_t\|^2 - \rho_1 - \rho_2 + C(\sum_{\mathbf{x}_i\in S_s}\xi_i + \sum_{\mathbf{x}_j\in S_t}\xi_j) \\ s.t.\ & (w_o + v_1)\cdot(\mathbf{x}_i + \triangle\overline{\mathbf{x}}_i) \geq \rho_1 - \xi_i, \quad \xi_i \geq 0 \quad \mathbf{x}_i \in S_s \\ & (w_o + v_2)\cdot(\mathbf{x}_j + \triangle\overline{\mathbf{x}}_j) \geq \rho_2 - \xi_j, \quad \xi_j \geq 0 \quad \mathbf{x}_j \in S_t \end{aligned} \tag{5}$$

We then have the following Theorem.

Theorem 1: By using Lagrangian function [22], the solution of the optimization problem (5) is to solve the following dual problem

$$\begin{aligned} F(\alpha) = {} & \tfrac{1}{2}\|w_o\|^2 + C_1\|v_1\|^2 + C_2\|v_2\|^2 - \textstyle\sum_{\mathbf{x}_i\in S_s}\alpha_i[(w_0 + v_1)\cdot\overline{\mathbf{x}}_i] \\ & - \textstyle\sum_{\mathbf{x}_j\in S_t}\alpha_j[(w_0 + v_2)\cdot\overline{\mathbf{x}}_j] \\ s.t. \quad & 0 \leq \alpha_i \leq C, \ \ 0 \leq \alpha_j \leq C, \ \ \textstyle\sum_{\mathbf{x}_i\in S_s}\alpha_i = 1, \ \ \sum_{\mathbf{x}_j\in S_t}\alpha_j = 1, \end{aligned} \tag{6}$$

in which

$$w_o = \textstyle\sum_{\mathbf{x}_i\in S_s}\alpha_i\cdot\overline{\mathbf{x}}_i + \sum_{\mathbf{x}_j\in S_t}\alpha_j\cdot\overline{\mathbf{x}}_j,$$
$$v_1 = \tfrac{1}{2}\textstyle\sum_{\mathbf{x}_i\in S_s}\alpha_i\cdot\overline{\mathbf{x}}_i, \quad v_2 = \tfrac{1}{2}\sum_{\mathbf{x}_j\in S_t}\alpha_j\cdot\overline{\mathbf{x}}_j,$$

where α_i and α_j are the Lagrange multipliers and $\overline{\mathbf{x}}_i = \mathbf{x}_i + \triangle\mathbf{x}_i$, $\overline{\mathbf{x}}_j = \mathbf{x}_j + \triangle\mathbf{x}_j$. After solve optimization problem (6), we obtain α_i and α_j, and $\overline{\rho}_1 = (w_0 + v_1)\cdot\mathbf{x}_i$ and $\overline{\rho}_2 = (w_0 + v_2)\cdot\mathbf{x}_j$.

Calculation of $\triangle\ \overline{\mathbf{x}}_i$ and $\triangle\ \overline{\mathbf{x}}_j$ by Fixing the Classifier. After setting $\triangle\overline{\mathbf{x}}_i$ and $\triangle\overline{\mathbf{x}}_j$ as a small values which are less than δ_i and δ_j respectively, and solving optimization problem (6), we obtain $\overline{\rho}_1$ and $\overline{\rho}_2$. The next step is to use the obtained $\overline{\rho}_1$ and $\overline{\rho}_2$ to calculate new $\triangle\overline{\mathbf{x}}_i$ and $\triangle\overline{\mathbf{x}}_j$. We then have Theorem 2 as follows.

Theorem 2: If the hyperplanes for the source and target tasks are denoted as $\overline{\rho}_1 = (\overline{w}_0 + \overline{v}_1)\cdot\mathbf{x}$, and $\overline{\rho}_2 = (\overline{w}_0 + \overline{v}_2)\cdot\mathbf{x}$, the solution of problem (4) over $\triangle\overline{\mathbf{x}}_i$ and $\overline{\mathbf{x}}_j$ are

$$\triangle\overline{\mathbf{x}}_i = \delta_i\frac{\overline{w}_0 + \overline{v}_1}{\| \overline{w}_0 + \overline{v}_1 \|}, \tag{7}$$

$$\triangle\overline{\mathbf{x}}_j = \delta_j\frac{\overline{w}_0 + \overline{v}_2}{\| \overline{w}_0 + \overline{v}_2 \|}. \tag{8}$$

This Theorem indicates that, for a given $\overline{\rho}_1$ and $\overline{\rho}_2$, the minimization of problem (4) over $\triangle\overline{\mathbf{x}}_i$ and $\triangle\overline{\mathbf{x}}_j$ is quite straightforward.

After that, we have one round of alternation and continue to update $\overline{\rho}_1$, $\overline{\rho}_2$, $\triangle\overline{\mathbf{x}}_i$, $\triangle\overline{\mathbf{x}}_2$ until the algorithm converges.

[1] We set $\triangle\ \overline{\mathbf{x}}_i = 0$ and $\triangle\ \overline{\mathbf{x}}_j = 0$ in the first step of the iterative framework.

Algorithm 1. Uncertain one-class transfer learning with uncertain data

```
Input: $S_s$, $S_t$ ; // source and target tasks
........... $C_1$, $C_2$ and $C$. // parameters
........... $\delta_i$, $\delta_j$ // bound value for samples in both tasks.
Output: $\overline{\rho}_1$ and $\overline{\rho}_2$.
Initialize each $\triangle\overline{\mathbf{x}}_i = 0$ and $\triangle\overline{\mathbf{x}}_j = 0$;
t=0;
Initialize $F_{va}(t) = \infty$;
repeat
    $t = t + 1$;
    Fix $\triangle\overline{\mathbf{x}}_i$ for $i = 1, 2, \ldots, |S_s|$ and $\triangle\overline{\mathbf{x}}_j$ for $j = 1, 2, \ldots, |S_t|$ to solve problem (5);

    Let $F_{va}(t) = F(\alpha)$;
    Obtain $\alpha_i, i = 1, 2, \ldots, |S_s|$;
    Obtain $\alpha_j, j = 1, 2, \ldots, |S_t|$;
    Obtain the hyperplane $\overline{\rho}_1 = (\overline{w}_0 + \overline{v}_1) \cdot \mathbf{x}$ for source task;
    Obtain the hyperplane $\overline{\rho}_2 = (\overline{w}_0 + \overline{v}_2) \cdot \mathbf{x}$ for the target task;
    Fix $\overline{\rho}_1$ and $\overline{\rho}_1$ to update each $\triangle\overline{\mathbf{x}}_i$ and $\triangle\overline{\mathbf{x}}_j$ according to Equation (7) and (8);
until $|F_{va}(t) - F_{va}(t-1)| < \varepsilon |F_{va}(t-1)|$
Return $\overline{\rho}_1 = (\overline{w}_0 + \overline{v}_1) \cdot \mathbf{x}$ and $\overline{\rho}_2 = (\overline{w}_0 + \overline{v}_2) \cdot \mathbf{x}$.
```

To utilize Theorem 1 and Theorem 2 iteratively to calculate $\overline{\rho}$ and $\triangle\overline{\mathbf{x}}$, we have Theorem 3 as follows.

Theorem 3: If optimal $\triangle\overline{\mathbf{x}}_i = \delta_i \frac{\overline{w}_0 + \overline{v}_1}{\|\overline{w}_0 + \overline{v}_1\|}$ and $\triangle\overline{\mathbf{x}}_j = \delta_j \frac{\overline{w}_0 + \overline{v}_2}{\|\overline{w}_0 + \overline{v}_2\|}$ are fixed, the solution of problem (4) is equivalent to optimization problem (6).

From $\triangle\overline{\mathbf{x}}_i = \delta_i \frac{\overline{w}_0 + \overline{v}_1}{\|\overline{w}_0 + \overline{v}_1\|}$, we have $\|\triangle\overline{\mathbf{x}}_i\| = \delta_i \frac{\|\overline{w}_0 + \overline{v}_1\|}{\|\overline{w}_0 + \overline{v}_1\|} = \delta_i$, then the constrains $\triangle\overline{\mathbf{x}}_i \leq \delta_i$ in problem (4) won't have any effect on problem (4). The same analysis can be used to $\triangle\overline{\mathbf{x}}_j = \delta_j \frac{\overline{w}_0 + \overline{v}_2}{\|\overline{w}_0 + \overline{v}_2\|}$. Thus, problem (4) equals to problem (6).

Iterative Framework. So far, we have introduced the framework to update $\overline{\rho}_1$, $\overline{\rho}_2$, $\triangle\overline{\mathbf{x}}_i$ and $\triangle\overline{\mathbf{x}}_j$ at a round, and we can use the above steps to obtain an uncertain one-class transfer learning classifier. By referring to the alternating optimization method in [6], we propose the usage of the iterative approach to solve problem (4) in Algorithm 1.

In Algorithm 1, ε is a threshold. Since the value of $F_{val}(t)$ is nonnegative, with the decreasing of $F_{val}(t)$, $|F_{val}(t) - F_{val}(t-1)|/|F_{val}(t-1)|$ will be smaller than a threshold. Thus, Algorithm 1 can converge in finite steps.

After that, we obtain the uncertain one-class transfer learning classifiers for the target task. We then utilize the learned classifier for prediction.

Note:(1): For the determination of δ_i for the sample $\mathbf{x}_i$ in S_s, we calculate the average distance of $\mathbf{x}_i$ between it and the its $k-$nearest neighbors. The same operation is utilized to the sample $\mathbf{x}_j$ in S_t. This setting is previously utilized in the previous work [14]. At the beginning of the framework, we initialize each $\triangle\overline{\mathbf{x}}_i = 0$, $\triangle\overline{\mathbf{x}}_j = 0$ and update them base on (7) and (8). Then, we can have

$\triangle\overline{\mathbf{x}}_i \leq \delta_i$ and $\triangle\overline{\mathbf{x}}_j \leq \delta_j$. (2): Above, we present the formulation of uncertain one-class transfer learning in the input space; while for the kernel space, we can utilize $K(\mathbf{x}, \mathbf{q}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{q})$ in the above formulation.

5 Experiments

5.1 Baseline and Metrics

In this section, we investigate the performance of our proposed UOCT-SVM method. In transfer learning, we expect the transferred knowledge from the source task to the target task can improve the performance of the classifier built on the target task. For comparison, another two methods are used as baselines.

1. The first method is the standard one-class SVM (OC-SVM), which determines a hyperplane to separate the target class and the origin of feature space. This baseline is used to show the improvement of our method over the standard one-class SVM.
2. The second baseline is the uncertain one-class SVM (UOC-SVM) [14], which builds one-class classifier on uncertain data. This baseline is utilized to investigate the ability of transferred knowledge contributed to the construction of classifier on the target data.

The performance of classification systems is typically evaluated in terms of F-measure [23], we use it as metrics. The F measure trades off precision p and recall r: $F = 2p \cdot r/(p + r)$. From the definition, we know only when both precision and recall are large, will the F-measure exhibit large value.

5.2 Dataset and Experiment Setting

One-Class Learning Data. To evaluate the properties of our approach, we conduct experiments on 20 Newsgroups [2] and Reuters-21578 [3]. Both data sets have hierarchical structures. The 20 Newsgroups corpus contains several top categories, and under the top categories, there are 20 sub-categories where each subcategory has 1000 samples. Similarly, Reuters-21578 contains Reuters news wire articles organized into five top categories, and each category includes different sub-categories.

Following the previous work [17,19] for one-class learning, we reorganize the original data in a way for the one-class transfer learning problem as follows. For the 20 Newsgroups, we consider one sub-category as target class, and select a number of example from other categories as non-target class. Specifically, we first choose a sub-category (a_1) from a top category (A), and consider this sub-category (a_1) as the target class and consider the examples from other top categories, i.e., except for category (A) as non-target class. Based on the this, we generate target class and non-target class for the source task. For the target

[2] Available at `http://people.csail.mit.edu/jrennie/20Newsgroups/`

[3] Available at `http://www.daviddlewis.com/resources/testcollections/`

task, we choose a sub-category (a_2) from the same top category (A) as that for the source task, and consider this sub-category (a_2) as the target class; while take the examples from other top categories except for (A) as non-target class.

For the Reuters-21578, each top category has many sub-categories, for example, "people" has 267 sub-categories and the size of each sub-category is not always large. We organize it as follows. For a top category (A), all of the subcategories are organized into two parts (denoted as $a(1)$ and $a(2)$), and each part is approximately equal in size. We then regard $a(1)$ and $a(2)$) as the target class of the source task and target task respectively. We also consider the examples from the other category except for (A) as the non-target class for the source and target task respectively.

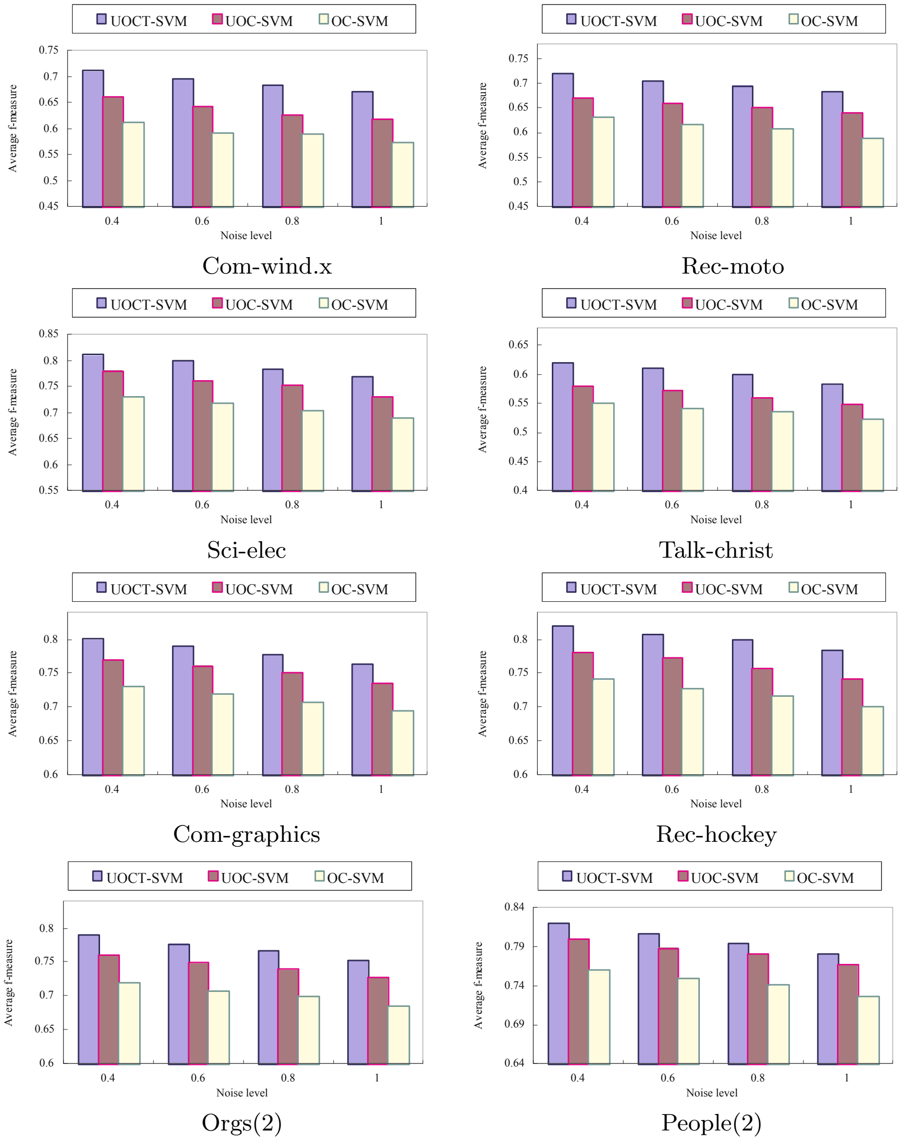
In the above operations, we generate target class from the same top category (A), that are $a(1)$ and $a(2)$, for the source task and target task, this is because we should guarantee the two tasks are related. Otherwise, the transfer learning may not, and may even hurt, the performance of a target task, which can be referred to as negative transfer [3].

Uncertain Information Generation. We note that the above data are deterministic, so we need to model and involve uncertainty to these data sets. Following the method in the previous work [1], we generate the uncertain data as follows.

For generate data, we first compute the standard deviation σ_i^0 of the entire data along the ith dimension, and then obtain the standard deviation of the Gaussian noise σ_i randomly from the range $[0, 2\cdot\eta\cdot\sigma_i^0]$. For the ith dimension, we add noise from a random distribution with standard deviation σ_i. Thus, a data example $\mathbf{x}_j$ is added with the noise, i.e., $\sigma^{\mathbf{x}_j} = [\sigma_1^{\mathbf{x}_j}, \sigma_2^{\mathbf{x}_j}, \cdots, \sigma_{r-1}^{\mathbf{x}_j}, \sigma_r^{\mathbf{x}_j}]$. Here, r denotes the number of dimensions for a data example $\mathbf{x}_j$, and $\sigma_i^{\mathbf{x}_j}$, $i = 1, \cdots r$ represents the noise added into the ith dimension of the data example. Fig. 1 (B) illustrates the basic idea of the method.

In the experiment, RBF kernel function ($K(\mathbf{x}, \mathbf{x}_i) = exp(- \| \mathbf{x} - \mathbf{x}_i \|_2^2 / 2\sigma^2)$) is used in the experiment since it is the most common kernel function. The σ in RBF kernel function is ranged from 2^{-10} to 2^{10}. In our method, C_1 and C_2 control the tradeoff between the source task and target task. Since we care about the target task more than the source task, we set $C_2 > C_1$ and C_1, C_2, C is chosen from 1 to 1000. For the k-nearest neighbors to generate bound score, we set k equal to ten percent of the training target class. We set ε is set as 0.15 in the experiment. All the experiments are on a laptop with a 2.8 GHz processor and 3GB DRAM.

Performance Comparison. For the target data set, we randomly choose around 20% to form a training set while the remaining example are used for testing. This is because transfer learning always assumes we do not have sufficient training data for the target task. We also conduct 10-fold cross validation on the test set. For the source data, since we are more concerned about the performance of the target task we incorporate around 80% them into training and



Com-wind.x

Rec-moto

Sci-elec

Talk-christ

Com-graphics

Rec-hockey

Orgs(2)

People(2)

Fig. 2. The performance of OC-SVM, UOC-SVM and UOCT-SVM at different noise level

Table 1. The average F-measure accuracy and standard deviation for the target task obtained by OC-SVM, UOC-SVM and UOCT-SVM methods

Number	Source task	Target task	OC-SVM	UOC-SVM	UOCT-SVM
1	Com-misc	Com-wind.x	0.61 ± 0.042	0.66 ± *0.035*	*0.71* ± *0.035*
2	Rec-autos	Rec-moto	0.62 ± 0.034	0.67 ± 0.031	*0.72* ± *0.029*
3	Sci-med	Sci-elec	0.73 ± 0.053	0.78 ± *0.042*	*0.81* ± *0.042*
4	Talk-religion	Talk-christ	0.53 ± 0.055	0.58 ± 0.052	*0.62* ± *0.049*
5	Com-wind.misc	Com-graphics	0.71 ± 0.045	0.77 ± *0.042*	*0.80* ± *0.042*
6	Rec-baseball	Rec-hockey	0.72 ± 0.032	0.78 ± 0.031	*0.82* ± *0.030*
7	Orgs(1)	Orgs(2)	0.70 ± 0.051	0.76 ± 0.049	*0.79* ± *0.045*
8	People(1)	People(2)	0.74 ± 0.047	0.80 ± *0.035*	*0.82* ± 0.042

the remaining are used for testing. To avoid a sampling bias, we repeat the above process 10 times, and report the average f-measure accuracy and the standard deviations in Table 1, in which we set the noise level at 0.4.

It can be seen that, our proposed UOCT-SVM method always provides a superior performance compared with UOC-SVM. Although both UOCT-SVM and UOC-SVM can handle data of uncertainty, our method can transfer knowledge from the source task to the target task such that we can develop an accurate classifier for the target task. In addition, both UOCT-SVM and UOC-SVM perform much better than the standard OC-SVM, this occurs because UOCT-SVM and UOC-SVM reduce the effect of the noise on the decision boundary; as results, they can deliver better one-class classifier compared with the standard OC-SVM. In addition, we find the standard deviation of our method is less than the UOC-SVM and OC-SVM for most data sets.

Performance on Different Noise Levels. We investigate the performance sensitivity of three methods on different noise level from 0.4 to 1. In Fig. 2, we illustrate the variation in effectiveness with increasing noise error. On the $x-$axis, we illustrate the noise level. On the $y-$axis, we illustrate the average f-measure value. It is clear that in each case, the f-measure value reduces with the increasing noise level. This occurs because when the level of noise increases, the target class potentially becomes less distinguishable from the non-target class. However, we can clearly see that, UOCT-SVM approach can still consistently yield higher f-measure value than OC-SVM and UOC-SVM. This indicates that, UOCL method can reduce the effect of noise. In addition, UOC-SVM performs better than OC-SVM since UOC-SVM can reduce the effect of noise on the decision classifier.

Average Running Time Comparison. So far, we have investigated the performance of the three methods, it is interesting to compare the running time of them. The average running time of OC-SVM, UOC-SVM and UOCT-SVM are 1553, 4763 and 6980 seconds respectively. We find that the standard one-class SVM performs much faster than UOC-SVM and UOCT-SVM since

OC-SVM does not consider the data of uncertainty and transfer knowledge in the learning; as a result, it performs faster while has the lowest accuracy compared with UOC-SVM and UOCT-SVM. In addition, UOC-SVM proceeds faster than UOCT-SVM, since the latter one transfers knowledge from the source task to the target task to benefit the classifier for target task, which takes time to fulfill the knowledge transfer.

6 Conclusion and Future Work

This paper proposes a novel approach, called UOCT-SVM, for one-class transfer learning with uncertain data. Our proposed UOCT-SVM first formulates the uncertain data and transfer learning to the one-class SVM learning as an optimization problem, and then puts forward an efficient framework to solve the optimization problem such that we can obtain an accurate classifier for the target task by transferring knowledge from the source task to the target task. Extensive experiment has investigated the performance of our proposed UOCT-SVM.

In the future, we would like to investigate how to design better methods to generate bound scores based on the data characteristics in a given application domain.

Acknowledgements. This work is supported in part by US NSF through grants IIS-0905215, CNS-1115234, IIS-0914934, DBI-0960443, and OISE-1129076, US Department of Army through grant W911NF-12-1-0066, Google Mobile 2014 Program and KAU grant, Natural Science Foundation of China (61070033, 61203280, 61202270), Natural Science Foundation of Guangdong province (9251009001000005, S2011040004187, S2012040007078), Specialized Research Fund for the Doctoral Program of Higher Education (20124420120004), Australian Research Council Discovery Grant (DP1096218, DP130102691) and ARC Linkage Grant (LP100200774, LP120100566).

References

1. Aggarwal, C.C., Yu, P.S.: A framework for clustering uncertain data streams. In: ICDE, pp. 150–159 (2008)
2. Aggarwal, C.C., Yu, P.S.: A survey of uncertain data algorithms and applications. TKDE 21(5), 609–623 (2009)
3. Cao, B., Pan, J., Zhang, Y., Yeung, D.Y., Yang, Q.: Adaptive transfer learning. In: AAAI (2010)
4. Fung, G.P.C., Yu, J.X., Lu, H., Yu, P.S.: Text classification without negative examples revisit. TKDE 18(6), 6–20 (2006)
5. Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., Kanamori, T.: Statistical outlier detection using direct density ratio estimation. KAIS 26(2), 309–336 (2011)
6. Huffel, S.V., Vandewalle, J.: The total least squares problem: Computational aspects and analysis. Frontiers in Applied Mathematics, vol. 9. SIAM Press, Philadelphia (1991)

7. Yang, J., Gunn, S.: Exploiting uncertain data in support vector classification. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007/ WIRN 2007, Part III. LNCS (LNAI), vol. 4694, pp. 148–155. Springer, Heidelberg (2007)
8. Kao, B., Lee, S.D., Lee, F.K.F., Cheung, D.W., Ho, W.: Clustering uncertain data using voronoi diagrams and r-tree index. TKDE 22(9), 1219–1233 (2010)
9. Kriegel, H.P., Pfeifle, M.: Hierarchical density based clustering of uncertain data. In: ICDE, pp. 689–692 (2005)
10. Lawrence, N.D., Platt, J.C.: Learning to learn with the informative vector machine. In: ICML (2004)
11. Li, J., Su, L., Cheng, C.: Finding pre-images via evolution strategies. Applied Soft Computing 11(6), 4183–4194 (2011)
12. Li, X., Liu, B.: Learning to classify texts using positive and unlabeled data. In: IJCAI, pp. 587–592 (2003)
13. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: ICDM, pp. 179–186 (2003)
14. Liu, B., Xiao, Y., Cao, L., Yu, P.S.: One-class-based uncertain data stream learning. In: SDM, pp. 992–1003 (2011)
15. Pan, S.J., Tsand, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE TNN 22(2), 199–210 (2011)
16. Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: ICML (2007)
17. Schölkopf, B., Platt, J., Taylor, J.S., Smola, A.J., Williamson, R.: Estimating the support of a high-dimensional distribution. Neural Computation 13, 1443–1471 (2001)
18. Takruri, M., Rajasegarar, S., Challa, S., Leckie, C., Palaniswami, M.: Spatio-temporal modelling-based drift-aware wireless sensor networks. Wireless Sensor Systems 1(2), 110–122 (2011)
19. Tax, D.M.J., Duin, R.P.W.: Support vector data description. Machine Learning 54(1), 45–66 (2004)
20. TEvgeniou, T., Pontil, M.: Regularized multi–task learning. In: KDD (2004)
21. Trung, L., Dat, T., Phuoc, N., Wanli, M., Sharma, D.: Multiple distribution data description learning method for novelty detection. In: IJCNN, pp. 2321–2326 (2011)
22. Vapnik, V.: Statistical learning theory. Springer, London (1998)
23. William, J., Shaw, M.: On the foundation of evaluation. American Society for Information Science 37(5), 346–348 (1986)
24. Xiao, Y., Liu, B., Yin, J., Cao, L., Zhang, C., Hao, Z.: Similarity-based approach for positive and unlabeled learning. In: IJCAI, pp. 1577–1582 (2011)
25. Yu, H., Han, J., Chang, K.C.C.: Pebl: Web page classification without negative examples. TKDE 16(1), 70–81 (2004)

Time Series Forecasting Using Distribution Enhanced Linear Regression

Goce Ristanoski[1,2], Wei Liu[2], and James Bailey[1,2]

[1] NICTA Victoria Laboratory
[2] Department of Computing and Information Systems, The University of Melbourne, Australia
g.ristanoski@student.unimelb.edu.au,
{wei.liu,baileyj}@unimelb.edu.au

Abstract. Amongst the wealth of available machine learning algorithms for forecasting time series, linear regression has remained one of the most important and widely used methods, due to its simplicity and interpretability. A disadvantage, however, is that a linear regression model may often have higher error than models that are produced by more sophisticated techniques. In this paper, we investigate the use of a grouping based quadratic mean loss function for improving the performance of linear regression. In particular, we propose segmenting the input time series into groups and simultaneously optimizing both the average loss of each group and the variance of the loss between groups, over the entire series. This aims to produce a linear model that has low overall error, is less sensitive to distribution changes in the time series and is more robust to outliers. We experimentally investigate the performance of our method and find that it can build models which are different from those produced by standard linear regression, whilst achieving significant reductions in prediction errors.

Keywords: Time series prediction, samples grouping, quadratic mean.

1 Introduction

The forecasting of time series is a well known and significant prediction task. Many methods have been proposed in this area, ranging from the simple to the very sophisticated. An important class of techniques includes methods which learn a model based on optimizing a regularized risk function, such as linear regression, Gaussian processes, neural networks and support vector machines. Common drawbacks in the development of time series forecasting methods are that many of them are applicable to only a specific type of time series, such as medical data or stock market series; they may require certain conditions to be fulfilled; or the improvement in performance is associated with a high increase in complexity.

Linear regression is one of the most popular and widely used techniques for time series prediction, since it is simple, intuitive and produces models with a

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 484–495, 2013.

high degree of interpretability. Its performance is also often surprisingly good compared to more complex methods. Nevertheless, reduction of prediction error is a key concern and it remains attractive to consider techniques for improving the behavior of standard linear regression, particularly for challenging circumstances. Such circumstances include non stationary and noisy time series, where changes in the distribution of the series often occur over time.

Given these issues, this paper investigates a modification of linear regression that takes a different perspective. Given a single univariate time series, instead of optimizing the arithmetic mean of the loss over all training samples, we propose to optimize the quadratic mean of the loss over groups of training samples. In particular, the univariate time series is segmented into a number of groups, where each group contains one or more samples. A linear model is then learned which simultaneously optimizes both the average loss of each group, as well as the variance of the loss across groups. In other words, there are two concurrent optimization objectives. First, the model which is produced should have low overall error rate - this is achieved by ensuring the average loss within each group is small. Second, the groups should not vary too much with respect to the error rate of each individual group - this ensures that there is no single group which can significantly bias the characteristics of the output model. A primary question is how should the univariate time series be segmented into groups ? Our proposal is to segment the samples according to their distribution characteristics. The intuition here is that we would like to learn a model which is not biased towards any single distribution, since we do not know which distribution the future behavior of the time series will most resemble.

We experimentally evaluate the performance of our technique using 20 real stock market datasets, 5 non-financial time series datasets and 5 synthetic datasets. We find that our new method (which we call QMReg) can produce linear models which are quite different from those of standard linear regression, and often have significantly less error, with empirical reductions typically in the range of 10%-30%. Our proposed method is an intuitive technique that ensures more evenly distributed, less volatile error and sensitivity to changes in the distribution of the series.

2 Related Work

Data mining researchers have a range of different types of classification and regression algorithms at their disposal, and many of them have been applied to the time series forecasting problem. Artificial Neural Networks(ANNs) [17], Support Vector Machines [13] and Hidden Markov Models [20] [9] are other methods that have been used with some success for financial time series forecasting, and clustering has been used as an aid in the forecasting process as well.

The practical use of linear regression along with statistical knowledge is embodied in the Autoregressive Integrated Moving Average (ARIMA) models, which are descriptive, intuitive and often perform as well as advanced models. Weighted linear regression and AutoRegressive Conditional Heteroskedasticity

(ARCH) models are more complex modifications of linear regression, which require some additional statistical expertise.

Robust regression is a more complex type of linear regression that assesses the statistical properties of the data samples when learning a model [15], by handling the outliers in the data, and assigning appropriate weights (or losses) to reduce their influence [18]. Incorrect labelling of a sample as an outlier can be a concern when using robust regression. Choosing the most appropriate approach often requires statistical expertise by the user.

Since our method is based on grouping of instances as input to a loss minimization function, a potentially related area of research is regularized multitask learning [16], where the objective is to learn multiple classification tasks simultaneously, rather than independently. However, to our knowledge, research in multitask learning has not used a quadratic mean loss function for the simultaneous optimization of loss across groups, as is done in this paper.

3 Distribution Based Quadratic Mean Convex Optimization

We propose an algorithm that has two phases: (1) detection of distribution change points to segment the time series into groups, and (2) training the regression model using a quadratic mean to minimize both the individual loss of each group and the variance of the loss across groups.

3.1 Time Series Segmentation - Distribution Change Points Detection

Distribution changes in time series variables are from a continuous process, subject to a set of external factors [1] [5]. The task of breaking up the samples of a dataset into segments or groups based on distribution or similarities is not an unfamiliar challenge - simple clustering be effective in many cases. When it comes to time series, as we may prefer to preserve the time element, distribution based methods can also be used. Potential candidate tests for non-parametric change point detection methods include the Wilcoxon rank sum method (WXN) and the kernel change method (KCD) [14]. They can be applied for this task as they are understandable and easy to introduce in the learning process. The Wilcoxon rank sum method (WXN) assess whether two sets of data samples follow the same distribution according to a statistical measure. It is an easy to implement statistical test, and no a-priori knowledge is required (other than specification of an appropriate p-value, commonly set to 0.05).

The WXN paradigm is as follows: for a fixed window of m points, [1, m], we appoint after it another sliding window of the same length, $[m+1,\ 2m]$. We move the second window and compare if the samples in both windows follow the same distribution: if that is the case, we continue moving the second window, until the distribution changes. The change point will be at the last sample of the second window (point $2m+p$), p>0, where we detect the group window v for that group of samples; we move the first window just after that point $[2m+p+1, 3m+p]$, the second window comes after the first one $[3m+p+1, 4m+p]$ and we repeat the process for the rest of the dataset(Figure 1). The choice of the window size m can vary, and we choose it to be the same size as the testing set.

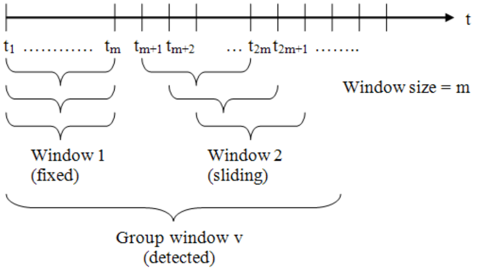


Fig. 1. The Wilcoxon method with a fixed reference window and a sliding window

3.2 Quadratic Mean Based Empirical Loss Function

The quadratic mean is defined as the square root of the average of the squares of each element in a set. In the case of just two errors, ϵ_1 and ϵ_2, the values of the quadratic mean QM and the arithmetic mean AM can be written as:

$$AM = \frac{\epsilon_1 + \epsilon_2}{2}, QM = \sqrt{\frac{\epsilon_1^2 + \epsilon_2^2}{2}} = \sqrt{AM^2 + (\frac{\epsilon_1 - \epsilon_2}{2})^2} \tag{1}$$

This shows that the quadratic mean is lower bounded by the arithmetic mean, and this bound is reached when ϵ_1=ϵ_2. This form of optimization was successfully tested for the scenario of imbalanced relational datasets [7] where there are only two groups (positive and negative classes). The more advanced form we use in our methods is specialised for the case of time series and permits any number of groups.

Many machine learning methods address the learning process as finding the minimum of the regularized risk function. For n training samples ($\mathbf{x}_i$, y_i) (i=1,..,n), where $\mathbf{x}_i \in \mathbb{R}^d$ is the feature vector of the i–th training sample, d is the number of features, and $y_i \in \{-1, 1\}$ is the true label for the i–th training sample (in the case of classification) and $y_i \in \mathbb{R}$ in the case of regression, the regularized risk function is:

$$\mathbf{w}^* = argmin_{\mathbf{w}} \ \lambda \mathbf{w}^T \mathbf{w} + R_{emp}(\mathbf{w}) \tag{2}$$

$\mathbf{w}$ is the weight vector which also includes the bias term b (which makes $\mathbf{x}$ having and additional bias feature $x_{d+1} \equiv 1$), and λ is a positive parameter that balances the two items in Equation 2, and $R_{emp}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} l(\mathbf{x}_i, y_i, \mathbf{w})$. The loss function $l(\mathbf{x}_i$,y_i,$\mathbf{w})$ in $R_{emp}(\mathbf{w})$ measures the distance between a true label y_i and the predicted label from the forecasting done using $\mathbf{w}$, and in the case of Linear Regression it has the form of $\frac{1}{2}(\mathbf{w}^T\mathbf{x}$-$y)^2$.

We investigate the use of the quadratic mean as a risk function which balances k values, instead of just two values. Each value represents the average loss for a group of instances. The grouping of instances can be conducted in a range of ways. A simple way is to segment the time series data into k groups by their distribution, and we use the Wilcoxon method for that purpose. Each group consists of consecutive data samples, and may vary in size depending on the distribution of the underlying data. The effect of using the quadratic mean is to produce a model which optimizes the average loss for each group, as well as the variance of the average loss across the k groups.

The details behind the groups error optimization are as follows: with n samples, we denote the sizes of k groups as n_1, n_2,..,n_k, where n_1+n_2+..+n_k=n. The empirical loss function of k group has the form of:

$$R^{QM}_{emp,k}(\mathbf{w}) = \sqrt{\frac{\sum_{j=1}^{k} f_j(\mathbf{w})^2}{k}} \tag{3}$$

where f_j is the average error for group j consisting of n_j consecutive samples following the same distribution

$$f_j(\mathbf{w}) = \frac{\sum_{i=1}^{n_j} l(x_{ji}, y_{ji}, \mathbf{w})}{n_j} \tag{4}$$

where x_{ji} is the i-th sample of group j, and y_{ji} is the i-th output of group j. After some manipulation, it can be rewritten as

$$R^{QM}_{emp,k}(\mathbf{w}) = \sqrt{\mu^2 + \sigma^2} \tag{5}$$

where μ is the mean error of the k groups $\frac{1}{k}\sum_{j=1}^{j=k} f_j(\mathbf{w})$ and σ is the standard deviation of the error across groups $1, \ldots, k$. This form clearly shows the overall loss is the sum of two components, the average loss per group and the variance across groups.

An ideal $\mathbf{w}$ minimizes the square root of the average of squares of errors per group, while keeping the structural risk minimization as well, therefore resulting in the final optimization function

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \ \lambda \mathbf{w}^T \mathbf{w} + R^{QM}_{emp}(\mathbf{w}) \tag{6}$$

This form of calculation of the loss function is the form we use for the proposed QMReg model, and this empirical loss function introduces robustness in the algorithm, making it capable of minimizing the effect of outliers and the error per distribution group. By using linear optimization methods, such as the bundle method [6], we can calculate the subradients of the empirical loss and use them to iteratively update the w vector in a direction that minimizes the quadratic mean loss presented at Formula 3. The loss for a given sample is $l(\mathbf{x}_i,y_i,\mathbf{w})=\frac{1}{2}(\mathbf{w}^T\mathbf{x}$-$y)^2$, and it's gradient will have the from of $l'(\mathbf{x}_i,y_i,\mathbf{w})=(\mathbf{w}^T\mathbf{x} - y)^2$. Using this in the calculation of the loss for a group in Equation 4, for the k groups of

Algorithm 1. Bundle methods for solving the k-group quadratic mean minimization problem

```
Input: convergence threshold $\epsilon$; initial weight vector $\mathbf{w}_0$;
// Step 1: Change point detection
Initialize distribution windows $Window1$ (starting form the first sample) and $Window2$ (which follows $Window1$)
Initialize iteration index $k \leftarrow 1$, initialize group window $v_k$ as empty
repeat
    if samples in $Window1$ and $Window2$ have different distribution (according to Wilcoxon rank sum test) - change point detected then
        $v_k \leftarrow$ samples starting $Window1$ till end of $Window2$
        Set $Window1$ after $v_k$, $Window2$ follows $Window1$
        $k \leftarrow k+1$
    else
        Move $Window2$ one sample further
    end if
until all samples passed
// Step 2: Weight vector training:
Initialize iteration index $t \leftarrow 0$;
repeat
    $t \leftarrow t + 1$;
    Compute subgradient $a_t \leftarrow \partial_{\mathbf{w}} \mathrm{R}^{Q}_{emp,k}(\mathbf{w}_{t-1})$;
    Compute bias $b_t \leftarrow \mathrm{R}^{Q}_{emp,k}(\mathbf{w}_{t-1}) - \mathbf{w}^T_{t-1} a_t$;
    Update the lower bound $\mathrm{R}^{lb}_t(\mathbf{w}) = \max_{1 \leq i \leq t} \mathbf{w}^T a_i + \mathrm{b}_i$;
    Update $\mathbf{w}_t \leftarrow \arg\min_w J_t(\mathbf{w}) = \lambda \mathbf{w}^T \mathbf{w} + \mathrm{R}^{lb}_t(\mathbf{w})$;
    Compute current gap $\epsilon_t \leftarrow \min_{0 \leq i \leq t} J(\mathbf{w}_i) - J_t(\mathbf{w}_t)$
until $\epsilon_t \leq \epsilon$ or $\epsilon_t - \epsilon_{t-1} \approx 0$
Return $\mathbf{w}_t$
```

samples, the subgradient function will have the form of Equation 7, and the detailed pseudo-code of the entire learning process is presented as Algorithm 1.

$$\partial_{\mathbf{w}} R^{Q}_{emp,k}(\mathbf{w}) = \partial_{\mathbf{w}} \sqrt{\frac{\sum_{j=1}^{k} f_j(\mathbf{w})^2}{k}} = \frac{1}{2}(\frac{\sum_{j=1}^{k} f_j(\mathbf{w})^2}{k})^{-\frac{1}{2}}(\sum_{j=1}^{k} \frac{2 f_j(\mathbf{w}) f'_j(\mathbf{w})}{k}) \quad (7)$$

3.3 "Every Sample as a Group" Strategy

We compare our method of k groups with 2 extreme cases - when there is only one group, and when every single instance is a group. It can be easily shown that in the case of 1 group, the quadratic mean is equivalent to the standard linear regression model. As a single instance can be represented as a group, we investigate this research direction as well. The resulting model is QMSampleGroup. In this case the quadratic mean will try to minimize the variance of error across the entire set of samples.

4 Experiments and Results

Evaluation of the performance of the QMReg method was the main target of the experimental work we conducted. To achieve this goal in the experiments both real datasets and synthetic datasets were used. We tested 20 real stock market time series datasets obtained from [11], in the time frame of 2000-2012. We obtained daily stock market closing prices, one of the most often analysed types of data [1]. The sizes of the datasets are between 200 and 600 samples, divided on training set and test set.

A synthetic dataset was created, and 4 different versions of it were also tested (Table 1). The initial set represented a visible deterministic trend, after which 2 types of changes were introduced: increasing or decreasing the last samples, and adding different amounts of noise, in order to test the newly proposed algorithm the ability to work with noisy data. We also performed testing on 5 non-financial time series, revealing opportunities for application of quadratic mean based approach in the non-financial time series domain [12].

Table 1. Synthetic datasets description

Dataset	Description
Simulated sample 1	Visible deterministic trend
Simulated sample 2	Deterministic trend, structural break(in same direction), caused by increasing the last samples, noise(10% of original value)
Simulated sample 3	Deterministic trend, structural break(in opposite direction), caused by decreasing the last samples, noise(10% of orig. value)
Simulated sample 4	Deterministic trend, structural break(in same direction), last samples increased further, noise(25% of original value)
Simulated sample 5	Deterministic trend, structural break(in opposite direction), last samples decreased further, noise(25% of original value)

4.1 Testing and Results

Comparison between 6 methods was conducted in order to evaluate the effect of the QMReg methodology: Standard Least Squares Linear Regression (LS, regressing to past 4 values), Distribution based Quadratic Mean Linear Regression (QMReg, regressing to past 4 values), Quadratic Mean Linear Regression with every sample as a group (QMSampleGroup, regressing to past 4 values), ARIMA(3,0,1), Robust Regression (Huber M-estimator) and SVM Regression(SVM, $\alpha = 1$, C=1, ϵ=0.001, ξ=ξ^*=0.001),). The Root Mean Square Error(RMSE) was chosen as a performance metric, and we also calculate the error reduction (ER) compared to the Least Squares models:

$$ER = \frac{\text{RMSE of LS} - \text{RMSE of QS methods}}{\text{RMSE of LS}} * 100.$$

From the results presented in Table 2, we can clearly see that the QMReg method performed significantly better than the standard Least Squares linear regression,

Table 2. Root Mean Square Error (RMSE) and Error reduction (ER) of QMReg and QMGroupSample methods compared to LS(in %)

	RMSE						ER
Datasets	LS	QMReg	QM Sample Group	ARIMA	Robust Regress.	SVM Regress.	QMReg
Amazon.com	0.725	0.625	0.628	1.85	0.6269	0.629	**13.8**
Apple	3.75	3.513	3.4	16.1	3.566	3.44	**6.3**
American Express	0.913	0.645	0.64	1.61	0.635	0.642	**29.4**
British Airways	0.129	0.1163	0.118	0.228	0.121	0.1182	**9.6**
Boeing	1.36	1.22	1.23	8.28	1.188	1.15	**10.3**
Coke	0.558	0.378	0.428	1.04	0.375	0.382	**32.3**
Colgate Plamolive	0.996	0.69	0.714	1.6	0.693	0.707	**30.7**
Ebay	0.66	0.615	0.622	2.17	0.616	0.614	**6.8**
Fedex	1.99	1.13	1.132	1.67	1.098	0.87	**43.2**
Ford	0.276	0.273	0.275	0.85	0.273	0.267	**1.1**
Hewlett-Packard	0.889	0.554	0.581	1.94	0.566	0.589	**37.7**
IBM	7.54	7.17	7.826	25.6	7.725	7.443	**4.9**
Intel	0.932	0.819	0.83	2.36	0.832	0.838	**12.1**
Island Pacific	1.18	1.12	1.085	1.11	1.113	1.1	**5.1**
Johnson & Johnson	0.439	0.382	0.413	1.1	0.392	0.399	**13.0**
McDonalds	0.756	0.71	0.716	2.5	0.719	0.704	**6.1**
Microsoft	0.439	0.256	0.383	0.91	0.258	0.27	**41.7**
Starbucks	0.619	0.618	0.614	1.86	0.621	0.624	**0.2**
Siemens	2.391	2.1	2.12	3.4	2.097	2.1	**12.2**
Walt Disney	0.75	0.565	0.54	3.2	0.542	0.552	**24.7**
Simulated sample 1	1.82	1.79	1.668	11.87	1.81	1.96	**1.6**
Simulated sample 2	5.6	1.838	1.949	12	1.781	1.843	**67.2**
Simulated sample 3	2.65	1.88	1.898	11.9	1.982	2.06	**29.1**
Simulated sample 4	2.52	2.14	2.81	12.8	2.31	2.17	**15.1**
Simulated sample 5	8.6	6.9	7.39	8.26	6.77	2.67	**19.8**
Chemical process	0.241	0.225	0.274	0.36	0.234	0.228	**6.6**
Temperature anomalies	14.38	13.52	13.57	16.3	13.53	13.57	**6.0**
Radioactivity	14.63	12.315	14.44	10.1	11.197	10.72	**15.8**
Airline passengers	49.78	45.81	46.6	118.8	48.398	56.3	**8.0**
Chocolate	1764	1382	1635	1970	1680	1686	**21.7**
Wilcoxon signed rank test p-value	base	1.83E-06 base	1.3E-04 0.00286 base	3.56E-05 1.30E-05 1.43E-05	9.78E-06 0.2563 0.1682	9.31E-05 0.2845 0.2845	

much better than the ARIMA model, and very similar to the Robust Regression and SVM, and was the method with the most stable convergence towards the optimal weight vector as well, which was not always the case with Least Squares. The performance of the QMReg method was mostly greater for datasets where higher number of groups was detected. The SVM output arguable less easy to interpret - support vector samples in the dataset are not as much of use as simple coefficients for the features. With similar performance as SVM regression,

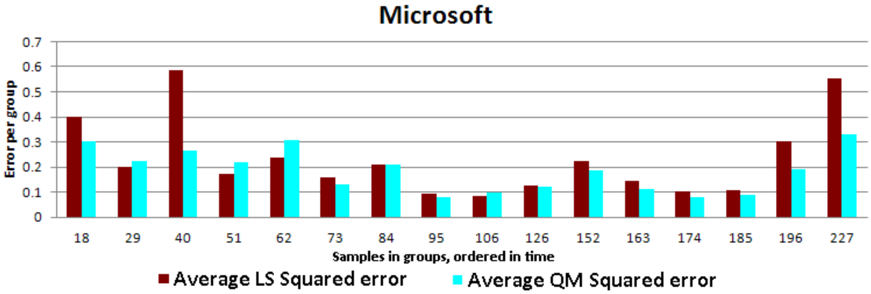


Fig. 2. Error between the groups of the loss, for Microsoft dataset

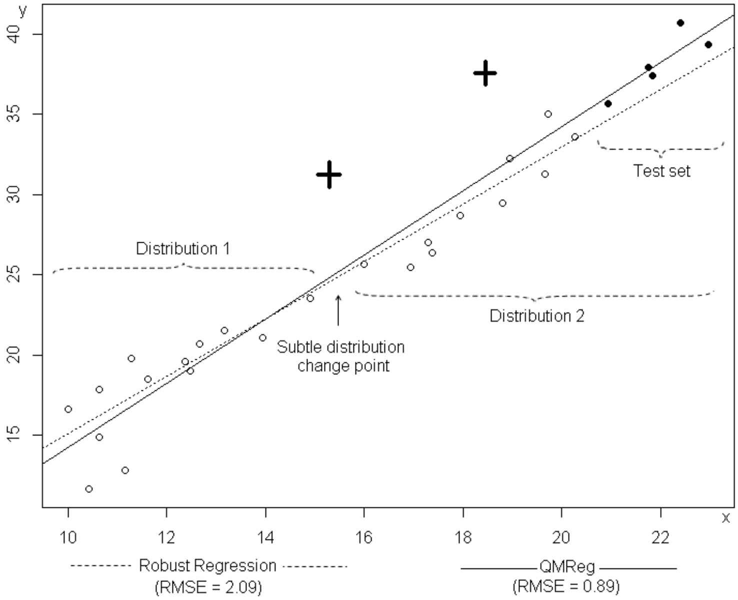


Fig. 3. Robust regression and QMReg comparison - the pluses are the outliers, and the black dots are future samples to be predicted. Robust regression considers the points at the very end as possible outliers, while QMReg detects the subtle change of direction and adjusts the line accordingly.

QMReg does show potential for use in cases when one is not familiar with more complex forecasting methods, but can interpret the output of a linear regression model. The objective of the QMReg to reduce the loss between groups can be seen from Figure 2 which graphically shows how the errors per group are minimized and made more even when QM grouping in performed.

The Robust regression is easy to interpret too, but a more statistical analysis of the data is required - the non-parametric QMReg can be used as a black-box method, and still deliver similar results. The difference with Robust regression can be seen from Figure 3: the crossed points are outliers, and red points are the test samples. We can notice that both methods are dealing successfully with the outliers, but the subtle change in the distribution is more correctly detected by the QMReg method, while robust regression is considering the end points as possible outliers and still keeps the line towards the overall mean. It is this subtle change detection that further leads to lower error for QMReg (RMSE=0.89) than the Robust regression error (RMSE=2.09). We can see the QMReg is highly competitive when compared to Robust regression, and based on Figure 3 we believe it may have the potential to deal with non-stationary behaviour better.

4.2 Loss Function Analysis

By performing grouping among the training set, the QM methods tend to minimize not only the overall training error, but the error per group. This results in a lower variance in the loss, as it can be seen from Table 3: the standard deviation in QMReg method was at its best up to 58% less than the one of the LS method. The grouping performed in the case of QMReg was also conducted for LS. The loss per group was calculated, and standard deviation among the loss per group showed that the QM method indeed minimizes the loss amongst the groups: Table 3 shows the standard deviation of the error per group is statistically lower in the case of QMReg when compared to LS.

5 Conclusion and Future Work

Time series forecasting is a classic prediction problem, for which linear regression is one of the best known and most widely used methods. In this paper, we have proposed a technique that enhances standard linear regression, by employing an optimization objective which explicitly recognises different groups of samples. Each group corresponds to a segment of the time series whose samples have similar distribution characteristics. Our objective simultaneously minimizes the expected loss of each group, as well as the variance of the loss across the groups. By doing so, we ensure a model that produces more stable, less volatile predictions, and capable of additionally minimizing the effect of outliers or noisy data.

The experimental study highlighted the promise of our approach, showing that it could produce linear models different to that of standard linear regression and which also achieved consistent reductions in error in the range 10% to 30% on average, up to 40% error reduction in some cases. As the performance of our proposed method is comparable to more advanced forecasting methods(SVM regression and Robust Regression), our model has an advantage that it improves the performance of linear regression while avoiding unnecessary complexity and unwanted parameters, so it can be used by more general practitioners on diverse types of time series.

For future work, we would like to investigate the use of weighted groupings, for cases where the quality of a group with respect to the prediction task can be estimated, and introduce the time element in the learning process even further.

Table 3. Loss function mean and standard deviation for entire training set, and standard deviation of loss across groups per training set

Dataset	Loss function				Standard deviation for loss across groups	
	LS		QMReg			
	mean	st. dev.	mean	st. dev.	LS	QMReg
Amazon.com	0.7074	1.1087	0.5081	0.8229	0.4888	0.3139
Apple	15.1928	27.2601	13.9676	26.0380	8.8954	8.3127
American Express	1.1356	2.2984	0.8372	1.5941	0.8179	0.5714
British Airways	0.0303	0.0483	0.0203	0.0398	0.0199	0.0156
Boeing	2.2065	3.5453	0.9353	1.5959	1.1735	0.4683
Coke	0.4660	1.1622	0.2504	0.8432	0.5104	0.2234
Colgate Plamolive	1.3929	2.8065	0.7759	2.1025	1.2410	0.5965
Ebay	0.2531	0.4295	0.2224	0.4208	0.1550	0.1060
Fedex	6.4903	9.3358	2.5329	3.9214	3.9518	1.3231
Ford	0.0680	0.1184	0.0681	0.1191	0.0485	0.0481
Hewlett-Packard	0.9649	2.2101	0.5336	1.2165	0.9802	0.4411
IBM	85.553	133.895	60.431	123.244	52.7541	56.8021
Intel	1.2986	2.4139	1.1098	2.0457	0.6658	0.4592
Island Pacific	0.9762	1.3706	0.5165	0.8315	0.4787	0.1660
Johnson & Johnson	0.2526	0.4612	0.2372	0.4137	0.1572	0.1343
McDonalds	0.3491	0.7640	0.2742	0.5731	0.2595	0.1650
Microsoft	0.2600	0.6304	0.1980	0.5074	0.1563	0.0867
Starbucks	0.1654	0.4354	0.1672	0.4239	0.1295	0.1232
Siemens	4.6880	6.1569	4.0694	5.4571	1.7561	1.2753
Walt Disney	0.4286	0.7747	0.2374	0.4255	0.3840	0.1434
Simulated sample 1	3.5835	3.2914	2.9072	2.7637	0.8706	0.6042
Simulated sample 2	14.5109	26.8793	4.6307	15.8900	13.8375	5.0620
Simulated sample 3	9.3680	18.8459	3.7615	8.3814	5.8793	1.6188
Simulated sample 4	40.6497	86.3151	31.5504	80.3505	38.3393	29.8716
Simulated sample 5	93.7317	133.0539	53.3164	104.7921	51.9541	29.3767
Chemical process	0.0680	0.1442	0.0756	0.1653	0.0339	0.0277
Temperature anomalies	232.47	368.44	206.57	318.60	16.7000	52.8821
Radioactivity	175.04	230.35	180.35	244.17	46.7919	31.9748
Airline passengers	809.82	1418.44	796.04	1130.33	490.8909	426.3732
Chocolate	1499216	2158796	952022	1486350	622544.00	467145.00
Wilcoxon signed rank test p-value	base		2.72E-05	3.90E-05	base	0.0001816

References

1. Tsay, R.S.: Analysis of Financial Time Series. Wiley-Interscience (2005)
2. Hulten, G., Spencer, L., et al.: Mining time-changing data streams. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, pp. 97–106 (2001)
3. Keogh, E., Kasetty, S.: On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. Data Mining and Knowledge Discovery 7(4), 349–371 (2003)

4. Dong, G., Han, J., et al.: Online mining of changes from data streams: Research problems and preliminary results. In: Proceedings of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams (2003)
5. Liu, X., Zhang, R., et al.: Incremental Detection of Distribution Change in Stock Order Streams. In: 26th International Conference on Data Engineering Conference, ICDE (2010)
6. Teo, C.H., Vishwanthan, S.V.N., Smola, A.J., Le, Q.V.: Bundle methods for regularized risk minimization. Journal of Machine Learning Research 11, 311–365 (2010)
7. Liu, W., Chawla, S.: A Quadratic Mean based Supervised Learning Model for Managing Data Skewness. In: Proceedings of the Eleventh SIAM International Conference on Data Mining, pp. 188–198 (2011)
8. Vellaisamy, K., Li, J.: Multidimensional decision support indicator (mDSI) for time series stock trend prediction. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 841–848. Springer, Heidelberg (2007)
9. Cheng, H., Tan, P.-N., Gao, J., Scripps, J.: Multistep-ahead time series prediction. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS (LNAI), vol. 3918, pp. 765–774. Springer, Heidelberg (2006)
10. Liu, Z., Yu, J.X., Lin, X., Lu, H., Wang, W.: Locating motifs in time-series data. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 343–353. Springer, Heidelberg (2005)
11. Web enabled scientific services and applications, `http://www.wessa.net/stocksdata.wasp`
12. Hyndman, R.J.: S&P quarterly index online database, `http://robjhyndman.com/tsdldata/data/9-17b.dat`
13. Muller, K.-R., Smola, A.J., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V.: Using Support Vector Machines for Time Series Prediction (2000)
14. Liu, X., Wu, X., Wang, H., Zhang, R., Bailey, J., Kotagiri, R.: Mining distribution change in stock order streams. In: IEEE 26th International Conference on Data Engineering, ICDE (2010)
15. Wilcox, R.R.: Introduction to Robust Estimation and Hypothesis Testing. Elsevier Academic Press, New York (2005)
16. Evgeniou, T., Pontil, M.: Regularized multi–task learning. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 109–117 (2004)
17. Adhikari, R., Agrawal, R.K.: A novel weighted ensemble technique for time series forecasting. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part I. LNCS, vol. 7301, pp. 38–49. Springer, Heidelberg (2012)
18. Khoa, N.L.D., Chawla, S.: Robust outlier detection using commute time and eigenspace embedding. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part II. LNCS, vol. 6119, pp. 422–434. Springer, Heidelberg (2010)
19. Widiputra, H., Pears, R., Kasabov, N.: Multiple time-series prediction through multiple time-series relationships profiling and clustered recurring trends. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 161–172. Springer, Heidelberg (2011)
20. Cheng, H., Tan, P.-N.: Semi-supervised learning with data calibration for long-term time series forecasting. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2008)
21. Meesrikamolkul, W., Niennattrakul, V., Ratanamahatana, C.A.: Shape-based clustering for time series data. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part I. LNCS, vol. 7301, pp. 530–541. Springer, Heidelberg (2012)

Twin Bridge Transfer Learning for Sparse Collaborative Filtering

Jiangfeng Shi[1,2], Mingsheng Long[1,2], Qiang Liu[1],
Guiguang Ding[1], and Jianmin Wang[1]

[1] MOE Key Laboratory for Information System Security;
TNLIST; School of Software
[2] Department of Computer Science & Technology,
Tsinghua University, Beijing, China
{shijiangfengsjf,longmingsheng}@gmail.com,
{liuqiang,dinggg,jimwang}@tsinghua.edu.cn

Abstract. Collaborative filtering (CF) is widely applied in recommender systems. However, the *sparsity* issue is still a crucial bottleneck for most existing CF methods. Although target data are extremely sparse for a newly-built CF system, some dense auxiliary data may already exist in other matured related domains. In this paper, we propose a novel approach, *Twin Bridge Transfer Learning* (TBT), to address the sparse collaborative filtering problem. TBT reduces the sparsity in target data by transferring knowledge from dense auxiliary data through two paths: 1) the *latent factors* of users and items learned from two dense auxiliary domains, and 2) the *similarity graphs* of users and items constructed from the learned latent factors. These two paths act as a *twin bridge* to allow more knowledge transferred across domains to reduce the sparsity of target data. Experiments on two benchmark datasets demonstrate that our TBT approach significantly outperforms state-of-the-art CF methods.

Keywords: Collaborative Filtering, Sparsity, Transfer Learning, Latent Factor, Similarity Graph, Twin Bridge.

1 Introduction

Recommender systems are widely deployed on the Web with the aim to improve the user experience [1]. Well-known e-commerce providers, such as Hulu (video) and NetFlix (movies), are highly relying on their developed recommender systems. These systems attempt to recommend items that are most likely to attract users by predicting the preference (ratings) of the users to the items. In the last decade, *collaborative filtering (CF)* methods [2–4], which utilize users' past ratings to predict users' future tastes, have been most popular due to their generally superior performance [5, 6]. Unfortunately, most of previous CF methods have not fully addressed the sparsity issue, i.e., the target data are extremely sparse.

Recently, there has been an increasing research interest in developed *transfer learning* methods to alleviate the data sparsity problem [7–13]. The key idea behind is to transfer the *common knowledge* from some dense auxiliary data

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 496–507, 2013.

to the sparse target data. The common knowledge can be encoded into some common latent factors, and extracted by the latent factorization models [14]. Good choices for the common latent factors can be the cluster-level codebook [8, 9], or the latent tastes of users/latent features of items [11–13]. The transfer learning methods have achieved promising performance over previous methods.

However, when the target data are *extremely* sparse, existing transfer learning methods may confront the following issues. 1) Since the target data are extremely sparse, the latent factors extracted from the auxiliary data may transfer *negative* information to the target data, i.e., the latent factors are likely to be inconsistent with the target data, which may result in overfitting [15]. We refer to this issue as *negative transfer*. 2) As the target data are extremely sparse, it is expected that more common knowledge should be transferred from the auxiliary data to reduce the sparsity of the target data [16]. We refer to this issue as *insufficient transfer*. Solving these two issues both can reduce the sparsity more effectively.

In this paper, we propose a novel approach, *Twin Bridge Transfer Learning* (TBT), for sparse collaborative filtering. TBT aims to explore a *twin bridge* corresponding to the *latent factors* which encode the latent tastes of users/latent features of items, and the *similarity graphs* which encode the collaboration information between users/items. TBT consists of two sequential steps. 1) It extracts the latent factors from dense auxiliary data and constructs the similarity graphs from these extracted latent factors. 2) It transfers both the latent factors and the similarity graphs to the target data. In this way, we can enhance sufficient transfer by transferring more common knowledge between domains, and alleviate negative transfer by filtering out the negative information introduced in the latent factors. Extensive experiments on two real-world data sets show promising results obtained by TBT, especially when the target data are extremely sparse.

2 Related Work

Prior works using transfer learning for collaborative filtering assume that there exist a set of common latent factors between the auxiliary and target data. Rating Matrix Generative Model (RMGM) [9] and Codebook Transfer (CBT) [8] transfer the cluster-level codebook to the target data. However, since codebook dimension cannot be too large, the codebook cannot transfer enough knowledge when the target data are extremely sparse. To avoid this limitation, Coordinate System Transfer (CST) [12], Transfer by Collective Factorization (TCF) [11], and Transfer by Integrative Factorization (TIF) [13] extract both latent tastes of users and latent features of items, and transfer them to the target data. However, these methods do not consider the negative transfer issue, which is more serious when the target data are extremely sparse. Also, they only utilize a single bridge for knowledge transfer. Different from all these methods, our approach explores a twin bridge for transferring more useful knowledge to the sparse target data.

Neighborhood structure has been widely used in memory-based CF [17] or graph regularized methods [4]. Neighborhood-Based CF Algorithm [17] predicts missing ratings of target users by their nearest neighbors. Graph Regularized Weighted Nonnegative Matrix Factorization (GWNMTF) [4] encodes the neighborhood

Table 1. Notations and their descriptions in this paper

Notation	Description	Notation	Description
τ	dataset index, $\tau \in \{1, 2\}$	$\mathbf{R}$	$m \times n$ rating matrix
m	#target users	$\mathbf{Z}$	$m \times n$ indicator matrix
n	#target items	$\mathbf{U}$	$m \times k$ latent tastes of users
p	#nearest neighbors	$\mathbf{V}$	$n \times k$ latent features of items
k	#latent factors	$\mathbf{B}$	$k \times k$ cluster-level codebook
λ_U, λ_V	regularization param of latent factors	γ_U, γ_V	regularization param of similarity graphs

information into latent factor models, which can preserve the similarity between user tastes or item features. However, these methods depend largely on dense ratings to compute accurate neighborhood structure, which is impossible when the data are extremely sparse. Different from these methods, our approach extracts the latent factors of users/items from some dense auxiliary data, and then constructs the neighborhood structure from these latent factors. In this way, the constructed neighborhood structure is more accurate for sparse CF.

3 Problem Definition

We focus on *sparse collaborative filtering* where the target data are *extremely* sparse. Suppose in the target data, we have m users and n items, and a set of integer ratings ranging from 1 to 5. $\mathbf{R} \in \mathbb{R}^{m \times n}$ is the rating matrix, where R_{ij} is the rating that user i gives to item j. $\mathbf{Z} \in \{0, 1\}$ is the indicator matrix, $Z_{ij} = 1$ if user i has rated item j, and $Z_{ij} = 0$ otherwise. In order to reduce the sparsity in the target data, we assume that there exist some dense auxiliary data, from which we can transfer some common knowledge to the target data. Specifically, we will consider two sets of auxiliary data, whose rating matrices are $\mathbf{R}_1$ and $\mathbf{R}_2$, with indicator matrices $\mathbf{Z}_1$ and $\mathbf{Z}_2$, respectively. $\mathbf{R}_1$ shares the common set of users with $\mathbf{R}$, while $\mathbf{R}_2$ shares the common set of items with $\mathbf{R}$. For clarity, the notations and their descriptions are summarized in Table 1.

Definition 1 (Learning Goal). *The goal of TBT is to construct 1) latent factors* $\mathbf{U}_0$,$\mathbf{V}_0$ *from* $\mathbf{R}_1, \mathbf{R}_2$ *and 2) similarity graphs* $\mathbf{L}_U, \mathbf{L}_V$ *from* $\mathbf{U}_0, \mathbf{V}_0$ *(Figure 1), and use them as the twin bridge to predict missing values in* $\mathbf{R}$ *(Figure 2).*

4 Twin Bridge Transfer Learning for Recommendation

In this section, we present our proposed TBT approach, which is comprised of two sequential steps: 1) twin bridge construction, and 2) twin bridge transfer.

4.1 Twin Bridge Construction

In the first step, we aim to construct the latent factors and similarity graphs of users and items, which will be used as the twin bridge for knowledge transfer.

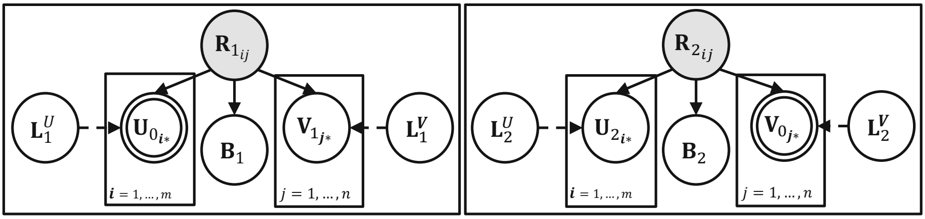


Fig. 1. Step 1 of TBT: twin bridge construction. The goal is to extract the latent factors and construct the similarity graphs from two sets of dense auxiliary data.

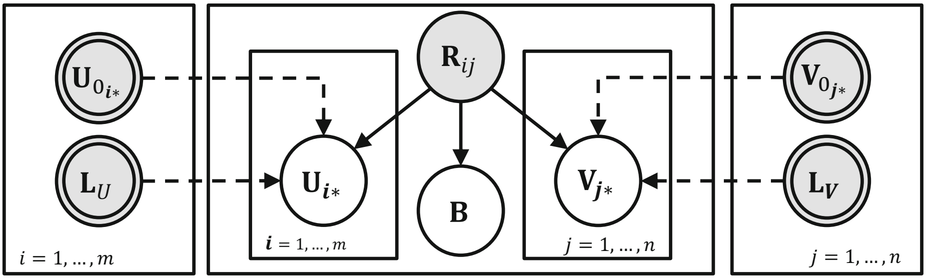


Fig. 2. Step 2 of TBT: twin bridge transfer. '$\longrightarrow$' represents matrix tri-factorization, '$- \rightarrow$' represents twin bridge transfer, ⊚ represents the transferred knowledge structure

Latent Factor Extraction. We adopt the *graph regularized weighted nonnegative matrix tri-factorization* (GWNMTF) method proposed by Gu et al. [4] to extract the latent factors from the two sets of dense auxiliary data $\mathbf{R}_1$ and $\mathbf{R}_2$

$$\mathcal{L} = \min_{\mathbf{U}_\tau,\mathbf{B}_\tau,\mathbf{V}_\tau \geq \mathbf{0}} \left\| \mathbf{Z}_\tau \odot \left(\mathbf{R}_\tau - \mathbf{U}_\tau \mathbf{B}_\tau \mathbf{V}_\tau^T \right) \right\|_F^2 + \gamma_U \mathrm{tr}\left(\mathbf{U}_\tau^T \mathbf{L}_\tau^U \mathbf{U}_\tau \right) + \gamma_V \mathrm{tr}\left(\mathbf{V}_\tau^T \mathbf{L}_\tau^V \mathbf{V}_\tau \right) \tag{1}$$

where γ_U and γ_V are graph regularization parameters, $\mathbf{L}_\tau^U$, $\mathbf{L}_\tau^V$ are the graph Laplacian matrices, $\|\cdot\|$ is the Frobenius norm of matrix. $\mathbf{U}_\tau = [u_{*1}^\tau, ..., u_{*k}^\tau] \in \mathbb{R}^{m \times k}$ are the k *latent tastes of users*, with each u_{i*}^τ representing the latent taste of user i. $\mathbf{V}_\tau = [v_{*1}^\tau, ..., v_{*k}^\tau] \in \mathbb{R}^{n \times k}$ are the k *latent features of items*, with each v_{i*}^τ representing the latent feature of item i. $\mathbf{B}_\tau \in \mathbb{R}^{k \times k}$ is the *cluster-level codebook* representing the association between $\mathbf{U}_\tau$ and $\mathbf{V}_\tau$, $\tau \in \{1, 2\}$.

Since auxiliary data $\mathbf{R}_1$ share the common set of users with the target data $\mathbf{R}$, we can share the latent tastes of users between them, that is, $\mathbf{U}_0 = \mathbf{U}_1$, where $\mathbf{U}_0$ denotes the common latent tastes of users. Similarly, since auxiliary data $\mathbf{R}_2$ share the common set of items with the target data $\mathbf{R}$, we can share the latent features of items between them, that is, $\mathbf{V}_0 = \mathbf{V}_2$, where $\mathbf{V}_0$ denotes the common latent features of items. $\mathbf{U}_0$ and $\mathbf{V}_0$ are the *common latent factors*, which will be used as the first type of bridge for knowledge transfer.

It is worth noting that, the latent factors extracted by GWNMTF can respect the intrinsic neighborhood structure better than those extracted by Sparse Singular Value Decomposition (SVD) [12], thus can fit sparse target data better.

Similarity Graph Construction. Collaboration information between users or items contained in the rating matrix has been widely explored for CF. However, in extremely sparse target data, two users may rate the same item with low probability though they have the same taste. To avoid this data sparsity issue, we construct the similarity graphs from dense auxiliary data since they share common users or items with target data. Because $\mathbf{U}_0$ and $\mathbf{V}_0$ are denser than $\mathbf{R}_1$ and $\mathbf{R}_2$ and can better respect the collaboration information underlying the target data, we propose to construct similarity graphs of users and items from $\mathbf{U}_0$ and $\mathbf{V}_0$, respectively. We define the distance between users/items as follows:

$$\begin{aligned}(\mathbf{W}_U)_{ij} &= \begin{cases} 1, & \text{if } \mathbf{u}_{i*}^0 \in \mathcal{N}_p\left(\mathbf{u}_{j*}^0\right) \text{ or } \mathbf{u}_{j*}^0 \in \mathcal{N}_p\left(\mathbf{u}_{i*}^0\right) \\ 0, & \text{otherwise} \end{cases} \\ (\mathbf{W}_V)_{ij} &= \begin{cases} 1, & \text{if } \mathbf{v}_{i*}^0 \in \mathcal{N}_p\left(\mathbf{v}_{j*}^0\right) \text{ or } \mathbf{v}_{j*}^0 \in \mathcal{N}_p\left(\mathbf{v}_{i*}^0\right) \\ 0, & \text{otherwise} \end{cases}\end{aligned} \tag{2}$$

where $\mathbf{u}_{i*}^0$ is the ith row of $\mathbf{U}_0$ to denote the latent taste of user i, $\mathbf{v}_{i*}^0$ is the ith row of $\mathbf{V}_0$ to denote the latent feature of item i, $\mathcal{N}_p(\mathbf{u}_{i*}^0)$ and $\mathcal{N}_p(\mathbf{v}_{i*}^0)$ are the sets of p-nearest neighbors of $\mathbf{u}_{i*}^0$ and $\mathbf{v}_{i*}^0$, respectively. The positive semi-definite symmetric matrices $\mathbf{W}_U$ and $\mathbf{W}_V$ are the weight matrices for the *similarity graphs*, which will be used as the second type of bridge for knowledge transfer.

4.2 Twin Bridge Transfer

In the second step, we introduce the objectives for latent factor transfer and similarity graph transfer respectively, and then integrate them into a unified optimization problem for twin bridge transfer learning.

Latent Factor Transfer. Since the latent factors of users/items may not be shared as a whole between domains, we propose to transfer latent factors $\mathbf{U}_0, \mathbf{V}_0$ to the target data via two regularization terms $\|\mathbf{U} - \mathbf{U}_0\|_F^2$ and $\|\mathbf{V} - \mathbf{V}_0\|_F^2$, instead of enforcing $\mathbf{U} \equiv \mathbf{U}_0, \mathbf{V} \equiv \mathbf{V}_0$. This leads to the *latent factor transfer*

$$\min_{\mathbf{U},\mathbf{V},\mathbf{B}\geq 0} \mathcal{O}_{UV} = \left\|\mathbf{Z} \odot \left(\mathbf{R} - \mathbf{U}\mathbf{B}\mathbf{V}^T\right)\right\|_F^2 + \lambda_U \left\|\mathbf{U} - \mathbf{U}_0\right\|_F^2 + \lambda_V \left\|\mathbf{V} - \mathbf{V}_0\right\|_F^2 \tag{3}$$

where λ_U, λ_V are regularization parameters indicating our confidence on the latent factors. Since the target data are extremely sparse, the latent factors $\mathbf{U}_0, \mathbf{V}_0$ may transfer some negative information to the target data. Therefore, we introduce similarity graph transfer as a complementary method in the sequel.

Similarity Graph Transfer. Based on the *manifold assumption* [18], similar users/items should have similar latent tastes/features. Therefore, preserving the neighborhood structure underlying the users/items in the target data are reduced to the following graph regularizers encoding the constructed similarity graphs

$$\begin{aligned}\mathcal{G}_U &= \frac{1}{2}\sum\nolimits_{ij} \|\mathbf{u}_{i*} - \mathbf{u}_{j*}\|^2 (\mathbf{W}_U)_{ij} = \operatorname{tr}\left(\mathbf{U}^T\left(\mathbf{D}_U - \mathbf{W}_U\right)\mathbf{U}\right) = \operatorname{tr}\left(\mathbf{U}^T \mathbf{L}_U \mathbf{U}\right) \\ \mathcal{G}_V &= \frac{1}{2}\sum\nolimits_{ij} \|\mathbf{v}_{i*} - \mathbf{v}_{j*}\|^2 (\mathbf{W}_V)_{ij} = \operatorname{tr}\left(\mathbf{V}^T\left(\mathbf{D}_V - \mathbf{W}_V\right)\mathbf{V}\right) = \operatorname{tr}\left(\mathbf{V}^T \mathbf{L}_V \mathbf{V}\right)\end{aligned} \tag{4}$$

where $\mathbf{D}_U = \operatorname{diag}\left(\sum_j (\mathbf{W}_U)_{ij}\right)$, $\mathbf{D}_V = \operatorname{diag}\left(\sum_j (\mathbf{W}_V)_{ij}\right)$. $\mathbf{L}_U = \mathbf{D}_U - \mathbf{W}_U$ and $\mathbf{L}_V = \mathbf{D}_V - \mathbf{W}_V$ are the graph Laplacian matrices for the similarity graphs.

Incorporating the graph regularizers into the objective function of nonnegative matrix tri-factorization, we obtain the following *similarity graph transfer*

$$\min_{\mathbf{U},\mathbf{V},\mathbf{B}\geq 0} \mathcal{O}_{\mathcal{G}_U\mathcal{G}_V} = \left\|\mathbf{Z} \odot \left(\mathbf{R} - \mathbf{U}\mathbf{B}\mathbf{V}^T\right)\right\|_F^2 + \gamma_U \mathcal{G}_U + \gamma_V \mathcal{G}_V \tag{5}$$

where γ_U, γ_V are regularization parameters indicating our confidence on the similarity graphs. With similarity graph transfer, we can essentially filter out the negative information in the latent factors by using the nearest neighbor rule. However, since the target data are extremely sparse, only transferring knowledge from the similarity graphs may be insufficient. Thus we seek an integrated model.

Twin Bridge Transfer. Considering the aforementioned limitations, we integrate the latent factor transfer and similarity graph transfer into a unified *twin bridge transfer* (TBT) method. In TBT, firstly, we extract the latent factors of users/items and construct the similarity graphs from them. Secondly, we transfer latent factors as the first bridge to solve the insufficient transfer issue. As mentioned before, these latent factors may contain negative information for the target data and result in the negative transfer issue. So thirdly, we transfer the similarity graphs as the second bridge to alleviate negative transfer and to transfer more knowledge to the target data. It is worth noting that, each of the twin bridge can enhance the learning of the other bridge during the optimization procedure due to their complementary property. Therefore, we incorporate Equations (3) and (5) into a unified *Twin Bridge Transfer* optimization problem

$$\min_{\mathbf{U},\mathbf{V},\mathbf{B}\geq 0} \mathcal{O} = \left\|\mathbf{Z} \odot \left(\mathbf{R} - \mathbf{U}\mathbf{B}\mathbf{V}^T\right)\right\|_F^2 + \lambda_U \|\mathbf{U} - \mathbf{U}_0\|_F^2 + \lambda_V \|\mathbf{V} - \mathbf{V}_0\|_F^2 + \gamma_U \mathcal{G}_U + \gamma_V \mathcal{G}_V \tag{6}$$

where $\lambda_U, \lambda_V, \gamma_U, \gamma_V$ are regularization parameters. If $\lambda_U, \lambda_V > \gamma_U, \gamma_V$, the transferred latent factors dominate the recommendation results; otherwise, the transferred similarly graphs dominate the recommendation results. With TBT, we can transfer more knowledge from dense auxiliary data to sparse target data to reduce the data sparsity, and substantially avoid the negative transfer issue.

4.3 Learning Algorithm

We solve the optimization problem in Equation (6) using the gradient descent method. The derivative of $\mathcal{O}$ with respect to $\mathbf{U}$ (the other variables are fixed) is

$$\frac{\partial \mathcal{O}}{\partial \mathbf{U}} = -2\mathbf{Z} \odot \mathbf{R}\mathbf{V}\mathbf{B}^T + 2\mathbf{Z} \odot \left(\mathbf{U}\mathbf{B}\mathbf{V}^T\right)\mathbf{V}\mathbf{B}^T + 2\lambda_U(\mathbf{U} - \mathbf{U_0}) + 2\gamma_U \mathbf{L}_U \mathbf{U}$$

Algorithm 1. TBT: Twin Bridge Transfer Learning for Collaborative Filtering

```
Input: Data sets $\mathbf{R}, \mathbf{R}_\tau, \mathbf{Z}, \mathbf{Z}_\tau, \tau \in \{1,2\}$, parameters $k, p, \lambda_U, \lambda_V, \gamma_U, \gamma_V$.
Ouput: Latent factors $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{B}$.
1: Extract latent factor $\mathbf{U}_0, \mathbf{V}_0$ by Equations (1).
2: Construct similarity graphs $\mathcal{G}_U, \mathcal{G}_V$ from $\mathbf{U}_0, \mathbf{V}_0$ by Equations (4).
3: for $it \leftarrow 1$ to $maxIt$ do
4:     update $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{B}$ by Equations (7)~(9), respectively.
5: end for
```

Using Karush-Kuhn-Tucker (KKT) condition [19] for nonnegativity of $\mathbf{U}$, we get

$$\left[-\mathbf{Z} \odot \mathbf{RVB}^T + \mathbf{Z} \odot \left(\mathbf{UBV}^T\right) \mathbf{VB}^T + \lambda_U (\mathbf{U} - \mathbf{U_0}) + \gamma_U \mathbf{L}_U \mathbf{U}\right] \odot \mathbf{U} = 0$$

Since $\mathbf{L}_U$ may take mixed signs, we replace it with $\mathbf{L}_U = \mathbf{L}_U^+ - \mathbf{L}_U^-$, where $\mathbf{L}_U^+ = \frac{1}{2}\left(|\mathbf{L}_U| + \mathbf{L}_U\right)$, $\mathbf{L}_U^- = \frac{1}{2}\left(|\mathbf{L}_U| - \mathbf{L}_U\right)$ are positive-valued matrices. We obtain

$$\mathbf{U} \leftarrow \mathbf{U} \odot \sqrt{\frac{\mathbf{Z} \odot \mathbf{RVB}^T + \lambda_U \mathbf{U_0} + \gamma_U \mathbf{L}_U^- \mathbf{U}}{\mathbf{Z} \odot \left(\mathbf{UBV}^T\right) \mathbf{VB}^T + \lambda_U \mathbf{U} + \gamma_U \mathbf{L}_U^+ \mathbf{U}}} \tag{7}$$

Likewise, we derive $\mathbf{V}$ and $\mathbf{B}$ in similar way and get the following update rules

$$\mathbf{V} \leftarrow \mathbf{V} \odot \sqrt{\frac{(\mathbf{Z} \odot \mathbf{R})^T \mathbf{UB} + \lambda_V \mathbf{V}_0 + \gamma_V \mathbf{L}_V^- \mathbf{V}}{\left(\mathbf{Z} \odot \left(\mathbf{UBV}^T\right)\right)^T \mathbf{UB} + \lambda_V \mathbf{V} + \gamma_V \mathbf{L}_V^+ \mathbf{V}}} \tag{8}$$

$$\mathbf{B} \leftarrow \mathbf{B} \odot \sqrt{\frac{\mathbf{U}^T \left(\mathbf{Z} \odot \mathbf{R}\right) \mathbf{V}}{\mathbf{U}^T \left(\mathbf{Z} \odot \left(\mathbf{UBV}^T\right)\right) \mathbf{V}}} \tag{9}$$

According to [4], updating $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{B}$ sequentially by Equations (7)∼(9) will monotonically decrease the objective function in Equation (6) until convergence.

The learning algorithm for TBT is summarized in Algorithm 1. The time complexity is $O(maxIt \cdot kmn + mn^2 + m^2 n)$, which is the sum of TBT cost plus the time cost for the latent factor extraction and similarity graphs construction.

5 Experiments

In this section, we conduct experiments to compare TBT with state-of-the-art collaborative filtering methods on benchmark data sets with high sparsity level.

5.1 Data Sets

MovieLens10M[1] MovieLens10M consists of 10,000,054 ratings and 95,580 tags given by 71,567 users to 10,681 movies. The preference of the user for a movie is rated from 1 to 5 and 0 indicates that the movie is not rated by any user.

[1] http://www.grouplens.org/node/73/

Table 2. Description of the data sets for evaluation

	Data Sets	MovieLens10M		Epinions	
		Size	Sparsity	Size	Sparsity
$\mathbf{R}$	Target (training)	5000×5000	0.01∼1%	1662×2078	0.01∼1%
	Target (test)		8.22∼8.13%		1.86∼0.87%
$\mathbf{R_1}$ (user side)	Auxiliary		2.31%		0.65%
$\mathbf{R_2}$ (item side)	Auxiliary		9.14%		1.28%

Epinions[2] In `epinions.com`, users can assign products with integer ratings ranging from 1 to 5. The dataset used in our experiments is collected by crawling the `epinions.com` website during March, 2012. It consists of 3,324 users who have rated a total of 4,156 items, and the total number of ratings is 138,133.

The data sets used in the experiments are constructed using the same strategy as [12]. For the MovieLens10M data set, we first randomly sample a $10^4 \times 10^4$ dense rating matrix $\mathbf{X}$ from the MovieLens data. Then we take the submatrices $\mathbf{R} = \mathbf{X_{1\sim5000,1\sim5000}}$ as the target rating matrix, $\mathbf{R_1} = \mathbf{X_{1\sim5000,5001\sim10000}}$ as the user-side auxiliary data and $\mathbf{R_2} = \mathbf{X_{5001\sim10000,1\sim5000}}$ as the item-side auxiliary data. In this way, $\mathbf{R}$ and $\mathbf{R_1}$ share common users but not common items, while $\mathbf{R}$ and $\mathbf{R_2}$ share common items but not common users. We use the same construction strategy for the Epinions data. In all experiments, the target ratings $\mathbf{R}$ are randomly split into a training set and a test set. To investigate the performance of each method on sparse data, we sampled training set randomly with a variety of sparsity levels from 0.01% to 1%, as summarized in Table 2.

5.2 Evaluation Metrics

We adopt two evaluation metrics, *Mean Absolute Error* (MAE) and *Root Mean Square Error* (RMSE), which are widely used for evaluating collaborative filtering algorithms [4, 12]. **MAE** and **RMSE** are defined as

$$MAE = \frac{\sum_{i,j} \left| R_{ij} - \widetilde{R}_{ij} \right|}{|T_E|} \qquad RMSE = \sqrt{\frac{\sum_{i,j} \left(R_{ij} - \widetilde{R}_{ij} \right)^2}{|T_E|}}$$

Where R_{ij} is the rating that user i gives to item j in test set, while $\widetilde{R}_{ij}$ is the predicted value of R_{ij} by CF algorithms, $|T_E|$ is the number of ratings in test set. We run each method 10 repeated trials with randomly generated training set and test set from $\mathbf{R}$, and report the average MAE and RMSE of each method.

5.3 Baseline Methods & Parameter Settings

We compare TBT with *Probabilistic Matrix Factorization* (PMF) [20], *Weight Nonnegative Matrix Factorization* (WNMF) [21], *Graph Regularized Weighted Nonnegative Matrix Tri-Factorization* (GWNMTF) [4], *Coordinate System Transfer* (CST) [12], our proposed $\mathbf{TBT}_{UV}$ (defined in Equation (3)) and $\mathbf{TBT}_{G}$

[2] `http://www.epinions.com/`

Table 3. MAE & RMSE comparison on MovieLens10M

	Sparsity	Methods Without Transfer			Method With Transfer			
		PMF	WNMF	GWNMTF	CST	TBT_G	TBT_{UV}	TBT
MAE	0.01%	1.171	2.7966	0.8321	0.9814	0.7535	0.7272	**0.6978±0.0006**
	0.05%	1.0256	1.3835	0.7663	0.8409	0.6945	0.7492	**0.6903±0.0007**
	0.10%	0.90232	0.9506	0.7407	0.7629	0.6853	0.7275	**0.6840±0.0005**
	0.50%	0.73368	0.697	0.6862	0.6799	0.6707	0.6785	**0.6690±0.0001**
	1%	0.70398	0.6743	0.6728	0.6677	0.6649	0.6677	**0.6640±0.0000**
	Average	0.9073	1.3004	0.7396	0.7866	0.6938	0.7100	**0.6810±0.0004**
RMSE	0.01%	1.3717	3.1434	1.0299	1.3926	0.9445	0.9397	**0.8995±0.0011**
	0.05%	1.2333	1.8779	0.97	1.1225	0.8896	0.9594	**0.8893±0.0012**
	0.10%	1.1097	1.3199	0.944	0.997	0.8812	0.935	**0.8817±0.0008**
	0.50%	0.93171	0.9076	0.8802	0.8755	0.8628	0.8734	**0.8608±0.0001**
	1%	0.90112	0.8718	0.8645	0.8605	0.8561	0.8601	**0.8550±0.0000**
	Average	1.1095	1.6241	0.9377	1.0496	0.8868	0.9135	**0.8772±0.0007**

Table 4. MAE & RMSE comparison on Epinions

	Sparsity	Methods Without Transfer			Method With Transfer			
		PMF	WNMF	GWNMTF	CST	TBT_G	TBT_{UV}	TBT
MAE	0.01%	1.6758	3.7617	0.9875	0.9781	0.9066	0.8806	**0.8737±0.0046**
	0.05%	1.6341	3.0304	0.9792	0.9629	0.8838	0.9122	**0.8740±0.0043**
	0.10%	1.5555	2.3353	0.9682	0.9506	0.8773	0.9258	**0.8682±0.0018**
	0.50%	1.1083	1.0465	0.8902	0.8837	0.8543	0.8846	**0.8441±0.0016**
	1%	0.9824	0.8769	0.8539	0.8467	0.8379	0.8478	**0.8294±0.0008**
	Average	1.3912	2.2102	0.9358	0.9244	0.8720	0.8902	**0.8579± 0.0026**
RMSE	0.01%	1.8474	3.999	1.2665	1.3063	1.1427	1.1366	**1.1164±0.0022**
	0.05%	1.8136	3.4811	1.2508	1.2695	1.118	1.1755	**1.1116±0.0022**
	0.10%	1.7464	2.9094	1.233	1.2403	1.1173	1.1919	**1.1105±0.0021**
	0.50%	1.3641	1.4222	1.1289	1.1422	1.0885	1.1369	**1.0790±0.0007**
	1%	1.218	1.1457	1.0932	1.0981	1.0766	1.095	**1.0688±0.0005**
	Average	1.5979	2.5915	1.1945	1.2113	1.1086	1.1472	**1.0973±0.0015**

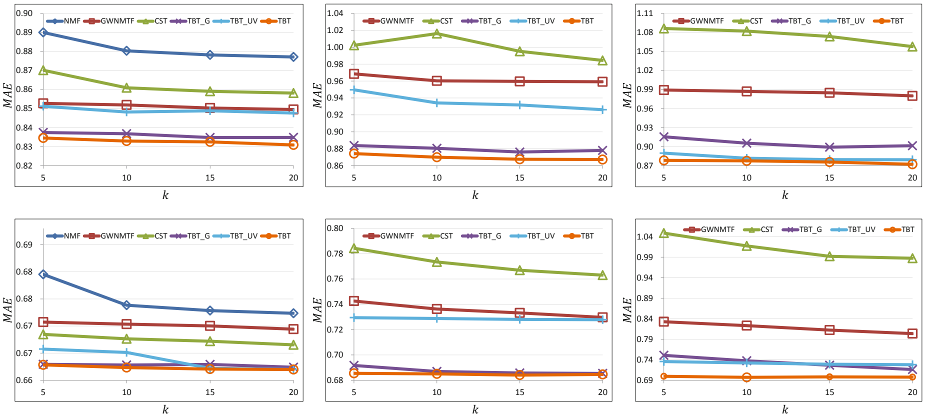
(defined in Equation (5)). Different numbers of latent factors $\{5, 10, 15, 20\}$ and different values of regularization parameters $\{0.01, 0.1, 1, 10, 100\}$ of each method are tried, with the best ones selected for comparison. Since all the algorithms are iterative ones, we run each of them 1000 iterations and report the best results.

5.4 Experimental Results

The experimental results on MovieLens10M and Epinions are shown in Table 3 and Table 4 respectively. From the tables, we can make several observations.

- TBT performs better than all baseline methods under all sparsity levels. It confirms that simultaneously transferring latent factors and similarity graphs as twin bridge can successfully reduce the sparsity in the target data.
- It is interesting to see that, for our method, the sparser the target data are, the more improvement can be achieved. This can be explained by two reasons.

First, when the rating data are sparse, matrix factorization methods that rely on neighborhood information is ineffective due to: 1) it is impossible to compute the similarity between users/items when data are extremely sparse, and 2) even if some similarity information is calculated, this information may be too limited to recommend correct items. So its prediction performance gets worse when rating



(a) 1% (b) 0.1% (c) 0.01%

Fig. 3. Impact of #latent factors on Epinions (above) and MovieLens10M (bottom)

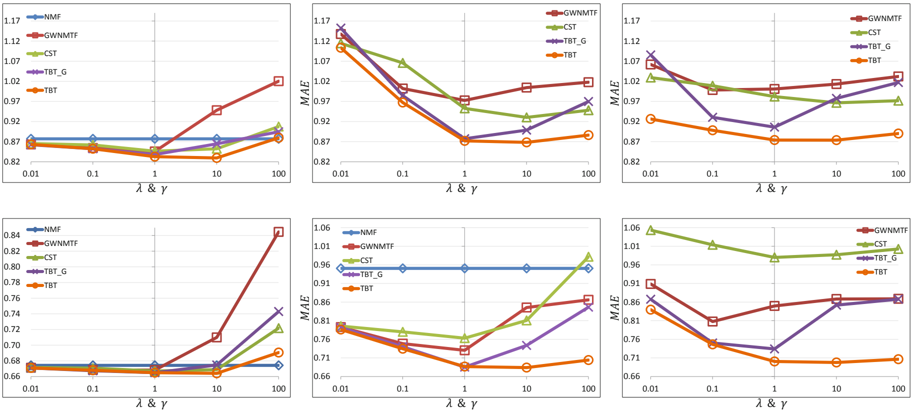
data are coming more spare. To handle this problem, we construct a twin bridge, which can transfer more knowledge from dense auxiliary data to the target data.

Secondly, towards the negative transfer issue, CST encounters the problem that the transferred latent factors extracted from the auxiliary data may contain some negative information for the target data. Our twin bridge transfer mechanism alleviates this problem by using similarity graphs as regularization when training the recommendation model. Notably, the graph regularization can filter out the negative information introduced by the transferred latent factors. With this advantage, TBT is made more robust to the extremely sparse data.

- Although TBT is a combination of our proposed TBT_{UV} and TBT_G, it outperforms both of them. This proves that the twin bridge transfer can reinforce the learning of both bridges. Each bridge emphasizes different properties of the data and enriches the matrix tri-factorization with complementary information.
- Unfortunately, transfer learning method CST has not consistently outperformed non-transfer learning methods GWNMTF. CST takes advantage on moderately sparse data (0.1%∼1%) by knowledge transfer. However, under extremely sparsity levels (0.01%∼0.1%), the latent factors transferred by CST may contain some negative information for the target data. In this case, GWNMTF performs better than CST, since it is not affected by the negative transfer issue.

5.5 Parameter Sensitivity

We investigate the parameter sensitivity by varying the values of k, λ and γ under different sparsity levels. As Figure 3 shows, under a given sparsity level, the more latent factors used, the better the performance is. Typically, we set $k = 20$ as used by baseline methods. For λ and γ, We set $\lambda = \gamma$ in TBT for easy comparison with baselines, and report average MAE in Figure 4. Each column corresponds



(a) 1% (b) 0.1% (c) 0.01%

Fig. 4. Regularization param impact on Epinions (above)/MovieLens10M (bottom)

to a specific sparsity level of training ratings while each row corresponds to a different dataset. We observe that, 1) TBT method performs much more stably under different parameter values and sparsity levels than the baseline methods, and 2) neither non-transfer method GWNMTF or single bridge transfer methods CST/TBT$_G$ can perform stably under different parameter values, sparsity levels, or data sets. Luckily, our TBT benefits from its twin bridge mechanism and performs much more robustly under all of these experimentation settings.

6 Conclusion

In this paper, we proposed a novel approach, Twin Bridge Transfer Learning (TBT), to reduce the sparsity in the target data. TBT consists of two sequential steps: 1) TBT extracts latent factors of users and items from dense auxiliary data and constructs similarity graphs from them; 2) TBT utilizes the latent factors and similarity graphs as twin bridge to transfer knowledge from dense auxiliary data to sparse target data. By using the twin bridge, TBT can enhance sufficient transfer by transferring more knowledge, while alleviate negative transfer by regularizing the learning model with the similarity graphs, which can naturally filter out the negative information contained in the latent factors. Experiments on real-world data sets validate the effectiveness of the proposed TBT approach.

Acknowledgement. The work is supported by the National HGJ Key Project (No. 2010ZX01042-002-002-01), the 973 Program of China (No. 2009CB320706), the National Natural Science Foundation of China (No. 61050010, No. 61073005, No. 60972096).

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. TKDE 17 (2005)
2. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: SIGIR (2009)
3. Song, Y., Zhuang, Z., Li, H., Zhao, Q., Li, J., Lee, W.C., Giles, C.L.: Real-time automatic tag recommendation. In: SIGIR (2008)
4. Gu, Q., Zhou, J., Ding, C.: Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In: SDM (2010)
5. Basilico, J., Hofmann, T.: Unifying collaborative and content-based filtering. In: ICML (2004)
6. Melville, P., Mooney, R., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: AAAI (2002)
7. Cao, B., Liu, N., Yang, Q.: Transfer learning for collective link prediction in multiple heterogenous domains. In: ICML (2010)
8. Li, B., Yang, Q., Xue, X.: Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In: AAAI (2009)
9. Li, B., Yang, Q., Xue, X.: Transfer learning for collaborative filtering via a rating-matrix generative model. In: ICML (2009)
10. Pan, S.J., Yang, Q.: A survey on transfer learning. TKDE 22 (2010)
11. Pan, W., Liu, N.N., Xiang, E.W., Yang, Q.: Transfer learning to predict missing ratings via heterogeneous user feedbacks. IJCAI (2011)
12. Pan, W., Xiang, E., Liu, N., Yang, Q.: Transfer learning in collaborative filtering for sparsity reduction. In: AAAI (2010)
13. Pan, W., Xiang, E., Yang, Q.: Transfer learning in collaborative filtering with uncertain ratings. In: AAAI (2012)
14. Bell, R.M., Koren, Y.: Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In: ICDM (2007)
15. Long, M., Wang, J., Ding, G., Shen, D., Yang, Q.: Transfer learning with graph co-regularization. In: AAAI (2012)
16. Long, M., Wang, J., Ding, G., Cheng, W., Zhang, X., Wang, W.: Dual transfer learning. In: SDM (2012)
17. Herlocker, J., Konstan, J.A., Riedl, J.: An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Information Retrieval 5 (2002)
18. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: NIPS, vol. 14 (2001)
19. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge University Press (2004)
20. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: NIPS (2008)
21. Zhang, S., Wang, W., Ford, J., Makedon, F.: Learning from incomplete ratings using non-negative matrix factorization. SIAM (2006)

Dimensionality Reduction with Dimension Selection

Yi Guo[1,⋆], Junbin Gao[2], and Feng Li[3]

[1] CSIRO Mathematics, Informatics and Statistics,
North Ryde, NSW 1670, Australia
yi.guo@csiro.au

[2] School of Computing and Mathematics,
Charles Sturt University, Bathurst, NSW 2795, Australia
jbgao@csu.edu.au

[3] Earth observation Technology Application Department,
Academy of Opto-Electronics, CAS, China
lifeng@aoe.ac.cn

Abstract. We propose a novel method called sparse dimensionality reduction (SDR) in this paper. It performs dimension selection while reducing data dimensionality. Different from traditional dimensionality reduction methods, this method does not require dimensionality estimation. The number of final dimensions is the outcome of the sparse component of this method. In a nutshell, the idea is to transform input data to a suitable space where redundant dimensions are compressible. The structure of this method is very flexible which accommodates a series of variants along this line. In this paper, the data transformation is carried out by Laplacian eigenmaps and the dimension selection is fulfilled by $l2/l1$ norm. A Nesterov algorithm is proposed to solve the approximated SDR objective function. Experiments have been conducted on images from video sequences and protein structure data. It is evident that the SDR algorithm has subspace learning capability and may be applied to computer vision applications potentially.

1 Introduction

Recent years have been witnessing large increase in studies on dimensionality reduction (DR). The purpose of DR is mainly to find the corresponding counterparts (or embeddings) of the input data of dimension M in a much lower dimensional space (so-called latent space, usually Euclidean) of dimension n and $n \ll M$ without incurring significant information loss. A number of new algorithms which are specially designed for nonlinear dimensionality reduction (NLDR) have been proposed such as Lapacian Eigenmaps (LE) [1], Isometric mapping (Isomap) [2], Local Tangent Space Alignment (LTSA) [3], Gaussian Process Latent Variable Model (GPLVM) [4] etc. to replace the simple linear

[⋆] The author to whom all the correspondences should be addressed.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 508–519, 2013.

methods such as Principal Component Analysis (PCA) [5], Linear Discriminant Analysis (LDA) [6] in which the assumption of linearity is essential.

Among these NLDR methods, it is worth mentioning those which can handle highly structured or so-called *non-vectorial* data (for example proteins, which are not readily converted to vectors) directly without vectorization. This category includes the "kernelized" linear methods. Typical methods are Kernel PCA (KPCA) [7], Generalized Discriminant Analysis (GDA or KLDA) [8] and so on. The application of the kernel function not only introduces certain nonlinearity implied by the feature mapping associated with the kernel which enables the algorithms to capture the nonlinear features, but also embraces much broader types of data including the aforementioned non-vectorial data. Meanwhile, kernels can also be regarded as a kind of similarity measurements which can be used in measurement matching algorithms like Kernel Laplacian Eigenmaps (KLE) [9] and Twin Kernel Embedding (TKE) [10]. Because these methods can directly use the structured data through kernel functions and hence bypass the vectorization procedure which might be a source of bias, they are widely used in complex input patterns like proteins, fingerprints, etc.

Although DR is proven to work well for some machine learning tasks such as classification [11], an inevitable question yet to be answered in applying DR is how to estimate the dimensionality which is the so-called intrinsic dimension estimation. Various methods have been presented in machine learning literature [12]. Nevertheless, a very simple way for dimension estimation is to reduce the dimension one at a time in a suitable space until significant information loss occurs and then stop. This procedure does the reduction and dimension estimation at the same time. There are two very important ingredients in this method: the proper space of the transformed data and the stop criterion of dimension reducing. These two should be combined seamlessly.

Interestingly, we can look at this problem the other way around. Instead of dropping dimensions, we can select dimensions in a suitable space. To do this properly, we refer to the variable selection framework. The nature of the variable selection problem is combinatorial optimization and hence NP hard. Nevertheless, there is a recent trend of using sparse approximation to solve this problem which has attracted attention in statistics and machine learning society. The earliest work is from [13] called the LASSO. By adding an $l1$ norm constraint on the coefficients of a linear regression model, the original combinatorial optimization problem was converted to a convex optimization problem which is much easier to solve. The optimization is normally cast as a regularization problem solved by lots of efficient algorithms such as coordinate descent [14], iterative shrinkage-thresholding [15] and etc. Several sparsity encouraging models have been proposed afterwards with various properties. For example, the group LASSO [16] has the group selection property by applying the $l2/l1$ norm to the group coefficients.

The above discuss leads to a novel method for dimensionality reduction which selects dimensions in transformed space and the selection is carried out by sparse methods. What follows is how to choose the transformed space? The research in this decade on dimensionality reduction provides many solutions to this question. We choose Laplacian eigenmaps in this paper because it approximates the embedded manifold through a mapping function with proximity preserving property. However, It is very convenient to extend this choice to other methods such as LLE, TKE and etc. depending on the application at hand. Another variable in this idea is the sparse method normally by using sparsity encouraging norms. Since different sparsity encouraging norms come with different features, they provide us flexibility for various machine learning tasks, for example subspace learning, feature extraction etc. We use $l2/l1$ norm here for its group selection capability which is suitable for our dimension selection purpose. In Section 3, we will briefly discuss some of its variants and show how this idea could be used for subspace learning. Since this method involves sparse models for dimensionality reduction, we call it sparse dimensionality reduction or SDR for short.

The most related work is [17] where a rank prior as a surrogate of data dimensionality was imposed on GPLVM. In [17] the transformation of data was carried out by GPLVM, which converted the data to a space where a low rank representation (measured by rank prior) was possible. The stopping criterion was the stationary point of the optimization process. The work in [18] is also similar to ours but it was built on a sparse linear regression model and a dictionary in high dimensional space is required.

The paper is organized as follows. Section 2 briefly reviews Laplacian eigenmaps. Section 3 explains the proposed SDR method in detail followed by a section for the optimization procedure. In Section 5, we present several experimental results using SDR on visualization to show its effectiveness. We conclude this paper in Section 6.

2 Laplacian Eigenmaps

In the following discussion, let $\mathbf{y}_i \in \mathbb{R}^M$ be the i-th data sample on a manifold embedded in M dimensional space. Laplacian eigenmaps (LE) [1] is a typical nonlinear method that belongs to the family of graph-based DR methods. It attempts to preserve proximity relations in the input data which is expressed by a weight matrix based on adjacency graph (or called neighborhood graph). This adjacency graph G is constructed by referring to ε neighborhood or n nearest neighbor criterion. An edge will connect $\mathbf{y}_j$ and $\mathbf{y}_i$ if $\|\mathbf{y}_i - \mathbf{y}_j\|^2 < \varepsilon$ (ε neighborhood), or if $\mathbf{y}_j$ is among n nearest neighbors of $\mathbf{y}_i$ and vice versa (n nearest neighbor is more commonly used). The adjacency graph plays an important role in dimensionality reduction which leads to a series of graph based methods [19,20].

After the construction of the adjacency graph, the weights on the edges are evaluated that stand for the proximity relations among the input data. There are also two variations of setting the weights in LE: (a) exponential decay function:

$$\text{the weight } w_{ij} = \begin{cases} e^{-\sigma\|\mathbf{y}_i - \mathbf{y}_j\|^2}, & \text{if } \mathbf{y}_i \text{ is connected with } \mathbf{y}_j; \\ 0, & \text{otherwise.} \end{cases}$$

(b) binary ($\sigma = 0$): $w_{ij} = 1$ if $\mathbf{y}_i$ and $\mathbf{y}_j$ are connected, and $w_{ij} = 0$ otherwise. This simplification avoids the need to choose σ.

The weight matrix $\mathbf{W}$ ($\{w_{ij}\}$) containing the proximity information is then used to construct the Laplacian of the graph $\mathbf{L} = \mathbf{D} - \mathbf{W}$ where $\mathbf{D}$ is a diagonal matrix and its ii-th element $[\mathbf{D}]_{ii} = \sum_{j=1}^{N} w_{ij}$. The reason for Laplican is that the optimal locality preserving maps of the data on manifold onto $\mathbb{R}^K$ (K is at most $M - 1$ because of the removal of arbitrary translation) can be approximated by obtaining the smallest K eigen vectors (excluding the eigen vector corresponding to eigenvalue 0) from the following generalized eigendecomposition

$$\begin{aligned} &\text{min. } \mathrm{tr}[\mathbf{X}^T\mathbf{L}\mathbf{X}] \\ &\text{s.t. } \mathbf{X}^T\mathbf{D}\mathbf{X} = \mathbf{I} \end{aligned} \tag{1}$$

where $\mathbf{X}$ of size $N \times K$ is the matrix of maps of $\mathbf{y}_i$'s in $\mathbb{R}^K$ and $\mathbf{I}$ is the identity matrix with compatible dimensions. For dimensionality reduction purpose, K is selected (somewhat arbitrarily) much less than M or by dimension estimation.

LE is a local method since it is based on the adjacency graph. Several variants have been derived from original LE such as the LPP (Locality Preserving Projection) [21] and the KLE (Kernel LE) [9]. LPP introduces a linear constraint between input data and embeddings, i.e $\mathbf{x}_i = \mathbf{A}\mathbf{y}_i$ while KLE replaces the weight matrix by a sparse kernel Gram matrix.

3 Data Reduction with Dimension Selection

We interpret the dimensionality reduction as a process of space transformation under the framework of Laplacian eigenmaps. The assumption is that the data lie on or near to a dimensional manifold embedded in M dimensional ambient space. The Laplacian eigenmaps is to unfold the manifold in a K dimensional subspace. As we do not know the intrinsic dimension of the manifold, We have to resort to other methods, which may be heterogeneous totally to LE, to estimate the dimensionality in advance.

However, if the unfolded manifold is indeed low dimensional, it should be "compressible", i.e. we can drop redundant dimensions while maintain the structure of the unfolded manifold in this transformed space. As discussed in Section 1, the force of compressing can be realized by introducing sparsity encouraging norms. The suitable one is the $l2/l1$ norm [16] defined as

$$||\mathbf{X}||_{2/1} = \sum_{i=1}^{K} ||\mathbf{X}^i||_2$$

where $\mathbf{X}^i$ is the i-th column of $\mathbf{X}$ and $||\cdot||_2$ is $l2$ norm. When $\mathbf{X}$ is a row vector, $l2/l1$ norm degenerates to normal $l1$ norm. An outstanding feature of $l2/l1$ norm is its group selection capability meaning that the elements in some $l2$ norms (groups) will be compressed towards zero altogether if the norm is minimized, for example in the group variable selection in [16,22]. For our purpose of dimension selection, we use it to vanish the whole dimension, which is regarded as group in terms of $l2/l1$ norm, if it is dispensable.

Our sparse dimensionality reduction (SDR) takes the following form

$$\begin{aligned} \min.\ & \mathrm{tr}[\mathbf{X}^T\mathbf{L}\mathbf{X}] \\ \text{s.t.}\ & \mathbf{X}^T\mathbf{D}\mathbf{X} = \mathbf{I} \\ & ||\mathbf{X}||_{2/1} \leq t \end{aligned} \tag{2}$$

where $t \in \mathbb{R}^+$ is the parameter to control the "dimension sparsity". Following the previous discussion, the rationale is quite clear, that is we unfold the manifold in such a way that some of the dimensions can be reduced or in other words the most important dimensions can be selected. The algorithm starts with a generalized eigen decomposition, i.e. (2) without $l2/l1$ norm constraint. Once the selection completes, we retain the selected dimensions only from the initialization only. Details about implementation will be presented in Section 4.

Interestingly, if we substitute the $l2/l1$ norm by the nuclear norm in (2) as follows

$$\begin{aligned} \min.\ & \mathrm{tr}[\mathbf{X}^T\mathbf{L}\mathbf{X}] \\ \text{s.t.}\ & \mathbf{X}^T\mathbf{D}\mathbf{X} = \mathbf{I} \\ & ||\mathbf{X}||_* \leq t, \end{aligned} \tag{3}$$

the idea would be unfolding the manifold such that the rank of the of the embeddings, i.e. $\mathbf{X}$, is restricted. This is equivalent to finding a subspace in $\mathbb{R}^K$ that reveals the lower dimensional structure of the manifold. It suggests that potentially SDR can be used as a tool for subspace learning [23]. More generally, we can use other sparsity encouraging norms denoted as lq, which certainly brings different interpretation to this method.

Furthermore, we can extend this idea to LPP which is in line with LE. LPP has another layer on top of LE which is a linear mapping from input data to embeddings. In this case, we have the following objective in variable $\mathbf{A}$

$$\begin{aligned} \min.\ & \mathrm{tr}[\mathbf{A}\mathbf{Y}^T\mathbf{L}\mathbf{Y}\mathbf{A}^T] \\ \text{s.t.}\ & \mathbf{A}\mathbf{Y}^T\mathbf{D}\mathbf{Y}\mathbf{A}^T = \mathbf{I} \\ & ||\mathbf{A}||_q \leq t, \end{aligned} \tag{4}$$

where $\mathbf{A} \in \mathbb{R}^{K\times M}$ is the linear transformation matrix. q varies depending on the purpose of the application.

The flexibility of SDR enables it to embrace a lot of existing DR methods as well as sparse methods. In this paper, we focus only on (2) for its simplicity and direct understanding of dimension selection.

4 SDR Implementation

We proceed to obtaining the solution for SDR in (2). Direct optimization of the objective of SDR in (2) is very difficult because the nonsmooth $l2/l1$ norm constraint and quadratic equality constraint, which makes it a quadratic programming with quadratic constraint (QPQC) problem with additional norm restriction. The quadratic equality constraint effectively excludes some popular alternating optimization schemes such as ADMM [24] since the augmented Lagrange term from the equality has no close form solution. It also throws some trouble to second order optimizer such as Newton-Raphson method because the Hessian is a tensor. To maintain the convexity of the original problem and also to make it easier to solve, we relax the equality constraint by converting it to a regularization term in the objective so that the first order algorithms are applicable. As a result, the original SDR problem has been converted to the following form

$$\begin{aligned} &\min.\ \frac{1}{2}\mathrm{tr}[\mathbf{X}^T\mathbf{L}\mathbf{X}] + \frac{\lambda}{4}||\mathbf{X}^T\mathbf{D}\mathbf{X} - \mathbf{I}||_F^2 + z\mathbf{e}^T\mathbf{t} \\ &\text{s.t. } ||\mathbf{X}||_{2/1} \preceq \mathbf{t} \end{aligned} \tag{5}$$

where $||\cdot||_F$ denotes the F norm of a matrix and $\mathbf{e}$ is an all one column vector. Note that we add another regularizer to the $l2/l1$ norm of $\mathbf{X}$, $z\mathbf{e}^T\mathbf{t}$ ($\mathbf{t} \in \mathbb{R}^K$), and the i-th element of $\mathbf{t}$ is responsible for $||\mathbf{X}^i||_2$. z has the same function as t in (2) controlling the dimension sparsity. The introduction of this additional regularization does not bring extra complexity; however, it enables us to use the efficient Euclidean projection explained in [25] with which (5) can be solved using Nesterov algorithm [26] easily. We will not go into too much detail of the algorithm but provide the necessary elements to make it work. Write

$$f(\mathbf{X}, \mathbf{t}) = \mathrm{tr}[\mathbf{X}^T\mathbf{L}\mathbf{X}] + \lambda||\mathbf{X}^T\mathbf{D}\mathbf{X} - \mathbf{I}||_F^2 + z\mathbf{e}^T\mathbf{t}$$

supposing λ and z are given. We have the derivatives

$$\frac{\partial f(\mathbf{X}, \mathbf{t})}{\partial \mathbf{X}} = \mathbf{L}\mathbf{X} + \lambda\mathbf{D}\mathbf{X}(\mathbf{X}^T\mathbf{D}\mathbf{X} - \mathbf{I}) \tag{6}$$

and $\frac{\partial f(\mathbf{X},\mathbf{t})}{\partial \mathbf{t}} = z\mathbf{e}$ for this first order algorithm. The detailed optimization procedures are shown in Table 1.

In Nesterov algorithm, there are two sequences of variables, the target in the optimization problem, $\mathbf{X}$ and $\mathbf{t}$ in this case, and assistant variables, $\mathbf{S}$ and $\mathbf{h}$ shown in Table 1 corresponding to $\mathbf{X}$ and $\mathbf{t}$ respectively. The assistant variable is a linear combination of current and previous estimation of the target, e.g. $\mathbf{S}_i = \mathbf{X}_i + \alpha_i(\mathbf{X}_i - \mathbf{X}_{i-1})$. The tentative new estimation is given by the gradient projection in Line 6, where $\mathcal{P}_\mathcal{C}(x)$ is the Euclidean projection of x onto the feasible convex set $\mathcal{C}$. In our case, $\mathcal{C}$ is the set of values satisfying $||\mathbf{X}||_{2/1} \preceq \mathbf{t}$. The Euclidean projection of the given pair $\mathbf{U}$ and $\boldsymbol{v}$ is the solution to the following

$$\begin{aligned} &\min_{\mathbf{X},\mathbf{t}}\ \frac{1}{2}||\mathbf{X} - \mathbf{U}||_2^2 + \frac{1}{2}||\mathbf{t} - \boldsymbol{v}||_2^2 \\ &\text{s.t. } ||\mathbf{X}||_{2/1} \preceq \mathbf{t}. \end{aligned} \tag{7}$$

Table 1. SDR algorithm implementation

Optimize SDR via Nesterov algorithm

```
Input: $\mathbf{L}$, $\mathbf{D}$, $\lambda$, $z$, $\mathbf{X}_0$, $\mathbf{t}_0$
Output: optimal $\mathbf{X}$ and $\mathbf{t}$
1. Initialization: $\mathbf{X}_1 = \mathbf{X}_0$, $\mathbf{t}_1 = \mathbf{t}_0$, $l_{-1} = l_0 = 1$, $\gamma_0 = 1$.
2. for $i = 1$ to ...
3.     $\alpha_i = \frac{l_{i-2}-1}{l_{i-1}}$, $\mathbf{S}_i = \mathbf{X}_i + \alpha_i(\mathbf{X}_i - \mathbf{X}_{i-1})$, $\boldsymbol{h}_i = \mathbf{t}_i + \alpha_i(\mathbf{t}_i - \mathbf{t}_{i-1})$
4.     for $k = 1$ to ...
5.         $\gamma = 2^{k-1}\gamma_{i-1}$
6.         $[\mathbf{X}_{i+1}, \mathbf{t}_{i+1}] = \mathcal{P}_{\mathcal{C}}([\mathbf{S}_i, \mathbf{t}_i] - \frac{f'([\mathbf{S}_i, \mathbf{t}_i])}{\gamma_i})$
7.         if $f([\mathbf{S}_{i+1}, \mathbf{t}_{i+1}]) \leq f_{\gamma,[\mathbf{S}_i,\mathbf{t}_i]}([\mathbf{S}_{i+1}, \mathbf{t}_{i+1}])$ then
8.             $\gamma_i = \gamma$, break
9.         end if
10.    end for
11.    $l_i = (1 + \sqrt{1 + 4l_{i-1}^2})/2$
12.    if convergent then stop and output $\mathbf{X}_i$ and $\mathbf{t}_i$ as the solution.
13. end for
```

$f_{\gamma,\mathbf{x}}(\mathbf{y})$ is the the linear approximation of the objective function $f(\mathbf{y})$ at the point $\mathbf{x}$ regularized by the proximality

$$f_{\gamma,\mathbf{x}}(\mathbf{y}) = f(\mathbf{x}) + f'(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \frac{\gamma}{2}||\mathbf{y} - \mathbf{x}||_2^2.$$

The algorithm in Table 1 has incorporated the Nemirovski line search in Line 4 to 10 for better efficiency. The initialization of $\mathbf{X}$ can be obtained from the solution of the generalized eigendecomposition $\mathbf{LX} = \mathbf{DX}\boldsymbol{\Lambda}$ where $\boldsymbol{\Lambda}$ is the diagonal matrix of the eigenvalues. Note that the last eigenvalue is 0 and its corresponding eigenvector should be removed to obtain translation invariant as in LE. The initial $\mathbf{t}$ can simply be the $l2$ norm of each column of $\mathbf{X}$.

The convergence is guaranteed by Nesterove algorithm. The computational complexity is mainly from matrix multiplications in the evaluation of the gradient in (6). The Euclidean projection in (7) is linear time. So the dominant complexity is $\mathcal{O}(K^3N^4)$ since $\mathbf{D}$ in (6) is a diagonal matrix. It may look very high. But as it iterates, many columns of $\mathbf{X}$ become zero, which effectively brings down K. Our experience with computation time is that it completes in several minutes on uptodate personal computer for $N = 2000$, $K = 600$.

5 Experimental Results

We applied the SDR to several data sets: COIL data set, Frey faces and SCOP protein structure data where LE has difficulties. They are widely used for machine learning and image processing tests. We mainly reduced the dimensionality to 2 so that we can plot the embeddings in 2D plane for interpretation. z is selected by bisection so that only required dimensions were selected. To construct LE initialization for the SDR algorithm and LE itself, we set the number of nearest neighbors to be 5 and used simple binary weight for images data. We used KLE with Mammoth kernel for protein data where the number of nearest neighbors was 8. All these parameters were frequently used or reported to be somewhat optimal. For other methods in protein experiment, we also set their parameters to their reported optimal if any. In SDR optimization procedure, we set the update tolerance to be 1e-10 and maximum number of iterations to be 100, whichever reaches first stops the algorithm. It turned out that these settings worked well on the data sets we have tested in this section.

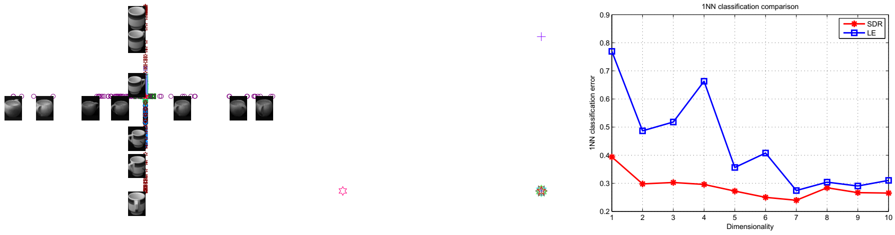
5.1 COIL 3D Images

We demonstrate SDR's dimension selection capability against LE in this experiment. We took the first 20 objects from Columbia Object Image Library (http://www1. cs.columbia.edu/CAVE/software/softlib/coil-100.php). Each object has 72 greyscale images of size 128 × 128 taken from video frames. Since all the images have been perfectly aligned and noise free, traditional methods like PCA can achieve good embedding. However, we focus on dimension selection capability of SDR here. In regard to 2D display, we expected that the objects to line up in the 2D plane somehow with some overlap. As we can see from Fig. 1 (b) where shapes stand for objects, LE's result is 3 isolated islands with heavy overlapping. However the 2D space selected by SDR reveals two cups classes clearly with overlap in the middle with other objets, which is closer to our expectation. This suggests SDR's subspace learning capability, which is further confirmed in the following experiment.

We further extended the target dimension from 1 to 10 and used the 1 nearest neighbor (1NN) classification errors rate as in [4] to compare the results quantitatively. The smaller the error rate, the better the method. The 1NN classification error rates are plotted in Figure 1 (c). It turned out that dimensions selected by SDR are consistently better although they are not optimized for classification task.

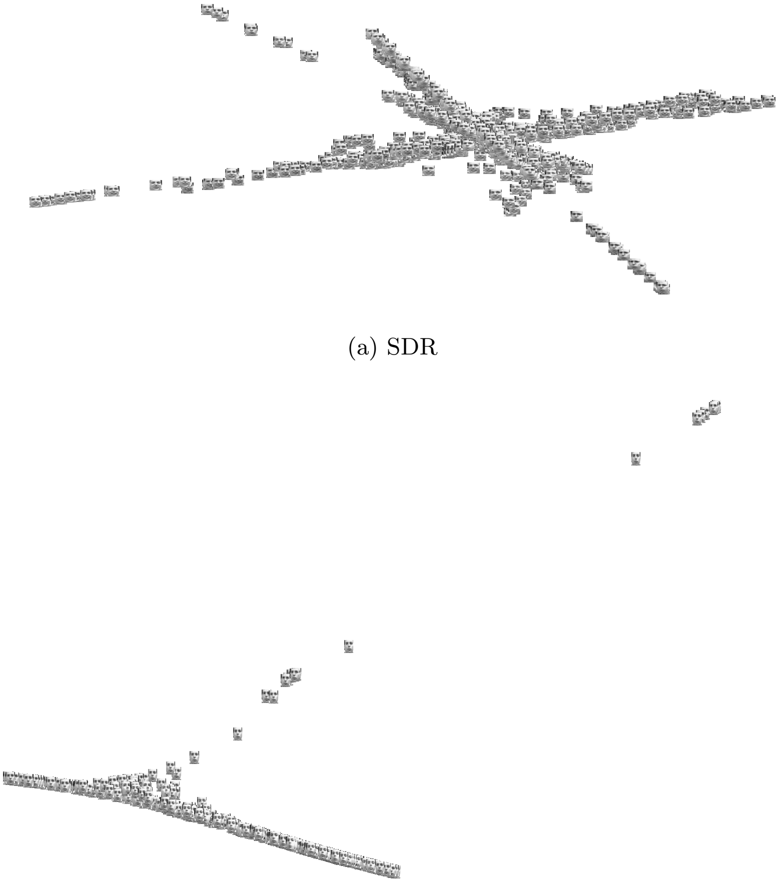
5.2 Frey Faces

In this subsection, the input data is 1,965 images (each 28 × 20 grayscale) of a single person's face taken from a video sequence which was also used in [27]. It is widely accepted that two parameters control the images, the face direction and facial expression. Intuitively, there should be two axes in 2D plane for these two parameters fused together somehow. However, the understanding like this is somewhat artificial. This may not even close to the truth. But we hope our algorithms can shed some light on these two parameters. Very interestingly as shown in Figure 2 (a) corresponding to SDR results, three axes for happy, sad and plain expressions respectively with continuously changing face direction can be clearly observed. It turns out that SDR identified the major dimensions as



(a) SDR (b) LE (c) 1NN error

Fig. 1. COIL 3D images



(a) SDR

(b) LE

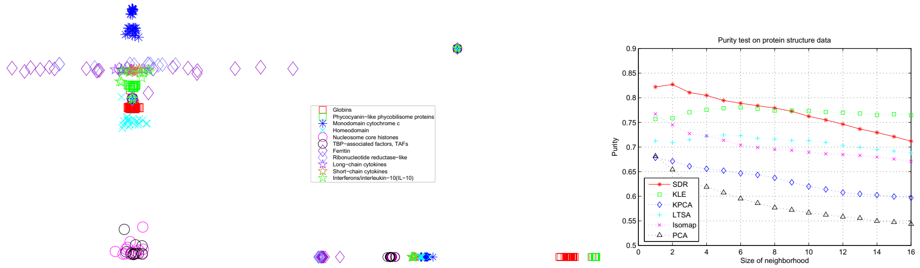
Fig. 2. Frey faces

facial expressions. The way that SDR axes are lined up once again pronounces its potential in subspace learning. The LE's result smeared out the facial expression and direction, which is not really informative.

5.3 Protein Structure Data

We move from image data (mainly video sequences) to protein structure data where KLE is more suitable because of the non-vectorial property of the protein data. Experiment was conducted on the SCOP (Structural Classification Of Protein). This database is available at http://scop.mrc-lmb.cam.ac.uk/scop/. It provides a detailed description of the structural and evolutionary relationships of the proteins of known structure. 292 protein sequences from different superfamilies and families were extracted for the test. The kernel we used is the Mammoth kernel, a so-called alignment kernel [28].

Only the results of SDR and KLE are plotted in Figure 3 for limited space. However other methods were tested. Each point (denoted as a shape in the figure) represents a protein. The same shapes with the same colors are the proteins from the same families (shown in legend) while the same shapes with different colors represent the proteins from different families but the same superfamilies. Except for better scattered clusters in SDR result, one noticeable difference is that one family dominates (diamonds) the horizontal axis in the middle in SDR result and others are projected to the other axis as 2D space is apparently not enough for this data set.



(a) SDR (b) KLE (c) Purity comparison

Fig. 3. 2D visualization of protein structure data

We used "purity" [10] to quantify the clusters produced by different methods. It uses the fraction of the number of samples from the same class as given point in a neighborhood of size n. The purity is the average of the fraction over all points. The higher the purity, the better the quality of the clusters. As we can see from Figure 3 (c), SDR has the purest clusters when $n < 9$. Although it drops below KLE when $n \geq 9$, it is still better than other methods in this test. Note that for linear methods like PCA we used the vectorization method derived form the kernel introduced in [10].

6 Conclusion

We proposed a novel method called sparse dimensionality reduction (SDR) in this paper along with a practical optimization algorithm to minimize an approximated SDR objective. SDR projects input data to a space where the redundant dimensions are compressible, and therefore it is not necessary to specify the dimensionality of the target space. The dimension sparsity parameter z in (5) is determined empirically. Bisection can be used if the target dimensionality is clear as shown in the experiments. If the final dimensionality is tied up with some quantitative standard such as MSE in regression, we can optimize z against it. It exhibits subspace learning property and the interesting results in images from video sequences suggested that SDR may be suitable for, and not confined to, computer vision applications such as subspace identification etc.

Acknowledgement. This project is sponsored by the Australian Research Council (ARC) under Discovery Project grant DP130100364 and also partially supported by the Commonwealth of Australian under the Australian-China Science and Research Fund (ACSRF01222).

References

1. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation 15(6), 1373–1396 (2003)
2. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(22), 2319–2323 (2000)
3. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimensionality reduction via tangent space. SIAM Journal on Scientific Computing 26(1), 313–338 (2005)
4. Lawrence, N.: Probabilistic non-linear principal component analysis with gaussian process latent variable models. Journal of Machine Learning Research 6, 1783–1816 (2005)
5. Jolliffe, M.: Principal Component Analysis. Springer, New York (1986)
6. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annals of Eugenics 7, 179–188 (1936)
7. Schölkopf, B., Smola, A.J., Müller, K.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10, 1299–1319 (1998)
8. Baudat, G., Anouar, F.: Generalized discriminant analysis using a kernel approach. Neural Computation 12(10), 2385–2404 (2000)
9. Guo, Y., Gao, J., Kwan, P.W.H.: Kernel laplacian eigenmaps for visualization of non-vectorial data. In: Sattar, A., Kang, B.-H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1179–1183. Springer, Heidelberg (2006)
10. Guo, Y., Gao, J., Kwan, P.W.: Twin kernel embedding. IEEE Transaction of Pattern Analysis and Machine Intelligence 30(8), 1490–1495 (2008)
11. Maillard, O.A., Munos, R.: Compressed least-squares regression. In: Advances in Neural Information Processing Systems 2011 (2011)
12. Farahmand, A.M., Szepesvári, C., Audibert, J.Y.: Manifold-adaptive dimension estimation. In: Proceedings of the 24th International Conference on Machine Learning (2007)

13. Tibshirani, R.: Regression shrinkage and selection via the Lasso. Journal of Royoal Statistical Society 1(58), 267–288 (1996)
14. Friedman, J.H., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software 33(1), 1–22 (2010)
15. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences 2(1), 183–202 (2009)
16. Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 68(1), 49–67 (2006)
17. Geiger, A., Urtasun, R., Darrell, T.: Rank priors for continuous non-linear dimensionality reduction. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 880–887 (2009)
18. Gkioulekas, I., Zickler, T.: Dimensionality reduction using the sparse linear model. In: Advances in Neural Information Processing Systems 2011 (2011)
19. Saul, L.K., Weinberger, K.Q., Sha, F., Ham, J., Lee, D.D.: Spectral methods for dimensionality reduction. In: Chapelle, O., Schölkopf, B., Zien, A. (eds.) Semi-Supervised Learning. MIT Press, MA (2006)
20. 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)
21. He, X., Niyogi, P.: Locality preserving projections. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems 16. MIT Press, Cambridge (2004)
22. Guo, Y., Gao, J., Hong, X.: Constrained grouped sparsity. In: Thielscher, M., Zhang, D. (eds.) AI 2012. LNCS, vol. 7691, pp. 433–444. Springer, Heidelberg (2012)
23. Lin, Z., Liu, R., Su, Z.: Linearized alternating direction method with adaptive penalty for low rank representation. In: Advances in Neural Information Processing Systems (2011)
24. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
25. Liu, J., Ji, S., Ye, J.: Multi-task feature learning via efficient $l2,1$-norm minimization. In: UAI, pp. 339–348 (2009)
26. Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course. Kluwer Academic Publishers (2003)
27. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(22), 2323–2326 (2000)
28. Qiu, J., Hue, M., Ben-Hur, A., Vert, J.P., Noble, W.S.: An alignment kernel for protein structures 23(9), 1090–1098 (2007)

Multi-View Visual Classification via a Mixed-Norm Regularizer

Xiaofeng Zhu[1], Zi Huang[1], and Xindong Wu[2,3]

[1] School of Information Technology & Electrical Engineering, The University of Queensland,
Brisbane, QLD4072, Australia
{zhux,huang}@itee.uq.edu.au

[2] School of Computer Science and Information Engineering,
Hefei University of Technology, China

[3] Department of Computer Science, University of Vermont, Burlington, Vermont 05405, USA
xwu@uvm.edu

Abstract. In data mining and machine learning, we often represent instances by multiple views for better descriptions and effective learning. However, such comprehensive representations can introduce redundancy and noise. Learning with these multi-view data without any preprocessing may affect the effectiveness of visual classification. In this paper, we propose a novel mixed-norm joint sparse learning model to effectively eliminate the negative effect of redundant views and noisy attributes (or dimensions) for multi-view multi-label (MVML) classification. In particular, a mixed-norm regularizer, integrating a Frobenius norm and an $\ell_{2,1}$-norm, is embedded into the framework of joint sparse learning to achieve the design goals, which include selecting significant views, preserving the intrinsic view structure and removing noisy attributes from the selected views. Moreover, we devise an iterative algorithm to solve the derived objective function of the proposed mixed-norm joint sparse learning model. We theoretically prove that the objective function converges to its global optimum via the algorithm. Experimental results on challenging real-life datasets show the superiority of the proposed learning model over state-of-the-art methods.

Keywords: Feature selection, Joint sparse learning, Manifold learning.

1 Introduction

In many real-world applications in data mining and machine learning, data are naturally described by multiple views [7]. For example, in document analysis, web pages can be represented by their content or the content of the pages pointing to them; In bioinformatics, genes can be described with the feature space (corresponding to the genetic activity under the biological conditions) as well as the term space (corresponding to the text information related to the genes); Images are represented by different kinds of low-level visual features, such as color histograms, bags of visual words, and so on.

Actually, different views describe different aspects of data. No one among them is absolutely better than others for describing the data [10]. Thus, a good alternative is to simultaneously employ multiple views to learn the data. This is well known as multi-view learning [2]. Multi-view learning has been shown to be more effective than single-view learning, particularly in the scenario where the weaknesses of a single view can be

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 520–531, 2013.

strengthened by others [14]. For example, in content analysis, color features have been shown to be sensitive to scaling, while SIFT features are robust to scaling. Combining color and SIFT features to perform multi-view learning can boost the performance by complementing each other's robustness on different aspects of the data.

Meanwhile, existing multi-view learning methods have several limitations. First, not all views of data are useful for some specific learning tasks since some of them may be redundant. However, existing methods are often designed to learn from all views of the data, without taking the redundancy issue into account. For example, canonical correlation analysis (CCA) and its kernel edition KCCA (e.g., [4,16]) were designed to learn a common latent feature space by learning from all views of the data. Second, multi-view data often contain noise, which easily affects the effectiveness of learning tasks while learning from all views of the data. Third, the intrinsic group structure of each individual view (i.e., view structure) in the data should also be preserved. Given multi-view data, each view of the data is a natural group to describe an aspect of the data. For example, a color histogram feature naturally forms a group for describing the color characteristics of image data.

Given that data are often represented by multiple views and associated with multiple object categories, this paper focuses on the problem of visual classification with multi-view multi-label (MVML) learning. In this paper, we propose a novel mixed-norm joint sparse learning model, which aims to select representative views and remove noisy attributes for MVML classification. More specifically, we first employ a least square loss function measured via a Frobenius norm (or F-norm in short) in each view to pursue a minimal regression error across all the views. We then introduce a new mixed-norm regularizer (i.e., combining an F-norm with an $\ell_{2,1}$-norm) to avoid redundant views and preserve the intrinsic view structure via the F-norm regularizer, and remove noisy attributes via the $\ell_{2,1}$-norm regularizer. We further devise a novel iterative algorithm to efficiently solve the objective function of the proposed mixed-norm joint sparse learning model, and then theoretically prove that the proposed algorithm enables the objective function to converge to its global optimum. After performing the iterative algorithm, the derived regression coefficient matrix only contains a few non-zero rows in a few selected views due to the mixed-norm regularizer. This makes the test process more efficient. Finally, we conduct an extensive experimental study on real-life datasets to compare the effectiveness of the proposed learning model with state-of-the-art methods for MVML classification.

We summarize the contributions of this paper as follows:

- We identify limitations in traditional multi-view learning, mainly caused by redundant visual features and noisy attributes. In this paper, we devise an effective solution to tackle the limitations via the proposed mixed-norm joint sparse learning model, which can be applied to many real-world applications, such as MVML visual classification
- The proposed model focuses on embedding a mixed-norm regularizer into the existing joint sparse learning framework. We solve the derived objective function by a simple yet efficient optimization algorithm, which theoretically guarantees that the object ive function converges to its global optimum.

- To perform MVML classification, the proposed model can be regarded as simultaneously performing two types of feature selection, i.e., view-selection and attribute-selection respectively. View-selection aims to discard redundant views and preserve the intrinsic view structure via the F-norm regularizer, while attribute-selection aims to select useful attributes in the selected views of the data via the $\ell_{2,1}$-norm regularizer. These two types of feature selection lead to a new mixed joint sparsity, i.e., the view sparsity and the row sparsity simultaneously. To the best of our knowledge, no research efforts have been proposed on performing two types of feature selection in the joint sparse learning framework. Moreover, extensive experimental results on the public datasets show that the proposed model is more effective than state-of-the-art methods.

2 Approach

In this paper, ℓ_p-norm of a vector $\mathbf{v} \in \mathbb{R}^n$ is defined as $||\mathbf{v}||_p = \left(\sum\limits_{i=1}^{n} |v_i|^p\right)^{\frac{1}{p}}$, where v_i is the i^{th} element of $\mathbf{v}$. $\ell_{r,p}$-norm over a matrix $\mathbf{M} \in \mathbb{R}^{n \times m}$ is defined as $||\mathbf{M}||_{r,p} = \left(\sum\limits_{i=1}^{n}\left(\sum\limits_{j=1}^{m} ||m_{ij}||^r\right)^{\frac{p}{r}}\right)^{\frac{1}{p}}$, where m_{ij} is the element of the i^{th} row and j^{th} column. The transpose of $\mathbf{X}$ is denoted as $\mathbf{X}^T$, the inverse of $\mathbf{X}$ is $\mathbf{X}^{-1}$, and the trace operator of a matrix is denoted by the symbol “tr”.

2.1 Loss Function

Given the g-th view $\mathbf{X}^g$ of the training data $\mathbf{X}$, we need to obtain its regression coefficients $\mathbf{W}^g$. In MVML learning, we wish to obtain the minimal difference between the training label $\mathbf{Y} = [\mathbf{Y}_1^T, ..., \mathbf{Y}_n^T]^T$ and the summation of all G views on the multiplication between $\mathbf{X}^g$ and $\mathbf{W}^g$, i.e., $\sum\limits_{g=1}^{G} \mathbf{X}^g\mathbf{W}^g$. Therefore, a least square loss function can be defined as follows:

$$\min_{\mathbf{W}} ||\mathbf{Y} - \mathbf{X}\mathbf{W}||_F^2 \tag{1}$$

where $\mathbf{X} = [\mathbf{X}^1, ..., \mathbf{X}^g, ..., \mathbf{X}^G]$. Obviously, Eq.1 meets our goal of minimizing the regression error across all views.

2.2 Mixed-Norm Regularizer

Given a loss function (such as in Eq.1), during the optimization process we also design a mixed-norm regularizer, aiming to meet other goals, such as removing redundant views and noisy attributes. In this paper, we achieve these goals by performing two types of feature selection, i.e., view-selection for removing redundant views and attribute-selection for deleting noisy attributes. To this end, we propose a new mixed-norm regularizer by integrating an F-norm regularizer with an $\ell_{2,1}$-norm regularizer.

More concretely, in the proposed joint sparse learning model, the F-norm regularizer generates the codes of redundant views as zeros and the others as non-zeros; the $\ell_{2,1}$-norm regularizer generates the codes of noisy attributes as zeros and the others as non-zeros. Then with the impact of sparse views and attributes, MVML classification can be effectively and efficiently performed. Moreover, the mixed-norm regularizer enables us to avoid the issue of over-fitting.

In this paper, the $\ell_{2,1}$-norm regularizer is defined as:

$$\|\mathbf{W}\|_{2,1} = \sum_{i=1}^{m} \|(\mathbf{W})^i\|_2 = \sum_{i=1}^{m} \sqrt{\sum_{j=1}^{c} m_{ij}^2} \tag{2}$$

where $(\mathbf{W})^j$ is the j-th row of matrix $\mathbf{W}$, and indicates the effect of the j-th attribute to all data points. As mentioned by existing literatures, e.g., [12], the $\ell_{2,1}$-norm regularizer was designed to measure the distance of the attributes via the ℓ_2-norm, while performing summation over all data points via the ℓ_1-norm. Thus the $\ell_{2,1}$-norm regularizer leads to the row sparsity as well as to consider the correlations of all attributes.

The F-norm regularizer is defined as:

$$\|\mathbf{W}\|_F = \sqrt{\sum_{g=1}^{m_g} \|\mathbf{W}^g\|_2^2} = \sqrt{\sum_{g=1}^{m_g}\sum_{j=1}^{c} w_{g,j}^2} \tag{3}$$

where $\mathbf{W}^g$ is the g-th block of matrix $\mathbf{W}$ (or the submatrix formed by all the rows belonging to the g-th view), and indicates the effect of the g-th block (i.e., there are sequential m_g rows in the g-th view) to all data points.

2.3 Objective Function

By considering three equations together, i.e., Eq.1, Eq.2 and Eq.3, we obtain the objective function of the proposed mixed-norm joint sparse learning model as follows:

$$\min_{\mathbf{W}} \tfrac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{W}\|_F^2 + \lambda_1\|\mathbf{W}\|_{2,1} + \lambda_2 \sum_{g=1}^{G} \|\mathbf{W}^g\|_F \tag{4}$$

where both λ_1 ($\lambda_1 > 0$) and λ_2 ($\lambda_2 > 0$) are tuning parameters.

Similar to the mixed sparsity using the ℓ_1-norm regularizer and the $\ell_{2,1}$-norm regularizer together in separable sparse learning, the proposed mixed regularizer leads to the mixed joint sparsity. That is, it first discriminates redundant views via the F-norm regularizer, and then detects noisy attributes in the selected views via the $\ell_{2,1}$-norm regularizer.

Actually, some literatures have focused on the mixed sparsity, such as elastic net [19] and sparse group lasso [11] in separable sparse learning, adaptive multi-task lasso [8] in joint sparse learning, and so on. For example, an elastic net combines the ℓ_1-norm regularizer with the ℓ_2-norm regularizer for achieving the element sparsity (via the ℓ_1-norm regularizer) and impact group effect (via the ℓ_2-norm regularizer). Sparse group lasso achieves the mixed sparsity, i.e., the element sparsity via the ℓ_1-norm regularizer as well as the group sparsity via the $\ell_{2,1}$-norm regularizer. As mentioned in Section 2, neither elastic net nor sparse group lasso benefit for feature selection. Recently, adaptive multi-task lasso combines the ℓ_1-norm regularizer with the $\ell_{2,1}$-norm regularizer in multi-task learning to achieve feature selection (via the $\ell_{2,1}$-norm regularizer) and deletes noisy elements (via the ℓ_1-norm regularizer). Obviously, existing literatures mentioned above were not designed to delete redundancy views and to perform feature selection at the same time, as the proposed method in this paper does.

Next we explain why the proposed mixed-norm regularizer leads to the mixed joint sparsity, i.e., simultaneously obtaining two types of sparsity. While the value of λ_2 is larger, the minimization process in Eq.4 drives the value of the F-norm (i.e., the third term in Eq.4) smaller. This tends to force the values of some blocks (e.g., the value of the g-th block is $\|\mathbf{W}^g\|_F$) with small values to be smaller. After several iterations, the values of these blocks in $\mathbf{W}$ are close to zero. Thus we obtain a sparse $\mathbf{W}$ with zero value in some blocks, e.g., the g-th block. This indicates that the corresponding views (e.g., the g-th view) of $\mathbf{X}$ are redundant views since the sparsity appears in those blocks (e.g., the g-th block) of $\mathbf{W}$. The sparse blocks of $\mathbf{W}$ remove the corresponding views of $\mathbf{X}$ from the test process. Meanwhile, we also notice that the larger the value of λ_2, the more the block sparsity. With the same principle, while the value of λ_1 is larger, the minimization process in Eq.4 forces some rows in $\mathbf{W}$ to be zero, i.e., the attributes corresponding to the sparse rows in $\mathbf{W}$ are not involved the test process. Hence, the proposed mixed-norm regularizer leads to the mixed joint sparsity, which achieves the block sparsity as well as the row sparsity.

According to above analysis, Eq.4 can be used to select a few useful attributes from a few representative (or significant) views of the data for the visual classification. This has the following advantages. First, it benefits for improving the efficiency of the test process due to the sparse $\mathbf{W}$. Second, these two kinds of feature selection help to avoid the impact of redundant views and noisy attributes in the test process, thus benefit for effectively performing the MVML classification. Third, it induces the mixed joint sparsity as well as leads to a hierarchical coding model (i.e., non-sparse attributes generated from non-sparse views), which plays an important role in many applications where a feature hierarchy exists. Last but not the least, views-selection via the F-norm regularizer also preserves the individual view structures of the non-sparse views since each view is regarded as a block.

2.4 Classification

By solving Eq.4, we obtain the optimal $\mathbf{W}$. Given a test dataset $\mathbf{X}_{test}$, we obtain the corresponding label set $\mathbf{Y}_{test}$ by $\mathbf{Y}_{test} = \mathbf{X}_{test}\mathbf{W}$ in the test process. Due to inducing by the proposed mixed-norm regularizer, only a few blocks in the derived $\mathbf{W}$ are non-zeros, and also only a few rows in these non-zero blocks are non-zeros. This makes the test process more efficient to be performed.

After ranking $\mathbf{Y}_{test}$ according to the label values, the top-k labels are assigned to the test data as the predicted labels. This rule is the same to existing multi-label methods, e.g., [18].

3 Optimization

Eq.4 is obviously convex since it consists of three norms, which have been shown to be convex [6]. Therefore, Eq.4 has the global optimum. However, its optimization is very challenging because both the $\|\mathbf{W}\|_F$-norm and the $\|\mathbf{W}\|_{2,1}$-norm in Eq.4 are convex but non-smooth. In this section we solve this problem by calculating sub-gradients of the mixed-norm regularizer, i.e., the $\|\mathbf{W}\|_F$-norm and the $\|\mathbf{W}\|_{2,1}$-norm respectively.

3.1 The Proposed Solver

By setting the derivative of Eq.4 with respect to $\mathbf{W}$ as zero, we obtain:

$$(\mathbf{X}^T\mathbf{X} + \lambda_1\mathbf{C} + \lambda_2\mathbf{D})\mathbf{W} = \mathbf{X}^T\mathbf{Y} \tag{5}$$

where $\mathbf{C}$ is a diagonal matrix with the i-th diagonal element:

$$C_{i,i} = \frac{1}{2\|(\mathbf{W})^i\|_2} \tag{6}$$

where $(\mathbf{W})^i$ denotes the i-th row of $\mathbf{W}$, $i = 1, ..., n$. $\mathbf{D} = diag(\mathbf{D}^1, ..., \mathbf{D}^G)$, where the symbol '*diag*' is the diagonal operator and each $\mathbf{D}^g$ $(g = 1, ..., G)$ is also a diagonal matrix with the i-th diagonal element as:

$$D_{j,j} = \frac{1}{2\|\mathbf{W}^g\|_F} \tag{7}$$

where $j = 1, ..., m_g$.

By observing Eq.5, we find that both the matrix $\mathbf{C}$ and the matrix $\mathbf{D}$ depend on the value of matrix $\mathbf{W}$. In this paper we design a novel iterative algorithm to optimize Eq.5 by alternatively computing the $\mathbf{W}$ and the $\mathbf{C}$ (with the $\mathbf{D}$). We first summarize the details in Algorithm 1, and then prove that in each iteration the updated $\mathbf{W}$ and the $\mathbf{C}$ (with the $\mathbf{D}$) make the value of Eq.4 decrease.

Algorithm 1. The proposed method for solving Eq.4

```
Input: $\mathbf{Y} \in \mathbb{R}^{n \times c}, \mathbf{X} \in \mathbb{R}^{n \times D}, \lambda_1$ and $\lambda_2$;
Output: $\mathbf{W} \in \mathbb{R}^{D \times c}$;
Initialize $t = 0$;
Initialize $\mathbf{C}_0$ as a $D \times D$ identity matrix;
Initialize $\mathbf{D}_0$ as a $D \times D$ identity matrix;
repeat
    $\mathbf{W}^{[t+1]} = (\mathbf{X}^T\mathbf{X} + \lambda_1\mathbf{C}^{[t]} + \lambda_2\mathbf{D}^{[t]})^{-1}\mathbf{X}^T\mathbf{Y}$;
    Update $\mathbf{C}^{[t+1]}$ via Eq.6;
    Update $\mathbf{D}^{[t+1]}$ via Eq.7;
    $t = t+1$;
until No change on the objective function value in Eq.4;
```

With Algorithm 1, at each iteration, given the fixed $\mathbf{C}$ and $\mathbf{D}$, the $\mathbf{W}$ is updated by Eq.5. Then the $\mathbf{C}$ and the $\mathbf{D}$ can be updated with the fixed $\mathbf{W}$. The iteration process is repeated until there is no change on the value of Eq.4.

3.2 Convergence

In this subsection we introduce Theorem 1 to guarantee that Eq.4 monotonically decreases in each iteration of Algorithm 1. Following the literature in [9,17], we first give a lemma as follows:

Lemma 1. *For any positive values a_i and b_i, $i = 1, ..., m$, the following holds:*

$$\sum_{i=1}^{m} \frac{b_i^2}{a_i} \leq \sum_{i=1}^{m} \frac{a_i^2}{a_i} \Longleftrightarrow \sum_{i=1}^{m} \frac{(b_i+a_i)(b_i-a_i)}{a_i} \leq 0$$
$$\Longleftrightarrow \sum_{i=1}^{m} (b_i - a_i) \leq 0 \Longleftrightarrow \sum_{i=1}^{m} b_i \leq \sum_{i=1}^{m} a_i \tag{8}$$

Theorem 1. *In each iteration, Algorithm 1 monotonically decreases the objective function value in Eq.4.*

Proof. According to the fifth line of Algorithm 1, we denote the $\mathbf{W}^{[t+1]}$ as the results of the $(t+1)$-th iteration of Algorithm 1, then we have:

$$\begin{aligned} \mathbf{W}^{[t+1]} = \min_{\mathbf{W}} \ & \frac{1}{2}\|\mathbf{Y} - \mathbf{XW}\|_F^2 + \lambda_1 tr(\mathbf{W}^T \mathbf{C}^{[t]} \mathbf{W}) \\ & + \lambda_2 \sum_{g=1}^{G} tr((\mathbf{W}^g)^T (\mathbf{D}^g)^{[t]} \mathbf{W}^g) \end{aligned} \tag{9}$$

Then we can get:

$$\begin{aligned} & \frac{1}{2}\|\mathbf{Y} - \mathbf{X}(\mathbf{W}^{[t+1]})^T\|_F^2 + \lambda_1 tr((\mathbf{W}^{[t+1]})^T \mathbf{C}^{[t]} \mathbf{W}^{[t+1]}) \\ & \quad + \lambda_2 \sum_{g=1}^{G} tr(((\mathbf{W}^g)^{[t+1]})^T (\mathbf{D}^g)^{[t]} (\mathbf{W}^g)^{[t+1]}) \\ & \leq \frac{1}{2}\|\mathbf{Y} - \mathbf{X}(\mathbf{W}^{[t]})^T\|_F^2 + \lambda_1 tr((\mathbf{W}^{[t]})^T \mathbf{C}^{[t]} \mathbf{W}^{[t]}) \\ & \quad + \lambda_2 \sum_{g=1}^{G} tr(((\mathbf{W}^g)^{[t]})^T (\mathbf{D}^g)^{[t]} (\mathbf{W}^g)^{[t]}) \end{aligned} \tag{10}$$

which indicates that:

$$\begin{aligned} & \frac{1}{2}\|\mathbf{Y} - \mathbf{X}(\mathbf{W}^{[t+1]})^T\|_F^2 + \sum_{i=1}^{n} \frac{\|(\mathbf{W}^{[t+1]})^{i}\|_2^2}{2\|(\mathbf{W}^{[t]})^{i}\|_2}) + \sum_{g=1}^{G} \frac{\|(\mathbf{W}^g)^{[t+1]}\|_F^2}{2\|(\mathbf{W}^g)^{[t]}\|_F}) \\ & \leq \frac{1}{2}\|\mathbf{Y} - \mathbf{X}(\mathbf{W}^{[t]})^T\|_F^2 + \sum_{i=1}^{n} \frac{\|(\mathbf{W}^{[t]})^{i}\|_2^2}{2\|(\mathbf{W}^{[t]})^{i}\|_2}) + \sum_{g=1}^{G} \frac{\|(\mathbf{W}^g)^{[t]}\|_F^2}{2\|(\mathbf{W}^g)^{[t]}\|_F}) \end{aligned} \tag{11}$$

Substituting b_i and a_i with $\left\|(\mathbf{W}^{[t+1]})^i\right\|_2$ (or $\|(\mathbf{W}^g)^{[t+1]}\|_F$) and $\left\|(\mathbf{W}^{[t]})^i\right\|_2$ (or $\|(\mathbf{W}^g)^{[t]}\|_F$) in Lemma 1, we have:

$$\begin{aligned} &\frac{1}{2}\|\mathbf{Y}-\mathbf{X}(\mathbf{W}^{[t+1]})^T\|_F^2+\lambda_1\sum_{i=1}^{n}\|(\mathbf{W}^{[t+1]})^i\|_2+\lambda_2\sum_{g=1}^{G}\|(\mathbf{W}^g)^{[t+1]}\|_F \\ &\leq\frac{1}{2}\|\mathbf{Y}-\mathbf{X}(\mathbf{W}^{[t]})^T\|_F^2+\lambda_1\sum_{i=1}^{n}\|(\mathbf{W}^{[t]})^i\|_2+\lambda_2\sum_{g=1}^{G}\|(\mathbf{W}^g)^{[t]}\|_F \end{aligned} \tag{12}$$

This indicates that the objective function value in Eq.4 monotonically decreases in each iteration of Algorithm 1. Therefore, due to the convexity of Eq.4, Algorithm 1 enables Eq.4 to converge to its global optimum.

4 Experimental Analysis

In order to evaluate the performance of the proposed mixed-norm joint sparse learning (denoted as $F2L21F$[1] for short from its objective function), we compare it with several state-of-the-art methods on public datasets (e.g., MIRFLICKR [5] and NUS-WIDE [3]) for MVML classification, by evaluating the average precision and Hamming loss.

4.1 Experiment Setup

We use four datasets, including MIRFlickr, NUS-WIDE, SCENE and OBJECT in our experiments for MVML classification. The comparison methods include the method in [15] (denoted as *F2F* from its objective function, for simplicity) which only considers the block sparsity, the method in [12] (denoted as *F2L21*) which only considers the row sparsity, the *MKCCA* method in [1] which does not consider the feature redundancy and noise, and the single view method *WorstS* (or *BestS*) which has the worst (or best) classification performance from the data represented by a single view via ridge regression (i.e., all single views are tested). We use two popular evaluation metrics (i.e., the average precision (AP) and Hamming loss (HL)) in multi-label learning [13] to evaluate the effectiveness of all the methods in our experiments.

Given the ground true label matrix $Y1 \in \{0,1\}^{n\times c}$ (where n is the number of instances and c is the number of labels) and the predicted one $Y2 \in \{0,1\}^{n\times c}$ obtained by the algorithm for performing MVML learning, average precision (AP) is defined as:

$$AP=\frac{1}{n\times c}\sum_{i=1}^{n}\frac{card(Y1_i\cap Y2_i)}{card(Y1_i\cup Y2_i)} \tag{13}$$

where the symbol “Card” means the cardinality operation.

HL measuring the recovery error rate is defined as:

$$HL=\frac{1}{n\times c}\sum_{i=1}^{n}\sum_{j=1}^{c}Y1_{i,j}\oplus Y2_{i,j} \tag{14}$$

where $\oplus$ is an XOR operation, a.k.a. exclusive disjunction.

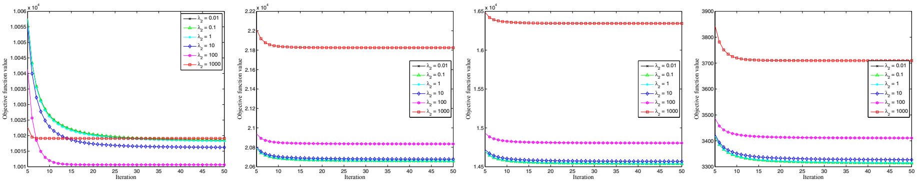
[1] $F2$ means the least square loss function, $L21$ means the $\ell_{2,1}$-norm regularizer, and F means the F-norm regularizer.

According to the literatures, e.g., [13,18], the larger (or smaller) the performance on AP (or HL) is, the better the method.

4.2 Experimental Results

In this subsection, we report the results on MVML classification. First, we evaluate the convergence rate of the proposed *F2L21F* on all four datasets, for evaluating the efficiency of our optimization algorithm, in terms of the objective function value in each iteration. Second, we test the parameters' sensitivity of the proposed model on λ_1 and λ_2, aiming at obtaining the best performance of the proposed *F2L21F*. Finally, we compare *F2L21F* with the comparison algorithms in terms of average precision and Hamming loss.

Convergence Rate. We solve Eq.4 by the proposed Algorithm 1. In this experiment, we want to know the convergence rate of Algorithm 1. Here we report some of the results in Fig.1 and Fig.2 due to the page limit. Fig.1 shows the results on the objective function value while fixing the value of λ_1 (i.e., $\lambda_1 = 1$) and varying λ_2. Fig.2 shows the results on the objective function value while fixing the value of λ_2 (i.e., $\lambda_2 = 1$) and varying λ_1. In both Fig.1 and Fig.2, the x-axis and y-axis denote the number of iterations and the objective function value respectively.



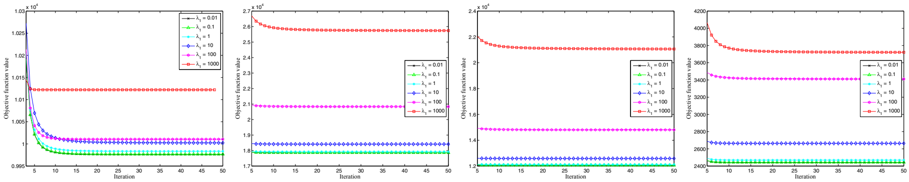
(a) MIRFlickr (b) NUS-WIDE (c) SCENE (d) OBJECT

Fig. 1. An illustration on convergence rate of Algorithm 1 for solving the proposed objective function with fixed λ_1, i.e., $\lambda_1 = 1$

We can observe from both Fig.1 and Fig.2 that: 1) the objective function value rapidly decreases at the first few iterations; and 2) the objective function value becomes stable after about 30 iterations (or even less than 20 in many cases) on all datasets. This confirms a fast convergence rate of Algorithm 1 to solve the proposed optimization problem in Eq.4. Similar results are observed for other λ_1 and λ_2 values.

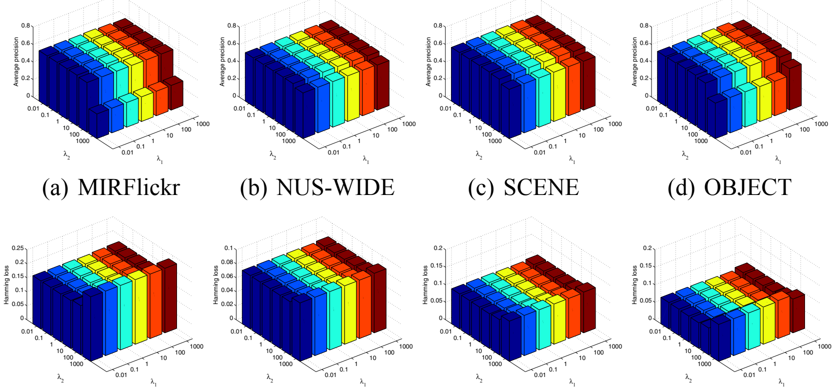
Parameters' Sensitivity. In this experiment, we test different settings on parameters λ_1 and λ_2 in the proposed *F2L21F*, by varying them as $\{0.01, 0.1, 1, 10, 100, 1000\}$. The results on average prediction and Hamming are illustrated in Fig.3.

It is clear that the proposed *F2L21F* is sensitive to the parameters' setting, similar to other sparse learning methods [11,18]. However, we find the worst performance is



(a) MIRFlickr (b) NUS-WIDE (c) SCENE (d) OBJECT

Fig. 2. An illustration on convergence rate of Algorithm 1 for solving the proposed objective function with fixed λ_2, i.e., $\lambda_2 = 1$



(a) MIRFlickr (b) NUS-WIDE (c) SCENE (d) OBJECT

(e) MIRFlickr (f) NUS-WIDE (g) SCENE (h) OBJECT

Fig. 3. The results of average precision (first row) and Hamming loss (second low) on various parameters' settings on different datasets

always obtained when both λ_1 and λ_2 have extremely large values. For example, when the values of parameters pair (λ_1, λ_2) are around (10,10), *F2L21F* achieves the best performance. Actually, in our experiments such a setting simultaneously leads to both the row sparsity (via the λ_1) and the block sparsity (via the λ_2).

Comparison. In this experiment, we compare our proposed method with state-of-the-art methods for MVML classification. We set the values of parameters for the comparison methods by following the instructions in their original papers. For all the methods, we randomly sample 60% of the original data as the training data, and leave the rest as the test data. We randomly generate ten runs, and report the average result and the standard deviation on the average precision and Hamming loss, as shown in Fig.4. Note that we do not use dataset MIRFlickr since it has only two views.

From Fig.4, we have the following observations: 1) The proposed *F2L21F* always achieves the best performance. Among six views in NUS-WIDE and five views in SCENE and OBJECT, the 64-D color histogram is detected as a redundant view in *F2L21F*. *F2L21* and *MKCCA* use all the views to perform MVML classificati n and obtain worse performance than *F2L21F*. This confirms that some views (e.g., the color histogram in the tested datasets) are not helpful in the learning process and may even

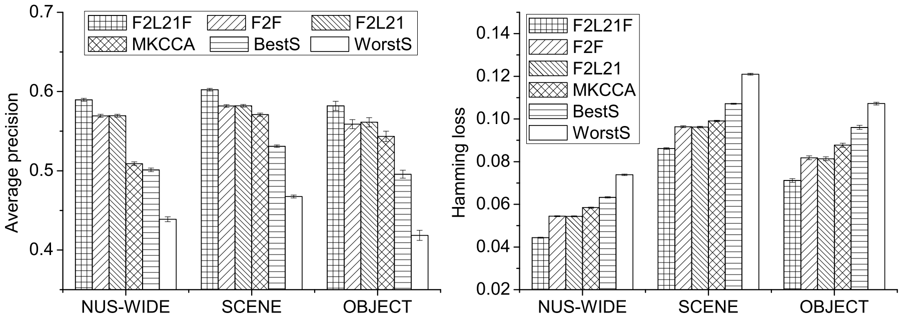


Fig. 4. Comparison on average precision (left) and Hamming loss (right) for all methods on different datasets. Note that the range shown at the top of the bar represents the performance standard deviation.

degrade the performance, especially when many views are available. *F2L21F* is able to identify those redundant views and avoid their negative impact on the classification. Although *F2F* can also discover those redundant views, it is not able to remove noisy attributes from the selected views, leading to worse performance than *F2L21F*. This result proves the effectiveness of *F2L21F* in removing redundant views and noisy attributes by employing the proposed mixed-norm regularizer in MVML classification. 2) The performance of single view learning methods (i.e., *BestS* and *WorstS*) is always worse than those multi-view learning methods. This again confirm the advantages of using multi-views in visual classification. 3) The sparse learning methods (i.e., *F2L21F*, *F2F*, and *F2L21*) consistently outperform *MKCCA*. This shows the superiority of sparse learning which encodes negligible elements as zeros and only selects important elements to perform MVML classification. Moreover, in our implementation the computational cost of *F2L21F* is about tens times faster than that of *MKCCA*, indicating much higher efficiency than *MKCCA*.

In sum, more views can help improve the performance of visual classificatio since more information can be utilized in the learning process. On the other hand, more views may also potentially introduce higher redundancy and more noise which compromise the performance. The proposed *F2L21F* is able to identify those redundant views and noisy attributes so that MVML classificatio can be performed for more effective performance.

5 Conclusion

In this paper we proposed a mixed-norm joint sparse learning model for multi-view multi-label (MVML) classification The proposed method, powered by a mixed-norm regularizer, can effectively avoid the negative impact of redundant views and noisy attributes from the multi-view representation of a large amount of data. Extensive experimental results have shown that the proposed method outperforms state-of-the-art learning methods for MVML classification. In the future, we will extend the proposed method into its kernel edition to project noise more clearly, and involve other learning models, such as semi-supervised MVML classificatio and transfer MVML classification, to leverage the widely available unlabeled data and heterogenous data.

References

1. Blaschko, M.B., Lampert, C.H., Gretton, A.: Semi-supervised laplacian regularization of kernel canonical correlation analysis. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 133–145. Springer, Heidelberg (2008)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Annual Conference on Computational Learning Theory, pp. 92–100 (1998)
3. Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: Nus-wide: A real-world web image database from national university of singapore. In: ACM International Conference on Image and Video Retrieval, p. 48 (2009)
4. Dhillon, P.S., Foster, D., Ungar, L.: Multi-view learning of word embeddings via cca. In: Neural Information Processing Systems, pp. 9–16 (2011)
5. Huiskes, M.J., Lew, M.S.: The mir flickr retrieval evaluation. In: ACM International Conference on Multimedia Information Retrieval, pp. 39–43 (2008)
6. Jenatton, R., Audibert, J.-Y., Bach, F.: Structured variable selection with sparsity-inducing norms. Journal of Machine Learning Research 12, 2777 (2011)
7. Kumar, A., DauméIII, H.: A co-training approach for multi-view spectral clustering. In: International Conference on Machine Learning, pp. 393–400 (2011)
8. Lee, S., Zhu, J., Xing, E.P.: Adaptive multi-task lasso: with application to eqtl detection. In: Neural Information Processing Systems, pp. 1306–1314 (2010)
9. Nie, F., Huang, H., Cai, X., Ding, C.: Efficient and robust feature selection via joint l2, 1-norms minimization. In: Neural Information Processing Systems, pp. 1813–1821 (2010)
10. Owens, T., Saenko, K., Chakrabarti, A., Xiong, Y., Zickler, T., Darrell, T.: Learning object color models from multi-view constraints. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 169–176 (2011)
11. Peng, J., Zhu, J., Bergamaschi, A., Han, W., Noh, D.-Y., Pollack, J.R., Wang, P.: Regularized multivariate regression for identifying master predictors with application to integrative genomics study of breast cancer. The Annals of Applied Statistics 4(1), 53–77 (2010)
12. Sun, L., Liu, J., Chen, J., Ye, J.: Efficient recovery of jointly sparse vectors. In: Neural Information Processing Systems, pp. 1812–1820 (2009)
13. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining Multi-label Data (2009)
14. Xie, B., Mu, Y., Tao, D., Huang, K.: m-sne: Multiview stochastic neighbor embedding. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 41(4), 1088–1096 (2011)
15. Yuan, X., Yan, S.: Visual classification with multi-task joint sparse representation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3493–3500 (2010)
16. Zhu, X., Huang, Z., Shen, H.T., Cheng, J., Xu, C.: Dimensionality reduction by mixed kernel canonical correlation analysis. Pattern Recognition (2012)
17. Zhu, X., Huang, Z., Yang, Y., Shen, H.T., Xu, C., Luo, J.: Self-taught dimensionality reduction on the high-dimensional small-sized data. Pattern Recognition 46(1), 215–229 (2013)
18. Zhu, X., Shen, H.T., Huang, Z.: Video-to-shot tag allocation by weighted sparse group lasso. In: ACM Multimedia, pp. 1501–1504 (2011)
19. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society Series B 67(2), 301–320 (2005)

Mining Specific Features for Acquiring User Information Needs

Abdulmohsen Algarni[1] and Yuefeng Li[2,*]

[1] College of Computer Science, King Khalid University
Saudi Arabia, B.O.Box 394, ABHA 61411
a.algarni@kku.edu.sa

[2] School of Electrical Engineering and Computer Science
Queensland University of Technology, Brisbane, QLD 4001, Australia
y2.li@qut.edu.au

Abstract. Term-based approaches can extract many features in text documents, but most include noise. Many popular text-mining strategies have been adapted to reduce noisy information from extracted features; however, text-mining techniques suffer from low frequency. The key issue is how to discover relevance features in text documents to fulfil user information needs. To address this issue, we propose a new method to extract specific features from user relevance feedback. The proposed approach includes two stages. The first stage extracts topics (or patterns) from text documents to focus on interesting topics. In the second stage, topics are deployed to lower level terms to address the low-frequency problem and find specific terms. The specific terms are determined based on their appearances in relevance feedback and their distribution in topics or high-level patterns. We test our proposed method with extensive experiments in the Reuters Corpus Volume 1 dataset and TREC topics. Results show that our proposed approach significantly outperforms the state-of-the-art models.

Keywords: Feature extraction, Pattern mining, Relevance feedback, Text classification.

1 Introduction

One of the objectives of knowledge extraction is to build user profiles by finding a set of features from feedback documents to describe user information needs. This is a particularly challenging task in modern information analysis, empirically and theoretically [11,13]. This problem has received much attention from the data mining, web intelligence, and information retrieval communities.

Information retrieval has deployed many effective term-based methods to find popular terms [14]. The advantages of term-based methods include efficient computational performance and mature theories for term weighting. However, many noisy terms can be extracted from the large-scale feedback documents. Words

[*] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 532–543, 2013.

and phrases have also been used as terms in many models. Many researchers believe phrases are more useful and crucial than words for query expansion in building effective ranking functions [14,4,22]. However, there are usually many redundant and noisy phrases [18,19].

Popular terms are useful for describing documents; however, they do not focus on the interesting topics in these documents. We argue that patterns (itemsets, or sets of terms) can be a good alternative form of terms for describing interesting topics in documents.

Data-mining techniques have been developed–e.g., maximal, closed, and master patterns–for removing redundant and noisy patterns [27,25]. By using the advantages of data-mining techniques, pattern taxonomy models (PTM) [24,23,12] have been proposed for using closed sequential patterns in text classification. These pattern mining-based approaches have improved the effectiveness for relevant (positive) feedback documents, but offer fewer significant improvements compared with term-based methods for using both relevant and irrelevant feedback.

Existing approaches focus more on extracting general or popular topics from feedback documents rather than what users really want. Several attempts have been made to determine terms specificity regarding to term distribution in documents. For example, Inverse Document Frequency (*IDF*) measures terms specificity in a set of documents, but much noisy information in text documents affects *IDF* for finding specific features.

This research proposes a specificity definition for mining specific features for user information needs. The proposed approach includes two stages. The first stage extracts high-level patterns (or topics) from text documents to focus on interesting topics in order to reduce the noise. The second stage deploys these topics (high-level patterns) to lower level terms to address the low-frequency problem in order to find specific features. The proposed approach can determine specific terms based on both their appearances in relevance feedback and their distribution in interesting topics (or high-level patterns).

The remainder of this paper is organized as follows. Section 2 introduces a detailed overview of related works. Section 3 reviews concepts of patterns in text documents. Section 4 proposes a method of mining specific features from positive feedback documents. Section 5 shows empirical results; Section 6 reports related discussions, followed by the final sections concluding remarks.

2 Related Work

Scientists have proposed many types of text representation. A well-known one is the bag-of-words model that uses keywords, or terms, in the vector of the feature space. In [9], the *tf*idf* weighting scheme was used for text representation in Rocchio classifiers. Enhanced from *tf*idf*, the global IDF and entropy weighting scheme was proposed in [5]. Various weighting schemes for the bag-of-words representation were given in [1,7].

Bag of words problem is how to select a limited number of feature terms to increase the system's efficiency and avoid *overfitting* [19]. To reduce the number of features, many dimensionality reduction approaches have been conducted using feature selection techniques like information gain, mutual information, chi-square, and odds ratio [19].

In [2], data-mining techniques analyzed text by extracting co-occurring terms as descriptive phrases from document collections. However, the effectiveness of the text classification systems using phrases as text representation showed no significant improvement. The likely reason is that a phrase-based method has lower consistency of assignment and lower document frequency for terms [8].

The data-mining community has extensively studied pattern mining for many years. Usually, existing data-mining techniques discover numerous patterns (e.g., sets of terms) from a training set, but many patterns may be redundant [25]. Nevertheless, the challenge is dealing effectively with the many discovered patterns and terms with much noise.

Regarding to these setbacks, closed patterns present a promising alternative to phrases [23,6]. Patterns, like terms, enjoy good statistical properties. To use closed patterns effectively in text mining, patterns have been evaluated by being deployed into a vector with a set of terms and term-weight distributions. The pattern-deploying method encouragingly improves effectiveness compared with traditional probabilistic models and Rocchio-based methods [23,12].

In summary, we can group the existing methods for finding relevance features into three approaches. The first one is to revise feature terms in both positive and negative samples, like Rocchio-based models [15]. The second approach is based on how often terms appear or do not appear in positive and negative samples like probabilistic-based models [26]. The third approach is to describe specific features based on their appearances in both patterns or/and documents [23,10]. In this paper, we further develop the third approach to utilize high-level patterns (or topics) extracted from only positive samples (relevant documents) for finding specific terms. The major research issue is how to determine the topics specificity of terms according their distributions in both documents and topics.

3 Definition

In this paper, we assume that all documents are split in paragraphs. So a given document d yields a set of paragraphs $PS(d)$. Let D be a training set of documents, which consists of a set of positive documents, D^+; and a set of negative documents, D^-. Let $T = \{t_1, t_2, \ldots, t_m\}$ be a set of terms (or keywords) which are extracted from the set of positive documents, D^+.

3.1 Frequent and Closed Patterns

Let $T = \{t_1, t_2, \ldots, t_m\}$ be a set of terms which are extracted from D^+. Given a *termset* X, a set of terms, in document d, $coverset(X) = \{dp | dp \in PS(d), X \subseteq dp\}$. Its *absolute support* $sup_a(X) = |coverset(X)|$; and its *relative support* $sup_r(X) = \frac{|coverset(X)|}{|PS(d)|}$. A termset X is called *frequent pattern* if its sup_a (or sup_r) $\geq min\_sup$, a minimum support.

Table 1. A set of paragraphs

Parapgraph	*Terms*
dp_1	t_1 t_2
dp_2	t_3 t_4 t_6
dp_3	t_3 t_4 t_5 t_6
dp_4	t_3 t_4 t_5 t_6
dp_5	t_1 t_2 t_6 t_7
dp_6	t_1 t_2 t_7

Table 2. Frequent patterns and covering sets

Freq. Pattern	*Covering Set*
$\{\mathbf{t_3}, \mathbf{t_4}, \mathbf{t_6}\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_3, t_4\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_3, t_6\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_4, t_6\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_3\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_4\}$	$\{dp_2, dp_3, dp_4\}$
$\{\mathbf{t_1}, \mathbf{t_2}\}$	$\{dp_1, dp_5, dp_6\}$
$\{t_1\}$	$\{dp_1, dp_5, dp_6\}$
$\{t_2\}$	$\{dp_1, dp_5, dp_6\}$
$\{\mathbf{t_6}\}$	$\{dp_2, dp_3, dp_4, dp_5\}$

Table 1 lists a set of paragraphs for a given document d, where $PS(d) = \{dp_1, dp_2, \ldots, dp_6\}$, and duplicate terms are removed. Let $min\_sup = 3$ giving rise to ten frequent patterns which are illustrated in Table 2. Normally not all frequent patterns are useful [24,25]. For example, pattern $\{t_3, t_4\}$ always occurs with term t_6 in paragraphs (see Table 1); therefore, we want to keep the larger pattern only.

Given a set of paragraphs $Y \subseteq PS(d)$, we can define its *termset*, which satisfies

$$termset(Y) = \{t | \forall dp \in Y \Rightarrow t \in dp\}.$$

Let $Cls(X) = termset(coverset(X))$ be the closure of X. We call X *closed* if and only if $X = Cls(X)$.

Let X be a closed pattern. We have

$$sup_a(X_1) < sup_a(X) \tag{1}$$

for all pattern $X_1 \supset X$.

3.2 Closed Sequential Patterns

A sequential pattern $s =< t_1, \ldots, t_r >$ $(t_i \in T)$ is an ordered list of terms. A sequence $s_1 =< x_1, \ldots, x_i >$ is a sub-sequence of another sequence $s_2 =< y_1, \ldots, y_j >$, denoted by $s_1 \sqsubseteq s_2$, iff $\exists j_1, \ldots, j_i$ such that $1 \leq j_1 < j_2 \ldots < j_i \leq j$ and $x_1 = y_{j_1}, x_2 = y_{j_2}, \ldots, x_i = y_{j_i}$. Given $s_1 \sqsubseteq s_2$, we usually say s_1 is a sub-pattern of s_2, and s_2 is a super-pattern of s_1. In the following, we simply say patterns for sequential patterns.

Given a pattern (an ordered *termset*) X in document d, $\ulcorner X \urcorner$ is still used to denote the covering set of X, which includes all paragraphs $ps \in PS(d)$ such that $X \sqsubseteq ps$, i.e., $\ulcorner X \urcorner = \{ps | ps \in PS(d), X \sqsubseteq ps\}$. Its *absolute support* and *relative support* are defined as the same as for the normal patterns.

A sequential pattern X is called *frequent pattern* if its relative support $\geq$ *min_sup*, a minimum support. The property of closed patterns can be used to define closed sequential patterns. A frequent sequential pattern X is called *closed* if not $\exists$ any super-pattern X_1 of X such that $sup_a(X_1) = sup_a(X)$.

4 Acquiring User Information Needs

Extracting high-level patterns from text documents would help us to focus on interesting topics in positive (relevant) feedback documents. In this paper, a feature's specificity describes the extent to which the feature focuses on interesting topics.

4.1 Two Levels of Features

Term-based user profiles are considered the most mature theories for term weighting to emerge over the last couple decades in information retrieval. A term-based model is based on the bag of words, which uses terms as elements and evaluates term weights based on terms' appearance or distribution in documents. This main drawback is that the relationship among words cannot be depicted [20]. Another problem in considering single words as features is semantic ambiguities, such as synonyms and polysemy. To overcome the limitations of term-based approaches, pattern mining-based techniques are used for information filtering systems, as patterns are less ambiguous and more discriminative than individual terms; but, pattern-based approaches suffer from low frequency problem.

To improve the efficiency of pattern taxonomy mining, an algorithm, *SP-Mining*(D^+, min_sup) [24], was proposed to find closed sequential patterns in paragraphes for all documents $\in D^+$ that used the well-known *Apriori* property to reduce searching space. For all positive documents $d \in D^+$, the *SPMining* algorithm discovered all closed sequential patterns based on a given *min_sup*.

Let SP_1, SP_2, ..., SP_n be the sets of discovered closed sequential patterns for all document $d_i \in D^+ (i = 1, \cdots, n)$, where $n = |D^+|$). All possible candidates of specific terms can be obtained from all SP_i $(i = 1, \cdots, n)$ as follows:

$$T = \bigcup_{i=1}^{n} \{t | t \in p, p \in SP_i\}$$

4.2 Specificity of Low-level Features

We assume a topic is a set of terms. We call a topic interesting if it is a closed sequential pattern. Usually large number of topics can be extracted from feedback documents, and there are overlaps among many topics. It is very difficult to determine which topic are useful to describe user information needs. Moreover, the user can be interested in one or many topics. Thus, in this paper we define the specificity of a term t based on the specificity of topics that contain t (ST); its distribution in topics (or term's frequency in patterns, TFP) and its appearance in documents (or called document frequency, DF)

The specificity of any given term t to topics ST can be measured by the topics size and the terms distribution in topics that contain t. The topic that contains more terms is unlikely used by other irrelevant documents, and then it is more likely a specific topic for user information needs. Thus, a large topic that contains term t is more important than a short topic. Moreover, the relative support of topics is significant for measure the term's specificity. Therefore, the specificity of a term to topics ST can be calculated as follows:

$$ST(t) = \sum_{i=1}^{n} \sum_{t \in p \subseteq SP_i} sup_r(p, d_i) \times |p|$$

where $sup_r(p, d_i)$ is the relative support of pattern p in document d_i and n is the number of positive feedback documents.

All positive documents in the user feedback describe user information needs. In other words, terms that appear in all positive documents are likely specific features. For example, if the user seeks information about Unicef, we expect the keyword Unicef appear in all positive feedback documents; however, many feedback documents contain noisy terms. To reduce noisy terms, the terms frequency will be calculated only according to their appearances in the extracted topics rather than in documents.

Based on the above analysis, in this paper, we propose the following equation to calculate the specificity of term t for all $t \in T$:

$$\begin{aligned} spe(t) &= TFP(t) \times DF(t) \times ST(t) \\ &= \left(\sum_{i=1}^{n} \sum_{t \in p \subseteq SP_i} sup_a(p, d_i) \right) \times \left(\frac{|coverage(t, D^+)|}{|D^+|} \right) \times \\ &\quad \left(\sum_{j=1}^{n} \sum_{t \in p \subseteq SP_j} \big(sup_r(p, d_j) \times |p| \big) \right) \\ &= \tfrac{1}{n} |coverage(t, D^+)| \times \left(\sum_{i=1}^{n} \sum_{t \in p \subseteq SP_i} sup_a(p, d_i) \right) \times \\ &\quad \left(\sum_{j=1}^{n} \sum_{t \in p \subseteq SP_j} \big(sup_r(p, d_j) \times |p| \big) \right) \\ &= \tfrac{r(t)}{n} \times \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\sum_{t \in p \subseteq SP_i} sup_a(p, d_i) \times \sum_{t \in q \subseteq SP_j} \big(sup_r(q, d_j) \times |q| \big) \right) \end{aligned}$$

where, $sup_a(p, d_i)$ is the frequency of patterns that contain term t in document d; $r(t) = |coverage(t, D^+)|$ is the number of positive documents that contain terms t; and n is the total number of positive documents.

For a term t, the higher of its *spe* score is, the more useful for describing the user information needs. The relevance of an incoming document d to the user information needs can be evaluated using the following ranking function:

$$rank(d) = \sum_{t \in T} spe(t) \tau(t, d)$$

where $\tau(t, d) = 1$ if $t \in d$; $\tau(t, d) = 0$ otherwise.

5 Evaluation

In this paper, we conduct binary text classification to test the proposed approach. We use routing filtering to avoid the need for threshold tuning, which is beyond our research scope. The proposed model in this paper is called Specific Feature Discovery (SFD). The SFD model uses positive relevance feedback to build user profiles. Unlike other models, it uses only positive feedback for selecting useful features.

According to Buckley and others [3], 50 topics are adequate to make a stable, high quality experiment. This evaluation used the 50 expert-designed topics in Reuters Corpus Volume 1 (RCV1) [21]. RCV1 corpus consists of 806,791 documents produced by Reuter's journalists. The document collection is divided into training sets and test sets. These topics were developed by human assessors of the National Institute of Standards and Technology (NIST). The documents are treated as plain text documents by preprocessing the documents. The tasks of removing stop-words according to a given stop-words list and stemming term by applying the Porter Stemming algorithm are conducted.

5.1 Baseline Models and Setting

The main baseline models were the well-known term-based methods: Rocchio, BM25 and SVM. The Rocchio algorithm [17] has been widely adopted in the areas of text categorization and information filtering. It can be used to build a profile for representing the concept of a topic which consists of a set of relevant (positive) and irrelevant (negative) documents. we set $\alpha = \beta = 1.0$ in this paper.

BM25 [16] is one of state-of-the-art term-based models. The values of k_1 and b are set as 1.2 and 0.75, respectively, in this paper.

Information filtering can also be regarded as a special form of text classification [19]. SVM is a statistical method that can be used to find a hyperplane that best separates two classes. SVM achieved the best performance on the Reuters-21578 data collection for document classification [28]. The decision function in SVM is defined as:

$$h(x) = sign(w \cdot x + b) = \begin{cases} +1 & \text{if } (w \cdot x + b) > 0 \\ \\ -1 & \text{otherwise} \end{cases}$$

where x is the input object; $b \in \Re$ is a threshold and $w = \sum_{i=1}^{l} y_i \alpha_i x_i$ for the given training data: $(x_i, y_i), ..., (x_l, y_l)$, where $x_i \varepsilon \Re^n$ and $y_i = +1(-1)$, if document x_i is labelled positive (negative). $\alpha_i \in \Re$ is the weight of the sample x_i and satisfies the constraint:

$$\forall_i : \alpha_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2}$$

To compare with other baseline models, SVM was used to rank documents rather than to make binary decisions. For this purpose, threshold b was ignored. For the documents in a training set, we knew only what were positive (or negative), but not which one was more important. To avoid this bias, we assigned the same α_i value (i.e., 1) to each positive document first, and then determined the same α_i (i.e., $\acute{\alpha}$) value to each negative document based on Eq.(2). Therefore, we used the following weighting function to estimate the similarity between a testing document and a given topic:

$$weight(d) = w \cdot d$$

where $\cdot$ means *inner product*; d is the term vector of the testing document; and

$$w = (\sum_{d_i \in D^+} d_i) + (\sum_{d_j \in D^-} d_j \acute{\alpha}).$$

For each topic, we also chose 150 terms in the positive documents, based on *tf*idf* values for all term-based baseline models.

5.2 Evaluation Measures

Precision p and recall r are suitable because the complete classification is based on the positive class. In order to evaluate the effectiveness of the proposed SFD method, we utilized a variety of existing methods; Mean Average Precision (MAP),*breakeven points(b/p)*, the precision of *top-20* returned documents, *F-scores* and recall at 11-points (IAP). These methods have been widely used to evaluate the performance of information filtering system.

A statistical method, t-test, was also used to analyse the experimental results. The t-test assesses whether the means of two groups are statistically different from each other. If the p-value associated with t is significantly low (<0.05), there is evidence to reject the null hypothesis, and the difference in means across the paired observations is significant.

In summary, the effectiveness is measured by five different means: the average precision of the top 20 documents, F_1 measure, Mean Average Precision (MAP), the break-even point (b/p), and Interpolated Average Precision (IAP) on *11-points.* The larger their values are, the better the system performs.

5.3 Results

We compared the proposed method, SFD, with baseline models, including Rocchio, BM25, and SVM. The experimental results for all 50 assessing topics are reported in Table 3, with the percentage changes *%chg*. The percentage changes *%chg* of the proposed SFD model were compared with the performance of the best baseline model (Rocchio). The SFD model outperforms all the baseline models, including the deployment of sequential closed patterns without using the specificity score (Seq. Cls). The average percentage of improvement over the

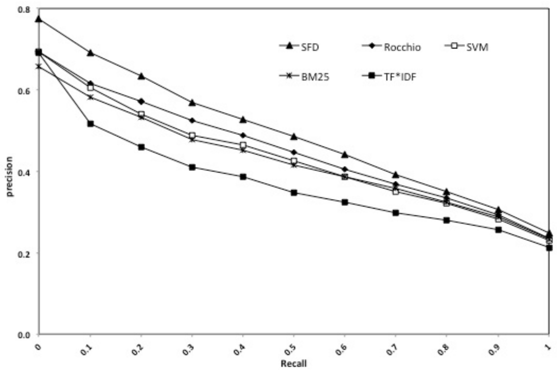


Fig. 1. Comparison of the results in all assessing topics

Table 3. Detailed comparisons of all models in all assessing topics

	top-20	*MAP*	$F_{\beta=1}$	*b/p*	*IAP*
SFD	**0.543**	**0.473**	**0.456**	**0.460**	**0.496**
Seq.Cls*	0.496	0.444	0.439	0.430	0.464
Rocchio	0.474	0.431	0.431	0.420	0.452
SVM	0.453	0.409	0.421	0.408	0.435
BM25	0.445	0.407	0.414	0.407	0.428
%chg	+12.71%	+9.05%	+ 5.70%	+8.59%	+8.84%

* Applying Deploying method to sequential closed patterns without weight revision.

standard measures is 8.98%, with a maximum of 12.71% and minimum 5.70% compared with the best results in Table 3.

The improvements are consistent and very significant on all five measures, as shown by *11-points* on all 50 assessing topics in Figure 1. The *t-test p* values in Table 4 indicate the significance of improvements in the SFD model statistically. Therefore, we conclude the SFD model is an exciting achievement in discovering high-quality features in text documents because it uses high-level patterns to get low-level terms and revises low-level terms based on specificity and distributions in positive relevance feedback.

5.4 Discussion

Generally, term-based approaches extract many terms from documents without considering terms relationships. The advantage of using patterns is that they carry more semantic information than single terms–but these suffer from low frequency [10]. Based on that observation, we used the patterns in this paper to consider the relationship among terms to reduce the extracted noise terms in features extracted from documents. As shown in Table 5, the number of extracted patterns is about 202 patterns with an average length of 2 terms in patterns.

Table 4. T-Test p-values for all models compared with the SFD model in all assessing topics

	top-20	*MAP*	$F_{\beta=1}$	*b/p*	*IAP*
Rocchio	0.02621	0.03650	0.05762	0.08629	0.02868
SVM	0.00269	0.00122	0.00658	0.02094	0.00135
BM25	0.00516	0.00424	0.00645	0.03155	0.00206

Table 5. Patterns statistical information for the proposed model

	$\|SP\|$	*Average length of P*	$\|T\|$	sup_r
50 Topics	202	2	156	86.393

Table 6. Statistical information for the proposed model

$\|D^+\|$	*Average No. terms*	*terms weight in topics*	*spe*
13	156	202.788	250.570

From that information, we expected about $404 = 202*2$ terms in all patterns. But the actual number of terms deployed from those patterns are 156 terms, which indicated that about $61.37 = \frac{404-156}{404}$ of the patterns overlap. This overlapping from the closed pattern indicates some terms importance in the documents.

As shown in Table 5, the average weight of patterns for each topic is 86.393 distributed into 202 patterns on average, which gave about $0.428 = \frac{86.393}{202}$. On the other hand, Table 6 shows a weight of about 202.788 distributed in 156 terms on average. That information indicates a weight of about $57.40\% = \frac{202.788-86.393}{202.788}$ is increased according to terms from the deploying method.

Using common sense, we know that positive terms with large *specificity* are more interesting than general terms with less *specificity* for a given topic. However, evaluating the specificity of a given term is challenging. The proposed model calculates the specificity of terms based on the specificity of each term to the topic ST, distribution of terms in topic TFP, and appearance of terms in the documents DF. Unlike other models, in this paper, specific terms appear in most positive patterns in positive documents. As shown in Table 6, before revision, 202.788 was distributed to all positive terms as weights; the *spe* increased the weight of terms to 250.570. The percentage of increase is $19.07\% = \frac{250.570-202.788}{250.570}$. However, the amount of increase differs for each term.

6 Conclusions

It has been proven that pattern-based approaches are useful for improving the quality of feature selection from text documents, although they suffer from low frequency. To solve that problem, deploying methods have been proposed to deploy high-level patterns into low-level terms. However, these deploying methods cause many low-level terms to have the same weight regardless of a terms specificity. The proposed SFD approach utilizes term distribution in high-level patterns (topics) and terms document frequency to calculate terms specificity according to their appearances and distribution in topics. The experimental results on RCV1 demonstrate that the proposed method has performed excitingly, with an average 8.98% improvement over the state-of-the-art benchmarks.

Acknowledgments. This paper was partially supported by Grant DP0988007 from the Australian Research Council (ARC Discovery Project).

References

1. Aas, K., Eikvil, L.: Text categorisation: A survey. Technical report, Norwegian Computing Center (June 1999)
2. 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), pp. 2–11 (1998)
3. Buckley, C., Voorhees, E.M.: Evaluating evaluation measure stability. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 33–40 (2000)
4. Cao, G., Nie, J.-Y., Gao, J., Robertson, S.: Selecting good expansion terms for pseudo-relevance feedback. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 243–250 (2008)
5. Dumais, S.T.: Improving the retrieval of information from external sources. Behavior Research Methods, Instruments, & Computers 23(2), 229–236 (1991)
6. Jindal, N., Liu, B.: Identifying comparative sentences in text documents. In: Proceedings of SIGIR 2006, pp. 244–251 (2006)
7. Joachims, T.: A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 143–151. Morgan Kaufmann Publishers Inc. (1997)
8. Lewis, D.D.: An evaluation of phrasal and clustered representations on a text categorization task. In: Proceedings of SIGIR 1992, pp. 37–50 (1992)
9. Li, X., Liu, B.: Learning to classify texts using positive and unlabelled data. In: Proceedings of IJCAI 2003, pp. 587–594 (2003)
10. Li, Y., Algarni, A., Zhong, N.: Mining positive and negative patterns for relevance feature discovery. Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 753–762 (2010)
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. Li, Y., Zhou, X., Bruza, P., Xu, Y., Lau, R.Y.: A two-stage text mining model for information filtering. In: Proceeding of the 17th ACM Conference on Information and Knowledge Management, pp. 1023–1032 (2008)
13. Ling, X., Mei, Q., Zhai, C., Schatz, B.: Mining multi-faceted overviews of arbitrary topics in a text collection. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 497–505 (2008)

14. Metzler, D., Croft, W.B.: Latent concept expansion using markov random fields. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 311–318 (2007)
15. Pon, R.K., Cardenas, A.F., Buttler, D., Critchlow, T.: Tracking multiple topics for finding interesting articles. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 560–569 (2007)
16. Robertson, S.E., Soboroff, I.: The trec 2002 filtering track report. In: Proceedings of TREC (2002)
17. Salton, G.: The SMART Retrieval System-Experiments in Automatic Document Processing. Prentice-Hall, Inc., Upper Saddle River (1971)
18. Scott, S., Matwin, S.: Feature engineering for text classification. In: The 16th International Conference on Machine Learning, pp. 379–388 (1999)
19. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. 34(1), 1–47 (2002)
20. Shen, D., Sun, J.-T., Yang, Q., Zhao, H., Chen, Z.: Text classification improved through automatically extracted sequences. In: Proceedings of the 22nd International Conference on Data Engineering, pp. 121–123. IEEE Computer Society (2006)
21. Soboroff, I., Robertson, S.: Building a filtering test collection for trec 2002. In: Proceedings of SIGIR 2003, pp. 243–250 (2003)
22. Wang, X., Fang, H., Zhai, C.: A study of methods for negative relevance feedback. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 219–226 (2008)
23. Wu, S.-T., Li, Y., Xu, Y.: Deploying approaches for pattern refinement in text mining. In: Proceedings of ICDM 2006, pp. 1157–1161 (2006)
24. Wu, S.-T., Li, Y., Xu, Y., Pham, B., Chen, P.: Automatic pattern-taxonomy extraction for web mining. In: Proceedings of WI 2004, pp. 242–248 (2004)
25. Xu, Y., Li, Y.: Generating concise association rules. In: Proceedings of CIKM 2007, pp. 781–790 (2007)
26. Xu, Z., Akella, R.: Active relevance feedback for difficult queries. In: Proceeding of the 17th ACM Conference on Information and Knowledge Management, pp. 459–468 (2008)
27. Yan, X., Cheng, H., Han, J., Xin, D.: Summarizing itemset patterns: a profile-based approach. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 314–323 (2005)
28. 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, pp. 42–49 (1999)

Ensemble-Based Wrapper Methods for Feature Selection and Class Imbalance Learning

Pengyi Yang[1,3], Wei Liu[2], Bing B. Zhou[1],
Sanjay Chawla[1], and Albert Y. Zomaya[1]

[1] School of Information Technologies, University of Sydney, NSW 2006, Australia
[2] Dept of Computing and Information Systems, University of Melbourne, Australia
[3] Garvan Institute of Medical Research, Darlinghurst, NSW 2010, Australia
yangpy@it.usyd.edu.au, wei.liu@unimelb.edu.au

Abstract. The wrapper feature selection approach is useful in identifying informative feature subsets from high-dimensional datasets. Typically, an inductive algorithm "wrapped" in a search algorithm is used to evaluate the merit of the selected features. However, significant bias may be introduced when dealing with highly imbalanced dataset. That is, the selected features may favour one class while being less useful to the adverse class. In this paper, we propose an ensemble-based wrapper approach for feature selection from data with highly imbalanced class distribution. The key idea is to create multiple balanced datasets from the original imbalanced dataset via sampling, and subsequently evaluate feature subsets using an ensemble of base classifiers each trained on a balanced dataset. The proposed approach provides a unified framework that incorporates ensemble feature selection and multiple sampling in a mutually beneficial way. The experimental results indicate that, overall, features selected by the ensemble-based wrapper are significantly better than those selected by wrappers with a single inductive algorithm in imbalanced data classification.

1 Introduction

Feature selection is a critical procedure for high-dimensional data classification. The benefits of feature selection are several-fold and dependent on the applications. For creating classification models, feature selection can often improve predictive accuracy and comprehensibility [1]. For many bioinformatics applications, feature selection is a critical procedure for identifying important biomarkers [2].

The techniques for feature selection are commonly classified as filter approach, wrapper approach, and embedded approach. Filter approach and embedded approach are relatively computationally efficient and are commonly applied as a fast feature ranking procedure [3]. In contrast, wrapper approach evaluates features by performing internal classification with a given inductive algorithm [4]. Therefore, they are much more computation intensive. Nevertheless, wrapper approach remains attractive for two reasons. Firstly, wrapper approach evaluates features iteratively with respect to an inductive algorithm. Therefore, features

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 544–555, 2013.

selected by wrapper approach are more likely to suit the inductive algorithm, and therefore, yield high classification accuracy [4]. Secondly, wrapper approach evaluates features jointly and are effective in capturing intrinsic relationships such as interactions among multiple features [5].

Learning from imbalanced data is an important problem in many data mining applications. Such a case arises when samples from one class significantly outnumber those from the other class. Imbalanced data are common in text mining [6] and bioinformatics where the minority class often represents the rare cases. It is well known that many classification algorithms are sensitive to the imbalanced class distribution [7]. Therefore, many strategies have been proposed to deal with class imbalance learning. Generally, they fall into two categories: cost-sensitive learning and data sampling [8]. With cost-sensitive learning, a given algorithm will receive a higher penalty when a mistake is made on the minority class than on the majority class. The advantage of cost-sensitive learning is that it does not modify the class distribution. However, an accurate cost-metric needs to be specified beforehand. As for data sampling, the learning instances in the majority class and minority class are manipulated in certain way so as to balance the class distribution. The downside is that sampling strategies may introduce noise or remove useful information while modifying class distribution.

The challenges of feature selection and imbalanced data classification meet when the dataset to be analysed is of both high-dimensionality and highly imbalanced class distribution [9]. In such a scenario, if wrapper approach is adopted for feature selection, the inductive algorithm may introduce significant bias because the merit of the feature subset is evaluated based on the performance of the inductive algorithm. Therefore, if the inductive algorithm favours a single class, the features selected will also bias to this class while being less useful to the adverse class.

In this study, we propose an ensemble-based wrapper approach for feature selection from highly imbalanced datasets. The proposed algorithm retains the advantages of wrapper feature selection while also maximises data usage and reduce feature selection bias simultaneously by training multiple base classifiers with balanced sample subsets. A hybrid multiple sampling procedure is employed to create balanced sample subsets. Together we introduce a unified framework that incorporates ensemble feature selection and multiple sampling in a mutually beneficial manner.

The paper is organised as follows. In Section 2, we outline the proposed framework and describe each component in details. Section 3 describes the experimental procedure. Results are presented in Section 4 and Section 5 concludes the paper.

2 Ensemble-Based Wrapper Approach

Wrapper algorithms, in general, consist of three main components [10]: (1) a search algorithm, (2) a fitness function, and (3) an inductive algorithm. The proposed system adheres to this structure. In this section, we outline the system and describe each component.

2.1 System Overview

A schematic representation of the proposed ensemble-based wrapper approach is shown in Figure 1. The imbalanced training dataset is balanced by a hybrid sampling approach (which will be explained in Section 2.3). Such a hybrid sampling procedure is applied multiple times producing multiple sets of balanced training data each of which is used to train a base classifier. The base classifiers trained on the balanced datasets are subsequently applied to classify an imbalanced test dataset. The classification distributions of each sample in the test dataset are normalised and combined, and the area under ROC curve (AUC) is calculated as the fitness indices for feature selection. The wrapper procedure terminates when it reaches a predefined number of iteration or a desired number of features is selected (i.e. greedy search), and the final feature subsets are used for further validation.

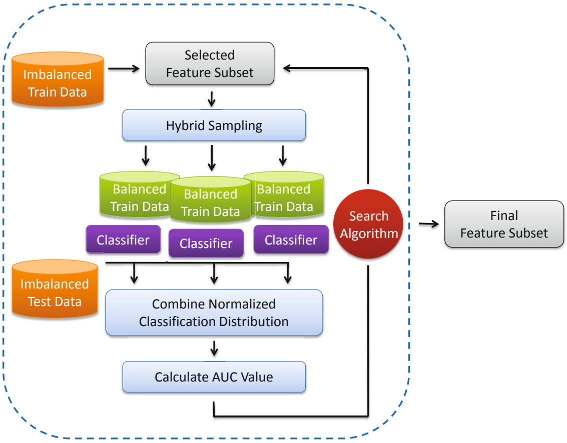


Fig. 1. A schematic representation of the ensemble-based wrapper approach

2.2 Search Algorithm

There are several popular search strategies, including hill climbing algorithms best exemplified by forward selection and backward elimination [11,12] and evolutionary algorithms such as genetic algorithm [13] and particle swarm optimisation [14].

In this study, we apply two search algorithms. The first one is a hill climbing algorithm that starts with an empty set and greedily selects a feature at a time that maximises the given fitness function. This is a typical greedy forward selection approach and at each step the best feature f^* is determined by:

$$f^* = \arg\max_{f \notin \mathbf{S}} fitness(\mathbf{S} \cup \{f\})$$

where $\mathbf{S}$ is the set that contains the features selected so far and f is a feature under evaluation according to a fitness function.

The second search algorithm is a simple elitism genetic algorithm. The feature size is pre-specified and the algorithm selects the best feature set that maximises the given fitness function through genetic operations such as crossover and mutation. Here each feature in the best set $\mathbf{S}^*$ is determined simultaneously:

$$\mathbf{S}^* = \arg\max_{i=1\ldots p} fitness(\mathbf{S}_i)$$

where p is the population size of the genetic algorithm.

The above two typical yet simple wrapper procedures offer a transparent way to compare different inductive components.

2.3 Hybrid Sampling from Imbalanced Data

Sampling is a popular approach to balance the dataset with imbalanced class distribution. The simplest methods are random under-sampling and random over-sampling [15]. The random under-sampling method balances the dataset by randomly removing samples in the majority class. On the contrary, the random over-sampling method balances the dataset by sampling from the minority class with/without replacement and reattaching them to the dataset. A more sophisticated approach is to synthesise "new" samples from the minority class (known as SMOTE) [16]. Several studies also found that better results can be achieved by increasing minority samples and decreasing majority samples simultaneously [17,18].

Here we apply our own hybrid approach in which the dataset (denoted as $\mathbf{D}$) is balanced by increasing minority class with SMOTE and decreasing majority class with random under-sampling as follows:

$$\begin{aligned} I_R &= Random(I_{maj},\ (N_{maj} - 3/2 \times N_{min})) \\ I_S &= SMOTE(I_{min},\ 1/2 \times N_{min}) \\ \mathbf{D}^* &= (I_{min} \cup I_S) \cup (I_{maj} \backslash I_R) \end{aligned}$$

where I_{maj}, I_{min}, N_{maj} and N_{min} are the majority samples, minority samples, and their sample sizes, respectively. $Random(.)$ randomly selects from I_{maj} a subset of samples I_R and $SMOTE(.)$ creates synthetic samples I_S using I_{min}. The balanced dataset $\mathbf{D}^*$ retains the original minority samples and introduces $1/2 \times N_{min}$ synthetic minority samples. The majority samples I_R are reduced to match the new set of minority samples in $\mathbf{D}^*$ and result in a class ratio of 1.

2.4 Ensemble Learning

The classic idea of ensemble is to generate multiple datasets using a sampling method such as bootstrap, and train a set of homogeneous learning algorithms which classify new instances in a consensus manner [19]. This idea has been extended both to imbalanced data classification [20] and feature filtering [21]. However, no work has been done to unify them as a single procedure which may be mutually beneficial.

Here, we extend the idea of ensemble to feature selection in a wrapper manner and provide a unified framework that incorporates ensemble feature selection as well as multiple sampling. Specifically, given a training dataset constrained by a set of features $\mathbf{S}$, suppose we apply the above hybrid sampling procedure L times, each time producing a balanced sampling dataset $\mathbf{D}_i^{*\mathbf{S}}$ $(i = 1...L)$, and each sampling dataset is used to train a base classifier denoted as h_i. Then, the ensemble classification distribution y of each test sample $\mathbf{x}$ is computed as follows:

$$p^E(y|\mathbf{x}, \mathbf{S}) = \frac{1}{L}\sum_{i=1}^{L} Prob(h_i(\mathbf{x}), \mathbf{D}_i^{*\mathbf{S}})$$

where $Prob(h_i(\mathbf{x}), \mathbf{D}_i^{*\mathbf{S}})$ is a probability vector computed by using an ensemble of L base classifiers (h_i), each is trained on a balanced sampling set $\mathbf{D}_i^{*\mathbf{S}}$ selected by the feature set $\mathbf{S}$. Therefore, both feature set information and data sampling information are incorporated in an ensemble framework.

2.5 Fitness Function

An inductive algorithm (classifier) is commonly used to generate fitness indices in wrapper algorithms. It is well known that the overall accuracy as a metric is biased when the class distribution is imbalanced in the data. A more reliable way to compute the fitness of a feature set in such a case is to use the area under the ROC curve (AUC). AUC is a numeric value summarising the trade-off between the true positive rate and the false positive rate across the entire sample classification distribution of a dataset.

When using a single inductive algorithm, the AUC value is directly calculated by sorting the classification probability of each sample, calculating trade-off value of the true positive rate and false positive rate at each classification threshold, and calculating the area under the trade-off values. As to the ensemble classifier, classification distribution of each sample is combined and normalised across all base classifiers. Then, the same procedure as those for a single inductive algorithm is applied to calculate the AUC value.

Accordingly, we define the fitness of a feature subset as follows:

$$fitness(\mathbf{S}) = AUC(p(y|\mathbf{x}_1, \mathbf{S})...p(y|\mathbf{x}_m, \mathbf{S}))$$

where $\mathbf{x}_1$ is the first sample in the test dataset and m is the total sample size. Function $AUC(.)$ calculates the AUC value.

2.6 Main Algorithm of Ensemble-Based Wrapper Approach

Algorithm 1 represents the core of the ensemble-based wrapper approach in pseudo-code:

Algorithm 1. Ensemble Component

```
Input: A feature subset $\mathbf{S}$; Imbalanced training set $\mathbf{D}_T$ and test set $\mathbf{D}_t$
Output: Fitness of $\mathbf{S}$
$Fit = 0$;
// constrain the data dimension using the input feature subset:
$\mathbf{D}_T^{\mathbf{S}}$ = constrainDataDimension($\mathbf{D}_T$, $\mathbf{S}$);
$E = \emptyset$;
for $i = 1$ to $L$ do
    // Sampling to create a balanced dataset using training set:
    $\mathbf{D}_i^{*\mathbf{S}}$ = hybridSampling($\mathbf{D}_T^{\mathbf{S}}$);
    // Train a base classifier using balanced dataset:
    $h_i$ = trainClassifier($\mathbf{D}_i^{*\mathbf{S}}$);
    // Add the base classifier to the ensemble:
    $\mathbf{E} = \mathbf{E} \cup h_i$;
end for
$\mathbf{D}_t^{\mathbf{S}}$ = constrainDataDimension($\mathbf{D}_t$, $\mathbf{S}$);
// Apply the ensemble of classifiers to the test set:
$Fit$ = calculateAUC($\mathbf{E}$, $\mathbf{D}_t^{\mathbf{S}}$);
return $Fit$;
```

The ensemble component is independent from the search algorithm. It is flexible and can be reused in different wrapper algorithms.

3 Experimental Procedure

In this section, we summarise the datasets used for evaluation and detail the algorithms and parameter settings. Following that, the performance evaluation is described.

3.1 Datasets and Data Partitioning

We used 5 datasets with high-dimensionality and highly imbalanced class distribution. Table 1 summarises the datasets.

Specifically, fbis, re0, and oh5 are text mining datasets extracted by Han and Karypis [22]. The ALL (acute lymphoblastic leukemia) dataset is from a leukemia study [23], and the oil dataset is from study [24].

For datasets with multiple classes, we reserved the class with the smallest number of samples as the minority class and combined the other classes as the majority class. To make the problem computationally less demanding,

Table 1. Summary of datasets

Name	# Sample	# Feature	Minority class ratio
fbis	1250	2000	0.0304
re0	1504	2886	0.0073
oh5	918	3012	0.0643
ALL	248	12626	0.0605
oil	937	50	0.0438

for datasets with very high dimensions, we applied a χ^2 filtering to reduce the feature size to 500.

The datasets are partitioned using the double-level cross-validation strategy. That is for each dataset, we partitioned it using a 2-fold stratified cross-validation to obtain the training and evaluation sets. For the training set, it is further partitioned using a 5-fold stratified cross-validation to obtain the internal training and internal testing sets for feature selection. The evaluation set is reserved from the feature selection procedure and is only used for evaluating the usefulness of the selected features after the feature selection procedure.

3.2 Algorithms and Parameter Settings

For the greedy forward feature selection algorithm, we specified it to search 20 steps in which 1 to 20 features are selected one after an other. As for the genetic algorithm, we set both the population size and the termination generation to 20. The crossover probability and the mutation probability are 0.7 and 0.1, respectively. The "chromosome" is coded as a string of feature indexes, and the chromosome size of 1 to 20 are tested which corresponds to the feature subset size of 1 to 20. Different from the greedy forward feature selection algorithm which builds the feature subset on previously selected features, the genetic algorithm tests different size of feature subsets separately.

The decision tree algorithm (J48) is used for induction in our wrapper algorithms. In ensemble learning, the decision tree algorithm is prevailingly used as the base classifier because it is relatively fast to train and unstable to small changes in the data [25]. These are the important merits to our wrapper algorithms since we need to evaluate features using multiple classifiers in an efficient manner. Yet, it is widely known that the decision tree algorithm is sensitive to the imbalance of the data class distribution [26]. Hence, it is of both theoretical and practical interests to use decision tree in our experimental settings. For the ensemble wrapper, we used the ensemble size of 20. That is 20 different sampling dataset are produced in each iteration and 20 decision tree classifiers are trained on these sampling dataset and then used for feature selection.

To evaluate the selected features, we used 6 different classification algorithms, including random forest (RF), nearest neighbour with k=3 (3-NN), nearest neighbour with k=7 (7-NN), logistic regression (LogReg), multiple layer perceptron (MLP), and alternating decision tree (ADTree). The rationale is that if the wrapper algorithm is able to select useful features, the selected features should be able to improve the classification result regardless what type of classification algorithm is used. Therefore, evaluating a wide range of different classifiers can better reflect the genuine usefulness of the selected features.

3.3 Performance Evaluation

In this study, we focus on comparing wrapper algorithms with ensemble-based imbalanced sampling and classification component to wrapper algorithms with a single inductive algorithm. We refer to the first approach as the ensemble approach and the latter as the single approach. To summarise the performance results, the AUC values obtained from each classifier using features selected by ensemble approach and single approach are compared. If the ensemble approach yields a higher AUC value compared to the single approach, we label it as "ensemble win". Similarly, if the ensemble approach yields a lower AUC value compared to the single approach, we label it as "single win". When the AUC values from these two approaches are equal, we obtain a "tie". The comparison is conducted from feature size 1 to 20.

In addition, the Friedman test [27] is applied to evaluate the performance of each classifier. The confidence of 95% is used under the null hypothesis that the performance of each classifier is not significantly different by using the features selected by the ensemble approach and the single approach. The null hypothesis is rejected if there are significant performance difference when using features selected by ensemble approach as to single approach.

4 Results

AUC comparison of ensemble wrapper and single wrapper using greedy forward feature selection with fbis and re0 dataset are plotted in Figure 2 and Figure 3, respectively. As can be seen, the ensemble wrapper approach exhibited a better performance compared to the single wrapper approach. We summarise results in Figures 2 and 3 and the rest of the comparison across using feature sets with size from 1 to 20 in Table 2 and Table 3 (see 3.3 for details of the summarisation method). Specifically, Table 2 shows the comparison of the ensemble approach and the single approach using greedy forward selection algorithm, and Table 3 shows the comparison using genetic algorithm. It is clear that across all datasets most classifiers achieves better classifications using features selected by ensemble approach than those selected by single approach. This implies that the ensemble approach is more robust to high-dimensionality and highly imbalanced class distribution. Hence, the features selected by the ensemble approach are likely to be more useful to both the majority class and the minority class.

The greedy forward selection appears to be more sensitive to ensemble component. In most cases, the improvements are significant. In comparison, genetic algorithm based selection is less sensitive to the ensemble component, and most improvements are moderate. This may attributed to their different feature selection styles. That is, greedy forward selection builds the feature subset on

Table 2. Comparison of ensemble and single approaches using greedy forward selection

	fbis dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	17	17	20	14	13	15
Single Win	3	3	0	6	7	5
Friedman Test	0.0017 ✓	0.0017 ✓	7.74e-6 ✓	0.073	0.1797	0.0253 ✓
	Re0 dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	20	20	20	20	20	20
Single Win	0	0	0	0	0	0
Friedman Test	7.74e-6 ✓	7.74e-6 ✓	7.74e-6 ✓	7.74e-6 ✓	7.74e-6 ✓	7.74e-6 ✓
	Oh5 dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	12	16	6	15	13	17
Single Win	7	3	13	4	6	2
Tie	1	1	1	1	1	1
Friedman Test	0.251	0.0029 ✓	0.108	0.011 ✓	0.108	5.79e-4 ✓
	ALL dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	15	15	15	17	15	6
Single Win	5	5	5	3	5	14
Friedman Test	0.025 ✓	0.025 ✓	0.025 ✓	0.0017 ✓	0.025 ✓	0.073
	Oil dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	19	7	10	15	18	19
Single Win	1	13	10	5	2	1
Friedman Test	5.69e-5 ✓	0.179	1	0.025 ✓	3.46e-4 ✓	5.69e-5 ✓

✓ Results with significant differences (p-value lower than 0.05) using Friedman test

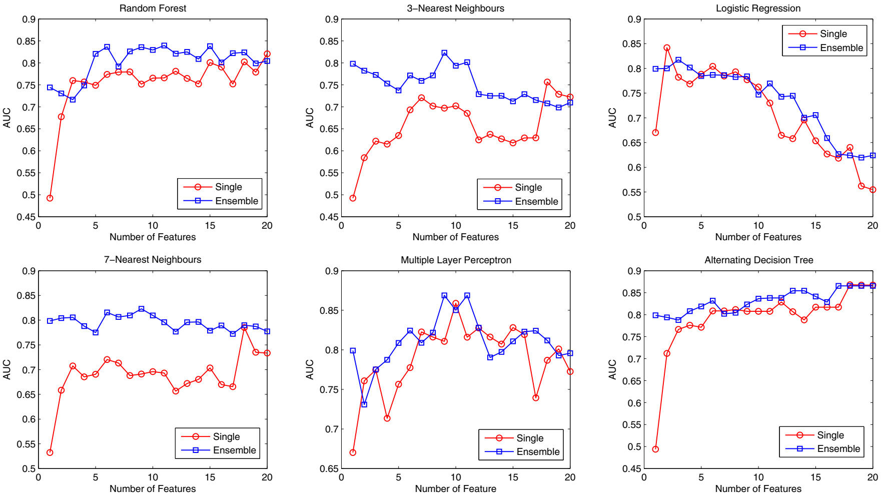


Fig. 2. AUC comparison of ensemble wrapper and single wrapper using greedy forward selection and fbis dataset. The feature size from 1 to 20 selected by ensemble and single wrappers are evaluated by 6 different classification algorithms.

Table 3. Comparison of ensemble and single approaches using genetic algorithm

	fbis dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	16	12	16	13	11	17
Single Win	4	8	4	7	9	3
Friedman Test	0.0073 ✓	0.3711	0.0073 ✓	0.1797	0.654	0.0017 ✓
	Re0 dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	13	16	18	15	11	15
Single Win	7	4	2	5	9	5
Friedman Test	0.1797	0.0073 ✓	3.46e-4 ✓	0.025 ✓	0.654	0.025 ✓
	Oh5 dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	11	12	12	14	12	11
Single Win	9	8	8	6	8	9
Friedman Test	0.654	0.3711	0.3711	0.1797	0.3711	0.654
	ALL dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	12	16	16	13	17	16
Single Win	8	4	4	7	3	4
Friedman Test	0.3711	0.0073 ✓	0.0073 ✓	0.1797	0.0017 ✓	0.0073 ✓
	Oil dataset					
	RF	3-NN	7-NN	LogReg	MLP	ADTree
Ensemble Win	12	15	12	13	19	15
Single Win	8	5	8	7	1	5
Friedman Test	0.3711	0.025 ✓	0.3711	0.179	5.69e-5 ✓	0.025 ✓

✓ Results with significant differences (p-value lower than 0.05) using Friedman test.

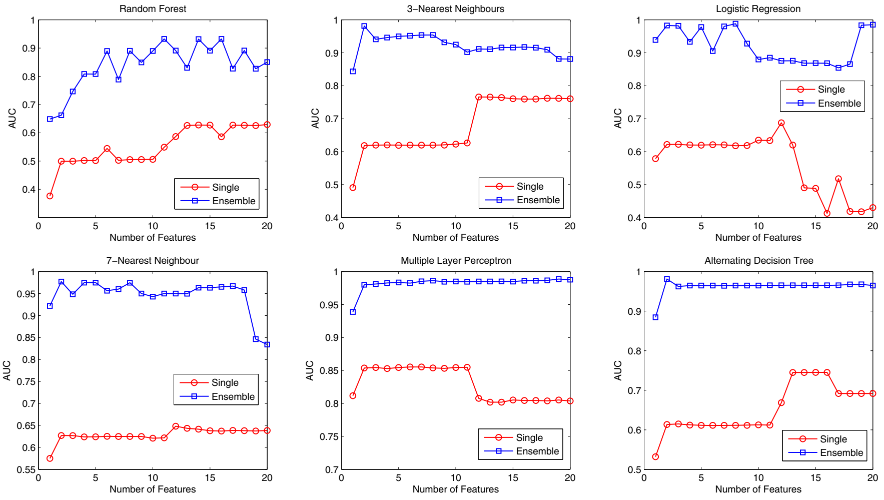


Fig. 3. AUC comparison of ensemble wrapper and single wrapper using greedy forward selection and re0 dataset. The feature size from 1 to 20 selected by ensemble and single wrappers are evaluated by 6 different classification algorithms.

previously selected features. Therefore, if a good feature is selected, it will continually be used in later iterations. Whereas, the genetic algorithm tries different size of feature subsets separately, and for each run the initiation, crossover, and mutation operations introduces randomness to the selection procedure. It follows that the greedy forward selection approach is likely to aggregate the effect of the ensemble through iterations, while the genetic algorithm approach may reduce the effect of the ensemble due to its stochastic behaviour.

It is interesting to see that different classification algorithms performed differently even with the same set of features. For the extreme case, in Table 2 the classification results of 7-NN on oh5 dataset and 3-NN and 7-NN on oil dataset contradict to the rest of the classifiers. Even for classifiers with similar comparison results, each of them may still behave differently throughout the feature subset size of 1 to 20. For example, in Figure 2, RF shows an increasing trend when more features are added. However, LogReg and ADTree indicate a decreasing trend when more features are included, whereas the performance of MLP increases first and then decreases. Note that the same sets of features and the same evaluation dataset are used for each classification algorithm. Therefore, it is clear that using a single classification algorithm for results evaluation is insufficient. Instead, multiple classification algorithms should be evaluated in order to reflect the general usefulness of the selected features.

5 Conclusion

In this study, we proposed an ensemble approach that incorporate feature selection and imbalanced data sampling in a wrapper framework. Using two search algorithms and several high-dimensional and highly imbalanced datasets, we demonstrated that features selected by the ensemble-based wrapper approach are more useful than the traditional approach (i.e. using single inductive algorithm) in terms of feature selection and imbalance learning. This implies that the traditional approach that uses a single inductive algorithm for feature evaluation may perform suboptimally when the dataset is of both high-dimensionality and highly imbalanced class distribution. By designing a multiple sampling and an ensemble feature evaluation components, we can correct the undesirable bias and identify more useful features and/or feature subsets.

References

1. Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on Knowledge and Data Engineering, 491–502 (2005)
2. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23(19), 2507–2517 (2007)
3. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. Artificial Intelligence 97(1-2), 245–271 (1997)
4. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
5. Freitas, A.: Understanding the crucial role of attribute interaction in data mining. Artificial Intelligence Review 16(3), 177–199 (2001)
6. Tang, L., Liu, H.: Bias analysis in text classification for highly skewed data. In: Proceedings of the Fifth IEEE International Conference on Data Mining, pp. 784–787 (2005)

7. He, H., Garcia, E.: Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 1263–1284 (2008)
8. Batista, G., Prati, R., Monard, M.: A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explorations Newsletter 6(1), 20–29 (2004)
9. Mladenic, D., Grobelnik, M.: Feature selection for unbalanced class distribution and naive bayes. In: Proceedings of the Sixteenth International Conference on Machine Learning, pp. 258–267 (1999)
10. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. The Journal of Machine Learning Research 3, 1157–1182 (2003)
11. Caruana, R., Freitag, D.: Greedy attribute selection. In: Proceedings of the Eleventh International Conference on Machine Learning, pp. 28–36 (1994)
12. Kudo, M., Sklansky, J.: Comparison of algorithms that select features for pattern classifiers. Pattern Recognition 33(1), 25–41 (2000)
13. Oh, I., Lee, J., Moon, B.: Hybrid genetic algorithms for feature selection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1424–1437 (2004)
14. Wang, X., Yang, J., Teng, X., Xia, W., Jensen, R.: Feature selection based on rough sets and particle swarm optimization. Pattern Recognition Letters 28(4), 459–471 (2007)
15. Japkowicz, N., Stephen, S.: The class imbalance problem: A systematic study. Intelligent Data Analysis 6(5), 429–449 (2002)
16. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: synthetic minority oversampling technique. Journal of Artificial Intelligence Research 16(1), 321–357 (2002)
17. Estabrooks, A., Jo, T., Japkowicz, N.: A multiple resampling method for learning from imbalanced data sets. Computational Intelligence 20(1), 18–36 (2004)
18. Khoshgoftaar, T., Seiffert, C., Van Hulse, J.: Hybrid Sampling for Imbalanced Data. In: Proceedings of IRI, pp. 202–207 (2008)
19. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
20. Li, C.: Classifying imbalanced data using a bagging ensemble variation (BEV). In: Proceedings of the 45th Annual Southeast Regional Conference, pp. 203–208 (2007)
21. Saeys, Y., Abeel, T., Van de Peer, Y.: Robust feature selection using ensemble feature selection techniques. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 313–325. Springer, Heidelberg (2008)
22. Han, E.-H(S.), Karypis, G.: Centroid-Based Document Classification: Analysis and Experimental Results. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 424–431. Springer, Heidelberg (2000)
23. Yeoh, E., Ross, M., Shurtleff, S., Williams, W., Patel, D., Mahfouz, R., Behm, F., Raimondi, S., Relling, M., Patel, A., et al.: Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. Cancer Cell 1(2), 133–143 (2002)
24. Kubat, M., Holte, R., Matwin, S.: Machine learning for the detection of oil spills in satellite radar images. Machine Learning 30(2), 195–215 (1998)
25. Dietterich, T.: An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine Learning 40(2), 139–157 (2000)
26. Liu, W., Chawla, S., Cieslak, D., Chawla, N.: A robust decision tree algorithms for imbalanced data sets. In: Proceedings SIAM International Conference on Data Mining, pp. 766–777 (2010)
27. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7, 1–30 (2006)

Exploring Groups from Heterogeneous Data via Sparse Learning

Huawen Liu[1,3], Jiuyong Li[2], Lin Liu[2], Jixue Liu[2],
Ivan Lee[2], and Jianmin Zhao[1]

[1] Zhejiang Normal University, Jinhua 321004, China
[2] University of South Australia, Adelaide, SA5095, Australia
[3] Academy of Mathematics and Systems Science, CAS, Beijing 100190, China

Abstract. Complexity networks, such as social networks, biological networks and co-citation networks, are ubiquitous in reality. Identifying groups from data is critical for network analysis, for it can offer deep insights in understanding the structural properties and functions of complex networks. Over the past decades, many endeavors from interdisciplinary fields have been attempted to identify groups from data. However, little attention has been paid on exploring groups and their relationships from different views. In this work, we address this issue by using canonical correlation analysis (CCA) to analyze groups and their interplays in the networks. To further improve the interpretability of results, we solve the optimization problem with sparse learning, and then propose a generalized framework of group discovery from heterogeneous data. This framework enables us to find groups and explicitly model their relationships from diverse views simultaneously. Extensive experimental studies conducted on both synthetic and DBLP datasets demonstrate the effectiveness of the proposed method.

Keywords: Group discovery, complexity network, canonical correlation analysis, LASSO, Sparse learning.

1 Introduction

It is well known that everything in the universe is relevant to others and nothing independently exists. According to different functions and properties, entities intertwined with each other form groups, resulting in complex networks [14]. Typical examples include functional regulation models of proteins in biology [10], trophic pathways of species in ecology [3,8], communities of people in sociology [16], interlinks of web pages in World Wide Web [5], collaboration relationships of authors in bibliography [11], and many others. Since complex networks are ubiquitous in reality, network learning now gains increasing attentions from a variety of disciplines including computer science, physics, economics, business marketing, biology, engineering, epidemiology, social and behavioral science [4].

Exploring the structures, functions, as well as the interactions of networks is very important, because it may provide us an insightful understanding how the

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 556–567, 2013.

networks work [16,4,20]. For example, identifying collaboration relationships of scientists provides us an indication to which topics are popular and what the research trends will be potentially studied in future [11]; uncovering functional modules of proteins helps us understand which causes lead to certain diseases and how the proteins are co-regularized [10]; revealing social communities of people aids us in finding out what the interests and opinions of people are [23].

An essential yet challenging task of network learning is to discover functional units in complex networks [4]. In literature, the functional unit is also known as community, module, clique, cluster or coalition, depending on the specific contexts or applications at hand [16]. These terms cover the entities only. Here we call the functional unit as *group*, for the interplays between the entities have also been taken into account. Since group discovery is helpful to analyze the network structures and further capture knowledge about the functions and properties of the network systems, it is not surprised that it has attracted many attentions from different domains [20,17,19].

Over the last decades, group discovery has been extensively investigated and dozens of discovery methods, including graph partition models, clique based models, clustering models, modularity maximization models, and so on, have been developed [4]. However, most of them place emphases merely on identifying function units. To the best of our knowledge, the interrelations of groups, which are very common in nature, have not been fully exploited. In some cases, identifying the interrelations is more crucial to understand the structural and functional properties, for it may offer us an insightful perspective to the networks.

Another issue is that the existing methods only take the entities with the same type into consideration and ignore other information with different types. Empirical studies show that the network structures in real world are complex and often involve the entities derived from heterogeneous data sources. The information with different types may bring benefits to explore and analyze the complex networks if it had been taken into account in network learning.

In this paper, we present a generalized framework to explicitly address the problems mentioned above. It adopts canonical correlation analysis (CCA) to analyze groups and their interrelations simultaneously. It not only allows to handle the groups with different types, but also provides an effective solution to scale their interrelations in a quantitative manner. Furthermore, we turn the objective function of CCA into a LASSO penalized least square problem (i.e., ℓ_1-norm penalty) by using complex linear algebra equivalent transformations. Consequently, the proposed method can obtain an optimal sparse solution for large-scale complex networks. Specifically, the contributions of this work are twofold: 1) Our method can handle the entities from the networks with different types, and is also extensible to the multi-view or multi-slice situations after some revisions have been made. 2) Since our method adopts LASSO (ℓ_1-norm penalty) to uncover groups and their interrelations, it has sparse property and the final results can be interpreted easily.

The rest of this paper is organized as follows. Section 2 briefly recalls previous related work. We presents the basic concept of CCA in Section 3. Section 4 proposes a new group discovery method by using sparse learning. The experimental results and discussions on artificial and real world networks have been provided in Section 5, followed by the conclusions in Section 6.

2 Related Work

Group discovery is a hot topic in network learning. Over the past years, a considerable number of discovery methods have been witnessed. Here only the latest methods will be discussed briefly. Interested readers can refer to good survey literature (see e.g., [4]) and references therein to get more information.

Since complex networks are often represented as graphs, graph analysis, which has solid mathematical and theoretical fundamentals, has been extensively investigated in network learning. This kind of discovery methods apply graph theory to explore groups, which tightly connected with each other by edges [4]. Typical examples of such kind include clique-based, graph partition-based, ratio cut-based, normalized cut-based and max-flow-min-cut detection approaches. Note that graph partitioning is a NP-hard problem. Moreover, it is also need the number of groups and even their sizes, which are usually unknown in advance.

Clustering is another solution for group discovery. The clustering techniques, such as hierarchical clustering, partitional clustering and spectral clustering, have been taken to identify groups from data. For instance, Chi et al. [2] disclosed communities and evaluated their evolution process by using the spectral clustering, which usually partitions nodes in a graph into clusters in terms of the eigenvectors of its matrix representation. Like the graph-based methods, the limitation of clustering is its relatively high computational cost, and the number of clusters should also be pre-specified in some situations.

Modularity is widely used as a stopping measure for clustering, resulting in the prevalence of the modularity maximization-based community discovery methods [13]. The great success of this framework relies on the modularity assumption, that is, the higher a modularity is, the better its corresponding partition is. This implies that the partition with maximum modularity on a given graph is the best group. As a typical example, Jiang and McQuay [7] exploited modularity Laplacian to discover communities by optimizing the modularity functions with additional nonnegative constraint. However, the modularity optimization is also NP-complete. Additionally, in real world the assumption of modularity maximum is not always true [4].

Statistical inference is a powerful tool to deduce properties of data in machine learning, and has also been used to model and analyze graph topological structures. The discovery methods based on block modeling, Bayesian inference, latent Dirichlet allocation and model selection belong to this kind of representative cases. For example, Yang et al. [22] estimated parameters with a Bayesian treatment in modeling networks and then developed a dynamic stochastic block model to find communities and their evolution in dynamic situations.

More recent studies of group discovery focus on exploring the evolution or behaviors of groups from the multi-slice or multi-dimension prospective. The representative examples of such framework have been illustrated in [14,22,12,18]. Pons and Latapy [15] integrated the communities discovered by different methods in a post-processing way. Tang et al. [20] exploited four integration strategies, i.e., network interactions, utility functions, structural features and community partitions, to fuse the communities derived from multi-dimension sources. Lancichinetti et al. [9] took edge directions, edge weights, overlapping communities, hierarchies and community dynamics into account to identify the significant communities.

It is worthy to notice that most of discovery algorithms mentioned above can not handle data from heterogeneous sources and the interrelations of groups. The networks in reality, however, often encounter groups with different types. Thus identifying groups with different types and their interrelations is important, because it may bring more information and provide us a deep insight to understand the working mechanisms of the networks. Recently, author-topic model (ATM) has gained much attraction in information retrieval [1]. It mainly adopts graph-based (e.g., LDA and pLDA), semantic-based (e.g., PCA, LSI and pLSI) or their extensions with other techniques (e.g., Gibbs sampling and HMM) to reveal the groups of authors and topics [1]. However, ATM only qualitatively describes the relationships of groups. As far as we are aware, little attention has been put on measuring the interrelations of groups in a quantitative way.

3 Canonical Correlation Analysis

Canonical correlation analysis (for short, CCA) proposed by Hotelling is a well-known multivariate technique [6]. Let $X = \{x_1, ..., x_p\}$ and $Y = \{y_1, ..., y_q\}$ be two sets of variables, and both of them are centralized, i.e., $\sum_{i=1}^{p} x_i = 0$ and $\sum_{i=1}^{q} y_i = 0$. CCA aims at obtaining two weighted linear combinations ω_X and ω_Y of X and Y, respectively, such that their correlation is maximal, i.e.,

$$\rho(\omega_X, \omega_Y) = \max \frac{< \omega_X, \omega_Y >}{||\omega_X|| ||\omega_Y||}, \tag{1}$$

where $\omega_X = Xu$ and $\omega_Y = Yv$ are canonical variates with the weight vectors $u' = (u_1, ..., u_p)$ and $v' = (v_1, ..., v_q)$.

The intuitive meanings of CCA is that it projects two sets of variables into a lower-dimensional space in which they are maximally correlated. Since solving the maximal value of $\rho(\omega_X, \omega_Y)$ is invariant to the scaling of u and v either together or independently, Eq. 1 can be rewritten as follows:

$$\begin{aligned} &\arg\max\nolimits_{u,v}\ u'X'Yv \\ &\qquad\qquad s.t.\ u'X'Xu = 1, v'Y'Yv = 1. \end{aligned} \tag{2}$$

For the optimization problem in Eq. 2, one of frequently used solutions is to formulate it as a Lagrangian optimization form. Due to the limitation of space, here we will not provide the details of inferences step by step. Eventually, the CCA formulation seeks for solving the eigenvectors and eigenvalues of the following generalized eigenvalue problem:

$$\begin{aligned} X'Y(Y'Y)^{-1}Y'Xu &= \lambda X'Xu \\ Y'X(X'X)^{-1}X'Yv &= \lambda Y'Yv \end{aligned} \tag{3}$$

where λ is the eigenvalue, and u and v are its corresponding eigenvectors with respect to X and Y, respectively. If $X'X$ is invertible (i.e., non-singular), the first formula of Eq. 3 can be further formulated as a standard eigenvalue problem of $(X'X)^{-1}X'Y(Y'Y)^{-1}Y'Xu = \lambda u$. Otherwise, other strategies, such as generalized inverses and regularization, should be considered to obtain the inverse of $X'X$ or $Y'Y$.

4 Group Relationship Discovery

4.1 Problem Statement

Assume $X = \{x_1, x_2, ..., x_p\} \in \Re^{n\times p}$ and $Y = \{y_1, y_2, ..., y_q\} \in \Re^{n\times q}$ are two sets of variables (or features) representing n instances (or samples), where $x_i \cap y_j = \emptyset$ for $i = 1, ..., p$ and $j = 1, ..., q$. They can be treated as the n instances observed from two different perspectives.

In this paper, the purpose of group discovery is twofold. The first one is, for each set of variables, e.g., X, to identify a set of groups $G_X = \{G_{Xi} | G_{Xi} \subseteq X, i = 1, ..., k\}$, such that the elements are highly relative to each other in the same group G_{Xi}, while irrelative to those in other groups G_{Xj} $(i \neq j)$. Similar operations can be performed on Y to obtain G_Y. The second task of this paper is to scale the relationships between groups derived from different types, e.g., $G_{Xi} \in G_X$ and $G_{Yj} \in G_Y$, in a quantitative manner.

Since the group interrelations are often hidden and implicitly observed through a large number of variables, it is not appropriate to evaluate correlations between pairs of variables individually, or simply calculate the accumulative total of dependencies between variables from groups with different views. A desirable solution is to take the groups as a whole, rather than their individual variables, into account in extracting the group interrelations.

4.2 Obtaining the First Group

Given two sets of variables X and Y, CCA can effectively obtain their canonical variates such that the correlations between them are maximal. However, one of the CCA problems is that the computational cost of matrix decomposition is relatively high, especially when the quantity of variables exceeds tens of thousands. The non-singular property of matrix is the second issue that should also be taken into consideration. Here we go further and efficiently solve it via ℓ_1-norm regularization.

For the Eq. 2, we have the following property:

Property 1. The optimization problem of CCA (i.e., Eq. 2) is equivalent to a distance minimization problem between two matrices, i.e.,

$$\begin{aligned} \arg\min_{u,v}\ f(u,v) = \|Xu - Yv\|^2 \\ s.t.\ u'X'Xu = 1, v'Y'Yv = 1. \end{aligned} \tag{4}$$

One may observe that this formulation is a least square if either Xu or Yv is fixed, i.e.,

$$\begin{aligned} f(u) = \|Xu - \alpha\|^2 \\ f(v) = \|\beta - Yv\|^2 \end{aligned} \tag{5}$$

To solve this optimization problem above, Partial Least Squares (PLS) seems to be an effective technique. The off-the-shelf method of PLS performs a regression operation on each formulation within Eq. 5 alternately, and ultimately obtains u and v. However, a possible drawback of PLS is that interpreting the derived results becomes impossible, since u and v are weighted combinations of all available variables in X and Y, respectively.

The interpretability of the obtained results is very important for the practical applications. It directly affects users understanding data, such as providing evidences for decision-makers, boosting product promotion for businessmen, uncovering pathogenies of diseases for doctors. An effective strategy is to make the results sparse via variable selection.

We resort to a least absolute shrinkage and selection operator (for short, LASSO) penalized model [21] to fulfill the purpose of variable selection. The underlying is that LASSO enables us estimating the objective function and achieving variable selection simultaneously in one stage, where variables will be selected by assigning zeros to the weights of variables with very small coefficients. Specifically, the first objective function $f(u)$ in Eq. 5 has been transformed to the following form after a ℓ_1-norm constraint has been performed on u.

$$f(u) = \|Xu - \alpha\|^2 + \lambda_u \sum |u_i| \tag{6}$$

where λ_u ($\lambda_u \geq 0$) is a tuning constant for u. Under this constraint, the weights of some variables become zero if λ_u is enough small. Specifically, each weight coefficient u_i decreases after comparing with a threshold. If u_i is lower than the threshold it will be set to zero, otherwise it will be modified or preserved. Thus the purpose of sparse solution can be achieved. Given λ_u, u_i is determined via the soft-thresholding strategy, i.e.,

$$u_i = sgn((\alpha'X)_i)(|(\alpha'X)_i| - \lambda_u)_+ \tag{7}$$

where $sgn(z)$ is the sign function of z, and $(z)_+$ is defined to z if $z > 0$ and 0 if $z \leq 0$. Similarly, $f(v)$ can also be handled by performing another ℓ_1-norm penalty regularization λ_v.

After u and v have been obtained, the new coordinate systems of X and Y are formed and represented as Xu and Yv, respectively. Since u and v are sparse, Xu and Yv have good interpretability and represent the first group derived from X and Y, respectively. More importantly, this pair of groups has maximal correlation, which can be measured in a quantitative way, i.e., $\rho =< Xu, Yv >$, after normalized.

4.3 Obtaining the Rest Groups

Assume the same instances are observed from two different views. Obtaining only one group from data is not enough, since it is unlikely to describe all relationships of variables, especially in high-dimensional settings. Therefore, additional groups should also be uncovered from data. To untie this knot, an alternative solution is to minimize the criterion of discovering the first group (i.e., Eq. 4) repeatedly, each time on the residual variables obtained by wiping off the information of the groups found previously.

To achieve the goal, a trick is available, where the original data will be updated according to the information of the obtained groups. The central idea is similar to the *whitening* step in image processing, which has often been used to lessen dependencies or correlations of images. Specifically, after the first group Xu has been disclosed, the original data X is extended as $X'_n = [X' \quad \frac{u}{\sqrt{\rho}}]$, where ρ is the canonical correlation. This extension aims to lessen the effect of Xu, so as not to frustrate the process of discovering other groups in succession. Y can also be handled in a similar way. Indeed, this extension process is lossless, because the original and extension matrices have the same canonical variates. Under the context of X_n (or Y_n), we can obtain its first canonical variate $X_n u_2$ (or $Y_n v_2$), which actually is the second group of X (or Y). Akin to u_2, other groups can be identified by updating X continuously, until the extension matrix contain no more useful information.

4.4 Group Discovery Algorithm

Based on the analysis above, we present a new group discovery algorithm shown as Alg. 1. It consists of two main loops, one nested within the other. The inner iteration aims to identify one group, whereas the outer loop obtains all groups hidden behind data.

The algorithm starts at initializing relative parameters, such as normalizing X and Y. In the outer loop, the initial weights of variables v_k (or u_k) is set, guaranteeing the inner iteration convergence. For simplicity, here we take the first right singular vector of Y_k as the initial value of v_k. Once the vector has been assign an initial value, the inner iteration can solve the optimization problem of CCA by using ℓ_1-norm penalty. The process is repeated until the residual matrices have no more useful information, or k is larger than a pre-specified threshold K.

5 Simulation Experiments

5.1 Experiments on Synthetic Datasets

The synthetic data consists of two datasets. Each one represents the same 100 samples observed from different views. The first dataset X has 60 variables

Algorithm 1. Identifying groups and their relationships

```
1). Initialize parameters, e.g., $X_1$=$X$ and $Y_1$=$Y$.
2). For $k = 1, ..., K$:
    2.1) Initialize $v$ and normalize it.
    2.2) Repeat until convergence
        2.2.1) Obtain $u$ by virtue of Eq. 7.
        2.2.2) Normalize $u$ as $u = \frac{u}{\|u\|}$.
        2.2.3) Obtain $v$ by virtue of Eq. 7.
        2.2.4) Normalize $v$ as $v = \frac{v}{\|v\|}$.
    2.3) Estimate the correlation $\rho(Xu, Yv)$.
    2.4) Update the residual matrices $X_k$ and $Y_k$.
```

$\{x_1, ..., x_{60}\}$ that belongs to six groups $G_X = \{xg_1, ..., xg_6\}$ with 10 variables in each group. The second dataset Y represents the same samples from another view. It contains 150 variables $\{y_1, ..., y_{150}\}$ also organized into six groups $G_Y = \{yg_1, ..., yg_6\}$ with the mean number of variables. Links between G_X and G_Y are generated as follows. For each sample in each dataset, it is randomly assigned to one of groups and the probabilities of variables corresponding to the group are larger than 0.2. For other groups, the values are less than 0.1.

The experimental results on the artificial datasets are presented in Table 1, where $|xg|$ denotes the number of variables contained within the i-th group xg_i. From this table, we know that the proposed method has excellent performance. It not only identified all groups from both datasets, but also correctly measured the interrelations between them. Additionally, the correlations of the groups were also simultaneously calculated in a quantitative way after the groups were obtained. For example, the group xg_5 (the 6th line) obtained from X was exactly matched with the group yg_5 from Y, and their correlation coefficient was 0.832. From the perspective of individual groups, the performance of the proposed method is also quite well (see the $|xg|$ and $|yg|$ columns in Table 1). It successfully identified all groups from the artificial data. More importantly, the disclosed groups contain the right members, and are subsets of the corresponding assumption ones. For instance, the xg_2 group is a subset of the assumption one in X, for it contains all members, i.e., $x_{21}, ..., x_{30}$, except x_{22}. Besides, the size of each disclosed group also approaches that of the original one.

Table 1. Experimental results on the synthetic data

No.	Group pair	$\|yg\|$	$\|xg\|$	Correlation coefficient
1	(xg_1, yg_1)	23	9	0.950
2	(xg_2, yg_2)	23	9	0.879
3	(xg_6, yg_6)	25	10	0.877
4	(xg_4, yg_4)	24	10	0.870
5	(xg_3, yg_3)	24	10	0.841
6	(xg_5, yg_5)	25	10	0.832

To further demonstrate the effectiveness, we calculated the similarity of the disclosed results to the assumption ones. Since the distributions of groups are known in advance, we took normalized mutual information (NMI) and Jaccard coefficient as our measurements in the simulation experiments. In our experiments, the values of NMI of the discovered groups $XG = \{xg_1, .., xg_6\}$ and $YG = \{yg_1, ..., yg_6\}$ to the assumption ones G_X and G_Y are 0.992 and 0.991, respectively. Correspondingly, the Jaccard coefficients are 0.967 and 0.96, respectively. These factors indicate that the proposed method is good at identifying groups from data and the groups identified by our method are quite similar the assumption ones.

5.2 Experiments on DBLP Datasets

A typical application of group discovery is science bibliography, where the groups of keywords and authors are quite ubiquitous. Here we also carried out experiments on the DBLP database [19]. For the convenience of discussion, we downloaded the DBLP database[1] and extracted 1071 papers published in five international conferences from 2000 to 2004. In our experiment, the papers were organized into two datasets, where the first one A involved 2022 authors, while the second one T contained 1065 terms extracted from paper titles.

After performing the proposed method on A and T, we obtained 143 groups of authors G_A and terms G_T, where the authors in the ith group G_{Ai} are associated with the terms within the ith group G_{Ti}. Table 2 lists 10 over 143 disclosed groups, where CC denotes the correlation coefficient of G_{Ai} to G_{Ti}, while $\#P, \#A$ and $\#T$ indicate how many papers, authors and terms were involved in each group. As an example, the 8th group of authors G_{A8} (G_{T8}) covers two papers involving 7 authors (7 terms). The topic of this pair mainly concerns *semantic* information by *parsing structures* of texts.

Table 2. Ten groups discovered from the DBLP corpus

No	CC	#P	#A	#T	Authors	Terms
Gp1	0.999	3	5	11	S Donoho, S Zhu, ...	option, markets, ...
Gp2	0.998	3	5	9	D Cau, S Acid, ...	effects, causality, ...
Gp4	0.994	3	6	6	C Borgelt, K Lou, ...	expectation, fuzzy,...
Gp5	0.993	4	6	5	ZW Ras, R Wong, ...	profit, rules, ...
Gp6	0.993	2	4	6	S Zhang, Y Zhu, ...	elastic, burst, ...
Gp8	0.991	2	7	7	S Pradhan, L Lloyd, ...	structure, parsing, ...
Gp11	0.990	2	3	4	M Sauban, BF White,...	profiling, document, ...
Gp25	0.986	2	3	5	J Adibi, E Oja,...	hidden, markov, ...
Gp38	0.979	3	3	6	PW Eklund, G Stumme,...	discovery, ontologies, ...
Gp107	0.946	3	9	6	P Kim, J Ye, ...	media, stream, ...

[1] `http://www.public.asu.edu/~{}ltang9/data/dblp.tar.gz`

An interesting discovery made by the proposed method is that co-authors were not always be included into the same author group. For instance, the three papers in the 107th author group had 10 authors totally, while only 9 was included within this author group. The missing co-author is *M. Vazirgiannis*. In fact, *M. Vazirgiannis* published more papers about *clustering* than *semantics*. This means that the results identified by our method rely on keywords in each group on the whole, rather simply collect co-occurrence information.

The interrelations between G_{Ai} and G_{Ti} were also estimated by the proposed method and illustrated as a dot graph (see Fig. 1), where the x-axis denotes the group pairs $Gp_i = (G_{Ai}, G_{Ti})$. The large values of correlation coefficients elucidate the effectiveness of our method, for in each group pair the disclosed groups with different types were tightly related with each other. Among the 143 coefficients, only six of them were lower than 0.90, whereas the minimal one was still larger than 0.81. This, however, is reasonable for the last six group pairs were identified upon the residual information the previously discovered groups left and covered more authors and terms.

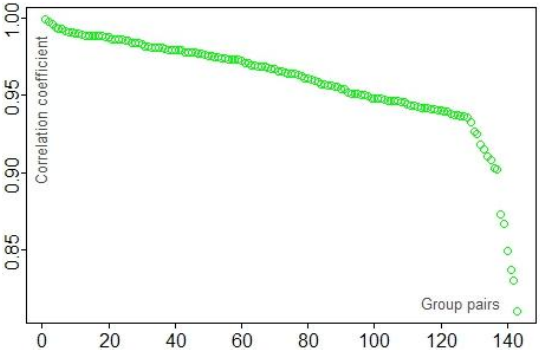


Fig. 1. The correlation coefficients of group pairs

Identifying hot topics is one of major tasks extensively studied in the field of author-topic models. This goal can also be achieved by our method. We obtained the hot topics in terms of the quantities of papers covered by the group pairs for the sake of simplification. Fig. 2 provides us an intuitive observation about the popularity of topics during the five years. From this graph, one may easily observe what topics are hot during the past years. For instance, *Gp97* followed by *Gp*140 and *Gp*135 was the most popular topic comprising 15 research papers. It concerned the study of *category*, *mining* and *relationship*, and included six authors, e.g., *P Yu*, *R Hilderman*, *H Hamilton*, etc. Another interesting fact is that identifying the groups and their relationships can also offer some insights into the research trends and the collaboration relationships between different research teams if the latest information is available.

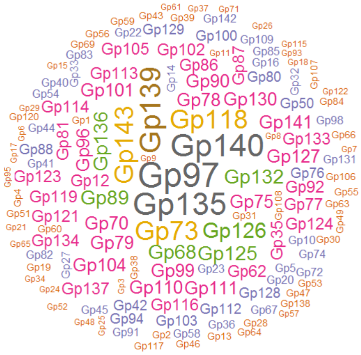


Fig. 2. The cloud graph of groups

6 Conclusions

Generally, additional information from heterogeneous sources is helpful to analyze and understand network structures and functions. In this paper, we propose a statistical learning framework to discover groups and capture their interrelations from different data sources. The central idea of our method is to formulate group discovery as an optimization problem of CCA, and then extend it to a LASSO problem to achieve the sparse purpose. Within this framework, group discovery and their relationship measurement are turned out to be easily fulfilled in one stage. Simulation experiments were conducted on both carefully designed synthetic datasets and the DBLP corpus. The experimental results show that the proposed method tends to identify accurate group information and reveal useful insights in a given network from two different views.

Acknowledgments. This work is partially supported by the Australian Research Council(ARC) (DP130104090), the National NSF of China (61100119, 61272130, 61272468), and the Open Project Program of the National Laboratory of Pattern Recognition (NLPR).

References

1. Blei, D.: Introduction to probabilistic topic models. Comm. of the ACM 55(4), 77–84 (2012)
2. Chi, Y., Song, X., Zhou, D., Hino, K., Tseng, B.: Evolutionary spectral clustering by incorporating temporal smoothness. In: Proc. of the 13th ACM SIGKDD Int'l Conf. on Know. Disc. and Data Mining, pp. 153–162. ACM (2007)
3. Chiua, G., Westveld, A.: A unifying approach for food webs, phylogeny, social networks, and statistics. Proc. Natl. Acad. Sci. USA 108(38), 15881–15886 (2011)
4. Fortunato, S.: Community detection in graphs. Phy. Rept. 486(3), 75–174 (2010)
5. Getoor, L., Diehl, C.: Link mining: a survey. SIGKDD Explor. Newsl. 7, 3–12 (2005)

6. Hotelling, H.: Relations between two sets of variables. Biometrika 28(3/4), 312–377 (1936)
7. Jiang, J.Q., McQuay, L.J.: Modularity functions maximization with nonnegative relaxation facilitates community detection in networks. Phys. A 391(2), 854–865 (2012)
8. Krause, A., Frank, K., Mason, D., Ulanowicz, R., Taylor, W.: Compartments revealed in food-web structure. Nature 426(6964), 282–285 (2003)
9. Lancichinetti, A., Radicchi, F., Ramasco, J., Fortunato, S.: Finding statistically significant communities in networks. PLoS ONE 6(4), e18961 (2011)
10. Liu, B., Liu, L., Tsykin, A., Goodall, G., Green, J., Zhu, M., Kim, C., Li, J.: Identifying functional mirna-mrna regulatory modules with correspondence latent dirichlet allocation. Bioinformatics 26(24), 3105–3111 (2010)
11. Michal, R.Z., Chaitanya, C., Thomas, G., Padhraic, S., Mark, S.: Learning author-topic models from text corpora. ACM Trans. Inf. Syst. 28(1), Article 4 (2010)
12. Mucha, P., Richardson, T., Macon, K., Porter, M., Onnela, J.P.: Community structure in time-dependent, multiscale, and multiplex networks. Science 328, 876–878 (2010)
13. Newman, M., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E 69(2), 026113 (2004)
14. Palla, G., Barabási, A.L., Vicsek, T.: Quantifying social group evolution. Nature 446(7136), 664–667 (2007)
15. Pons, P., Latapy, M.: Post-processing hierarchical community structures: Quality improvements and multi-scale view. Theo. Comp. Sci. 412(8), 892–900 (2011)
16. Scott, J.: Social Network Analysis: A Handbook. SAGE Publications, London (2000)
17. Serrour, B., Arenas, A., Gómez, S.: Detecting communities of triangles in complex networks using spectral optimization. Comp. Comm. 34(5), 629–634 (2011)
18. Shen, H.W., Cheng, X.Q., Fang, B.X.: Covariance, correlation matrix, and the multiscale community structure of networks. Phy. Rev. E 82(1), 016114 (2010)
19. Tang, L., Liu, H., Zhang, J., Nazeri, Z.: Community evolution in dynamic multi-mode networks. In: Proc. of the 14th ACM SIGKDD Intl' Conf. on Knowl. Disc. and Data Mining, pp. 677–685. ACM (2008)
20. Tang, L., Wang, X., Liu, H.: Community detection via heterogeneous interaction analysis. Know. Dis. Dat. Min. 25(1), 1–33 (2012)
21. Tibshirani, R.: Regression shrinkage and selection via the lasso: a retrospective. J. R. Statist. Soc. B 73(3), 273–282 (2011)
22. Yang, T., Chi, Y., Zhu, S., Gong, Y., Jin, R.: Detecting communities and their evolutions in dynamic social networks—a bayesian approach. Mach. Learn. 82(2), 157–189 (2011)
23. Yang, Z., Tang, J., Li, J.: Social community analysis via factor graph model. IEEE Intelligent Sys. 26(3), 58–65 (2011)

Multiplex Topic Models

Juan Yang[1], Jia Zeng[1], and William K. Cheung[2]

[1] School of Computer Science and Technology,
Soochow University, Suzhou,215006, China
yangjuancc@gmail.com, j.zeng@ieee.org
[2] Department of Computer Science,
Hong Kong Baptist University, Kowloon Tong, Hong Kong
william@comp.hkbu.edu.hk

Abstract. Multiplex document networks have multiple types of links such as citation and coauthor links between scientific papers. Inferring thematic topics from multiplex document networks requires quantifying and balancing the influence from different types of links. It is therefore a problem of considerable interest and represents significant challenges. To address this problem, we propose a novel multiplex topic model (MTM) that represents the topic influence from different types of links using a factor graph. To estimate parameters in MTM, we also develop an approximate inference algorithm, multiplex belief propagation (MBP), which can estimate the influence weights of multiple links automatically at each learning iteration. Experimental results confirm the superiority of MTM in two applications, document clustering and link prediction, when compared with several state-of-the-art link-based topic models.

Keywords: Multiplex topic models, belief propagation, factor graph.

1 Introduction

Documents contain not only content information but also relational information such as coauthors, citations and geographic locations, leading to the *multiplex* network structures. For instance, the bibliographic field of a scientific paper has a list of entries: authors, venues, time, and references, representing different types of relations between papers. Citation and coauthor links exist widely in both scientific papers and web pages, and thus attract considerable interest in the topic modeling community [1–5]. Intuitively, the papers cited each other, written by the same author, or published in the same venue, will have stronger topic correlations. Previous topic models consider only equal-weighted multiplex structures in document networks, i.e., citation and coauthor links have been treated equally in topic formation. However, in real-world applications it is obvious that different links may play different roles in topic modeling. For example, two authors who often collaborate on some papers tend to be interested in the same topics, while the cited paper may be from an interdisciplinary area with a quite different topic. How to quantify and balance different types of links in document networks for a better topic modeling performance still remains a challenging and unsolved problem.

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 568–582, 2013.

In this paper, we propose a novel multiplex topic model (MTM) for multiplex document networks. MTM uses a factor graph [6, 7] to combine multiple types of links into the topic modeling process. The factor graph provides a natural graphical description that factorizes the joint probability of MTM into a product of local functions. The topic label for each word is represented as the variable node in the factor graph, and different links between documents can be encoded by the factor nodes with parameterized functions to encourage or penalize specific topic labeling configurations through links. To maximize the joint probability of MTM, we develop a multiplex belief propagation (MBP) algorithm for approximate inference and parameter estimation. MBP is a message passing algorithm [8] based on multiple links. We quantify and balance the link influence by consistency of messages passed through the link. Intuitively, if the passed messages are more consistent, they will influence the topic labeling configuration more through the link. To summarize, the methodological contributions of our work are: (1) MTM that incorporates different types of links using the factor graph, (2) MBP that maximizes the joint probability of MTM by passing messages through multiple links, and (3) balance of link influences by the consistency of messages passed through links. Experimental results confirm that MTM has the superior performance in two applications, document clustering and link prediction, when compared with several state-of-the-art link-based topic models including author-topic models (ATM) [1], relational topic models (RTM) [2] and multirelational topic models (MRTM) [3].

2 Related Work

As one of the simplest topic models, latent Dirichlet allocation (LDA) [9] has achieved great successes in text analysis due to its ability in reducing the dimensionality of text data. Since LDA ignores multiple types of links in multiplex document networks, there have been many link-based topic models that fall broadly into two categories. The first is to build generative models that generate both content and link such as ATM [1], RTM [2] and topic-link LDA (TLLDA) [4]. ATM and RTM are typical single-link topic models for coauthor and citation links, respectively. Neither of them combines coauthor and citation links together. TLLDA generates citation links by incorporating the coauthor link contribution. But it does not clearly show how these links mutually influence each other. The second focuses on depicting topic influences between documents through links. For example, NetPLSA [10] regularizes the topic labeling configuration with the citation links. Furthermore, TMBP [11] develops a joint regularization framework to combine the multiplex network structure with topic modeling. MRTM [3] and iTopic model [5] describe multiple dependencies between documents within the Markov random field (MRF) [8] framework. MRTM combines both coauthor and citation links in either the parallel or the cascade manner. iTopic model transforms multiple types of links into one single type, so that it cannot directly deal with multiplex document networks. Note that most previous studies face the same unsolved problem: how to automatically determine the influence of different types of links for a better topic modeling performance? This motivates the MTM model in the following sections.

3 Multiplex Topic Models

In this section, we will address three questions: 1) how to represent the joint probability of MTM by a factor graph, 2) how to perform approximate inference based on the factor graph, and 3) how to learn the weights of different types of links.

3.1 Factor Graph Representation for MTM

Probabilistic topic modeling can be interpreted as a labeling problem [7]. LDA [9] assigns a set of thematic topic labels $\mathbf{z}_{W\times D} = \{z^k_{w,d}\}$ to explain the observed non-zero elements in the document-word matrix $\mathbf{x}_{W\times D} = \{x_{w,d}\}$, where the notation $1 \le k \le K$ is the topic index, the notations $1 \le w \le W$ and $1 \le d \le D$ are the word index in the vocabulary and the document index in corpus. The topic label satisfies $\sum_k z^k_{w,d} = 1$ and $z^k_{w,d} = \{0, 1\}$. The non-zero element $x_{w,d} \neq 0$ denotes the number of word counts at index $\{w, d\}$. The document-specific topic proportion $\boldsymbol{\theta} = \{\theta_d(k)\}$ generates the topic label $z^k_{w,d}$, and the topic-specific multinomial distribution $\boldsymbol{\phi} = \{\phi_w(k)\}$ generates the word token w forming the word counts $x_{w,d}$. In this paper, the Dirichlet hyperparameters α, β in LDA are assumed to be fixed for simplicity [12]. The best topic labeling configuration $\mathbf{z}^*$ is inferred by maximizing the joint probability $p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}, \boldsymbol{\phi}; \alpha, \beta)$ in terms of $\mathbf{z}$. Note that LDA does not consider the multiplex network structure that influences the topic labeling configuration $\mathbf{z}$.

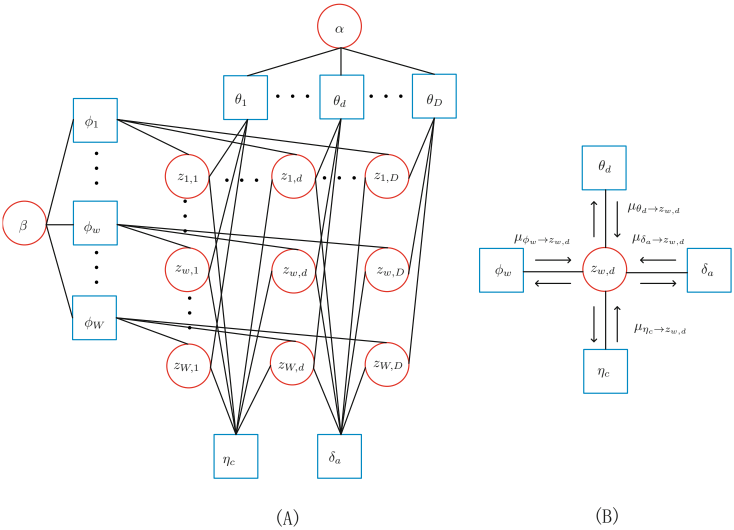
To address this problem, MTM takes the observed multiplex network structure $\mathbf{G}$ into account. So, the objective of MTM turns to maximizing the joint probability $p(\mathbf{x}, \mathbf{G}, \mathbf{z}|\boldsymbol{\theta}, \boldsymbol{\phi}; \alpha, \beta)$. Clearly, the best topic labeling configuration $\mathbf{z}^*$ depends not only on $\mathbf{x}$ but $\mathbf{G}$ as well. We assume that $\mathbf{x}$ and $\mathbf{G}$ are conditionally independent with regard to $\mathbf{z}$. According to the Bayes' rule and LDA assumption [9], this joint probability can be decomposed into three parts:

$$p(\mathbf{x}, \mathbf{G}, \mathbf{z}|\boldsymbol{\theta}, \boldsymbol{\phi}; \alpha, \beta) = p(\mathbf{x}|\mathbf{z}, \boldsymbol{\phi}; \beta)p(\mathbf{z}|\boldsymbol{\theta}; \alpha)p(\mathbf{G}|\mathbf{z}), \tag{1}$$

where the first two parts are standard LDA objective function [7], and the third part $p(\mathbf{G}|\mathbf{z})$ shows that the multiplex document network structure is conditioned only on $\mathbf{z}$ for simplicity. Unlike LDA, MTM (1) explains how the content $\mathbf{x}$ and the link $\mathbf{G}$ influence the topic labeling configuration $\mathbf{z}$.

Using the Multinomial-Dirichlet conjugacy [13], we can integrate out $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ in (1) and obtain the objective function in the *collapsed* $\mathbf{z}$ space,

$$p(\mathbf{x}, \mathbf{G}, \mathbf{z}; \alpha, \beta) = p(\mathbf{x}, \mathbf{z}; \alpha, \beta)p(\mathbf{G}|\mathbf{z}). \tag{2}$$



(A) (B)

Fig. 1. (A) Factor graph representation for MTM, (B) multiplex message passing process

The first term of (2) is the joint probability of LDA [14],

$$p(\mathbf{x},\mathbf{z};\alpha,\beta) \propto \prod_d \prod_k \Gamma\Big(\sum_w x_{w,d} z^k_{w,d} + \alpha\Big) \times \\ \prod_w \prod_k \Gamma\Big(\sum_d x_{w,d} z^k_{w,d} + \beta\Big) \Gamma\Big(\sum_{w,d} x_{w,d} z^k_{w,d} + W\beta\Big)^{-1}, \qquad (3)$$

where $\Gamma(\cdot)$ is the gamma function. The second term of (2) is still the likelihood function of $\mathbf{G}$ given the topic labeling configuration $\mathbf{z}$.

The collapsed LDA in (3) can be represented by a factor graph in Fig. 1A without looking at the factors η_c and δ_a. In the collapsed space $\{\mathbf{z},\alpha,\beta\}$, the factors θ_d and ϕ_w are denoted by squares, and their connected variables $z_{w,d}$ are denoted by circles. The factor θ_d connects the neighboring topic labels $\{z^k_{w,d}, \mathbf{z}^k_{-w,d}\}$ at different word indices within the same document d. We see that the factor θ_d describes the dependency between $z^k_{w,d}$ and $\mathbf{z}^k_{-w,d}$ in the first term $\Gamma(\sum_w x_{w,d} z^k_{w,d} + \alpha)$ of (3). The factor ϕ_w connects the neighboring topic labels $\{z^k_{w,d}, \mathbf{z}^k_{w,-d}\}$ at the same word index w but in different documents. We see that the factor ϕ_w describes the dependency between $z^k_{w,d}$ and $\mathbf{z}^k_{w,-d}$ in the second term $\Gamma(\sum_d x_{w,d} z^k_{w,d} + \beta)$ of (3). Since the third term $\Gamma(\sum_{w,d} x_{w,d} z^k_{w,d} + W\beta)^{-1}$ of (3) is a normalization factor in terms of k [7], we do not explicitly connect ϕ_w with all hidden variables $\mathbf{z}^k$ for a better illustration. The hyperparameters $\{\alpha,\beta\}$ can be viewed as pseudo counts having the same layer with hidden variables $\mathbf{z}$ [7].

The second part $p(\mathbf{G}|\mathbf{z})$ of (2) is the likelihood function of the topic labeling configuration under the constraints of multiple links. Without loss of generality, we consider two types of links in this paper: citation and coauthor links, i.e., $\mathbf{G} = \{c, a\}$. Thus, we can re-write $p(\mathbf{G}|\mathbf{z})$ as

$$p(\mathbf{G}|\mathbf{z}) = \prod_{c} p(c|\mathbf{z}_c) \prod_{a \in \mathcal{A}_d} p(a|\mathbf{z}_{\mathcal{A}_d}), \tag{4}$$

where $\mathbf{z}_c$ is a pair of topic labeling configurations $\{\mathbf{z}_{\cdot,d}, \mathbf{z}_{\cdot,d_c}\}$ connected by the citation link c, and $\mathbf{z}_{\mathcal{A}_d}$ is the topic labeling configuration of all documents written by the coauthors $a \in \mathcal{A}_d$ of the document d. The notations $1 \leq c \leq C$ and $1 \leq a \leq A$ denote citation link and author indices in corpus. Eq. (4) shows that each citation link c is generated by the pairwise topic configurations of two cited documents $\mathbf{z}_c$, while each author a is conditioned on the topic configurations from all coauthors $\mathbf{z}_{\mathcal{A}_d}$ in the document d. As a result, Eq. (4) describes the multiplex document network structure in terms of citation and coauthor links. In the next section, we shall design the citation likelihood as a function of similarity between two linked documents, and design the coauthor likelihood as a function of similarity among coauthor topic proportions.

Incorporating (4), we can represent the complete MTM by the factor graph as shown in Fig. 1A. The factors η_c and δ_a denote the citation and coauthor links, respectively. For example, the factor η_c connects topic labels $\mathbf{z}_{\cdot,1}$ and $\mathbf{z}_{\cdot,d}$ on different word indices if the document pair $\{1, d\}$ has a citation link, and the factor δ_a connects topic labels $\mathbf{z}_{\cdot,d}$ and $\mathbf{z}_{\cdot,D}$ on different word indices if the document pair $\{d, D\}$ has the same author a. From Fig. 1A, we see that MTM assigns a set of thematic topic labels, $\mathbf{z} = \{z^k_{w,d}\}$, to explain the nonzero elements in the document-word matrix $\mathbf{x} = \{x_{w,d}\}$ by taking multiple types of links into consideration. The factor θ_d connects topic labels $\mathbf{z}_{\cdot,d}$ on different word indices within the same document d, while the factor ϕ_w connects topic labels $\mathbf{z}_{w,\cdot}$ on the same word index w but in different documents. Through four types of factor nodes $\{\theta_d, \phi_w, \eta_c, \delta_a\}$, the topic label $z_{w,d}$ depends on its neighbors $\{\mathbf{z}_{-w,d}, \mathbf{z}_{w,-d}, \mathbf{z}_c, \mathbf{z}_{\mathcal{A}_d}\}$. The notations $-w$ and $-d$ represent all word and document indices except for w and d. Because the factor graph Fig. 1A contains loops, we usually can make only approximate inference for the conditional posterior probability $p(z^k_{w,d} = 1|\mathbf{z}_{-w,d}, \mathbf{z}_{w,-d}, \mathbf{z}_c, \mathbf{z}_{\mathcal{A}_a})$, referred to as the *message*. In the next subsection, we propose a novel MBP algorithm for passing messages over the factor graph Fig. 1A, which maximizes the joint probability (2) in the collapsed space.

3.2 Multiplex Belief Propagation

To maximize the objective (2) with hidden variables $\mathbf{z}$, we often use the iterative expectation-maximization (EM) algorithm [8]. Fortunately, the factor graph representation in Fig. 1A facilitates a special EM algorithm called MBP for approximate inference and parameter estimation. There are two important steps in MBP within the EM framework. In the E-step, MBP infers the conditional posterior probability called message $\mu_{w,d}(k) = p(z^k_{w,d} = 1|\mathbf{z}_{-w,d}, \mathbf{z}_{w,-d}, \mathbf{z}_c, \mathbf{z}_{\mathcal{A}_d})$. The message is a K-tuple vector satisfying $0 \leq \mu_{w,d}(k) \leq 1, \sum_{k=1}^{K} \mu_{w,d}(k) = 1$.

In the M-step, MBP uses the messages to update parameters in MTM, including the document-specific topic proportions $\boldsymbol{\theta}$ and topic-specific multinomial parameters $\boldsymbol{\phi}$. These two steps repeat for several iterations until convergence.

Fig. 1B shows the message passing process based on the factor graph. Messages are passed from factor nodes to variables, and the message $\mu_{w,d}(k)$ is influenced by the four types of messages: $\mu_{\eta_c \to z_{w,d}}(k)$, $\mu_{\theta_d \to z_{w,d}}(k)$, $\mu_{\phi_w \to z_{w,d}}(k)$ and $\mu_{\delta_a \to z_{w,d}}(k)$. The factors θ_d, ϕ_w and δ_a are parameterized functions, and their values can be estimated using EM algorithm. The message $\mu_{\theta_d \to z_{w,d}}(k)$ is from the neighboring words in the same document d by the factor θ_d, implying that different words in the same document tend to have similar topic labels. The message $\mu_{\phi_w \to z_{w,d}}(k)$ is associated with the messages from the same word in different documents, which indicates that the same word in different documents are likely to be assigned the same topic label. The message $\mu_{\eta_c \to z_{w,d}}(k)$ receives the messages from all cited documents from the factor node η_c, which encourages citing and cited documents to have similar topics. The message $\mu_{\delta_a \to z_{w,d}}(k)$ is from the neighboring words written by the same author a, which encourages topic smoothness among topic labels $z_{w,d}$ attached to the author a. Generally, a document d has multiple coauthors $a \in \mathcal{A}_d$. So, we sum and pass all messages from coauthors attached to the document d, i.e., $\sum_{a \in \mathcal{A}_d} \mu_{\delta_a \to z_{w,d}}(k)$, accumulating the influence from all coauthors. Note that the normalization factor in (3) prevents all words from having the same topic label, which forms different thematic topic groups.

In the E-step, according to the standard sum-product algorithm [6, 8], the message $\mu_{w,d}(k)$ is proportional to the product of all incoming messages from factors in Fig. 1B,

$$\mu_{w,d}(k) \propto \mu_{\theta_d \to z_{w,d}}(k) \times \mu_{\phi_w \to z_{w,d}}(k) \times \mu_{\eta_c \to z_{w,d}}(k) \times \sum_{a \in \mathcal{A}_d} \mu_{\delta_a \to z_{w,d}}(k), \quad (5)$$

However, in practice, the direct product operation cannot balance the messages from different sources. For example, the message $\mu_{\theta_d \to z_{w,d}}(k)$ is from the neighboring words within the same document d, the message $\mu_{\eta_c \to z_{w,d}}(k)$ is from all cited documents, and the message $\mu_{\delta_a \to z_{w,d}}(k)$ is from the words in different documents written by the same author a. If we use the product of these three types of messages, we cannot distinguish which one influences more on the topic label $z_{w,d}$. To quantify and balance different types of messages, we use the weighted sum of the three types of message,

$$\mu_{w,d}(k) \propto \Bigg[\lambda_d \mu_{\theta_d \to z_{w,d}}(k) + \lambda_c \mu_{\eta_c \to z_{w,d}}(k) + \lambda_a \sum_{a \in \mathcal{A}_d} \mu_{\delta_a \to z_{w,d}}(k)\Bigg] \times \mu_{\phi_w \to z_{w,d}}(k), \quad (6)$$

where $\lambda_d, \lambda_c, \lambda_a \in [0, 1]$, $\lambda_d + \lambda_c + \lambda_a = 1$ are the weights to balance the three messages $\mu_{\theta_d \to z_{w,d}}(k)$, $\mu_{\eta_c \to z_{w,d}}(k)$ and $\mu_{\delta_a \to z_{w,d}}(k)$. To infer the message $\mu_{w,d}(k)$, we have to determine the weight vector $\boldsymbol{\lambda} = \{\lambda_d, \lambda_c, \lambda_a\}$ and compute the four messages sending to the variable node $z_{w,d}$ from its connected factor nodes. If $\lambda_c = \lambda_a = 0$, Eq. (6) reduces to the following message update equation,

$$\mu_{w,d}(k) \propto \mu_{\theta_d \to z_{w,d}}(k) \times \mu_{\phi_w \to z_{w,d}}(k), \tag{7}$$

which is the message passing algorithm for LDA [7]. Comparing (6) with (7), we find that MTM is a natural extension of LDA by considering multiplex document structure $\mathbf{G} = \{c, a\}$. We shall discuss how to automatically learn the weight vector $\boldsymbol{\lambda}$ to achieve the desired topic modeling performance in the next subsection.

Eq. (6) transforms the standard sum-product [6] to the sum-sum message update equation. In practice, such a transformation often works because the weighted sum is a widely used method to combine different sources of information [15]. Eq. (6) also follows our intuition on topic modeling. The message $\mu_{\theta_d \to z_{w,d}}(k)$ reflects the content information from the document d, the message $\mu_{\eta_c \to z_{w,d}}(k)$ encodes the citation information from cited documents, and the message $\sum_{a \in \mathcal{A}_d} \mu_{\delta_a \to z_{w,d}}(k)$ is from documents written by authros/coauthors. Since the message $\mu_{\phi_w \to z_{w,d}}(k)$ is vocabulary word-specific rather than document-specific, it does not need to have a balancing weight as the above document-level messages. Therefore, we still multiply $\mu_{\phi_w \to z_{w,d}}(k)$ by the weighted messages as the standard sum-product algorithm.

Following [7], the messages from factors $\{\theta_d, \phi_w\}$ to variables are the normalized sum of all incoming messages from the neighboring variables.

$$\mu_{\theta_d \to z_{w,d}}(k) = f_{\theta_d}\Bigg(\boldsymbol{\mu}_{-w,d}(k) + \alpha\Bigg), \tag{8}$$

$$\mu_{\phi_w \to z_{w,d}}(k) = f_{\phi_w}\Bigg(\boldsymbol{\mu}_{w,-d}(k) + \beta\Bigg), \tag{9}$$

where $\boldsymbol{\mu}_{-w,d}(k) = \sum_{-w} x_{w,d}\mu_{w,d}(k)$, $\boldsymbol{\mu}_{w,-d}(k) = \sum_{-d} x_{w,d}\mu_{w,d}(k)$, f_{θ_d} and f_{ϕ_w} are the factor functions that encourage or penalize the incoming messages. Based on the topic smoothness prior, we follow [7] and design f_{θ_d} and f_{ϕ_w} as follows: $f_{\theta_d} = 1/[\sum_k [\mu_{-w,d}(k) + \alpha]]$, $f_{\phi_w} = 1/[\sum_w [\mu_{w,-d}(k) + \beta]]$.

In Fig. 1A, the citation link c connects a document pair $\{d, d_c\}$, and the factor η_c connects word topic labels $\mathbf{z}_{\cdot,d}$ and $\mathbf{z}_{\cdot,d_c}$. We assume that the cited documents are more likely to have similar topics, and use the similarity matrix $L(d, d_c)$ to encourage similar topics. The similarity matrix is built using the standard cosine similarity between two vectors $\mathbf{x}_{\cdot,d}$ and $\mathbf{x}_{\cdot,d_c}$. The higher similarity $L(d, d_c)$ the more influence of the passed message. Note that L is a pre-computed similarity matrix as one of inputs for the MBP algorithm. The message $\mu_{\eta_c \to z_{w,d}}(k)$ receives all the incoming message sending to the factor η_c from the variable node $\mathbf{z}_{\cdot,d_c}$, which can be calculated as follows,

$$\mu_{\eta_c \to z_{w,d}}(k) = \frac{\sum_{d_c} L(d, d_c)\mu_{\cdot,d_c}(k)}{\sum_k \sum_{d_c} L(d, d_c)}, \tag{10}$$

where the denominator is a normalization function in terms of k.

Fig. 1A shows that coauthor links can be also encoded into the factor graph. For example, the document pair $\{d, d_a\}$ is connected by the author index a. The factor δ_a connects the documents written by the same author, which in turn indicates that the research expertise of an author could be characterized by the document topics. We assume that documents written by the same author are more correlated than other documents, so we design the factor function f_{δ_a} as follows,

$$f_{\delta_a} = \frac{\sum_{d,d_a \in D_a} \boldsymbol{\mu}_{\cdot,d} \circ \boldsymbol{\mu}_{\cdot,d_a}}{|d, d_a \in D_a|}, \tag{11}$$

where the notation $\circ$ denotes the Hadamard (element-wise) product [2] between two vectors, D_a is a set of documents that are associated with author a, $|d, d_a \in D_a|$ indicates the total number of document pairs connected with the author a. The Hadamard product captures similarity between topic proportions of the connected documents with the author a. Consequently, Eq. (11) is the average Hadamard product of all pairs of documents connected with the author a, which encourages that documents written by the same author tend to have similar topic proportions. Based on (11), the message from factor δ_a is calculated as follows,

$$\mu_{\delta_a \to z_{w,d}}(k) = f_{\delta_a} \sum_{d_a \in D_a \setminus d} \boldsymbol{\mu}_{\cdot,d_a}. \tag{12}$$

where $d_a \in D_a \setminus d$ denotes all connected documents with the author a except the current document d in Fig. 1A.

In (8),(9),(10) and (12), we have shown how to calculate the four types of messages from factors to variables in Fig. 1B. Then, we update the message $\mu_{w,d}(k)$ using (6). Note that we have to estimate the weight vector $\boldsymbol{\lambda}$ to balance different sources of incoming messages.

In the M-setup, we estimate the parameters θ_d and ϕ_w based on the inferred messages. To estimate parameters θ and ϕ, we follow [7],

$$\theta_d(k) = \frac{\boldsymbol{\mu}_{\cdot,d}(k) + \alpha}{\sum_k [\boldsymbol{\mu}_{\cdot,d}(k) + \alpha]}, \tag{13}$$

$$\phi_w(k) = \frac{\boldsymbol{\mu}_{w,\cdot}(k) + \beta}{\sum_w [\boldsymbol{\mu}_{w,\cdot}(k) + \beta]}, \tag{14}$$

which are actually the normalized sum of messages flowing to factors θ_d and ϕ_w. Similar to the document-specific proportion θ_d, we can view the factor δ_a as the author-specific topic proportion,

$$\delta_a(k) = \frac{\sum_{d_a \in D_a} \boldsymbol{\mu}_{\cdot,d_a}(k)}{|D_a|}, \tag{15}$$

The underlying intuition behind the factor δ_a is that the topic proportion of each author is determined by the average topic proportions of his/her written documents.

3.3 Learning Link Weights

Learning link weights is a challenging problem because different links play different roles in multiplex document networks. One possible method is to optimize the objective (2) in terms of the weight vector $\boldsymbol{\lambda}$. But this strategy does not work in practice for two main reasons. First, $\boldsymbol{\lambda}$ introduces more free parameters in (2) that make the optimization problem complicated. Second, $\boldsymbol{\lambda}$ often varies during the topic modeling process to reflect the dynamical change of influence from different types of links. This dynamic property of $\boldsymbol{\lambda}$ may make the optimization problem even harder.

Here we propose a consistency-based method to quantify $\boldsymbol{\lambda}$ at each learning iteration. Our intuition is that if the messages received by the factor node are more consistent, their influence weight should be also larger. For example, if two cited papers have quite different topics implying inconsistency, their influence to the citing document will be small. If two coauthored papers have quite similar topics implying consistency, their influence to the current document will be large. As a result, the link weight is proportional to the consistency of messages.

Inspired by [16–18], we use the absolute difference of messages at successive learning iterations $t-1$ and t to quantify the consistency of messages,

$$\lambda_d = \sum_d \sum_k \sum_w |\mu_{\theta_d \rightarrow z_{w,d}}(k)^t - \mu_{\theta_d \rightarrow z_{w,d}}(k)^{t-1}|, \tag{16}$$

$$\lambda_c = \sum_d \sum_k \sum_w |\mu_{\eta_c \rightarrow z_{w,d}}(k)^t - \mu_{\eta_c \rightarrow z_{w,d}}(k)^{t-1}|, \tag{17}$$

$$\lambda_a = \sum_d \sum_k \sum_w |\mu_{\delta_a \rightarrow z_{w,d}}(k)^t - \mu_{\delta_a \rightarrow z_{w,d}}(k)^{t-1}|. \tag{18}$$

The larger accumulated difference means that the passed messages are more consistent owing to the fast convergence speed. So, we can update the weights $\boldsymbol{\lambda}$ of different types of links by (16), (17) and (18). Note that we need to normalize weights to satisfy $\lambda_d + \lambda_a + \lambda_c = 1$ at each learning iteration. Our experiments confirm that the consistency-based link weights work very well in practice.

Fig. 2 summarizes the MBP algorithm for MTM. For each learning iteration t, the E-step updates the message $\mu_{w,d}(k)$, and in the meanwhile estimate the link weight $\boldsymbol{\lambda}$. The M-step estimates the parameters including the document-specific topic proportion $\theta_d(k)$, the topic-specific distribution $\phi_w(k)$, and the author-specific topic proportion $\delta_a(k)$. The computational complexity of MBP is $\mathcal{O}(TKDCA)$, where T is the number of learning iterations, K the number of topics, D the number of documents, C the number of citation links, and A the number of authors. Theoretical proof of the convergence of MBP is out of the scope of this paper. Generally, MBP can be viewed as a generalized EM algorithm [8]. By repeatedly running E step and M step, MBP can converge to a local maximum of objective (2) under a fixed number of iterations.

```
input     : $W, T, K, L(d, d_c), \lambda_d, \lambda_a, \lambda_c$.
output    : $\theta_d, \phi_w, \delta_a$.
initialize: $\mu_{z_{w,d}}(k) \rightarrow random\ initialization\ and\ normaization, \lambda_d = \lambda_a = \lambda_c = 1/3$.
begin
    for $t \leftarrow 1$ to $T$ do
        1. E-step:
            $\mu_{\theta_d \rightarrow z_{w,d}}(k) = \frac{\mu_{-w,d}(k)+\alpha}{\sum_k[\mu_{-w,d}(k)+\alpha]}$;
            $\mu_{\phi_w \rightarrow z_{w,d}}(k) = \frac{\mu_{w,-d}(k)+\beta}{\sum_w[\mu_{w,-d}(k)+\beta]}$;
            $\mu_{\eta_c \rightarrow z_{w,d}}(k) = \frac{\sum_{d_c} L(d,d_c)\mu_{\cdot,d_c}(k)}{\sum_k \sum_{d_c} L(d,d_c)\mu_{\cdot,d_c}(k)}$;
            $\mu_{\delta_a \rightarrow z_{w,d}}(k) = f_{\delta_a} \sum_{d_a \in D_a \setminus d} \mu_{d,a}(k)$;
            $\mu_{w,d}(k) \propto \left[\lambda_d \mu_{\theta_d \rightarrow z_{w,d}}(k) + \lambda_c \mu_{\eta_c \rightarrow z_{w,d}}(k) + \lambda_a \sum_{a \in \mathcal{A}_d} \mu_{\delta_a \rightarrow z_{w,d}}(k)\right] \times \mu_{\phi_w \rightarrow z_{w,d}}(k)$;
            $\lambda_d = \sum_d \sum_k \sum_w |\mu_{\theta_d \rightarrow z_{w,d}}(k)^t - \mu_{\theta_d \rightarrow z_{w,d}}(k)^{t-1}|$;
            $\lambda_c = \sum_d \sum_k \sum_w |\mu_{\eta_c \rightarrow z_{w,d}}(k)^t - \mu_{\eta_c \rightarrow z_{w,d}}(k)^{t-1}|$;
            $\lambda_a = \sum_d \sum_k \sum_w |\mu_{\delta_a \rightarrow z_{w,d}}(k)^t - \mu_{\delta_a \rightarrow z_{w,d}}(k)^{t-1}|$;
        2. M-step:
            $\theta_d(k) = [\mu_{\cdot,d}(k) + \alpha] / \sum_k[\mu_{\cdot,d}(k) + \alpha]$;
            $\phi_w(k) = [\mu_{w,\cdot}(k) + \beta] / \sum_w[\mu_{w,\cdot}(k) + \beta]$;
            $\delta_a(k) = \sum_{d_a \in D_a} \mu_{\cdot,d_a}(k) / |D_a|$;
    end
end
```

Fig. 2. The multiplex belief propagation (MBP) algorithm

Table 1. Statistics of document data sets

Data sets	CL	D	W	C	A
CORA	7	2410	2961	8651	2480
DBLP	3	8023	3387	38165	6974

4 Experiments

We compare MTM with some sate-of-the-art link-based topic models such as RTM [2], ATM [1] and MRTM [3] on two publicly available data sets CORA [19] and DBLP[1]. For a fair comparison, the open source codes are implemented using MATLAB/MEX C++ platform [20]. For simplicity, we use the same hyperparameters $\alpha = 50/K, \beta = 0.01$, where K is the number of topics. Using the same $T = 500$ learning iterations, we compare these topic models in two tasks: document clustering and link prediction.

Table 1 summarizes the statistics of the two data sets, where CL is the number of document categories, D is the number of documents, W is the vocabulary size, C is the total number of citation links and A is the total number of authors. In the following experiments, we randomly divide the entire CORA and DBLP in halves, and choose one half as the training set and the other one as the test set for the link prediction task.

4.1 Document Clustering

Topic modeling techniques can be used as dimensionality reduction methods. The reduced document-specific topic proportions could be viewed as a soft clustering result. We can directly extract the clusters by assigning the cluster label with the highest topic proportion to each document, i.e., $C = \arg max_k \theta_d(k)$.

[1] `http://dblp.uni-trier.de/xml/`

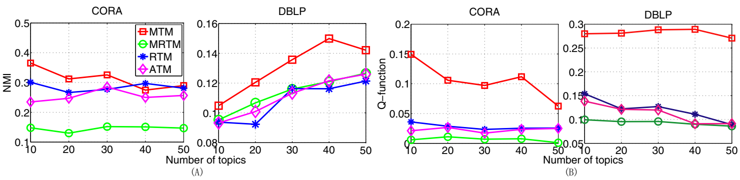


Fig. 3. Document clustering results: (A) NMI as a function of $K \in \{10, 20, 30, 40, 50\}$, (B) Q-function as a function of $K \in \{10, 20, 30, 40, 50\}$

We use two performance metrics to compare the document clustering results: normalized mutual information (NMI) [21] and Q-function [5, 22]. The former compares the predicted cluster labels C with the correct labels manually assigned by experts, and the latter measures the consistency of the clustering results over the network without using pre-existing labels. The value of NMI is in the range $[0, 1]$. It is higher if the predicted cluster labels are more consistent with the correct class labels. The value of Q-function lies in the range $[-1, 1]$. It is positive if the number of edges within groups exceeds the expected number on the basis of chance.

Fig. 3 shows the document clustering results of different link-based topic models. In Fig. 3A, on the CORA data set, MTM outperforms ATM and RTM around 23% and 10% in terms of NMI, respectively. Also, on the DBLP data set, MTM outperforms ATM and RTM more than 20% in terms of NMI, respectively. Such a salient improvement has been largely attributed to the ability of MTM in handling multiple types links in document networks. Although MRTM can also handle coauthor and citation links it has a sightly bad overall performance on document clustering. One reason is that MRTM is very sensitive to the hyperparameters α and β, and it is difficult to determine the best hyperparameters to achieve the overall good performance. Another reason is that MRTM is unable to balance topic influences from citation and coauthor links. In contrast, MTM dynamically tunes the link weight vector $\boldsymbol{\lambda}$ to reflect the truth that different links play different roles in topic modeling. Fig. 3B shows the Q-function on CORA and DBLP. Clearly, MTM achieves twice or three times higher Q-function values than those of RTM, ATM and MRTM, which implies that MTM can yield much more consistent document clustering results. As a summary, in the document clustering application, MTM significantly outperforms several state-of-the-art link-based topic models like RTM, ATM and MRTM.

4.2 Link Prediction

In this subsection, we perform two tasks of link prediction: citation link prediction and article recommendation. For the first task, we examine all algorithms on CORA and DBLP and compare MTM with RTM, ATM and MRTM. We follow the experimental setup in RTM [2] by first fitting the model to the training set with citation links, and then use a logistic regression model to predict the links on the test set. The input to the logistic regression model is the Hadamard

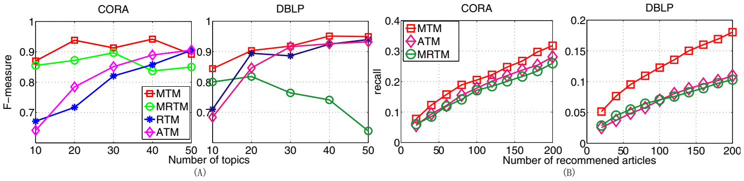


Fig. 4. Link prediction results: (A) F-measure as a function of topics $K \in \{10, 20, 30, 40, 50\}$, (B) $recall@M$ on out-of-matrix prediction tasks by varying the number of recommended articles $M \in \{20, 40, 60, \cdots, 200\}$

product of the topic proportions of each pair of documents. Citation link prediction results in terms of F-measure [3] are presented in Fig. 4A. MTM performs better than ATM, RTM and MRTM with 13%, 16% and 5% higher F-measure on CORA. Also, MTM yields 7%, 5% and 22% higher F-measure than ATM, RTM and MRTM on DBLP. The reason why the performance improvement on DBLP is much higher than that on CORA is that DBLP contains more citation links and authors than CORA, which shows the advantage of MTM in handling multiple types of links. Overall, the citation link prediction result demonstrates the effectiveness of MTM for modeling multiplex document networks.

Besides citation link prediction, MTM can address the out-of-matrix prediction problem [23] in recommending scientific articles to researchers very well. In this task, we consider how to recommend the newly published articles to the readers. We assume that the author a writes a paper d indicating that the author a likes reading the paper d. Thus, we form the prediction of whether an author a will like a paper d with the inner product between their topic proportions, $r_{a,d} = \delta_a^T \theta_d$. In our model, δ_a denotes the author-specific topic proportion and θ_d is the document-specific topic proportion. We will present each author with M articles sorted by their predicted ratings $r_{a,d}$ and evaluate based on which of these papers were actually written by the authors. We use the $recall@M$ [23] metric to evaluate the article recommendation performance. For each author, the definition of $recall@M$ is

$$recall@M = \frac{\text{number of artices the author likes in top M}}{\text{total number of article the author likes}}, \tag{19}$$

The above equation calculates the user-specific recall, and the overall recall for the entire system can be summarized using the average recall from all authors. We only consider the rated articles within the top M articles. A higher recall with lower M indicates a better system. When we use MTM, ATM and MRTM to recommend articles, we set $K = 50$ and $M \in \{20, 40, 60, \cdots, 200\}$. Note that RTM can only process citation links so it cannot do article recommendation task.

Fig. 4B shows the performance of different models for out-of-matrix prediction. When compared with ATM and MRTM, MTM works much better for out-of-matrix prediction with 16% and 21% recall enhancement on CORA. Also, MTM has a 43% and 44% higher recall than ATM and MRTM on DBLP, when

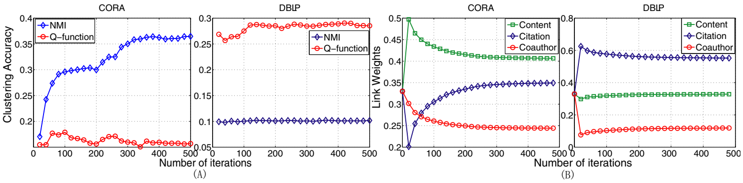


Fig. 5. (A) Clustering accuracy of the number of iterations when the topic $K = 10$, (B) Link weights by varying the number of iterations $T \in \{20, 40, 60, \cdots, 500\}$

the number of recommended documents M is small. The reason why MTM is superior to ATM and MRTM is that weighted citation links are incorporated into article recommendation. Although MRTM also uses citation links, it does not quantify and balance the influence of citation and coauthor links accounting for a relatively worse performance.

4.3 Analysis of Link Weights

One of major contributions of this work is to automatically estimate weights to balance citation and coauthor link information. Fig. 5A shows the clustering accuracy progresses along with the changes of the weights of different types of links. Fig. 5B shows how the learned weights change at different iterations and finally converge to the fixed values for the two data sets. We see that the content information has the highest weight in the topic modeling process for the CORA data set, while the citation has the highest weight for the DBLP data set. The major reason lies in that CORA uses the paper abstract but DBLP uses the paper titles as its content. Clearly, the content information of CORA is more important than that of DBLP. Moreover, DBLP has a much more number of citation links than CORA, so that the citation link information may play a major role in DBLP during the topic modeling process. Since the two metrics F-measure and Q-function can capture the characteristic of the citation network, the higher citation link weight for DBLP explains the much better citation link prediction results in Fig. 4A. As we see in Fig. 4B, the learned weights reflect the multiplex structure of document networks.

5 Conclusions

In this paper, we propose a novel MTM for multiplex document networks. This model has the following advantages. First, MTM naturally represent both citation and coauthor relations using the factor graph, which clearly illustrates the topic labeling dependencies in the multiplex document networks. Furthermore, this factor graph can be extended to represent more types of relational information of scientific papers such as venue and time. Second, this factor graph facilitates efficient MBP algorithm for approximate inference and parameter estimation within the EM framework. In the E-step, we infer the posterior topic distribution over each word. In the M-step, we estimate parameters based on the inferred messages. Finally, MBP uses a consistency-based method to automatically learn the link weights that can balance different types of link information during the message passing process. Experimental results on document clustering and link prediction show that MTM achieves the best performance among several state-of-the-art link-based topic models. Since our model has shown promising results on article recommendation, we are interested in designing more accurate recommendation system by incorporating more meta-data into the proposed MTM based on the factor graph representation.

Acknowledgements. This work is supported by NSFC (Grant No. 61003154), Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 12KJA520004), and a grant from Baidu to JZ, and General Research Fund (HKBU210410) from the Research Grant Council of the Hong Kong Special Administrative Region, China to WKC.

References

1. Steyvers, M., Smyth, P., Zvi, M.R., Griffiths, T.: Probabilistic author-topic models for information discovery. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 306–315 (2004)
2. Chang, J., Blei, D.M.: Hierarchical relational models for document networks. The Annals of Applied Statistics 4, 124–150 (2010)
3. Zeng, J., Cheung, W.K.,Hung Li, C., Liu, J.: Multirelational topic models. In: ICDM, pp. 1070–1075 (2009)
4. Liu, Y., Niculescu-Mizil, A., Gryc, W.: Topic-link LDA: joint models of topic and author community. In: ICML, p. 84 (2009)
5. Sun, Y., Han, J., Gao, J., Yu, Y.: iTopicModel: Information network-integrated topic modeling. In: ICDM, pp. 493–502 (2009)
6. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 47(2), 498–519 (2001)
7. Zeng, J., Cheung, W.K., Liu, J.: Learning topic models by belief propagation. IEEE TPAMI abs/1109.3437 (2012)
8. Bishop, C.M.: Pattern recognition and machine learning. Springer (2006)
9. Blei, D.M., Ng, A., Jordan, M.: Latent dirichlet allocation. JMLR 3, 993–1022 (2003)
10. Mei, Q., Cai, D., Zhang, D., Zhai, C.: Topic modeling with network regularization. In: WWW, pp. 101–110 (2008)
11. Deng, H., Han, J., Zhao, B., Yu, Y., Lin, C.X.: Probabilistic topic models with biased propagation on heterogeneous information networks. In: KDD, pp. 1271–1279 (2011)
12. Griffiths, T.L., Steyvers, M.: Finding Scientific Topics. PNAS 101 (suppl. 1), 5228–5235 (2004)
13. Heinrich, G.: Parameter estimation for text analysis. Technical report (2004)
14. Teh, Y.W., Newman, D., Welling, M.: A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In: NIPS, pp. 1353–1360 (2006)
15. Zeng, J., Liu, Z.Q.: Markov random field-based statistical character structure modeling for handwritten Chinese character recognition. IEEE TPAMI 30(5), 1193–1198 (2008)

16. Elidan, G., McGraw, I., Koller, D.: Residual belief propagation: Informed scheduling for asynchronous message passing. In: UAI, pp. 165–173 (2006)
17. Zeng, J., Cao, X.-Q., Liu, Z.-Q.: Residual belief propagation for topic modeling. In: Zhou, S., Zhang, S., Karypis, G. (eds.) ADMA 2012. LNCS, vol. 7713, pp. 739–752. Springer, Heidelberg (2012)
18. Zeng, J., Liu, Z.Q., Cao, X.Q.: A new approach to speeding up topic modeling. CoRR abs/1204.0170 (2012)
19. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automaticing the construction of internet portals with machine learning. Information Retrival 3(2), 127–163 (2000)
20. Zeng, J.: A topic modeling toolbox using belief propagation. JMLR 13, 2233–2236 (2012)
21. Strehl, A., Ghosh, J.: Cluster ensembles — a knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research 3, 583–617 (2002)
22. Newman, M., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69, 026113 (2004)
23. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: KDD, pp. 448–456 (2011)

Integrating Clustering and Ranking on Hybrid Heterogeneous Information Network

Ran Wang[1], Chuan Shi[1], Philip S. Yu[2,3], and Bin Wu[1]

[1] Beijing University of Posts and Telecommunications, Beijing, China
{wangran51,shichuan,wubin}@bupt.edu.cn
[2] University of Illinois at Chicago, IL, USA
[3] King Abdulaziz University Jeddah, Saudi Arabia
psyu@cs.uic.edu

Abstract. Recently, ranking-based clustering on heterogeneous information network has emerged, which shows its advantages on the mutual promotion of clustering and ranking. However, these algorithms are restricted to information network only containing heterogeneous relations. In many applications, networked data are more complex and they can be represented as a hybrid network which simultaneously includes heterogeneous and homogeneous relations. It is more promising to promote clustering and ranking performance by combining the heterogeneous and homogeneous relations. This paper studied the ranking-based clustering on this kind of hybrid network and proposed the ComClus algorithm. ComClus applies star schema with self loop to organize the hybrid network and uses a probability model to represent the generative probability of objects. Experiments show that ComClus can achieve more accurate clustering results and do more reasonable ranking with quick and steady convergence.

Keywords: Clustering, Ranking, Heterogeneous Information Network, Probability Model.

1 Introduction

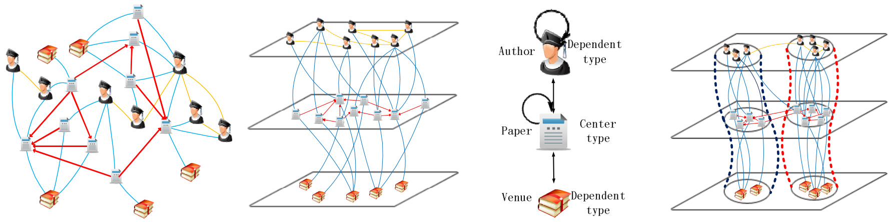
Information network analysis is an increasingly important direction in data mining in the past decade. Many analytical techniques have been developed to explore structures and properties of information networks, among which clustering and ranking are two primary tasks. The clustering task [1] partitions objects into different groups with similar objects gathered and dissimilar objects separated. Spectral method [1,4] is widely used in graph clustering. The ranking task [6,10,12] evaluates the importance of objects based on some ranking function, such as PageRank [12] or MultiRank [10]. Clustering and ranking are often regarded as two independent tasks and they are applied separately to information network analysis. However, integrating clustering and ranking makes more sense in many applications [2-3,11]. On one hand, the knowledge of important objects in a cluster helps to understand this cluster; on the other hand, knowing clusters is benefited to make more elaborate ranking. Some preliminary works have explored this issue [11].

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 583–594, 2013.

Although it is a promising way to do clustering and ranking together, previous approaches confine it to a “pure” heterogeneous information network which does not consider the homogeneous relations among same-typed objects. For example, RankClus [2] only considers relations between two-typed objects; NetClus [3] just considers relations among center type and attribute types. However, in many applications, the networked data are more complex. They include heterogeneous relations among different-typed objects as well as homogeneous relations among same-typed objects. Taking bibliographic data as an example which is shown in Fig. 1(a), papers, venues, authors and their relations construct a heterogeneous information network. Simultaneously, the network also includes the citation relations among papers and the social network among authors. It is important to cluster on such a hybrid network which includes heterogeneous and homogeneous relations at the same time. The hybrid network can more authentically represent real networked data. Moreover, more information from heterogeneous and homogeneous relations is promising to promote the performance of clustering and ranking.

Although it is important to integrate clustering and ranking on the hybrid network, it is seldom studied due to the following challenges. 1) It is difficult to effectively organize networked data. The hybrid network is more complex than either of them. The way to organize the network not only needs to effectively represent objects and their relations but also benefits for clustering and ranking analysis. 2) It is not easy to integrate information from heterogeneous and homogeneous relations to improve clustering and ranking performances. It is obvious that more information from different sources can help to obtain better performances. However, we need to design an effective mechanism to make full use of information from these two networks.

In this paper, we study the ranking based clustering problem on a hybrid network and propose a novel ComClus algorithm to solve it. A star schema with self loop is applied to organize the hybrid network. The ComClus employs a probability model to represent the generative probability of objects and the experts model and generative method are used to effectively combine the information from heterogeneous and homogeneous relations. Moreover, through applying the probability information of objects, we propose ComRank to identify the importance of objects based on ComClus. Experiments on DBLP show that ComClus achieves better clustering and ranking accuracy compared to well-established algorithms. In addition, ComClus has better stability and quicker convergence.



(a) Bibliographic data (b)Hybrid network (c)Star schema with self loop (d)Clusters on hybrid network

Fig. 1. An example of clustering on bibliographic data

2 Problem Formulation

In this section, we give the problem definition and some important concepts used in this paper.

Definition 1. ***Information Network.*** Given $K+1$ types of nodes, V^k is a vertex set, denoted by $V^k = \{v_0^k, v_1^k, v_2^k, \ldots\ldots, v_n^k\}$, where v_n^k represents the $n\text{-}th$ node belonging to the $k\text{-}th$ type. An information network can be represented as a weighted network $G=< V,E,W >$, if $V = \bigcup_{k=0}^{K} V^k$. E is a binary relation on V, and W is a weight mapping from an edge $e\in E$ to a real number $w\in R^+$. If $K \geq 2$ the information network G is ***heterogeneous information network;*** *and* ***homogeneous information networ*k** when $K=1$.

For a network with multiple types of nodes, *K*-partite network [7,9] and star schema [3] are widely used. These network structures only have heterogeneous relations among different-typed nodes, without considering the homogeneous relations among same-typed nodes. However, real networked data are more complex hybrid networks where links exist not only in heterogeneous nodes but also in homogeneous nodes. So we propose the star schema with self loop for this kind of networks.

Definition 2. *Star schema with self loop network.* An information network $G =< V,E,W >$ on *K+1* types of nodes $V = \bigcup_{k=0}^{K} V^k$ is called star schema with self loop network, $E = E_{homo} \cup E_{hete}$ and $E_{homo} \cap E_{hete} = \emptyset$. If $\forall e =< v_i^0, v_j^k > \in E_{hete}$, $v_i^0 \in V^0 \wedge v_j^k \in V^k (k \neq 0)$. If $\forall$e $=< v_i^k, v_j^k > \in E_{homo}$, $v_i^0 \in V^0 \wedge v_j^0 \in V^0$ ($k = 0$) or $v_i^k \in V^k \wedge v_j^k \in V^k (k \neq 0)$. Type $V^0$is called the center type (denoted as V^c), and $V^k (k \neq 0)$ is called dependent types (denoted as V^d).

E_{homo} is the links set among the same-typed nodes (called homo-link) and E_{hete} is the links set among the different-typed nodes (called hete-link). Then the hete-link can be written as e<v_i^c, v_j^d>, representing the link between center node and dependent node. The homo-link is the link between two same-typed nodes, which is denoted as e<v_i^c,v_j^c> or e<v_i^d,v_j^d>.

Fig. 1 shows such an example. For a complex bibliographical data (see Fig. 1(a)), we can organize it as a hybrid network which includes heterogeneous network among different layers and homogeneous network on the same layer in Fig.1 (b). As shown in Fig. 1(c), the hybrid network can be represented with a star schema with self loop where "paper" is the center type, while "venue" and "author" are dependent types.

Now, we can formulate the problem of clustering on hybrid network. Given a network $G =< V,E,W >$, $V = \bigcup_{k=0}^{K} V^k$ and the cluster number N, our goal is to find a clusters set $C = \bigcup_{n=1}^{N} C_n$, where C_n is defined as $C_n =< G', P_n >$. $G'$is a subnet of G, $E' \subseteq E, V' \subseteq V$ and $\forall e =< v_i^p, v_j^q > \in E'$. The probability function P_n represents the possibility that node v_i^p belongs to clusterC_n, $P_n(v_i^p) \in [0,1]$, and $\sum_{n=1}^{N} P_n(v_i^p) = 1$. In our solution, we restrict probability function of center node $P_n(v_i^p) \in \{0,1\}$, and for dependent node v_j^d, P_n is the successive probability measure from 0 to 1.

3 The ComClus Algorithm

After introducing the basic framework of ComClus, this section describes the ComClus in detail and then proposes ComRank for estimating the importance of objects.

3.1 The Framework of ComClus

The basic idea of ComClus is to determine the memberships of center nodes and then estimate the memberships of dependent nodes by center nodes. We consider that the probability of center node is estimated by two probabilities: homogeneous probability and heterogeneous probability. The homogeneous probability of center node depends on its homo-links. The heterogeneous probability of center node is generated by the dependent nodes that are correlated with it. In order to co-consider the heterogeneous and homogeneous probability for center nodes, generative method and experts model are used to mix these two types information. Finally, we estimate the posterior probability for center node according to the Bayesian rule and reassign the memberships of center nodes. The ComClus will iteratively calculate posterior probability until the memberships do not change. Algorithm 1 shows the basic framework of ComClus.

Algorithm 1. ComClus: Detecting *N* clusters on hybrid information network

```
Input: Cluster number N and hybrid network G
Output: Membership of center node, the posterior probability of dependent node
Begin:
    Randomly partition on network G
    Calculate global probability of center node for smoothing: $p(v_i^c|G)$
    repeat
        foreach subnet $G_n \subseteq G$
            Calculate the homogeneous probability of center node: $p(v_i^c|C, G_n)$
            Calculate the conditional probability of dependent node: $p(v_i^d|G_n)$
            Calculate the heterogeneous probability of center node: $p(v_i^c|D^i, G_n)$
            Calculate the mixed probability: $p(v_i^c|G_n)$
        end
        Calculate the center node posterior probability: $p(G_n|v_i^c)$ and Reassign
    until $\vec{D}(V_i^c)$ convergence obtained
    Calculate the dependent node posterior probability: $p(G_n|v_i^d)$
End
```

3.2 Homogeneous Probability for Center Node

The homogeneous probability of v_i^c depends on its homo-links and denotes as $p(v_i^c|C, G)$. $p(v_i^c|C, G)$ represents the fraction of links that the center node v_i^c connects to other center nodes on G. This idea is inspired by a general phenomenon that a node has higher probability to connect with nodes within the same cluster.

For convenience, $hodeg(\ v_i^c|G)$ denotes the number of homo-links of v_i^c and the number of in- degree of center node v_i^c on homogeneous network is denoted as $in(v_i^c|G)$.

$$p(\ v_i^c|C,G) = \frac{hodeg(\ v_i^c|G)}{\sum_{i=1}^{|V^c|} hodeg(\ v_i^c|G)} \tag{1}$$

$$QuotedRate(\ v_i^c|G) = \frac{in(\ v_i^c|G)}{\sum_{i=1}^{|V^c|} in(\ v_i^c|G)} \tag{2}$$

The value of $QuotedRate(v_i^c|G)$ is calculated by the quoted times of v_i^c on G, which will be used to rank (in Sect.3.7 Eq. (11)) and filter the unimportant nodes (in Sect.3.3 Eq. (3)) in our algorithm. The center node v_i^c has higher possibility to be assigned into a cluster with higher $p(\ v_i^c|C,G)$. Therefore, the clustering result will benefit from the homogeneous information.

3.3 Conditional Probability for Dependent Node

We consider that the heterogeneous probability of center node v_i^c is generated by its related dependent nodes v_i^d. Therefore, we need to estimate the probability of v_i^d, which can be represented as $p(v_i^d|G) = p(d|G) \times p(v_i^d|d\ ,G)$. The probability of dependent type d being selected is $p(d\ |G) = \frac{|V^d|}{|V|}$, where $|V^d|$ is the number of nodes in dependent type d layer, and $|V|$ is the number of all nodes in G. After the type d being selected, the probability $p(v_i^d|d\ ,G)$ can be estimated. We utilize the two dependent types d_a, d_b to mutually estimate the probability for $p(v_i^{d_a}|d_a\ ,G)$ and $p(v_i^{d_b}|d_b\ ,G)$. D^i is the related dependent type set of v_i^c. Take $p(v_i^{d_a}|d_a\ ,G)$ as an instance. By taking advantage of the homogeneous information of $v_i^{d_a}$, we set $p(v_i^{d_a}|d_a\ ,G) = \frac{hodeg(v_i^{d_a}|G)}{\sum_{i=1}^{|V^{d_a}|} hodeg(v_i^{d_a}|G)}$ at the beginning of iteration. We consider the center node v_i^c is the medium between $v_i^{d_a}$ and $v_i^{d_b}$. Naturally, an important medium v_i^c should have a higher $QuotedRate(\ v_i^c|G)$ than an ordinary one. Besides, we use θ as a filter factor to expand the $QuotedRate(\ v_i^c|G)$ gap among ent v_i^c. Repeat calculating (4) and (5) until the convergence is obtained.

$$\theta = \begin{cases} 1\ if\ QuotedRate(\ v_i^c|G) < avgQuotedRate(V^c|G) \\ \theta\ if\ QuotedRate(\ v_i^c|G) \geq avgQuotedRate(V^c|G) \\ 0\ otherwise \end{cases} \tag{3}$$

$$score(\ v_i^c|G) = \ \theta \times QuotedRate(\ v_i^c|G) \times \sum_{i=1}^{|V^{d_b}|} \frac{e<v_i^c, v_i^{d_b}> \times p(v_i^{d_b}|d_b\ ,G)}{hedeg(v_i^{d_b})} \tag{4}$$

$$p(v_i^{d_a}|d_a\ ,G) = \sum_{i=1}^{|V^c|} \frac{e<v_i^c,v_i^{d_a}> \times score(\ v_i^c|G)}{hedeg(v_i^{d_a})} \tag{5}$$

where $hedeg(v_i^{d_a})$ is the number of hete-links of $v_i^{d_a}$on G. We run the same process for $v_i^{d_b}$ to get the probability $p\left(v_i^{d_a}\middle|d_a, G\right)$. As a res ult, the "productive" dependent nodes and the "barren" nodes can be distinguished obviously. Normalization method can be used when necessary.

3.4 Heterogeneous Probability for Center Node

After conditional probability of dependent nodes being figured out, we can estimate the heterogeneous probability for v_i^c. Here, we make an independency assumption that the dependent nodes generate the heterogeneous probability of center node independently. Given dependent node probabilities which are related to v_i^c, the heterogeneous probability of center node v_i^c can be denoted as $p(\, v_i^c|D^i, G)$.

$$p(v_i^c|D^i, G) = \prod_d^{D^i} \prod_{i=1}^{|V^d|} p\left(v_i^d\middle|d\,, G\right) \tag{6}$$

3.5 Mixed Probability for Center Node

Until now, we obtain the homogeneous and heterogeneous probability of center node v_i^c. Next, the major difficulty in estimating the probability measure is how to jointly consider the homogeneous and heterogeneous distribution of center nodes. To mix the two distributions, we employ two methods: a generative method of center node and a mixture of experts model [5].

In the generative method, we consider the center n ode v_i^c is generated by two parts: the homogeneous and h eterogeneous information of v_i^c . The former is $p(\, v_i^c|C, G)$ and the latter is $p(v_i^c|D^i, G)$. We can calculate the conditional probability on hybrid network G as follows:

$$p(v_i^c|G) = p(\, v_i^c|C, G) \times p(v_i^c|D^i, G) \tag{7}$$

In experts model, we regard the homogeneous and heterogeneous information of v_i^c as "homogeneous expert" and "heterogeneous expert". Then we can evaluate mixed probability of center node according to its own distribution. The mixture of experts model is denoted as follows:

$$p(v_i^c|G) = \sum_{m=1}^{M} \pi_m p_m(v_i^c|G) \tag{8}$$

where $M\; = 2$ represents the number of experts. If $m = 1,$ the homogeneous expert takes into effect: $p_1(v_i^c|G)$=$p(\, v_i^c|C, G)$. If $m = 2$, the heterogeneous expert is activated as: $p_2(v_i^c|G) = p(v_i^c|D^i, G)$. $\pi_m = \frac{p_m(v_i^c|G)p(E_m)}{\sum_{m=1}^{M} p_m(v_i^c|G)p(E_m)}$ can be seen as the weight of corresponding expert, and we adopt *Softmax* function to compute it. $p(E_m)$ is the weight of expert *m*, which is proportional to the number of heter-links or homo-links of v_i^c. For example, the weight of homogeneous expert of v_i^c is calculated by the following formula: $p(E_1) = \frac{hodeg(v_i^c|G)+1}{hodeg(v_i^c|G)+\sum_d^{|D^i|} hedeg(v_i^d|G)}$. Because we only have two experts, the $p(E_2)$ is simply set as 1- $p(E_1)$. Obviously, the weight is dynamic for each v_i^c.

Both methods can evaluate the conditional probability of center node, which can be applied to different scenarios. The generative method equally treats the homogeneous and heterogeneous information, because it simply products homogeneous and heterogeneous probability. Therefore, the generative method is suitable for the hybrid network with the same scales of homogeneous and heterogeneous relations. The mixture of experts model can dynamically adjust the weights of distributions (by π_m). As a result, the method is more suitable for the hybrid network of which the homogenous and heterogeneous parts have different size.

Besides, to avoid zero probabilities, we smooth the distribution by the following formula: $p(v_i^c|G_n) = \lambda p(v_i^c|G_n) + (1-\lambda)p(v_i^c|G)$, where λ is a smoothing parameter. *G* is the whole hybrid network and G_n is the *n*-th subnet.

3.6 Posterior Probability for Nodes

In the previous subsection, we get the conditional probability of center node v_i^c by mixing two distributions. Now, we need to calculate the posterior probability $p(G_n|v_i^c)$ for each v_i^c, and reassign the memberships for center nodes. The posterior probability of center node can be calculated by Bayesian rule: $p(G_n|v_i^c) \propto p(v_i^c|G_n) \times p(G_n)$, where $p(v_i^c|G_n)$ is the conditional probability in cluster G_n and $p(G_n)$ represents the cluster size. However, the size of cluster G_n is not fixed. For the purpose of getting the $p(G_n)$, the EM algorithm can be used to get the local optimum $p(G_n)$ by maximizing the log likelihood of center nodes in different areas.

$$log\, P = \sum_{i=1}^{|V^c|} \log[\sum_{n=1}^{N+1} p(v_i^c|G_n) \times p(G_n)] \tag{9}$$

where $|V^c|$ is the size of V^c, and *N+1* represents the global distribution on *G*. The target is to maximize $log\, P$ and two iterative steps can be set to optimize the value *P*. We set $p^0(G_n) = \frac{1}{N+1}$ before the first iteration. The following two steps run iteratively until the convergence is obtained. $p^t(G_n|v_i^c) \propto p(v_i^c|G_n) \times p(G_n)$; $p^{t+1}(G_n) = \sum_{i=1}^{|V|} \frac{p^t(G_n|v_i^c)}{|V|}$. Finally, we will have a N dimensional indicator vector $\vec{D}(v_i^c)$, which is made up of posterior probability of v_i^c. Then we can calculate the indicator of membership for each center node with *K*-means.

After the iterative process is finished, the posterior probability of dependent node $p(G_n|v_i^d)$ can be evaluated by the average posterior probability of center nodes connecting with v_i^d. The notation S^i is a set o f center nodes connecting with v_i^d and $|S^i|$ is the size of set S^i.

$$p(G_n|v_i^d) = \sum_{i=1}^{|S^i|} \frac{p(G_n|v_i^c)}{|S^i|} \tag{10}$$

3.7 Ranking for Nodes

As an additional benefit for ComClus, the posterior probabilities of nodes can be used for ranking nodes. Once the cluster process is finished, we can further figure out the rank of nodes in their cluster. We proposed a function (called ComRank) to evaluate the importance of nodes.

$$Rank(v_i^c|G_n) = QuotedRate(v_i^c|G) \times p(G_n|v_i^c) \tag{11}$$

where $p(v_i^c|G_n)$ is the probability of center node v_i^c. Generally, the rank of center node is proportional to its $QuotedRate(v_i^c|G)$. It is natural in many applications. Taking bibliographic network as an example, the goodness of a paper is decided by the number of citations to a large extent. Another factor of rank function is the posterior probability, which can be seen as a cluster coefficient and represents the degree of membership in that cluster. The rank of dependent node v_i^d can be computed according to the rank of center nodes connecting with it.

$$Rank(v_i^d|G_n) = \sum_{i=1}^{|S^i|} Rank(v_i^c|G_n) \times p(G_n|v_i^d) \tag{12}$$

4 Experiment

In this section, we evaluate the effectiveness of our ComClus algorithm, and compare it with the state-of-the-art methods on two data sets.

4.1 Data Set

The DBLP is a dataset of bibliographic information in computer science domain. We use it to build a hybrid network with three-typed nodes: papers (center type), venues (dependent type) and authors (dependent type). Homo-links among authors form a co-author network, and homo-links among papers form a paper citation network. Hete-links are the writing relation between authors and papers and the publication relation between venues and papers. We extract venues from different areas according to the categories of China Computer Federation (http://www.ccf.org.cn). Moreover, CCF provides three levels for ranking venues: A, B, C. The class A is top venues, such as KDD in *data mining* (DM). The class B is some famous venues such as SDM, ICDM. The class C is admitted venues such as WAIM. In the experiments, we extract two different-scaled subsets of the DBLP which are called DBLP-L and DBLP-S.

The DBLP-S is a small size dataset and it includes three areas in computer domain: *database, data mining, and information retrieval.* There are 21venues (7 venues for each area, covering three levels), 25,020 papers and 10,907 authors in DBLP-S. Two or three venues for each level are picked out.

The DBLP-L is a large dataset. There are eight areas included, which are *computer network, information security, computer architecture, theory, software engineering & programming language, artificial intelligence& pattern recognition, computer graphics, data mining& information retrieval &database*. There are 280 venues

(35 venues for each area), 275,649 papers, and 238,673 authors. For each area, five venues are in A level and fifteen venues are selected in B or C level.

In these two datasets, venues are labeled with their research areas. Moreover, in DBLP-S, we randomly label 1031 papers and 1295 authors with three research areas, which are used to evaluate the clustering accuracy. All the results are based on 20 runnings, and average results are shown.

4.2 Clustering Accuracy Comparison Experiments

For accuracy evaluation, we apply our method to cluster on both DBLP-S and DBLP-L. We compare ComClus with the representative ranking-based clustering algorithm NetClus which can be applied in heterogeneous networks organized as star schema. The smoothing parameter λ is fixed at 0.7 in both two algorithms. The filter factor θ in ComClus is 3. The clustering accuracy of paper is the fraction of nodes identified correctly. For author and venue nodes, the accuracy is the posterior probability fraction of nodes identified correctly. Results are shown in Table 1. The two different mixture methods of ComClus both have higher accuracy than NetClus. The lower deviation of ComClus implies that ComClus is steadier than NetClus. The results show that, the additional homogeneous relation utilized by ComClus is helpful for improving its accuracy as well as stability. In addition, ComClus with experts model achieves better performance than ComClus with generative method. We think the reason is that experts model considers the weight of heterogeneous and homogeneous information. In the following experiments, we use ComClus with experts model as the standard version of ComClus.

Table 1. Clustering accuracy comparison for different-typed nodes

Accuracy	ComClus(experts method)		ComClus(generative method)		NetClus	
	Mean	Dev.	Mean	Dev.	Mean	Dev.
Paper(DBLP-S)	**0.774**	0.019	0.766	0.021	0.715	0.066
Venue(DBLP-S)	**0.855**	0.018	0.777	0.028	0.739	0.067
Author(DBLP-S)	**0.731**	0.018	0.680	0.016	0.697	0.052
Venue(DBLP-L)	**0.681**	0.041	0.648	0.046	0.579	0.084

Since the hybrid network includes homogeneous network, we compare ComClus with those clustering algorithms on homogeneous network, where a representative spectral clustering algorithm Normalize Cut [4] is employed. We design the similarity of two nodes (i,j) as: $S(i,j) = cos(V_i, V_j)$, where V_i is the adjacent vector of node i. The result is shown in Table 2, which clearly illustrates that ComClus is better than Normalized Cut. ComClus combines the information from homogeneous and heterogeneous relations. It makes ComClus outperform Normalized Cut which only uses homogeneous network information.

Table 2. Clustering accuracy comparison on homogeneous network

Accuracy	ComClus	Normalized Cut
Paper Accuracy	**0.787**	0.457

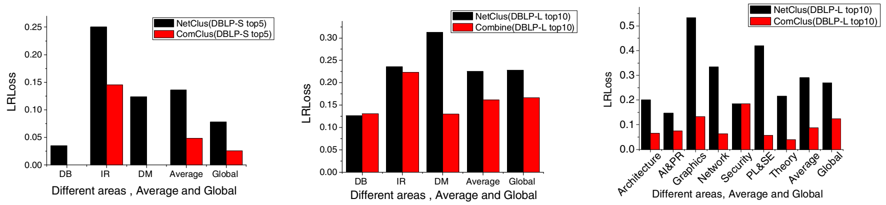
4.3 Ranking Accuracy Comparison Experiment

On DBLP-L, we make a ranking accuracy comparison between ComRank and AuthorithyRank which is a rank method in NetClus[3]. In this application, it is hard to definitely compare the goodness of two venues, whereas we can roughly distinguish their levels. For example, it is difficult to compare the ranking of SDM and ICDM. But we can safely say that SDM and ICDM are on the same level and they are worse than the top level venues (e.g., KDD) and better than the common level venues (e.g., WAIM). Inspired by *RankingLoss* measure [8], we define *LevelRankingLoss* to evaluate the disorder ratio of object pairs on their levels and it is abbreviated as *LRLoss*. Without loss of generality, we define *LRLoss* on bibliographic data. First, we define a triple to represent a venue:$RankTuple_i =< C_i, L_i, R_i >$, where C_i represents a venue, L_i is the level of C_i, $L_i \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$ (the recommended level of CCF). R_i is the rank number of C_i generated by the algorithms(the smaller, the better). The *LRLoss* is defined as follows.

$$LRLoss = \frac{1}{R}\sum_{i=1}^{R}\frac{|LossPair_i|}{|LossPair_i|+|\overline{LossPair_i}|} \tag{13}$$

where R is the size of Cartesian product of $RankTuple$ set and $LossPair_i = \{< RankTuple_i, RankTuple_j > | L_i < L_j, R_i > R_j \; or \; L_i > L_j, R_i < R_j\}$. Here, $\overline{LossPair_i}$ denotes the complementary set. $|LossPair_i|$ is the number of misordered pairs for $RankTuple_i$. For example, $RankTuple_1$= <KDD, A, 2>, $RankTuple_2$=<ICDM, B, 1> can be seen as one *LossPair* for $RankTuple_1$.

We select the top 5 and top 10 venues in different areas and then calculate *LRLoss* for them. Additionally, we also compare the accuracy of the global rank on both ComRank and NetClus. Results are shown in Fig2.



(a) 3 areas top5 venues on DBLP-S (b) 3 areas top10 venues on DBLP-L (c) 8 areas top 10 venues on DBLP-L

Fig. 2. Ranking accuracy comparison (The smaller *LRLoss*, the better)

The results clearly show that ComRank better ranks these venues, since its *LRLoss* is lower than that of AuthorityRank on all research areas. We think the additional homogeneous information utilized by ComRank contributes to its better ranking performance.

4.4 Case Study

In this section, we further show the performance of ComRank with a ranking case study.

Table 3. Top 15 venues with global rank on DBLP-S

ComRank	1	2	3	4 5		6	7	8	9	10	11	12	13	14	15
Venue	SIGMOD	VLDB	SIGIR	ICDE	KDD	PODS	WWW	CIKM	ICDM	EDBT	PKDD	WSDM	PAKDD	WebDB	DEXA
#Papers	2428	2444	2509	2832	1531	940	1501	2204	1436	747	680	198	1030	972	1731
Level	A	A	A	A A		A	B	B	B	B	B	B	B	C	C
AuthorityRank	1	2	3	4 5		6	7	8	**9**	**10**	11	12	13	14	15
Venue	VLDB	ICDE	SIGMOD	SIGIR	KDD	WWW	CIKM	ICDM	**PODS**	**DEXA**	PAKDD	EDBT	PKDD	WSDM	ECIR
#Papers	2444	2832	2428	2509	1531	1510	2204	1436	**940**	**1731**	1030	747	680	198	575
Level	A	A	A	A A		B	B	B	**A**	**C**	B	B B B			C

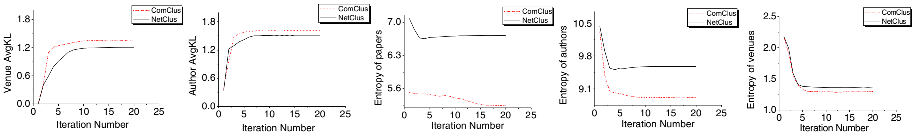
Table 3 sho ws the top 15 venues ranked by ComRank and AuthorityRank on DBLP-S. The results show that the ranks of venues generated by ComRank are all consistent with the recommended level by CCF. However, there are some disordered venues in AuthorityRank, which implies that AuthorityRank is sensitive to the number of papers. That is, AuthorityRank tends to rank a venue publishing many papers with a higher value. For example, AuthorityRank ranks PODS with a low value and DEXA with a relati vely high value because PODS published not many papers and DEXA published so many papers. In contrast, ComRank considers the citation information from homogeneous network. So ComRank avoids these shortcomings.

4.5 Convergence and Stability Experiments

For observing the convergence, we compare each cluster probability distribution with global distribution by average KL divergence [3]. Next, we use entropy to measure the unpredictability of cluster and prove the algorithm stability.

$$AvgKL(V^d) = \frac{1}{N}\sum_{n=1}^{N} D_{KL}(p(v_i^d|G_n)||p(v_i^d|G)) \quad (14)$$

$$AvgEntropy(V^p) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{|V^p|} p\left(v_i^p\middle|G_n\right) \times log\, p\left(v_i^p\middle|G_n\right) \quad (15)$$



(a)*AvgKL* of venues (b) *AvgKL* of authors (c)*AvgEntropy* of papers (d) *AvgEntropy* of authors (e)*AvgEntropy* of venues

Fig. 3. The change of *AvgKL* and *AvgEntropy* of nodes with iteration number

As shown in Fig. 3(a) and (b), the convergence of our algorithm is faster than NetClus. From the results shown in Fig. 3(c), (d) and (e), we can observe that ComClus achieves lower $gEntropy$. The reason is that ComClus prevents the negative effects of unimportant paper by the factor θ. Besides, in ComRank, the distribution information of objects comes from heterogeneous and homogeneous relations. However, the distribution information of objects in NetClus is o nly from heterogeneous network. More information helps ComClus fast converge and achieve steady solution.

5 Conclusions

In this paper, we proposed a new ranking-based clustering algorithm ComClus on heterogeneous information networks. Different from conventional clustering methods, ComClus can group different-typed objects on a hybrid network which includes the homogeneous network and heterogeneous relations together. Through applying probability information in ComClus, ComClus can also rank the importance of objects. The experiments on real datasets have demonstrated that our algorithm can generate more accurate cluster and rank with quicker and steadier convergence.

Acknowledgments. It is supported by the National Natural Science Foundation of China (No. 60905025, 61074128, 71231002). This work is also supported by the National Basic Research Program of China (2013CB329603) and the Fundamental Research Funds for the Central Universities.

References

1. Shen, H., Cheng, X.: Spectral Methods for the Detection of Network Community Structure: a Comparative Analysis. J. Stat. Mech., P10020 (2010)
2. Sun, Y., Han, J., Zhao, P., Yin, Z., Cheng, H., Wu, T.: Rankclus: Integrating Clustering with Ranking for Heterogeneous Information Network Analysis. In: EDBT, pp. 565–576 (2009)
3. Sun, Y., Yu, Y., Han, J.: Ranking-based Clustering of Heterogeneous Information Networks with Star Network Schema. In: KDD, pp. 797–806 (2009)
4. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. In: CVPR, pp. 731–737 (1997)
5. Jacobs, R.A., Jordan, M.I., Nowlan, S., Hinton, G.E.: Adaptive Mixtures of Local Experts. Neural Computation 3, 79–87 (1991)
6. Zhou, D., Orshanskiy, S., Zha, H., Giles, C.: Co-ranking Authors and Documents in a Heterogeneous Network. In: ICDM, pp. 739–744 (2007)
7. Liu, X., Murata, T.: Detecting Communities in K-partite K-uniform (Hyper) Networks. JCST 26(5), 778–791 (2011)
8. Zhang, M.L., Zhang, K.: Multi-label Learning by Exploiting Label Dependency. In: KDD, pp. 999–1008 (2010)
9. Long, B., Wu, X., Zhang, Z.M., Yu, P.S.: Unsupervised Learning on K-partite Graphs. In: KDD, pp. 317–326 (2006)
10. Michael, K.N., Li, X., Ye, Y.: MultiRank: Co-ranking for Objects and Relations in Multi-relational Data. In: KDD, pp. 1217–1225 (2011)
11. Ailon, N., Charikar, M., Newman, A.: Aggregating Inconsistent Information: Ranking and Clustering. J. ACM 55(5) (2008)
12. Brin, S., Page, L.: The Anatomy of a Large-scale Hyper Textual Web Search Engine. Comput. Netw. ISDN Syst. 30(1-7), 107–117 (1998)

Learning from Multiple Observers with Unknown Expertise

Han Xiao, Huang Xiao, and Claudia Eckert

Institute of Informatics
Technische Universität München, Germany
{xiaoh,xiaohu,claudia.eckert}@in.tum.de

Abstract. Internet has emerged as a powerful technology for collecting labeled data from a large number of users around the world at very low cost. Consequently, each instance is often associated with a handful of labels, precluding any assessment of an individual user's quality. We present a probabilistic model for regression when there are multiple yet some unreliable observers providing continuous responses. Our approach simultaneously learns the regression function and the expertise of each observer that allow us to predict the ground truth and observers' responses on the new data. Experimental results on both synthetic and real-world data sets indicate that the proposed method has clear advantages over "taking the average" baseline and some state-of-art models.

1 Introduction

With the recent advent of social web services, the data can now be shared and processed by a large number of users. As a consequence, researchers are faced with data sets that are labeled by multiple users. For example, Wikipedia provides a feedback tool to engage readers in the assessment of article quality based on four criteria, i.e. "trustworthy", "objective", "complete" and "well-written". The Amazon Mechanical Turk is an online system that allows the requesters to hire users from all over the world to perform crowdsourcing tasks. Galaxy Zoo is a website where visitors label astronomical images. While providing large amounts of cheap labeled data in a short time, these platforms usually have little quality control over users. Thus, the response of each user can vary widely, and in some cases may even be adversarial. A natural question to ask is how to integrate opinions from multiple users for obtaining an objective opinion. The commonly used "majority vote" and "take the average" heuristics completely ignore the individual expertise and may fail in the settings with non-Gaussian or adversarial noise. This casts a challenge of *learning from multiple sources* for the machine learning and data mining researchers [2].

Despite these web applications, one can find this problem in wide range of domains. Recently, *sensor networks* have been deployed for the scientific monitoring of remote and hostile environments. For example, researchers deployed a 16-node sensor network on a tree to study its elevation under different weather fronts [9]. Each node samples climate data at regular time intervals and the statistics are collected. Using sensor data in this manner presents many novel challenges, such as fusing noisy readings from several sensors, detecting faulty and aging sensors. Importantly, it is necessary to use the

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 595–606, 2013.

trends and correlations observed in previous data to predict the value of environmental parameters into the future, or to predict the reading of a sensor that is temporarily unavailable (e.g. due to network outages). However, these tasks may have to be performed with only limited knowledge of the location, reliability, and accuracy of each sensor.

In this work, the labeler (including user, annotator and sensor) mentioned above is referred to as the *observer*. Given an *instance*, the label (e.g. annotation, reading) provided by an observer is called the *response*. Unlike the conventional supervised learning scenario, in our setting each instance is associated with a set of responses, yet the *ground truth* is unknown as some responses may be subjective or come from unreliable observers. We concentrate on the regression problem with continuous responses from multiple observers. Specificall , our method provides a principled way to answer the following questions:

1. How to learn a regression function to predict the ground truth precluding the prior knowledge of observers?
2. How to estimate the expertise of each observer without knowing the ground truth?

2 Related Work and Novel Contributions

There is a number of studies dealing with the setting involving multiple labelers, yet most of them focus on the classificatio problem. Early work such as [3,4,8] focus on estimating the error rates of observers. In the machine learning community, the problem of estimating the ground truth from multiple noisy labels is addressed in [7]. Instead of estimating the ground truth and learning the classifier separately, recent interest has shifted towards on learning classifiers directly from such data. Authors of [2] provide a general theory of selecting the most informative samples from each source for model training. Later, a probabilistic framework is presented by [5,6] to address the classification regression and ordinal regression problem with multiple annotators. The framework is based on a simple assumption that the expertise of each annotator does not depend on the given data. This assumption is infringed in [10,13] and later is extended to the active learning scenario [12]. There are some other related work that focus on different settings [1,11].

The above studies paid little attention to the regression problem under multiple observers, which is the main core of this paper. Moreover, our work differs from the related work in various aspects. First, we employ a less-parametric method, i.e. the *Gaussian process* (GP), to model the observers and the regression function. This allows us to associate the observer's expertise with both ground truth and input instance. Moreover, our model is presented in an extensible probabilistic framework. The missing data and prior knowledge can be straightforwardly incorporated into the model.

The rest of this paper is organized as follows. Section 3 formulates the problem and introduces a probabilistic framework. The framework consists of two parts. The regression model is introduced in Section 3.2. A linear and a non-linear observer model is proposed in Section 3.3 and Section 3.4, respectively. Section 4 reports the experimental results on both synthetic and real-world data sets. Conclusions are drawn in Section 5.

3 Probabilistic Formulation

Denote the *instance space* $\mathcal{X} \subseteq \mathbb{R}^L$ and the *response space* $\mathcal{Y} \subseteq \mathbb{R}^D$ and the *ground truth space* $\mathcal{Z} \subseteq \mathbb{R}^D$. Given N instances $\mathbf{x}_1, \ldots, \mathbf{x}_N$ where $\mathbf{x}_n \in \mathcal{X}$, denote the *objective ground truth* for $\mathbf{x}_n$ as $\mathbf{z}_n \in \mathcal{Z}$. In our setting, the ground truth is unknown. Instead, we have multiple responses $\mathbf{y}_{n,1}, \ldots, \mathbf{y}_{n,M} \in \mathcal{Y}$ for $\mathbf{x}_n$ provided by M different observers. For compactness, the $N \times L$ matrix of instance $x_{n,l}$ is represented as $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$. The $N \times M \times D$ tensor of observers' responses $y_{n,m,d}$ is denoted by $\mathbf{Y} := [\mathbf{y}_{1,1}, \ldots, \mathbf{y}_{1,M}; \ldots; \mathbf{y}_{N,1}, \ldots, \mathbf{y}_{N,M}]$. The $N \times D$ matrix of ground truth $z_{n,d}$ is denoted by $\mathbf{Z} := [\mathbf{z}_1, \ldots, \mathbf{z}_N]^\top$.

Given the training data $\mathbf{X}$ and $\mathbf{Y}$, our goal is threefold. First, it is of interest to get an estimate of the unknown ground truth $\mathbf{Z}$. The second goal is to learn a regression function $f : \mathcal{X} \to \mathcal{Z}$ which generalizes well on unseen instances. Finally, for each observer we want to model its *expertise* as a function of the input instance and the ground truth, i.e. $g : \mathcal{X} \times \mathcal{Z} \to \mathcal{Y}$.

3.1 Probabilistic Framework

To formulate this problem from the probabilistic perspective, we consider the training data $\mathbf{X}$ and $\mathbf{Y}$ as random variables. The ground truth $\mathbf{Z}$ is unknown and hence is a latent variable. In general, the observed response $\mathbf{Y}$ depends both on the unknown ground truth and the instance. That is, observers may exhibit varying levels of expertise on different instances. On Wikipedia the assumption is particularly true for the novice readers, whereas the rating from an expert reader is consistent across different types of articles. Figure 1 illustrates the conditional dependence between $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ with a graphical model. As a consequence, the joint conditional distribution can be expressed as

$$\begin{aligned} p(\mathbf{Y}, \mathbf{Z}, \mathbf{X}) &= p(\mathbf{Z} \,|\, \mathbf{X}) p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}) p(\mathbf{X}) \\ &\propto \prod_{n=1}^{N} \prod_{d=1}^{D} p(z_{n,d} \,|\, \mathbf{x}_n) \prod_{m=1}^{M} p(y_{n,m,d} \,|\, \mathbf{x}_n, z_{n,d}), \end{aligned} \tag{1}$$

where the term $p(\mathbf{X})$ is dropped as we are more interested in the other two conditional distributions. There are two underlying assumptions in this model. First, each dimension of the ground truth is independent, but is not identically distributed. Second, all observers respond independently.

Note that the first term in (1) indicates the probabilistic dependence between the ground truth and the input instance, whereas the second term characterizes the observers' expertise. Previous work have explored different parametric methods to model these two conditional distributions [10,13,5,12,6]. A distinguishing factor in this paper is that, we employ the Gaussian process as the backbone to construct the model. Specifically, the generative process of $\mathbf{Y}$ can be interpreted as follows

$$z_{n,d} = f_d(\mathbf{x}_n) + \epsilon_n, \tag{2}$$

$$y_{n,m,d} = g_{m,d}(\mathbf{x}_n, z_{n,d}) + \xi_{m,d}, \tag{3}$$

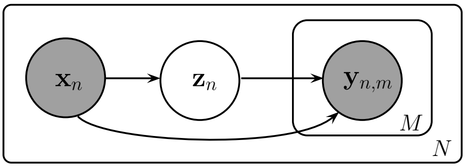


Fig. 1. Graphical model of instances $\mathbf{X}$, unknown ground truth $\mathbf{Z}$ and responses $\mathbf{Y}$ from M different observers. Only the shaded variables are observed.

where ϵ and ξ is independent identically distributed Gaussian noise, respectively. Note that the choice of $\{f_d\}$ and $\{g_{m,d}\}$ characterizes the regression function and the observers, respectively. In particular, an ideal observer would have $g_{m,d}(z_{n,d}) = z_{n,d}$ on every d. Therefore, our goal can be understood as searching $\{f_d\}$ and $\{g_{m,d}\}$ given the training data. Intuitively, if two instances are close to each other in $\mathcal{X}$, then their corresponding ground truth should be close in $\mathcal{Z}$ through the mapping of $\{f_d\}$, which in turn restricts the searching space of $\{g_{m,d}\}$ when $\mathbf{Y}$ is known.

3.2 Regression Model

We first concentrate on Eq. (2) and represent functions $\{f_d\}$ by the Gaussian process with some non-linear kernel. Specificall , the conditional distribution of the ground truth given the training instances is assumed to be

$$p(\mathbf{Z} \,|\, \mathbf{X}) = \prod_{d=1}^{D} \mathcal{N}\left(\mathbf{z}_{:,d} \,|\, \mathbf{0}, \mathbf{K}_d\right), \tag{4}$$

where the d^{th} dimension of the ground truth is denoted as $\mathbf{z}_{:,d}$. We introduce a $N \times N$ kernel matrix $\mathbf{K}_d$ that depends on $\mathbf{X}$, where each element is given by the value of a composite covariance function $k_d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{0+}$, made up of several contributions as follows

$$k_d(\mathbf{x}_i, \mathbf{x}_j) := \kappa_{1,d}^2 \exp\left(-\frac{\kappa_{2,d}^2}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right) + \kappa_{3,d}^2 + \kappa_{4,d}^2 \mathbf{x}_i^\top \mathbf{x}_j + \kappa_{5,d}^2 \delta(\mathbf{x}_i, \mathbf{x}_j). \tag{5}$$

The noise term ϵ in Eq. (2) is folded into the Kronecker delta function $\delta(\mathbf{x}_i, \mathbf{x}_j)$. The covariance function involves an exponential of a quadratic term, with the addition of a constant bias, a linear and a noise terms. For each dimension, the parameters need to be learned from the data are $\kappa_{1,d}, \ldots, \kappa_{5,d}$.

3.3 Linear Observer Model

To model the observer's expertise, we now concentrate on (3) and assume that $\{g_{m,d}\}$ is a linear mapping from $\mathcal{Z}$ to $\mathcal{Y}$, which does not depend on the instance at all.

Denote $\mathbf{y}_{:,m,d}$ the d^{th} dimension response of all training instances provided by the m^{th} observer. The second conditional distribution in (1) is assumed to be

$$p(\mathbf{Y}\,|\,\mathbf{Z},\mathbf{X}) = p(\mathbf{Y}\,|\,\mathbf{Z}) = \prod_{m=1}^{M}\prod_{d=1}^{D}\mathcal{N}\left(\mathbf{y}_{:,m,d}\,\middle|\,w_{m,d}\mathbf{z}_{:,d} + \mu_{m,d}\mathbf{1}, \sigma_{m,d}^{2}\mathbf{I}\right), \tag{6}$$

where $\mathbf{1}$ is an all-ones vector with length N and $\mathbf{I}$ is a $N \times N$ identity matrix. Each observer is characterized by $3 \times D$ parameters, i.e. $w_{m,d}, \mu_{m,d}, \sigma_{m,d} \in \mathbb{R}$.

Parameter Estimation. Now we can combine Eq. (6) with Eq. (4) and estimate the set of all parameters, i.e. $\boldsymbol{\Theta} := \{\{\kappa_{1,d}, \ldots, \kappa_{5,d}\}, \{w_{m,d}\}, \{\mu_{m,d}\}, \{\sigma_{m,d}\}\}$, by maximizing the likelihood function $p(\mathbf{Y}\,|\,\mathbf{X}, \boldsymbol{\Theta})$. In the linear observer model, the latent variable $\mathbf{Z}$ can be marginalized out, which yields

$$p(\mathbf{Y}\,|\,\mathbf{X}, \boldsymbol{\Theta}) = \prod_{m=1}^{M}\prod_{d=1}^{D}\mathcal{N}\left(\mu_{m,d}\mathbf{1}, w_{m,d}^{2}\mathbf{K}_{d} + \sigma_{m,d}^{2}\mathbf{I}\right).$$

The maximum likelihood estimator of $\mu_{m,d}$ is given by $\widetilde{\mu}_{m,d} = \frac{1}{N}\sum_{n=1}^{N} y_{n,m,d}$. We hereinafter use the short-hand $\overline{\mathbf{y}}_{:,m,d} := \mathbf{y}_{:,m,d} - \widetilde{\mu}_{m,d}\mathbf{1}$. As a consequence, the log-likelihood function is given by

$$\begin{aligned} F^{\text{LOB}} &:= \log p(\mathbf{Y}\,|\,\mathbf{X}, \boldsymbol{\Theta}) = \sum_{m=1}^{M}\sum_{d=1}^{D}\log p(\mathbf{y}_{:,m,d}\,|\,\mathbf{X}, \boldsymbol{\Theta}) \\ &= \sum_{m=1}^{M}\sum_{d=1}^{D} -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{C}| - \frac{1}{2}\text{tr}\left(\overline{\mathbf{y}}_{:,m,d}^{\top}\mathbf{C}^{-1}\overline{\mathbf{y}}_{:,m,d}\right), \end{aligned} \tag{7}$$

where $\mathbf{C} := w_{m,d}^{2}\mathbf{K}_{d} + \sigma_{m,d}^{2}\mathbf{I}$. To find the parameters by maximizing Eq. (7), we take the partial derivatives of F^{LOB} with respect to the parameters and obtain

$$\frac{\partial F^{\text{LOB}}}{\partial w_{m,d}} = w_{m,d}\text{tr}\left(\mathbf{B}\mathbf{C}^{-1}\mathbf{K}_{d}\right), \tag{8}$$

$$\frac{\partial F^{\text{LOB}}}{\partial \sigma_{m,d}} = \sigma_{m,d}\text{tr}\left(\mathbf{B}\mathbf{C}^{-1}\right), \tag{9}$$

$$\frac{\partial F^{\text{LOB}}}{\partial \kappa_{i,d}} = \sum_{m=1}^{M}\frac{1}{2}w_{m,d}^{2}\text{tr}\left(\mathbf{B}\mathbf{C}^{-1}\frac{\partial \mathbf{K}_{d}}{\partial \kappa_{i,d}}\right), \tag{10}$$

where $\mathbf{B} := \mathbf{C}^{-1}\overline{\mathbf{y}}_{:,m,d}\overline{\mathbf{y}}_{:,m,d}^{\top} - \mathbf{I}$ and $\frac{\partial \mathbf{K}_{d}}{\partial \kappa_{i,d}}$ is a matrix of element-wise partial derivatives of Eq. (5) with respect to $\kappa_{1,d}, \ldots, \kappa_{5,d}$. As there exists no closed-form solution, we resort to L-BFGS quasi-Newton method to maximize F^{LOB}. Essentially, in each iteration the gradients are computed by Eqs. (8) to (10) and the parameters are updated accordingly.

Estimate of Ground Truth. Note that the ground truth $\mathbf{Z}$ is marginalized out from Eq. (7) and still remains unknown. To estimate the ground truth of all training instances, we need to find the posterior of $\mathbf{Z}$, i.e. $p(\mathbf{Z} \,|\, \mathbf{Y}, \mathbf{X}) = p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}) p(\mathbf{Z} \,|\, \mathbf{X}) / p(\mathbf{Y} \,|\, \mathbf{X})$. By using the property of Gaussian distribution, one can show that the posterior of $\mathbf{z}_{:,d}$ follows $\mathcal{N}(\mathbf{u}, \mathbf{V})$, where

$$\mathbf{u} = \mathbf{V} \left(\sum_{m=1}^{M} \frac{w_{m,d}}{\sigma_{m,d}^2} \overline{\mathbf{y}}_{:,m,d} \right), \quad \mathbf{V} = \left(\sum_{m=1}^{M} \frac{w_{m,d}^2}{\sigma_{m,d}^2} \mathbf{I} + \mathbf{K}_d^{-1} \right)^{-1}. \tag{11}$$

The above computation is repeated D times on every dimension to obtain the estimate of ground truth $\widetilde{\mathbf{Z}}$.

Prediction on New Instance. Given a new instance $\mathbf{x}_*$, we are interested in predicting the ground truth $\mathbf{z}_*$ by using the learned regression function. This can be derived from the joint distribution

$$\begin{bmatrix} \widetilde{\mathbf{z}}_{:,d} \\ z_{*,d} \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_d & \mathbf{k}_*^\top \\ \mathbf{k}_* & k_d(\mathbf{x}_*, \mathbf{x}_*) \end{bmatrix} \right), \tag{12}$$

where $\mathbf{k}_* := [k_d(\mathbf{x}_*, \mathbf{x}_1), \ldots, k_d(\mathbf{x}_*, \mathbf{x}_N)]$. It turns out that $p(z_{*,f} \,|\, \mathbf{X}, \widetilde{\mathbf{z}}_{:,d}, \mathbf{x}_*)$ follows a Gaussian distribution. Hence, the best estimate for the ground truth is

$$\widetilde{z}_{*,d} = \mathbf{k}_* \mathbf{K}_d^{-1} \widetilde{\mathbf{z}}_{:,d}, \tag{13}$$

and the uncertainty is captured in its variance

$$\mathrm{var}(\widetilde{z}_{*,d}) = k_d(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_* \mathbf{K}_d^{-1} \mathbf{k}_*^\top. \tag{14}$$

As a consequence, the response from an observer can be also predicted by

$$\widetilde{y}_{*,m,d} = (1 + \widetilde{w}_{m,d}) \widetilde{z}_{*,d} + \widetilde{\mu}_{m,d}, \tag{15}$$

with variance $\widetilde{\sigma}_{m,d}$.

Priors on Parameters. Note that $w_{m,d}$ is an important indicator of the observer's expertise. On the one hand, a genuine observer would have $w_{m,d}$ close to 1, whereas an adversarial observer gives $w_{m,d}$ close to -1. On the other hand, we encourage $w_{m,d}$ to be a small value unless supported by the data. Without any knowledge on observers, we can only expect that $w_{m,d}$ takes value either around 1 or -1, which inspires the following penalty function

$$\mathrm{penalty}(w_{m,d}) := \begin{cases} \eta(w_{m,d} - 1)^2 & \text{if } w_{m,d} > 1; \\ 0 & \text{if } -1 \leq w_{m,d} \leq 1; \\ \eta(w_{m,d} + 1)^2 & \text{if } w_{m,d} < -1, \end{cases} \tag{16}$$

where η controls the value of penalty as shown in Fig. 2 (see "general"). When $w_{m,d}$ takes value between $[-1, 1]$, there is no penalty and the gradient is given by Eq. (8) directly. When $|w_{m,d}| > 1$ we penalize $w_{m,d}$ and keep it from being too large. This allows our model to search a reasonable solution for $w_{m,d}$ without over-fittin on the training data.

In the case that observers are highly reliable, the learned $w_{m,d}$ should be close to 1 and $\mu_{m,d}, \sigma_{m,d}$ close to 0. One can add a Laplacian prior for observers' parameters, which leads to an L_1 regularization. The penalty term induced by the Laplacian prior for $w_{m,d}$ is $-(\frac{1}{2}\log\lambda + \sqrt{\frac{2}{\lambda}}|w_{m,d} - 1|)$, where a smaller value of λ suggests that the observer is more reliable. The maximization of F^{LOB} can be carried out by computing the sub-gradient of $w_{m,d}$, $\mu_{m,d}$ and $\sigma_{m,d}$, respectively.

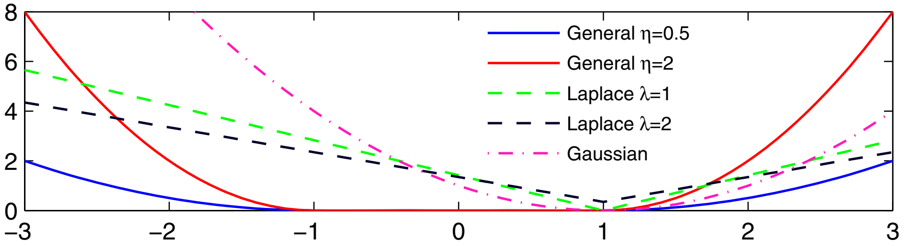


Fig. 2. Penalty functions of $w_{m,d}$ induced by different prior models. The "general" penalty function corresponds to Eq. (16). Similar penalty functions can be added to $\mu_{m,d}$ and $\sigma_{m,d}$ as well.

The relationship between observers can be incorporated into the model as well. For example, the demographic information of users or the geographic location of sensors can be represented as a $M \times M$ proximity matrix $\mathbf{P}$. In particular, we expect two observers have similar parameters if they are highly correlated in $\mathbf{P}$. Assuming $\mathbf{P}$ is a positive definite matrix, we can set the prior distribution of $\mathbf{w}_{:,d}$ set as $\mathcal{N}(\mathbf{w}_{:,d} \mid \mathbf{1}, \mathbf{P})$. As a consequence, we add a penalty term $-\sum_{d=1}^{D} \operatorname{tr}(\mathbf{w}_{:,d}^{\top}\mathbf{P}\mathbf{w}_{:,d})$ to Eq. (6). The gradient of $w_{m,d}$ is computed by Eq. (8) with an additional term $-2\mathbf{P}_{m,:}\mathbf{w}_{:,d}$. Figure 2 illustrates different penalty functions of $w_{m,d}$.

Missing Responses. The model can be extended to handle the training data with missing responses. First of all, we partition the responses $\mathbf{Y} = (\mathbf{Y}^o, \mathbf{Y}^u)$, where $\mathbf{Y}^o$ represents the observed part and $\mathbf{Y}^u$ is the missing part of the responses. Consequently, the latent variables in our model consists of $\mathbf{Z}$ and $\mathbf{Y}^u$. The *expectation maximization* (EM) algorithm can be developed for estimating the model parameters. In the E-step, we fix the model parameter $\boldsymbol{\Theta}$ and compute the sufficient statistics of $\widetilde{\mathbf{Z}}$ by Eq. (11) and then update $\widetilde{\mathbf{Y}}^u$ by its prediction using Eq. (15). In the M-step, we use L-BFGS to maximize $\log p(\widetilde{\mathbf{Y}}, \widetilde{\mathbf{Z}} \mid \mathbf{X}, \boldsymbol{\Theta})$ and update $\boldsymbol{\Theta}$. The two steps are repeated until the likelihood reaches a local maximum.

3.4 Non-linear Observer Model

The assumptions behind the linear observer model may not be appropriate in some scenarios. For instance, if the thermistor is being used to measure the temperature of the environment, due to the self-heating effect the electrical heating may introduce a significant error, which is known as a nonlinear function of the actual environment temperature. Moreover, the observers' responses may depend on the input instance. With these considerations in mind, we propose a more sophisticated model which assumes that $\{g_{m,d}\}$ is a nonlinear mapping from $\mathcal{X} \times \mathcal{Z}$ to $\mathcal{Y}$. By representing $\{g_{m,d}\}$ as the Gaussian process, the second conditional distribution in (1) has the form of

$$p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}) = \prod_{m=1}^{M} \prod_{d=1}^{D} \mathcal{N}\left(\mathbf{y}_{:,m,d} \,|\, \mathbf{0}, \mathbf{S}_{m,d}\right), \tag{17}$$

where $\mathbf{Y}$ is connected with $\mathbf{X}$ and $\mathbf{Z}$ by a $N \times N$ kernel matrix $\mathbf{S}_{m,d}$. The $(i,j)^{\text{th}}$ element in $\mathbf{S}_{m,d}$ is given by

$$\begin{aligned} s_{m,d}\left(\{\mathbf{z}_i, \mathbf{x}_i\}, \{\mathbf{z}_j, \mathbf{x}_j\}\right) := \phi_{m,1,d}^2 \exp\left[-\frac{\phi_{m,2,d}^2}{2}(z_{i,d} - z_{j,d})^2\right] + \phi_{m,3,d}^2 \\ + \phi_{m,4,d}^2 z_{i,d} z_{j,d} + \phi_{m,5,d}^2 \delta(z_{i,d}, z_{j,d}) \\ + \phi_{m,6,d}^2 \exp\left[-\frac{1}{2}\sum_{l=1}^{L} \eta_{m,l,d}^2 (x_{i,l} - x_{j,l})^2\right], \end{aligned} \tag{18}$$

where $x_{i,l}$ is the l^{th} dimension of the instance $\mathbf{x}_i$. This covariance function has a similar form as Eq. (5), but with the addition of an *automatic relevance determination* kernel on $\mathbf{X}$. By incorporating a separate parameter $\eta_{m,l,d}$ for each input dimension l, we can optimize these parameters to infer the relative importance of different dimensions of an instance from the data. One can see that, as $\eta_{m,l,d}$ becomes small, the response $y_{n,m,d}$ becomes relatively insensitive to $x_{n,l}$. This allows us to detect the dimensions of $\mathcal{X}$ that substantially affect the observer's response.

Parameter Estimation. The observer model in Eq. (17) can be combined with Eq. (4) to form our new model,

$$p(\mathbf{Y} \,|\, \mathbf{X}, \boldsymbol{\Theta}) = \int p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}, \boldsymbol{\Theta}) p(\mathbf{Z} \,|\, \mathbf{X}, \boldsymbol{\Theta}) \mathrm{d}\mathbf{Z},$$

where $\boldsymbol{\Theta} := \{\{\kappa_{1,d}, \ldots, \kappa_{5,d}\}, \{\phi_{m,1,d}, \ldots, \phi_{m,6,d}\}, \{\eta_{m,l,d}\}\}$ is the set of model parameters to be inferred from the data. Unfortunately, such marginalization of $\mathbf{Z}$ intractable as the latent variable $\mathbf{z}$ appears nonlinear in the kernel matrix. Instead, we seek a *maximum a posterior* (MAP) solution by maximizing

$$\log p(\mathbf{Z}, \boldsymbol{\Theta} \,|\, \mathbf{Y}, \mathbf{X}) = \log p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}, \boldsymbol{\Theta}) + \log p(\mathbf{Z} \,|\, \mathbf{X}, \boldsymbol{\Theta}) + \text{constant}, \tag{19}$$

with respect to $\mathbf{Z}$ and $\boldsymbol{\Theta}$. Substituting Eq. (17) and Eq. (4) into Eq. (19) gives

$$\begin{aligned} F^{\text{NLOB}} := \log p(\mathbf{Z}, \boldsymbol{\Theta} \,|\, \mathbf{Y}, \mathbf{X}) = -\frac{1}{2}\sum_{d=1}^{D}\sum_{m=1}^{M}\Big(\ln|\mathbf{S}_{m,d}| + \operatorname{tr}(\mathbf{S}_{m,d}^{-1}\mathbf{y}_{:,m,d}\mathbf{y}_{:,m,d}^{\top})\Big) \\ -\frac{1}{2}\sum_{d=1}^{D}\left(\ln|\mathbf{K}_d| + \operatorname{tr}(\mathbf{K}_d^{-1}\mathbf{z}_{:,d}\mathbf{z}_{:,d}^{\top})\right) + \text{constant}. \end{aligned} \tag{20}$$

The partial derivative of F^{NLOB} with respect to the latent variable is given by

$$\frac{\partial F^{\text{NLOB}}}{\partial \mathbf{z}_{:,d}} = \operatorname{tr}\left(\left(\mathbf{S}^{-1}_{m,d}\mathbf{y}^{\top}_{:,m,d}\mathbf{y}_{:,m,d}\mathbf{S}^{-1}_{m,d} - \mathbf{S}^{-1}_{m,d}\right)\frac{\partial \mathbf{S}_{m,d}}{\partial \mathbf{z}_{:,d}}\right) - \mathbf{K}_d^{-1}\mathbf{z}_{:,d}. \qquad (21)$$

The gradients with respect to the parameters of kernel matrix can be likewise derived as in the linear observer model. Finally, these gradients are used in the L-BFGS algorithm for maximizing F^{NLOB}.

When the algorithm converges, the estimate of ground truth is directly given by the stationary point of F^{NLOB}. Predicting the response of a new instance can be carried out in the same way as in Eq. (11). Moreover, the estimation of the m^{th} observer's response is given by

$$\widetilde{y}_{*,m,d} = \mathbf{s}_*\mathbf{S}^{-1}_{m,d}\widetilde{\mathbf{y}}_{:,m,d},$$

where $\mathbf{s}_* := [s_{m,d}(\widetilde{\mathbf{z}}_*, \widetilde{\mathbf{z}}_1, \mathbf{x}_*, \mathbf{x}_1), \ldots, s_{m,d}(\widetilde{\mathbf{z}}_*, \widetilde{\mathbf{z}}_N, \mathbf{x}_*, \mathbf{x}_N)]$.

Initialization. Note that seeking the MAP solution of $\mathbf{Z}$ and $\boldsymbol{\Theta}$ simultaneously may lead to a bad local optimum. Specificall , the model may stuck in a solution where $\{f_d\}$ is too trivial (e.g. close to a constant) and $\{g_{m,d}\}$ is too complicated (e.g. highly non-linear), which contradicts our intuition. To mitigate this problem, we first fit the training data with the linear observer model. The idea is to find an initial approximation of $\{f_d\}$ by restricting $\{g_{m,d}\}$ as linear. Then, we take $\widetilde{\mathbf{Z}}$ estimated by the linear observer model as the initialization of the ground truth, and train the nonlinear observer model to further refine $\{f_d\}$ and $\{g_{m,d}\}$.

4 Experimental Results

To evaluate the performance of our algorithm on predicting the ground truth and the observers' responses, we set up two experiments[1]. First, the effectiveness of our models is demonstrated on the synthetic data. The second experiment is conducted on the real-world data. In both experiments, the ground truth is known and observers' responses are simulated by mapping the ground truth with some random nonlinear functions. As a consequence, the performance can be evaluated straightforwardly. Two metrics are considered here, i.e. the mean absolute normalized error (MANE) and the Pearson correlation coefficient (PCC). In MANE, we first rescale the actual value and its predicted value into $[0, 1]$ respectively, and then measure the mean absolute error. MANE value close to 0 and PCC value close to 1 indicate that the algorithm performs well. In particular, the expected MANE of a random predictor is 0.5.

The proposed linear observer model (`LOB`) and nonlinear observer model (`NLOB`) are compared with several baselines. We first refer `SVR` and `GPR` as the Support Vector Regression and Gaussian Process Regression trained with the ground truth, respectively. Then we combine responses from multiple observers by taking the average and then using it for training, which we denote as `SVR-AVG` and `GPR-AVG`, respectively.

[1] For reproducing the experimental results, our MATLAB implementation is available at `http://home.in.tum.de/~xiaoh`.

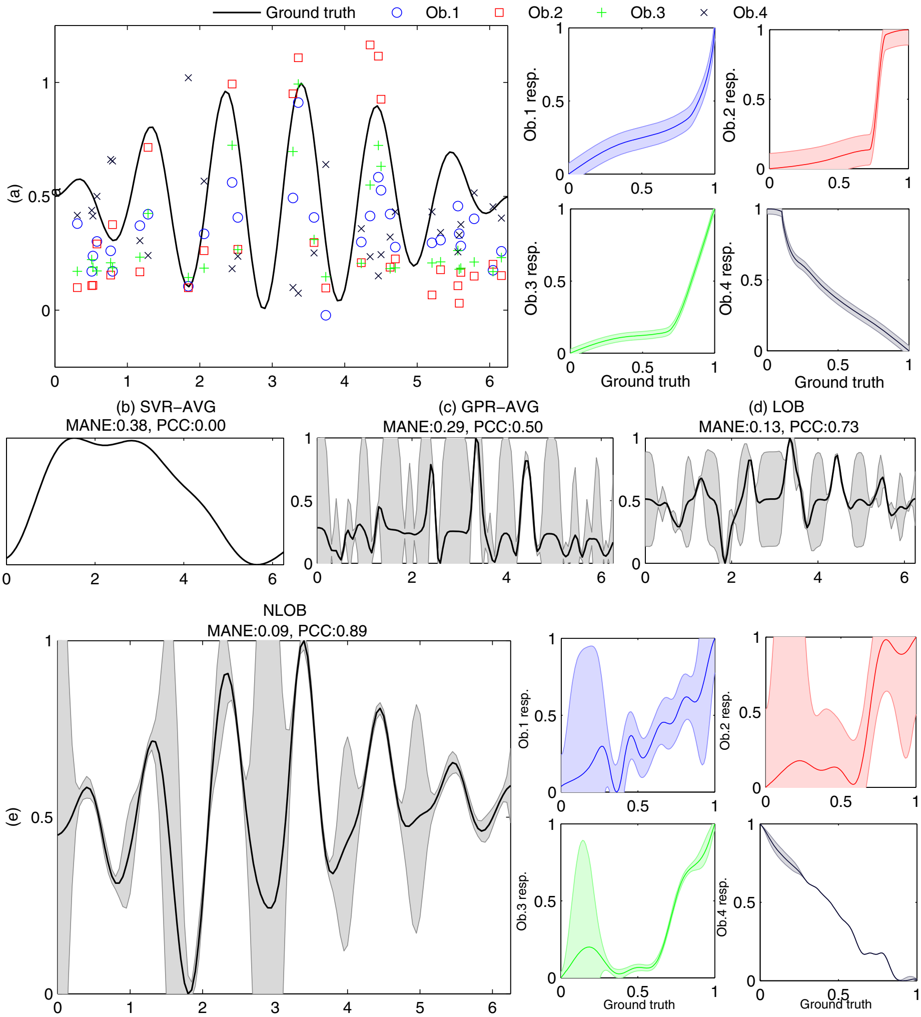


Fig. 3. (a) Synthetic data generated for the experiment. Responses from observers are represented by markers with different colors. The right panel illustrates randomly generated $\{g_m\}$ used for simulating four observers. Shaded area represents the pointwise variance. Note that the 4^{th} observer is *adversarial*, as his response tends to be the *opposite* of the ground truth. **(b, c, d)** Predicted ground truth on the test set by applying SVR−AVG, GPR−AVG and LOB, respectively. **(e)** Predicted ground truth and learned observer functions given by NLOB.

For a fair comparison, the covariance function of $\mathbf{x}$ in GPR and GPR−AVG has the same composite form as in Eq. (5). In addition to these non-parametric methods, Raykar refers to the model in which both $p(\mathbf{Z} \,|\, \mathbf{X})$ and $p(\mathbf{Y} \,|\, \mathbf{Z})$ are Gaussian in the spirit of [6].

4.1 Synthetic Examples

To create one-dimensional synthetic data (i.e. $L := 1$ and $D := 1$), we set $f(x) := \sin(6x)\sin(\frac{x}{2})$. The training instances $\mathbf{X}$ are generated by randomly sampling 30 points in $[0, 2\pi]$ from the uniform distribution. The test instances are obtained using a discretization of $[0, 2\pi]$ with equal space of 0.05, which results in 126 points. Four simulated observers are obtained by setting the corresponding $\{g_m\}$ as a random nonlinear monotonic function. For a training instance x, the m^{th} observer provides its response by $g_m(f(x))$ plus some Gaussian noise. An illustration of our synthetic data is depicted in Fig. 3(a). Figure 3(b, c, d, e) shows the results given by the baselines and our method. Not surprisingly, taking the average of observers' responses is not an effective solution. In contrast, our LOB and NLOB models outperform baseline methods significantly, which yield lower MANE and higher PCC. Moreover, the observers' functions learned by NLOB are very close to those predefine $\{g_m\}$ in Fig. 3(a).

4.2 On Real-World Data

We download four real-world data sets from UCI Machine Learning Repository, namely AUTO, COMMUNITY, CONCRETE and WINE. On each data set, we randomly select 500 instances and generate 20 observers in the same manner as in Section 4.1. The number of adversarial observers is fixed to 6. The experiment is conducted with 10-fold cross-validation. The prediction result of the ground truth and observers' responses is summarized in Table 1. It is notable that the proposed LOB and NLOB significantly outperform SVR/GPR-AVG and Raykar on inferring the ground truth. In general, additional improvements are observed when NLOB is used. Comparing it with the SVR/GPR column, one can see that the regression function learned by NLOB is almost as good as the one trained using the ground truth. We remark that the promising performance of NLOB is achieved by merely learning from a set of observers without any prior knowledge of their expertise and the ground truth. Furthermore, LOB and NLOB also show encouraging performance on predicting responses of observers, which can be proved useful in many applications such as the recommendation system.

Table 1. Prediction of the ground truth and observers' responses. In each cell, the upper value is MANE, while PCC is at the bottom. For the ground truth and the average baselines we only report the best performance, where a superscript $^{\mathrm{S}}$ denotes that the performance is achieved by SVR or SVR-AVG; for GPR and GPR-AVG we use the superscript $^{\mathrm{G}}$. The best model on each data set is highlighted by bold font. Note that only LOB and NLOB can predict observers' responses.

Data set	Ground truth					Observers' responses	
	SVR/GPR	SVR/GPR-AVG	Raykar	LOB	NLOB	LOB	NLOB
AUTO	$0.19 \pm 0.05^{\mathrm{G}}$	$0.21 \pm 0.07^{\mathrm{G}}$	0.25 ± 0.08	0.26 ± 0.05	$\mathbf{0.20 \pm 0.04}$	0.26 ± 0.04	$\mathbf{0.25 \pm 0.09}$
	$0.84 \pm 0.07^{\mathrm{G}}$	$0.63 \pm 0.43^{\mathrm{G}}$	0.50 ± 0.22	$\mathbf{0.84 \pm 0.05}$	0.82 ± 0.08	$\mathbf{0.75 \pm 0.05}$	0.70 ± 0.11
COMMUNITY	$0.15 \pm 0.03^{\mathrm{G}}$	$0.27 \pm 0.08^{\mathrm{S}}$	0.22 ± 0.10	0.17 ± 0.03	$\mathbf{0.16 \pm 0.03}$	0.26 ± 0.04	$\mathbf{0.25 \pm 0.09}$
	$0.80 \pm 0.08^{\mathrm{G}}$	$0.44 \pm 0.38^{\mathrm{S}}$	0.70 ± 0.13	0.76 ± 0.04	$\mathbf{0.77 \pm 0.04}$	$\mathbf{0.62 \pm 0.09}$	0.55 ± 0.15
CONCRETE	$0.15 \pm 0.02^{\mathrm{G}}$	$0.22 \pm 0.08^{\mathrm{G}}$	0.20 ± 0.08	0.18 ± 0.07	$\mathbf{0.17 \pm 0.06}$	0.26 ± 0.04	$\mathbf{0.15 \pm 0.06}$
	$0.76 \pm 0.08^{\mathrm{G}}$	$0.60 \pm 0.46^{\mathrm{G}}$	0.66 ± 0.21	0.78 ± 0.11	$\mathbf{0.79 \pm 0.09}$	0.66 ± 0.18	$\mathbf{0.72 \pm 0.15}$
WINE	$0.20 \pm 0.06^{\mathrm{G}}$	$0.30 \pm 0.05^{\mathrm{S}}$	0.29 ± 0.06	0.27 ± 0.09	$\mathbf{0.25 \pm 0.07}$	0.32 ± 0.07	$\mathbf{0.24 \pm 0.07}$
	$0.67 \pm 0.12^{\mathrm{G}}$	$0.52 \pm 0.30^{\mathrm{G}}$	0.38 ± 0.19	0.58 ± 0.20	$\mathbf{0.61 \pm 0.17}$	0.47 ± 0.18	$\mathbf{0.48 \pm 0.15}$

5 Conclusion

This paper investigates the regression problem under multiple observers providing responses that are not absolutely accurate. The problem involves learning a regression function and observers' expertise from such data without any prior information of the observers. Based on the Gaussian process, we propose a probabilistic framework and develop two models. Our approach provides an estimate of the ground truth and also predicts the responses of each observer given new instances. Experiments show that the proposed method outperforms several baselines and leads to a performance close to the model trained with the ground truth.

There are many opportunities for future research. One possible direction is to extend our model with *multiple kernel learning*. The idea is to let the algorithm pick or composite different covariance functions instead of fixing the combination in advance. As a consequence, the algorithm may learn complex fits for the observers by selecting multiple kernels in a data-dependent way. Moreover, it would be highly beneficial to design *active sampling* methods for selecting which instance and whose response should be learned next.

References

1. Chen, S., Zhang, J., Chen, G., Zhang, C.: What if the irresponsible teachers are dominating? In: Proc. 24th AAAI (2010)
2. Crammer, K., Kearns, M., Wortman, J.: Learning from multiple sources. JMLR 9, 1757–1774 (2008)
3. Dawid, A., Skene, A.: Maximum likelihood estimation of observer error-rates using the em algorithm. Applied Statistics, 20–28 (1979)
4. Hui, S., Walter, S.: Estimating the error rates of diagnostic tests. Biometrics, 167–171 (1980)
5. Raykar, V., Yu, S., Zhao, L., Jerebko, A., Florin, C., Valadez, G., Bogoni, L., Moy, L.: Supervised learning from multiple experts: Whom to trust when everyone lies a bit. In: Proc. 26th ICML, pp. 889–896. ACM (2009)
6. Raykar, V., Yu, S., Zhao, L., Valadez, G., Florin, C., Bogoni, L., Moy, L.: Learning from crowds. JMLR 11, 1297–1322 (2010)
7. Smyth, P., Fayyad, U., Burl, M., Perona, P., Baldi, P.: Inferring ground truth from subjective labelling of venus images. In: Proc. 9th NIPS, pp. 1085–1092 (1995)
8. Spiegelhalter, D., Stovin, P.: An analysis of repeated biopsies following cardiac transplantation. Statistics in Medicine 2(1), 33–40 (1983)
9. Tubaishat, M., Madria, S.: Sensor networks: an overview. IEEE Potentials 22(2), 20–23 (2003)
10. Whitehill, J., Ruvolo, P., Wu, T., Bergsma, J., Movellan, J.: Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In: Proc. 23rd NIPS, vol. 22, pp. 2035–2043 (2009)
11. Wu, O., Hu, W., Gao, J.: Learning to rank under multiple annotators. In: Proc. 22nd IJCAI (2011)
12. Yan, Y., Rosales, R., Fung, G., Dy, J.: Active learning from crowds. In: Proc. 28th ICML (2011)
13. Yan, Y., Rosales, R., Fung, G., Schmidt, M., Hermosillo, G., Bogoni, L., Moy, L., Dy, J., Malvern, P.: Modeling annotator expertise: Learning when everybody knows a bit of something. In: Proc. AISTATS (2010)

Author Index