LNAI 6440

Longbing Cao
Yong Feng
Jiang Zhong (Eds.)

Advanced Data Mining and Applications

6th International Conference, ADMA 2010
Chongqing, China, November 2010
Proceedings, Part I

1 Part I

Springer

Lecture Notes in Artificial Intelligence 6440

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

Subseries of Lecture Notes in Computer Science

Longbing Cao
Yong Feng
Jiang Zhong (Eds.)

Advanced Data Mining and Applications

6th International Conference, ADMA 2010
Chongqing, China, November 19-21, 2010
Proceedings, Part I

Springer

Volume Editors

Longbing Cao
University of Technology Sydney
Faculty of Engineering and Information Technology
Sydney, NSW 2007, Australia
E-mail: longbing.cao-1@uts.edu.au

Yong Feng
Chongqing University
College of Computer Science
Chongqing, 400030, China
E-mail: fengphd@msn.com

Jiang Zhong
Chongqing University
College of Computer Science
Chongqing, 400030, China
E-mail: zhongjiang@cqu.edu.cn

Library of Congress Control Number: 2010939048

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

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN	0302-9743
ISBN-10	3-642-17315-2 Springer Berlin Heidelberg New York
ISBN-13	978-3-642-17315-8 Springer Berlin Heidelberg New York

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

With the ever-growing power of generating, transmitting, and collecting huge amounts of data, information overload is now an imminent problem to mankind. The overwhelming demand for information processing is not just about a better understanding of data, but also a better usage of data in a timely fashion. Data mining, or knowledge discovery from databases, is proposed to gain insight into aspects of data and to help people make informed, sensible, and better decisions. At present, growing attention has been paid to the study, development, and application of data mining. As a result there is an urgent need for sophisticated techniques and tools that can handle new fields of data mining, e.g., spatial data mining, biomedical data mining, and mining on high-speed and time-variant data streams. The knowledge of data mining should also be expanded to new applications.

The 6th International Conference on Advanced Data Mining and Applications (ADMA 2010) aimed to bring together the experts on data mining throughout the world. It provided a leading international forum for the dissemination of original research results in advanced data mining techniques, applications, algorithms, software and systems, and different applied disciplines. The conference attracted 361 online submissions from 34 different countries and areas. All full papers were peer reviewed by at least three members of the Program Committee composed of international experts in data mining fields. A total number of 118 papers were accepted for the conference. Amongst them, 63 papers were selected as regular papers and 55 papers were selected as short papers. The Program Committee worked very hard to select these papers through a rigorous review process and extensive discussion, and finally composed a diverse and exciting program for ADMA 2010. The ADMA 2010 program was highlighted by three keynote speeches from outstanding researchers in advanced data mining and application areas: Kotagiri Ramamohanarao, Chengqi Zhang, and Vladimir Brusic.

September 2010

Longbing Cao
Yong Feng
Jiang Zhong

Organization

ADMA 2010 was organized by Chongqing University, China, and the School of Information Technology and Electrical Engineering, the University of Queensland, Australia, sponsored by the National Natural Science Foundation of China, Chongqing Science and Technology Commission,Chongqing Academy of Science and Technology, and technically co-sponsored by IEEE Queensland Section.

Organizing Committee

Steering Committee Chair

Xue Li	University of Queensland, Australia

Keynote Speakers

Kotagiri Ramamohanarao	University of Melbourne, Australia
Chengqi Zhang	University of Technology Sydney, Australia
Vladimir Brusic	Dana Farber Cancer Institute, USA

General Co-chairs

Charles Ling	The University of Western Ontario, Canada
Shangbo Zhou	Chongqing University, China
Jie Xu	Leeds University, UK

Program Co-chairs

Jinyan Li	Nanyang Technological University, Singapore
Longbing Cao	University of Technology Sydney, Australia
Zhongyang Xiong	Chongqing University, China

Publicity Chair

Xueming Li	Chongqing University, China

Regional Organization Co-chairs

Jiang Zhong	Chongqing University, China
Yong Feng	Chongqing University, China

Finance Chair

Yong Feng	Chongqing University, China

China Registration Chair

Li Wan	Chongqing University, China

China Web Master

Quanji Qiu	Chongqing University, China
Xiaoran Lin	Chongqing University, China

China Secretariat

Min Zheng	Chongqing University, China
Yanyan Zou	Chongqing University, China
Haoyang Ren	Chongqing University, China
Junhui Wang	Chongqing University, China

Program Committee

Hua Li, Canada
Arlindo Oliveira, Portugal
Dragan Gamberger, Croatia
Andre Ponce Leao, Brazil
Andrew Kusiak, America
Wang Shuliang, China
Christophe Giraud-Carrier, USA
Daniel Neagu, UK
Liu Zhen, Japan
Daniel Sanchez, Spain
Dianhui Wang, Australia
Fernando Berzal, Spain
Gang Li, Australia
Jan Rauch, Czech Republic
Jean-Gabriel Ganascia, France
Joseph Roure, UK
Juho Rousu, USA
Junbin Gao, Australia
Paul Vitanyi, USA
Petr Berka, Czech Republic
Rui Camacho, Portugal
Wanquan Liu, Australia
Christophe Rigotti, India
Xiaochun Cheng, UK
Yonghong Peng, UK
Zhaoli Zhu, China
Peng Han, China
Cai Yueping, Japan
Yu Qiao, China
Guang Chen, China
Xinyang Ying, China
Guobin Zhou, China
Yun Li, China
Jun Zhao, China
Hong Tang, China
Hao Wang, China
Hong Yu, China
Li Li, China
Ling Ou, China
Zili Zhang, China
Xingang Zhang, China
Xiaofeng Liao, China
Kaigui Wu, China
Yufang Zhang, China
Hua Li, China
Xiaofan Yang, China
Jiang Zhong, China
Yong Feng, China
Ji Li, China
Li Wan, China
Chengliang Wang, China
Chunxiao Ye, China
Huiping Cao, USA
Yan Qi, USA
Yanchang Zhao, Australia
Sumon Shahriar, Australia

Senthil Kumar, India
Alfredo Cuzzocrea, Italy
Rong Xie, China
Ji Liu, China
Changze Wu, China
Yixiong Chen, China
Qi Xie, China
Pan He, China
BaoHua Qiang, China
Zhixing Li, China
Fenghua Tu, China
Xiaohong Zhang, China
Chuan Li, China

Table of Contents – Part I

I Data Mining Foundations

II Data Mining in Specific Areas

Table of Contents – Part II

III Data Mining Methodologies and Processes

IV Data Mining Applications and Systems

Cost Sensitive Classification in Data Mining

Zhenxing Qin, Chengqi Zhang, Tao Wang, and Shichao Zhang

Faculty of Information Technology, University of Technology, Sydney
PO Box 123, Broadway, Sydney, NSW 2007, Australia
{zqin,chengqi,wangtao,zhangsc}@it.uts.edu.au

Abstract. Cost-sensitive classification is one of mainstream research topics in data mining and machine learning that induces models from data with unbalance class distributions and impacts by quantifying and tackling the unbalance. Rooted in diagnosis data analysis applications, there are great many techniques developed for cost-sensitive learning. They are mainly focused on minimizing the total cost of misclassification costs, test costs, or other types of cost, or a combination among these costs. This paper introduces the up-to-date prevailing cost-sensitive learning methods and presents some research topics by outlining our two new results: lazy-learning and semi-learning strategies for cost-sensitive classifiers.

Keywords: Cost sensitive learning, misclassification cost, test cost.

1 Introduction

Standard classification is a technique of training classifiers from a given dataset and predicting a query with the trained classifiers. The principal objective of standard classification is a high overall classification accuracy, or the lowest overall misclassification rate. This must lead to an inherent bias in favor of the majority classes because the rare (or minority) class has less impact on accuracy. Although standard classification has successfully been used in many real applications, it was found to, however, fail to meet the need of medical diagnosis data analysis. This is because standard classification is based on the assumption that the class distribution is not very skewed and all classification errors involve the same cost. However, in most data mining and machine learning applications, different misclassification errors often involve different costs.

For example, in medical diagnosis domain, there are 1,479,350 US people have been confirmed as cancer patients. Comparing to the US population, 300 millions, it is rare (about 0.48%). If you classify all people as non-cancer, you could get 99.5% accuracy. However, it is obvious not a solution for our application because the class distribution and impacts are hugely different here. On the other word, misclassifying a Cancer patient as normal could lead his/her death while misclassifying a normal patient as with Cancer may only cost more time/money for further examinations.

We can easily find similar situations in terrorist detection, home loan or credit applications where some classes are rare but with much great impact. Traditional data mining methods that aimed at minimizing error rate will perform poorly in these

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 1–11, 2010.

areas, as they assume equal misclassification cost and relatively balanced class distributions. This leads to the development of domain-driven learning techniques, referred to *cost-sensitive learning*, aiming to address classification problems with non-uniform costs.

Cost-sensitive learning is a procedure of inducing models from data with unbalance class distributions and impacts by quantifying and tackling the unbalance. It has attracted extensive attentions and become one of active research topics in data mining and machine learning, since the Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning (ICML-2000). Consequently, diverse learning methods were reported on minimizing the total cost of misclassification costs, test costs, and other types of cost. To make the grasp of these techniques and algorithms easier and smarter, in this paper we survey the up-to-date prevailing cost-sensitive learning methods and classify them into five current research directions. This survey is focused on some basic cost-sensitive learning concepts such as cost function and cost matrix, class probability estimate, misclassification costs and other types of costs involved in the learning process.

2 Settings and Definitions of Cost-Sensitive Learning

2.1 Types of Cost

According to Turney's paper [1], there are nine major types of cost involved in cost-sensitive learning. Two of most studied costs are listed below:

Misclassification Costs: In data mining, different types of misclassification errors usually involve different costs. These costs are the most important costs in cost-sensitive learning. They can either be stationary (represented as a cost matrix) or example dependent.

Test Costs: In some domains, such as medical diagnosis, many tests involve costs. Some tests are more costly than other. If the misclassification costs surpass the test costs greatly, then all tests should be performed. If the test costs are much more than the misclassification costs, then it is rationale not to do any tests.

In addition to the above costs, there are other types of costs such as teacher costs, computation costs, intervention costs, unwanted achievement costs, human-computer interaction costs, costs of cases and costs of instability. In this survey, we concentrate on the cost-sensitive learning methods which minimize the misclassification costs and the test costs.

2.2 Cost Matrix and Cost Function

Most of the cost-sensitive learning methods surveyed in this paper assume that for an M-class problem, an M by M cost matrix C is available at learning time. The value of $C(i, j)$ is the cost involved when a test case is predicted to be class i but actually it belongs to class j.

In reality, *C*(*i, j*) can be example dependent, can be represented by a function, but in this survey, most cost-sensitive learning methods assume that *C*(*i, j*) does not change during the learning or decision making process. So the cost matrix *C* is static.

A static cost matrix always has the following structure when there are only two classes:

Table 1. Two-Class Cost Matrix

	Actual negative	Actual positive
Predict negative	C(0, 0) = C00	C(0, 1) = C01
Predict positive	C(1, 0) = C10	C(1, 1) = C11

As per above cost matrix, the cost of a false positive is C_{10} while the cost of a false negative is C_{01}. Conceptually, the cost of labelling an example incorrectly should always be greater than the cost of labelling it correctly. Mathematically, it should always be the case that $C_{10} > C_{00}$ and $C_{01} > C_{11}$ [3].

As we just mentioned, the cost values of a static cost matrix are always fixed, usually defined by dollar values associated with the correct or incorrect decisions. The learning task is not altered if all the cost values are scaled by a constant factor.

Cost values can represent either costs or benefits, or both, and a careful analysis is needed prior to designing the matrix so that all potential costs that are incurred by a decision are captured. Costs are represented by positive values, whereas benefits are represented by negative values [4].

As per Elkan [3], if a cost matrix *C* is known in advance, let the (*i, j*) entry in *C* be the cost of predicting class *i* when the true class is *j*. If *i* = *j* then the prediction is correct, while if the prediction is incorrect. The optimal prediction for an example *x* is the class *i* that minimizes:

$$L(x,i)=\sum_{j=1}^{n} p(j\,|\,x)C(i,j)\ . \tag{1}$$

In this framework, a test example should always be predicted to have the class that leads to the lowest expected cost, where the expectation is computed using the conditional probability of each class given the example. The role of a learning algorithm is to produce a classifier that for any example can estimate the probability *P(j*|*x)* of each class *j* being the true class of *x*. For an example *x*, making the prediction *i* means acting as if *i* is the true class of *x*. The essence of cost-sensitive decision-making is that it can be optimal to act as if one class is true even when some other class is more probable.

2.3 Traditional Cost-Sensitive Learning Methods

Cost-sensitive learning is an extension of traditional non-cost-sensitive data mining. Cost-sensitive learning methods are also developed based on the existing non-cost-sensitive data mining methods. To make an error-based classifier cost-sensitive, a common method is to introduce biases into an error based classification system in the following different ways [12]:

1) By changing the class distribution of the training data, including:

- Re-sampling
- Instance weighting
- Metacost

2) By modifying the learning algorithms, including:

- Modifying Decision Tree algorithm
- Modifying Naïve Bayes algorithm
- Modifying Neural Network algorithm
- Modifying Support Vector Machine algorithm

3) By taking the boosting approaches, including:

- AdaBoost / AdaCost
- Cost boosting
- Asymmetric boosting

4) Direct cost-sensitive learning which uses the conditional probability estimates provided by error based classifiers to directly compute the optimal class label for each test example using cost function, including:

- Laplace correction
- Smoothing
- Curtailment
- Binning NB
- Platt Calibration
- Isotonic Regression

5) Other cost-sensitive learning methods such as:

- Cost-sensitive specification
- Cost-sensitive CBR system
- Cost-sensitive genetic programming

Among these methods, the “Changing the class distribution of the training data” approach incorporates the misclassification cost into the data pre-processing step. It does this by re-sampling or re-weighting the training data in proportion of their misclassification cost. While the “Direct cost-sensitive learning” approach incorporates the misclassification cost into the data post-processing step. It uses the probability estimation generated by error based classifiers and the cost function to directly compute the optimal class labels for each test example. This approach is easy to implement, but needs good calibration methods to generate accurate probability estimation. The “Modifying the learning algorithms” approach is more straightforward, it modifies the error based classifiers directly to handle misclassification cost, but each classifier needs to be modified separately. The “Boosting” approach is more complicated compared to other approaches. It generates a set of different weak classifiers in sequential trial and then constructs a composite classifier by voting them. The advantage of this approach is that it is applicable to any kind of error based classifiers.

3 Improvement Efforts for Cost-Sensitive Learning

3.1 Cost-Sensitive Learning with Test Costs

The cost-sensitive learning methods mentioned in last section mainly focus on reducing the misclassification cost. Recently, researchers started to consider both test cost and misclassification cost [1, 2, 8-10]. The objective is to minimize the expected total costs of test and misclassification.

Test cost is the cost for obtaining the attribute's value. Current test cost sensitive learning framework combines both of the misclassification cost and test cost together. It aims to minimize the sum of the two kinds of costs. When we combine the test cost and misclassification cost into the classification framework, we need to add extra test cost in the formula defined in Section 2.2, i.e. total cost of all tested attributes for making a decision. Assume there are *m* attributes for test and each attribute *k* is with a test cost t_k, the cost set is noted as T= {t_1, $t_{2, \ldots}$ t_m}. The optimal prediction for an example x in test cost sensitive learning is class *i* that minimizes:

$$L'(x,i) = \sum_{j=1}^{n} p(j \mid y) C(i,j) + \sum_{k=1}^{m} p(k) * t_k \quad . \tag{2}$$

Where L(x, i) is defined as the formula in Section 2.2; *p*(k) is the probability of performing a test for the value attribute k while making the decision.

Test cost-sensitive learning is an extension of classic cost-sensitive learning. When all the test costs are set as zero, the objective of test cost-sensitive learning is the same as that of classic cost-sensitive learning. Classic cost-sensitive learning framework is a special case of test cost-sensitive learning framework.

Turney [2] developed a learning system, called ICET, a cost-sensitive algorithm that employs genetic search to tune parameters used to construct decision trees. Each decision tree is built using Nunez' ICF criterion (described at the end of this section), which selects attributes greedily, based on their information gain and costs. Turney's method adjusts the test costs to change the behavior of Nunez' heuristic so that it builds different trees. These trees are evaluated on an internal holdout data set using the real test costs and misclassification costs. After several trials, the best set of test costs found by the genetic search is used by the Nunez' heuristic to build the final decision tree on the entire training data set. Because Turney simply modifies the attribute selection in C4.5 to add attribute costs when implementing the Nunez' criterion, his algorithm can deal with continuous attributes and with missing attribute values. Turney [2] is also a seminal work laying the foundations of cost-sensitive learning with both attribute costs and misclassification costs. Turney compares his algorithm with C4.5 and with algorithms sensitive only to attribute costs [5-7]. He does not compare ICET with algorithms sensitive to misclassification costs only, because in his experiments he used simple misclassification cost matrices (equal costs on diagonal, equal costs off diagonal) which make algorithms sensitive only to misclassification costs equivalent to minimizing 0/1 loss. ICET outperformed the simpler greedy algorithms on several medical domains from the UCI repository.

In [8], the cost-sensitive learning problem is cast as a Markov Decision Process (MDP), and an optimal solution is given as a search in a state space for optimal policies. For a given new case, depending on the values obtained so far, the optimal policy can suggest a best action to perform in order to both minimize the misclassification and the test costs. While related to other work, their research adopts an optimal search strategy, which may incur very high computational cost to conduct the search.

Similar in the interest in constructing an optimal learner, Greiner et al. [9] studied the theoretical aspects of active learning with test costs using a PAC learning framework. It is a theoretical work on a dynamic programming algorithm (value iteration) searching for best diagnostic policies measuring at most a constant number of attributes. Their theoretical bound is not applicable in practice, because it requires a specified amount of training data in order to obtain close-to-optimal policies.

Ling et al. [10] proposed a new method for building and testing decision trees involving misclassification cost and test cost. The task is to minimize the expected total cost of tests and misclassifications. It assumes a static cost structure where the cost is not a function of time or cases. It also assumes the test cost and the misclassification cost have been defined on the same cost scale, such as the dollar cost incurred in a medical diagnosis. In the later part of this section, we will provide some details of this work because most of our work is based on this work.

Following the work in [10], further research has been done by us and our collaborators. Qin et al. [11] proposed a general framework for involving multiple costs in different cost scales. The task is to minimize one cost scale and control other cost scales in specified budgets. Chai et al. [13] proposed a test cost sensitive Naive Bayes network. Ling and Yang have done much work in test strategies in test cost sensitive learning [15, 16], and they aim to seek the best test attribute set for decision making. Zhang et al. [17] considers the cost sensitive learning in data with missing value and conclude that some data are left as unknown in domain of test cost sensitive learning and could be useful for decision.

Some representative strategies for test-cost-sensitive learning are briefly outlined as follows.

- **EG2**

EG2 [6] is a decision tree induction algorithm that uses the Information Cost Function (ICF) for selection of attributes. It is a modified version of ID3 (Quinlan 1989), the predecessor of a popular decision tree induction algorithm C4.5. ICF selects attributes based on both their information gain and their cost. The ICF for the *i*-th attribute, ICF*i*, is defined as follow:

$$ICFi = \frac{2^{Ii} - 1}{(Ci + 1)^{w}} . \tag{3}$$

Where *Ii* is the information gain associated with the *i*-th attribute at a given stage in the construction of the decision tree and *Ci* is the cost of measuring the *i*-th attribute, and *w* is an adjustable parameter between 0 and 1.

EG2 is able to reduce the overall cost by selecting attributes with less test cost and larger information gain to split.

- **CS-ID3**

CS-ID3 [7] is a decision tree algorithm that selects the split attribute which maximizes the following function:

$$\frac{Ii^2}{Ci} . \tag{4}$$

It is very similar to EG2. Where *Ii* is the information gain associated with the *i*-th attribute at a given stage in the construction of the decision tree and *Ci* is the cost of measuring the *i*-th attribute. However, CS-ID3 does not build a full decision tree then classify examples. Instead, it only constructs a lazy tree (a decision path) for each test example to classify them.

- **IDX**

IDX [5] is also a decision tree algorithm that selects the split attribute that maximizes the following function:

$$\frac{Ii}{Ci} . \tag{5}$$

Same as EG2 and CS-ID3, in the above function, *Ii* is the information gain associated with the *i*-th attribute at a given stage in the construction of the decision tree and *Ci* is the cost of measuring the *i*-th attribute. In C4.5, at each step, a greedy search strategy is used to choose the attribute with the highest information gain ratio. IDX uses a look-ahead strategy that looks n tests ahead, where n is a parameter that may be set by the user.

- **ICET**

ICET is a hybrid of a genetic algorithm and a decision tree induction algorithm. The genetic algorithm evolves a population of biases for the decision tree induction algorithm. The genetic algorithm is GENESIS [2]. The decision tree induction algorithm is EG2. ICET manipulates the bias of EG2 by adjusting the parameters C*i* and *w*. In the original design of EG2, C*i* is the attribute test cost. But in ICET, it is treated as a bias parameter.

In ICET, the genetic algorithm GENESIS begins with a population of 50 individuals. EG2 is run on each one of them to build a corresponding decision tree. The "fitness" of the individual is the total of test and misclassifications costs averaged over the number of cases. In the next generation, the population is replaced with new individuals generated from the previous generation. The fittest individuals in the first generation have the most offspring in the second generation. After a fixed number of generations, ICET stops and outputs the decision tree generated by the fittest individual.

- **MDP**

In Zubek and Dietterich's paper [8], cost-sensitive learning problem is cast as a Markov Decision Process (MDP), and solutions are given as searching in a state space for optimal policies. For a given new case, depending on the values obtained, the resulting policy can suggest an optimal action to perform in order to minimize both the misclassification and the test costs. Their admissible search heuristic is shown to reduce the problem search space remarkably. However, it may take very high computational cost to conduct the search process. In addition, to reduce over-fitting, they have introduced a supplementary pruning heuristic named statistical pruning.

- **Cost-sensitive Naive Bayes**

Chai et al. [13] presented a test Cost-sensitive Naive Bayes (CSNB) classifier which modifies the Naive Bayes classifier by including a test strategy which determines how unknown attributes are selected to perform test on in order to minimize the sum of misclassification cost and test cost. In the framework of CSNB, attributes are intelligently selected for testing to get both sequential test strategies and batch test strategies.

- **Cost-sensitive Decision Tree**

Cost-sensitive Decision Tree (CSDT) is based on C4.5. When building a decision tree, at each step, instead of choosing an attribute that minimizes the entropy (as in C4.5), CSDT chooses an attribute that reduces and minimizes the total of misclassification cost and test cost, for the split. Similar to C4.5, CSDT chooses a locally optimal attribute without backtracking. To make a decision tree cost-sensitive, the decision on which attribute to split on is determined by calculating the misclassification cost for every possible split, and, of course, choosing the lowest. Elkan (2001) points out that this approach may lead to a classification model that minimizes the cost of misclassification of the training set, but does not produce an optimal model when applied to unseen data, mainly because of over-fitting.

- **Multiple Scale Cost-sensitive Decision Tree**

Qin et al. [11] argue that cost sensitive decision tree algorithm must consider the resource budget when building trees and classifying test examples. Based on the decision tree built by Ling et al. [10], they propose a new decision tree algorithm which considers multiple cost scales in the tree building and testing process. In their algorithm, misclassification cost and test cost can be on the same scale if both of them can be converted to dollar values. They can be on different scales if one of them cannot be converted. Other resource costs such as time cost are always on different scales. When building decision trees, instead of using total cost as the split criterion, they use cost gain ratio (cost gain / resource cost) to split. In their paper, a resource budget is set for each resource. During the testing, if an example is run out of resource, it has to stop at the internal node which represents the attribute. Missing values are handled in the same way as that in Ling et al. [10].

The aim of Qin et al.'s decision tree algorithm [10] is to minimize the misclassification cost with limited resource budget. In their framework, misclassification cost does not have a budget. All other resources have limited budget. Multiple resource costs, such as test cost, time cost and computation cost, can be involved in the decision process. The decision tree built by Ling et al. (2004) becomes a special case of this more general cost sensitive decision tree building framework.

3.2 Lazy Cost-Sensitive Learning with Medical History

When dealing with complex and versatile medical data, the different nature of individual attributes of the data may require different measures and usage requirements, i.e. an attribute value can be measured at costs distinguished from other attributes and this value is with its specific usage period. Below Example 1 shows an actual and new setting of cost-sensitive learning.

Example 1: Assume that the cost of testing X needs 1 day, $3000 and 100ml blood; and Y 5 days, $1000 and 50ml blood. The three costs (testing time, testing fee and blood need) are valued in distinct scales. If a patient is too weak to provide more than 60ml blood, then his/her doctor can only choose test Y for him; or if the patient needs an urgent decision within 3 days, then test X should be much more proper for him. On the other hand, if the patient holds a valid test result of X in his/her medical history, the doctor can certainly reset X's test cost to zero before considering further tests.

This setting faced by cost-sensitive learning has two new features: the multiple-scale cost constraint and combination of test data with medical history. Unfortunately, existent cost sensitive learning methods do not handle well the above complicate but real problem because they often simplify the costs in a uniform scale, and in particular, they are not designed for those cases that patients are with certain history record.

Our research attacks this new setting of cost-sensitive learning with a new cost model based on existing cost-sensitive learning framework. This model is introduced with some new concepts, such as target cost and resource cost, and a multiple-scale cost structure which represents the interrelationship between the new cost concepts and the multiple costs involved during a learning process. With the new cost structure, an attribute selection strategy is incorporated to a lazy decision tree induction, so as to minimize the total cost of multiple-scale costs when medical history is dynamically utilized to current test tasks.

3.3 Semi-cost-Sensitive Learning

Another research my research group is to apply cost-sensitive learning techniques to semi-supervised environments.

In many real world applications, labeled examples are often very difficult, time consuming or expensive to obtain, as they require the efforts of human annotators. At the mean time, unlabeled data sometimes is easy to get, but they need to be used carefully. Semi-supervised learning addresses this problem by using large amount of unlabeled data, together with the labeled data, to build classifiers with higher accuracy and less cost [14]. However, semi-supervised classification still needs to tackle he unbalance class distribution problem and it is not well studied. We apply semi-supervised learning techniques to learn cost-sensitive models from datasets with inadequate labeled data.

We propose two classification strategies for learning cost-sensitive classifier from training datasets with both labeled and unlabeled data, based on Expectation Maximization (EM). The first method, Direct-EM uses EM to build a semi-supervised classifier, then directly compute the optimal class label for each test example using the class probability produced by the learning model. The second method, CS-EM modifies EM by incorporating misclassification cost into the probability estimation process.

4 Conclusions

We have introduced the up-to-date prevailing cost-sensitive learning methods. Some open research topics include

- Semi-learning strategies for cost-sensitive classifiers;
- Instance-based (Lazy) learning strategies for cost-sensitive classifiers;

- Semi-lazy learning strategies for cost-sensitive classifiers;
- Cold deck learning strategies for cost-sensitive classifiers;
- Cost-sensitive learning from multiple data sources.

Acknowledgement. This work was supported in part by the Australian Research Council (ARC) under grant DP0985456, the Nature Science Foundation (NSF) of China under grant 90718020, and the Guangxi NSF (Key) grants.

References

1. Turney, P.: Types of cost in inductive concept learning. In: Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning, pp. 15–11. Stanford University, California (2000)
2. Turney, P.D.: Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. Journal of Artificial Intelligence Research 2, 369–409 (1995)
3. Elkan, C.: The foundations of cost-sensitive learning. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, pp. 973–978. Morgan Kaufmann Publishers Inc., Seattle (2001)
4. Margineantu, D.D., Dietterich, T.G.: Improved class probability estimates from decision tree models. Lecture Notes in Statistics, vol. 171, pp. 169–184. Springer, New York (2002)
5. Norton, S.W.: Generating better decision trees. In: Proceedings of the Eleventh International Conference on Artificial Intelligence, pp. 800–805. Morgan Kaufmann Publishers Inc., Detroit (1989)
6. Núñez, M.: The use of background knowledge in decision tree induction. Machine Learning 6(3), 231–250 (1991)
7. Tan, M.: Cost-sensitive learning of classification knowledge and its applications in robotics. Machine Learning 13(1), 7–33 (1993)
8. Zubek, V.B., Dietterich, T.G.: Pruning Improves Heuristic Search for Cost-Sensitive Learning. In: Proceedings of the Nineteenth International Conference on Machine Learning, pp. 27–34. Morgan Kaufmann Publishers Inc., San Francisco (2002)
9. Greiner, R., Grove, A.J., Roth, D.: Learning cost-sensitive active classifiers. Artificial Intelligence 139(2), 137–174 (2002)
10. Ling, C.X., Yang, Q., Wang, J., Zhang, S.: Decision trees with minimal costs. In: ICML 2004, p. 69. ACM, Banff (2004)
11. Qin, Z., Zhang, C., Zhang, S.: Cost-sensitive Decision Trees with Multiple Cost Scales. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS (LNAI), vol. 3339, pp. 380–390. Springer, Heidelberg (2004)
12. Wang, T., Qin, Z., Zhang, S.: Cost-sensitive Learning - A Survey. Accepted by International Journal of Data Warehousing and Mining (2010)
13. Chai, X., Deng, L., Yang, Q., Ling, C.X.: Test-Cost Sensitive Naive Bayes Classification. In: ICDM 2004, pp. 51–58. IEEE Computer Society Press, Brighton (2004)
14. Zhu, X., Wu, X.: Cost-Constrained Data Acquisition for Intelligent Data Preparation. IEEE Transactions on Knowledge and Data Engineering 17(11), 1542–1556 (2005)
15. Sheng, S., Ling, C.X., Yang, Q.: Simple Test Strategies for Cost-Sensitive Decision Trees. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 365–376. Springer, Heidelberg (2005)

16. Sheng, V.S., Ling, C.X., Ni, A., Zhang, S.: Cost-Sensitive Test Strategies. In: Proceedings of 21st National Conference on Artificial Intelligence (AAAI 2006), pp. 482–487. AAAI Press, Boston (2006)
17. Zhang, S., Qin, Z., Ling, C.X., Sheng, S.: Missing Is Useful: Missing Values in Cost-Sensitive Decision Trees. IEEE Transactions on Knowledge and Data Engineering 17(12), 1689–1693 (2005)
18. Qin, Z., Zhang, S., Liu, L., Wang, T.: Cost-sensitive Semi-supervised Classification using CS-EM. In: IEEE 8th International Conference on Computer and Information Technology, pp. 131–136. IEEE Computer Society Press, Sydney (2008)

Web Users Access Paths Clustering Based on Possibilistic and Fuzzy Sets Theory

Hong Yu, Hu Luo, and Shuangshuang Chu

Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, 400065, P.R. China
yuhong@cqupt.edu.cn

Abstract. Web users access paths clustering is important to conduct Web page prediction. In this paper, a novel Web users access paths clustering method is proposed based on possibilistic and fuzzy sets theory. Firstly, a similarity measure method of access paths is proposed based on differences between paths' factors, such as the length of time spent on visiting a page, the frequency of a page accessed and the order of pages accessed. Furthermore, considering that clusters tend to have vague or imprecise boundaries in the path clustering, a novel uncertain clustering method is proposed based on combining advantages of fuzzy clustering and possibility clustering. A λ_cut set is defined here to process the overlapping clusters adaptively. The comparison of experimental results shows that our proposed method is valid and efficient.

Keywords: Web mining, access path clustering, possibilistic theory, fuzzy sets theory, overlapping.

1 Introduction

Over the past decade, the World Wide Web has evolved from being an information source to a global hub for business and socializing. Web pages are personalized based on the factors of an individual, which are the something like interests, social category and context, etc. Personalization implies that the changes are based on implicit data, such as the pages viewed. The sequence of the accessed pages of a user is also called a Web user access path, an access path shorted. Usually, the access path implies the interests of the user. Therefore, clustering the users access paths is useful to predict the next set of pages that will be accessed by users, and is useful to perform successful Web page prefetching.

Obviously, the better similarity measure of Web users access paths can produce the better clustering results. The usual similarity measure are mostly based on the intersection operation of sets such as the Cosine method, or based on sequence alignment methods of non-Euclidean distance such as Jacobian coefficients method. Banerjee and Ghosh [2] defined a similarity measure by a function of the longest common subsequence of the paths, that takes into account both the trajectory taken through a Web site and the time-length spent on each page. Ye etc. [14] proposed a similarity measure that combined Jacobin coefficients

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 12–23, 2010.

and CM method(common road length divide max road length). However, the method does not take into account the time-length spent on the page and the frequency of the page accessed. Guo etc. [6] used the factors of the time-length on the page and the frequency of the page accessed for Web page prediction.

To combat the similarity measure, we will propose the SM-DPF (Similarity Measure based on the Differences of Paths' Factors) method in this paper, which not only think over the differences of the time-length spent on pages and the differences of the frequencies of pages accessed, but also think over the differences of accessed pages' order and the longest common subsequence between the paths. The comparative analysis show that the new results is much more reasonable.

On the other hand, Web data is unstructured, semi-structured or incomplete, and full of noise data, and clustering Web data are not strictly assigned to a certain class in many cases. The clusters tend to have vague or imprecise boundaries. There is a likelihood that an object may be a candidate for more than one cluster. In other word, there are clusters overlapping [9].

Some researchers have focused on solving the uncertain clustering with fuzzy sets theory, possibilistic theory, rough set theory and so on. Asharaf and Narasimha [1] proposed an adaptive rough fuzzy leader algorithm based on the rough sets theory, where the upper and lower approximates are expressed by the threshold values which are decided by the experiential experts. Chen and Miao [4] defined a concept of roughness, the ratio of similarity coefficient, and the overlapping clusters are obtained here, but there are too many manual coefficients used. Wu and Zhou [13] proposed a possibilistic fuzzy algorithm based on c-means clustering. Lingras [9] proposed a k-means cluster algorithm based on the rough sets theory. Chen etc. [5] proposed a possibilistic fuzzy algorithm based on the uncertainty membership. Yu and Luo [15] proposed a possibilistic fuzzy leader algorithm. Actually, there are many work for uncertain clustering, but there are few for the overlapping in Web users access paths clustering. For example, there is only literature [4] is for paths clustering in the above mentioned literatures.

In order to process the vague or imprecise boundaries of clusters, we will focus on the Web users access paths clustering based on possibilistic and fuzzy theory in this paper. Considering that the leader clustering algorithm [10] makes only a single scan through the data set, we choose the framework of leader cluster in our work to improve the time efficiency. A λ_cut set is defined to partition the recorders to more than one clusters adaptively. In short, based on combination of fuzzy clustering and possibilistic clustering, a novel WUAPC-PF (Web Users Access Paths Clustering based on Possibilistic and Fuzzy sets theory) algorithm will be developed in this paper.

2 Similarity Measure Method Based on the Differences of Paths Factors

In this section, we will introduce some basic concepts and the new similarity measure method based on the differences of paths factors.

We discuss the similarity measure method in view of system theory. We take a Web user access path as a system, and the system can be divided into different levels of subsystems. In order to make the similarity measure, we divide a Web user access path system into the page subsystems and the longest common sub-path subsystem. The subsystems also have many of properties and factors. Through calculating the similarity on the same factor between different subsystems and analyzing the quantity of factors, we can get the similarity between systems.

A Web user access path is denoted as $s_i = (p_1, p_2, \cdots, p_m, \cdots, p_n)$, which is a ordered sequence of n Web pages accessed by the user i, and the p_m means the mth page accessed by the user i.

Obviously, the s_i describes the pages accessed by the user i. We define the s_i as a Web user access path system as follows.

Web User Access Path System. A Web user access path system is a Web user access path s_i, which is a ordered sequence of triple tuples,
$s_i =< (p_1, t_1^i, f_1^i), \cdots, (p_m, t_m^i, f_m^i), \cdots, (p_n, t_n^i, f_n^i) >$.
where the length of the s_i is n, and the p_m means the mth page accessed by the user i . The t_m^i means that the user i spend the length of time on visiting the page p_m, and the f_m^i describes the frequency that the page p_m is visited. Sometimes, the path system is called for short.

User Access Page Subsystem. The user access page subsystem, page subsystem for short, is the triple tuple (p_m, t_m^i, f_m^i), where $m < n$.

Sometimes, we also call p_m as a page subsystem, since p_m means the page visited.

Longest Common Sub-path Subsystem. The longest common subsequence between the two path systems s_i and s_j is the longest common sub-path subsystem between them.

An object can be pictured by pairs of attributes and values in data analyzing. If the two objects are similar, that means the values of some attributes are much similar. The access time-length and frequency are the factors of pages, and the order of access pages is the factors of the longest common sub-path subsystem. The differences between time-length of visits, the frequency of visits and the order of the pages show the differences between the users' access behavior and interesting in Web pages. Thus, we study the similarity measure of the access paths from the differences of access paths' factors in this paper.

2.1 Similarity Measure of Page Subsystems

The similarity of page subsystems is mainly decided by the similar factors of pages, which include the time-length and the frequency. That is, if a user stay longer on a page than other pages, we think that the user is more interesting in this page than the other pages. And if a page is accessed with high frequency, we think that this page interests the most of users.

Set s_i and s_j are the user access path systems of user i and j, respectively. (p_m, t_m^i, f_m^i) and (p_m, t_m^j, f_m^j) are page subsystems.

Firstly, we scale the page subsystem similarity based on the differences of page factors. The similarity of time-length factor and the similarity of frequency factor are denoted as r_{ijt} and r_{ijf}, respectively.

We define the similarity of time-length factor on a page p_m as following:

$$r_{ijt} = 1 - \frac{|t_m^i - t_m^j|}{max(t_m^i, t_m^j)}, \tag{1}$$

where $0 \leq r_{ijt} \leq 1$.

We define the similarity of frequency factor on the page p_m as following:

$$r_{ijf} = 1 - \frac{|f_m^i - f_m^j|}{max(f_m^i, f_m^j)}, \tag{2}$$

where $0 \leq r_{ijf} \leq 1$.

Let $q(r_{ij})_m$ means the similarity measure of page subsystem p_m based on the differences of page factors. Obviously, it is decided by the time-length and frequency factors, and calculated by the following equation.

$$q(r_{ij})_m = r_{ijf} * r_{ijt}, \tag{3}$$

where $0 \leq q(r_{ij})_m \leq 1$.

2.2 Similarity Measure of User Access Path Systems

Let there are N common sub-path subsystems between the access path s_i and s_j. The similarity measure of common sub-path subsystems is denoted by $Sim'(s_i, s_j)$. Let the similar weight coefficient between the same page subsystems accessed is λ_m, then we have:

$$Sim'(s_i, s_j) = \frac{N}{|s_i| + |s_j| - N} \sum_{m=1}^{N} \lambda_m q(r_{ij})_m, \tag{4}$$

where $\sum_{m=1}^{N} \lambda_m = 1$, $0 \leq Sim'(s_i, s_j \leq 1)$, and $|.|$ means the cardinality.

On the other hand, the order of the access pages is a important factor to measure the similarity of paths [11,12]. That is, there exists an order relationship between the user access path and the user's interest. In other words, it means that user groups' interest associated with the access sequences, and the web site be accessed firstly has greater interesting for user.

Then we define the similarity measure of the longest common sub-path subsystem between the s_i and the s_j, $Sim''(s_i, s_j)$, which is based on the differences of common sub-path subsystems' factors.

$$Sim''(s_i, s_j) = |comm(s_i, s_j)|/max(|s_i|, |s_j|), \tag{5}$$

where $|comm(s_i, s_j)|$ is the length of the longest common sub-path, $max(|s_i|, |s_j|)$ is the maximum length of access paths.

Therefore, the similarity between access paths system s_i and s_j can be defined as following, where the differences of page factors and common sub-path factors are considered:

$$Sim(s_i, s_j) = \alpha Sim'(s_i, s_j) + \beta Sim''(s_i, s_j), \tag{6}$$

$\alpha + \beta = 1$ is the adjustable coefficient. Adjusting the values of α and β changes the different factors' weight in the similarity. In other words, not only the factors of the page subsystems but also the factors of common sub-path systems have contributions to the similarity measure of access paths.

2.3 Algorithm for Similarity Measure

Here is the description of the similarity measure algorithm based on the differences of paths' factors, which is called SM-DPF algorithm in short.

Algorithm 1. SM-DPF Algorithm

```
Input  : User Access Paths $s_i$ and $s_j$.
Output: the Similarity $Sim(s_i, s_j)$.
begin
    $N = 0$;//the length of the longest common sub-path
    for ($i = 0$, $i \leq |s_i|$, $i++$) do
       for ($j = 0$, $j \leq |s_j|$, $j++$) do
          if exist the same page between the two paths then
             Calculate the number of the same page subsystem;
             Calculate the similarity of time-length factor according to Equa.(1);
             Calculate the similarity of frequency factors according to Equa.(2);
             Calculate the similarity of page subsystem according to Equa.(3);
       if exist the common sub-paths between the two paths then Calculate $N$;
       Calculate $Sim'(s_i, s_j)$,$Sim''(s_i, s_j)$ and $Sim(s_i, s_j)$ according to Equa.(4),
       (5) and (6), respectively.
end
```

2.4 Examples Analysis for Similarity Measure

In the literature [14], the Jacobin coefficients and CM coefficients' weight in the paths similarity measure are decided by users. There the coefficients are set to 1 considering the comparability with our method. The weight coefficients of the SM-DPF algorithm are $\alpha = \beta = 1/2$. If there are m same page subsystems, then assume the similarity weight coefficients of each of them are $\lambda_1 = \lambda_2 = ... = \lambda_m = 1/m$.

Table 1 gives the similarity calculated by the two methods in different cases as following: Case 1-Two users(paths) access the same pages, but the access time-length and frequency are different; Case 2: For two users access paths, the time-length and frequency are the same, but the access order of the pages are different; Case 3: The accessed pages of two paths are not the same at all times, but the time-length and frequency of the same accessed pages are high; Case 4:

Table 1. Comparison on Computing Similarities of Access Paths

Cases	Samples of User Access Path	SM-DPF	Literature [14]
1	$s_1 =< (p_1, 3, 1), (p_2, 4, 2) >$ $s_2 =< (p_1, 4, 2), (p_2, 3, 1) >$	0.69	1
2	$s_1 =< (p_1, 3, 1), (p_2, 4, 2) >$ $s_2 =< (p_2, 4, 2), (p_1, 3, 1) >$	0.5	0
3	$s_1 =< (p_1, 3, 1), (p_2, 4, 2), (p_3, 10, 1) >$ $s_2 =< (p_1, 3, 2), (p_2, 4, 1), (p_4, 2, 1), (p_3, 10, 1) >$	0.5	0.38
4	$s_1 =< (p_1, 3, 1), (p_2, 4, 2), (p_4, 10, 2) >$ $s_2 =< (p_1, 4, 2), (p_2, 3, 1), (p_5, 11, 2)) >$	0.427	0.444
5	$s_1 =< (p_1, 3, 1), (p_3, 15, 2) >$ $s_2 =< (p_2, 4, 2), (p_3, 15, 2) >$	0.17	0

The accessed pages of two users are not the same at all times, but the time-length and frequency of the different accessed pages are high; Case 5: The accessed pages of two users are not the same at all times, and the access order of the pages are different, but the time-length and frequency of the same accessed pages are high. And the second column are some samples under the corresponding cases. The results of the new method SM-DPF and the method in literature [14] are shown in column 3 and 4, respectively.

Observe the samples, we can make a conclusion that the SM-DPF method is much more reasonable in practice. For example, it is obviously that the s_2 is some similar with the s_1 in Case 5, but the similarity is 0 in [14], which is unreasonable. Otherwise, the similarity is 0.17 in the new method, which conform to reality.

3 Clustering Method for Access Paths

As we have discussed, the SM-DPF algorithm is proposed to compute the $Sim(s_i, s_j)$. Let $Sim(s_i, L_k)$ is the similarity of the path s_i to the cluster $\{L_k\}$. Then, it can be calculated by:

$$Sim(s_i, L_k) = \frac{\sum_{j=1}^{|L_k|} Sim(s_i, s_j)}{|L_k|}.$$

Let $DSim(s_i, L_k)$ is the difference degree of the current path s_i to the cluster $\{L_k\}$. Then, $DSim(s_i, L_k) = 1 - Sim(s_i, L_k)$.

On the other hand, considering Web users access paths clustering tends to have vague or imprecise boundaries, that is, there is a likelihood that an object may be a candidate for more than one cluster. We will solve the uncertain clustering based on the possibilistic theory and the fuzzy sets theory in this section.

3.1 Membership Function and Possibility Distribution Function

Clearly, the data set $U = \{s_1, s_2, ..., s_n\}$ composed of a series of user access paths s_i is the universe need to cluster. The literature[1] puts forward the leader clustering based on the Fuzzy sets theory. The membership function can help the objects group to the cluster center, which is a good clustering. Hence we extend the equation used in [3] to define the *membership function* as following.

$$u_{ik} = (\sum_{j=1}^{N_l} (DSim(s_i, L_k)/DSim(s_i, L_j))^{2/(m-1)})^{-1}. \tag{7}$$

The $DSim(s_i, L_k)$ is the difference degree of the current path s_i to the cluster $\{L_k\}$. Let N_l is the number of the current clusters(leaders), and $1 \leq k \leq N_l$. $m \in [1, \infty)$ is a weighting exponent called the fuzzifier.

The clustering algorithm based on the possibilistic can overcome the impact of the noise-data and isolating points [8,13]. To extend the equation used in [8], we have the *possibility distribution function* defined as following.

$$t_{ik} = [1 + (\frac{DSim^2(s_i, L_k)}{\eta_i})^{1/(p-1)}]^{-1}. \tag{8}$$

Here, the value of p determines the fuzziness of the final possibilistic c-partition and the shape of the possibility distribution. A value of 2 for p yields a very simple equation for the membership updates. Fortunately, $p = 2$ seems to give good results in practice.

The value of η_i needs to be chosen depending on the desired bandwidth of the possibility distribution for each cluster. According to the theorem in [8], we define the η_i as followed.

$$\eta_i = K\frac{\sum_{k=1}^{N_l} u_{ik}^m DSim^2(s_i, L_k)}{\sum_{k=1}^{n} u_{ik}^m}. \tag{9}$$

Eq.(9) makes η_i proportional to the average fuzzy intra-cluster distance of cluster $\{L_i\}$. Here $K > 0$, typically K is chosen to be 1.

3.2 λ_cut Set

Let $\mathcal{A}$ be a fuzzy set on universe $\mathbf{X}$, given a number λ in $[0, 1]$, an λ_cut, or λ-level set, of a fuzzy set is defined by[16]: $\mathcal{A}_\lambda = \{x \in \mathbf{X} \mid u_\mathcal{A} \geq \lambda\}$, which is a subset of $\mathbf{X}$.

Let $u_{ic_1} = max\{u_{ik}\}$, $t_{ic_2} = max\{t_{ik}\}$. Obviously, the s_i is most likely belonging to the leader L_{c_1} or L_{c_2}. Hence, we assign the current path s_i to the L_{c_1} or L_{c_2} determinately. Then, we meet the following questions. Does the s_i belong to others cluster? Which clusters are they? According to the quality of the λ_cut set of a fuzzy set, $u_{ik} \geq \lambda$ means that the current object s_i is likelihood belonging to the cluster $\{L_k\}$, so the s_i can be assigned to the cluster $\{L_k\}$. To reduce the number of thresholds, we define the adaptive λ formula as followed.

$$\lambda = (1 - (\frac{DSim(s_i, L_{c^*})}{AU})) \times min(u_{ic_1}, u_{ic_2}) \tag{10}$$

Here, $u_{ic^*} = min(u_{ic_1}, u_{ic_2})$, $AU = \sum_{s=1}^{N_u} DSim(s_i, L_s)$, and L_{c^*} means the cluster which have the smaller u_{ik}. N_u means the number of unaccessed leaders(clusters) after the s_i is assigned to L_{c_1} or L_{c_2} determinately. That is:

$$N_u = \begin{cases} N_l - 1 \; c_1 = c_2, \; s_i \text{ is assigned to } \{L_{c_1}\} \\ N_l - 2 \; c_1 \neq c_2, \; s_i \text{ is assigned to } \{L_{c_1}\} \text{ and } \{L_{c_2}\} \end{cases}$$

3.3 WUAPC-PF Algorithm Description

This subsection presents a Web users access paths clustering algorithm based on possibilistic and fuzzy sets theory, WUAPC-PF algorithm for short.

Algorithm 2. WUAPC-PF Algorithm

```
Input: $\mathbf{U} = (s_1, s_2, \ldots, s_i, \ldots, s_n)$.
Output: $\mathbb{L} = \{L_1, \ldots, L_k, \ldots, L_c\}$.
begin
    Step 1: Produce a random number $r \in [1, n]$, exchange $s_1$ and $s_r$;
    Initially, the leader $\{L_1\} = \{s_1\}$; $N_l = 1$;   //the number of leaders
    Step 2: For each object $s_i$ do:
    Calculate the difference degree $DSim(s_i, L_k)$;
    Calculate the $u_{ik}$, $\eta_i$ and $t_{ik}$ according to Equa.(7),(9) and (8), respectively.
    $u_{ic1} = max(u_{ik}), t_{ic2} = max(t_{ik})$;
    if $DSim(s_i, L_{c1}) < \tau \&\& DSim(s_i, L_{c2}) < \tau$ then
        if $c1 = c2$ then Assign the $s_i$ to $L_{c1}$; $N_u = N_l - 1$;
        else Assign the $s_i$ to $L_{c1}$ and $L_{c2}$; $N_u = N_l - 2$;
        if $N_u > 1$ then
            Calculate $\lambda$ according to Equa. (10);
            for $k = 1$, $k \leq N_u$, $k++$ do
                if $u_{ik} \geq \lambda$ Assign the $s_i$ to the $L_k$;
    else  $N_l = N_l + 1$; Set $s_i$ as a new leader(cluster).
    Step 3: Output all the $\{L_k\}$.
end
```

Firstly, the algorithm calculates the difference degree of the current user path with the existing clusters (leaders) according to $DSim = 1 - Sim(s_i, L_k)$. Only a threshold is used here, which is the global threshold τ. In other words, when $DSim(s_i, L_{c1}) < \tau$ and $DSim(s_i, L_{c2}) < \tau$, the s_i will be assigned to the cluster which indicated by L_{c1} or L_{c2}. Namely, we decide the current path whether belonging to a cluster or two clusters through fuzzy membership functions and probability distribution functions. Through adjusting parameters according to the adaptive λ formula, Equa. (10), then we can decide the current object whether belonging to other clusters. The framework of the algorithm is based on the leader algorithm, whose advantage is scanning the data set only once. Because of the adaptive λ_cut used here, the number of thresholds is reduced to compare with other methods of solving the overlapping clustering. Here is the description of the WUAPC-PF algorithm.

4 Experimental Results

The DePaul University's standard data set [7] is used here. This data is based on a random sample of users visiting this site for a 2 week period during April of 2002, and it has 20950 sessions. Each session begins with a line of the form: SESSION #n (USER_ID = k), where n is the session number, and k is the user id. Each line in a session is a tab delimited sequence of 3 fields: time stamp, pageview accessed, and the referrer. The time stamp represents the number of seconds relative to January 1, 2002.

Here is an example of a session #250:

```
SESSION #250 USER_ID = 109)
10131799 /news/default.asp                -
10131811 /people/                         /news/default.asp
10139445 /people/                         /news/default.asp
10139451 /people/search.asp?sort=ft           /people/
10139463 /people/facultyinfo.asp?id=779       /people/search.asp?sort=ft
10139721 /people/search.asp?sort=ft           /people/facultyinfo.asp?id=779
10139754 /people/search.asp?sort=ft           /people/facultyinfo.asp?id=779
10139770 /people/facultyinfo.asp?id=100       /people/search.asp?sort=ft
```

The ordered sequence of pageviews buildups one access path of a user. For example, the access path of the above session can be described as: (/news/default, 1, 1) → (/people/, 7646, 2) → (/people/search, 297, 3) → (/people/facultyinfo, 28, 2).

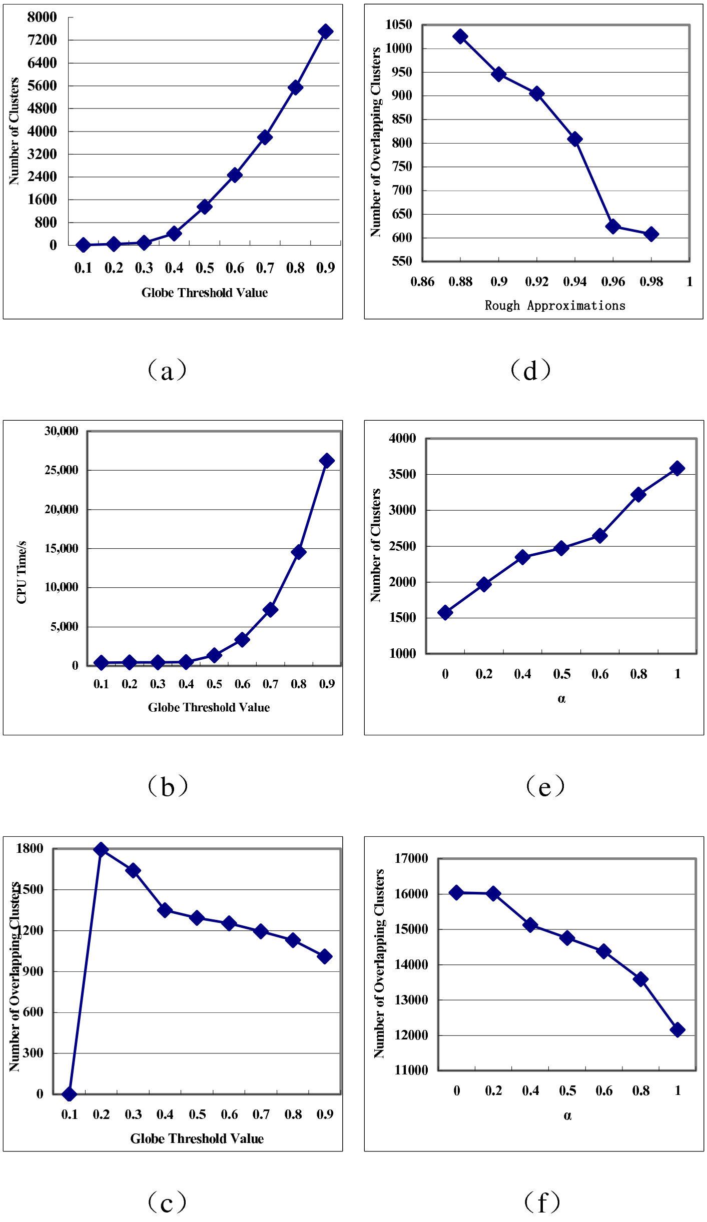
Experiment 1. In order to observe the performance of the new method WUAPC-PF, all of the paths from the above mentioned data set used firstly.

Part (a) of Fig.1 shows the number of clusters curve of the global threshold τ and the number of clusters. Part (b) of Fig.1 shows the time-consuming curve of the CPU time and the global threshold τ. It's obvious that, with the changes of global threshold τ, the different clustering results of Web access paths can be obtained. The browser whose purpose is uncertainty may be assigned to more than one cluster. For a mass of Web access paths, the clustering method is feasible in terms of time performance. From the time-consuming curve and the cluster number curve, we find that when the global threshold τ value is 0.6, the curves change greatly. Then we get a better threshold $\tau = 0.6$ which come from the experiment.

Experiment 2. In order to compare the algorithms, we have done another experiment. A subset including random 2000 paths of the standard data set [7] is used here, and the comparative algorithm is from [4].

Literature [4] introduce a roughness to achieve the overlapping clustering. Part (d) of Fig.1 shows the curve of the number of overlapping clusters with the artificial threshold(Rough Approximations) under a fixed global threshold value τ. The value of τ is 0.3 in the experiment.

The method WUAPC-PF achieves overlapping clustering through the adaptive λ_cut set, and reduces the use of artificial thresholds. Part (c) of Fig.1 shows



(a) (d)

(b) (e)

(c) (f)

Fig. 1. Results of Experiments. **(a)** Globe Threshold Value and the Number of Clusters **(b)** Globe Threshold Value and the CPU Time **(c)** Globe Threshold Value and the Number of Overlapping Clusters **(d)** Rough Approximations and the Number of Overlapping Clusters **(e)** α and the Number of Clusters **(f)** α and the Number of Overlapping Clusters.

the relationship between the global threshold τ and the number of overlapping clusters in WUAPC-PF algorithm.

Comparative analysis between Part (c) and (d) of Fig.1 shows that: [4] use the threshold value of human experience to control the number of overlapping, and the results is obtained only under a fixed threshold, which affects the accuracy of clustering results. Our method reduces the manual intervention and improve the accuracy of the experiment. In addition, the method meets the uncertainty of users access, allows the access path analysis more intuitively. From Part (c), the number of overlapping clusters change in the most stable way, when the global threshold is 0.6. That is, the point $\tau = 0.6$ is the best clustering threshold point. But the method in [4] can not get the relationship between the number of overlapping clusters and the global threshold, which results in reducing the power of recommendation.

Experiment 3. To further validate the correctness of the new method, all of the standard data set [7] are used again.

Part (e) of Fig.1 gives the relation curve between weight coefficients α (or β) in Equa.(6) and the number of clusters, and Fig.6 gives the relation curve between weight coefficients α (or β) in Equa.(6) and the number of overlapping clusters, where the global threshold $\tau = 0.6$.

From Part (e), the experimental results show that the number of clusters increase with the weight coefficient α value increasing, and decrease with the weight coefficient β value increasing. Part (f) shows that the number of overlapping clusters decrease with the weight coefficient α value increasing. We can adjust the weight coefficients α and β according to specific application.

5 Conclusion

It is favorable to discover the user interest models by clustering. Web users access paths clustering plays an important role in the recommendation service of Web intelligence. Generally speaking, users' interests are changeable at many cases, and it is not very reasonable to assign a user only to a cluster, especially for business service. Therefore, in this paper, a novel uncertain clustering method is proposed after taking into account the factors of users' browsing actions. Firstly, a novel similarity measure method between Web users access paths is proposed using the time factors, the frequency factors and the order factors in view of system theory. Furthermore, based on combination of fuzzy clustering and possibility clustering, a possibilistic fuzzy clustering algorithm used for clustering Web users access paths is proposed. A λ_cut set is defined here to process the overlapping clusters adaptively. Considering the advantages of the leader algorithm in time efficiency, the framework of the leader algorithm is used. The comparison of experimental results shows that the new method is valid and efficient. How to realize personalized recommendation using the access paths clustering results would be our further work.

Acknowledgments. This work was supported in part by the Natural Science Foundation of Chongqing of China grant CSTC2009BB2082 and the Science

& Technology Research Program of the Municipal Education Committee of Chongqing of China grant KJ080510.

References

1. Asharaf, S., Narasimha, M.N.: An adaptive rough fuzzy single pass algorithm for clustering large data sets. Pattern Recognition 36, 3015–3018 (2003)
2. Banerjee, A., Ghosh, J.: Click stream clustering using weighted longest common Subsequences. In: Proceedings of the Web Mining Workshop at the 1st SIAM Conference on Data Mining, Chicago, pp. 33–40 (2001)
3. Bezdek, J.C.: Pattern recognition with fuzzy objective function algorithms. Plenum Press, New York (1981)
4. Chen, M., Miao, D.Q., Duan, Q.Q.: Clustering Web users based on users' browsing action. Computer Science 35(3), 186–1879 (2008) (in Chinese)
5. Chen, J., Lu, H., Song, Y., Song, S., Xu, J., Xie, C., Ni, W.: A Possibility Fuzzy Clustering Algorithm Based on the Uncertainty Membership. Journal of Computer Research and Development 45(9), 1486–1492 (2008)
6. Guo, Y.Z., Ramamohanarao, K., Park, L.A.F.: Personalized PageRank for Web Page Prediction Based on Access Time-Length and Frequency. In: IEEE/WIC/ACM International Conference on Web Intelligence (WI 2007), pp. 687–690 (2007)
7. `http://facWeb.cs.depaul.edu/mobasher/classes/ect584/resource.html` (2008)
8. Krishnapuram, R., Keller, J.: A possibilistic approach to clustering. IEEE Trans Fuzzy systerms 1(2), 98–110 (1993)
9. Lingras, P.: Interval set c1ustering of Web users with rough k-means. Journal of Intelligent Information System 23(1), 5–16 (2004)
10. Spath, H.: Cluster analysis algorithm for date reduction and classification of objects. Ellis Horwood Publ., Chichester (1980)
11. Sun, J.G., Liu, J., Zhao, L.Y.: Clustering algorithms research. Journal of Software 19(1), 48–61 (2008) (in Chinese)
12. Wang, S., Gao, W., Li, J.T., Xie, H.: Path clustering: discovering in the Web site. Journal of Computer Research and development 38(4), 482–486 (2001) (in Chinese)
13. Wu, X.H., Zhou, J.J.: A novel possibilistic fuzzy c-means clustering. ACTA Electronica Sinica 36(10), 1996–2000 (2008)
14. Ye, N., Li, W., Liang, Z.P., Dong, Y.S.: Web User Action Clustering Algorithm. MINI-MICRO SYSTEMS 25(7), 1364–1367 (2004) (in Chinese)
15. Yu, H., Luo, H.: A Novel Possibilistic Fuzzy Leader Clustering Algorithm RSFDGrC 2009: Delhi, India. In: Sakai, H., Chakraborty, M.K., Hassanien, A.E., Ślezak, D., Zhu, W. (eds.) RSFDGrC 2009. LNCS, vol. 5908, pp. 423–430. Springer, Heidelberg (2009)
16. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)

Discriminative Markov Logic Network Structure Learning Based on Propositionalization and χ^2-Test

Quang-Thang Dinh, Matthieu Exbrayat, and Christel Vrain

LIFO, Université d'Orléans, Rue Léonard de Vinci,
B.P. 6759, 45067 ORLEANS Cedex 2, France
{thang.dinh,matthieu.exbrayat,christel.vrain}@univ-orleans.fr
http://www.univ-orleans.fr/lifo/

Abstract. In this paper we present a bottom-up discriminative algorithm to learn automatically Markov Logic Network structures. Our approach relies on a new propositionalization method that transforms a learning dataset into an approximative representation in the form of boolean tables, from which to construct a set of candidate clauses according to a χ^2-test. To compute and choose clauses, we successively use two different optimization criteria, namely pseudo-log-likelihood (PLL) and conditional log-likelihood (CLL), in order to combine the efficiency of PLL optimization algorithms together with the accuracy of CLL ones. First experiments show that our approach outperforms the existing discriminative MLN structure learning algorithms.

Keywords: Markov Logic Network, Structure Learning, Relational Learning, Propositionalization, Inductive Logic Programming.

1 Introduction

In Machine Learning, many real-world applications require both probability and first-order-logic in order to deal with uncertainty and complex relational structures. Statistical learning focuses on the former, and relational learning on the latter. Statistical relational learning (SRL) seeks to combine the power of both. In recent years, research in SRL has expanded rapidly. Several SRL approaches have been proposed such as Probabilistic Relational Models [1], Bayesian Logic Programs [2], Relational Dependency Networks [3], Markov Logic Network [4], and others.

Markov Logic Networks (MLNs) [4] are a recently developed SRL model that generalizes both full first-order logic and Markov Networks [5]. A Markov Network (MN) is a graph, where each vertex corresponds to a random variable. Each edge indicates that two variables are conditionally dependent. Each clique of this graph is associated with a weight, which is a real number. A Markov Logic Network consists of a set of pairs (F_i,w_i), where F_i is a formula in First Order Logic, to which a weight w_i is associated. The higher w_i, the more likely a grounding

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 24–35, 2010.

of F_i to be true. Given a MLN and a set of constants $C = \{c_1, c_2, \ldots c_{|C|}\}$, a MN can be generated. The nodes (vertices) of this latter correspond to all ground predicates that can be generated by grounding any formula F_i with constants of C.

Both generative and discriminative learning can be applied to MLNs. Regarding MLN weights learning, generative approaches optimize the log-likelihood or the pseudo-log-likelihood (PLL) [4] using the iterative scaling [6] algorithm or a quasi-Newton optimization method such as *L-BFGS* [7]. The PLL of the possible world x is given by: $\log P_w(X = x) = \sum_{l=1}^{n} \log P_w(X_l = x_l | MB_x(X_l))$, where X_l is a ground atom, x_l is the truth value (0 or 1), $MB_x(X_l)$ is the state of the Markov blanket of X_l in x. Discriminative approaches rely on the optimization of the conditional log-likelihood (CLL) of query given evidence [4]. Let Y be the set of query atoms and X be the set of evidence atoms, the CLL of Y given X is: $\log P(Y = y|X = x) = \sum_{j=1}^{n} \log P(Y_j = y_j|X = x)$. Former methods for MLN discriminative weights learning were based on the voted-perceptron algorithm [8], the scaled conjugate gradient method (PSCG) [9] and the max-margin [10]. All these algorithms only focus on parameter learning, the structure being supposed given by an expert or previously learned. This can lead to suboptimal results when these clauses do not capture the essential dependencies in the domain in order to improve classification accuracy [11]. This hence requires to learn MLN structure directly from data. Further, MLN structure learning is a very important task as it allows to discover new knowledge. However, it is also a challenging one because of its large search space, hence only a few practical approaches have been proposed to date. For generative structure learning, we can cite the top-down approach [12], the bottom-up BUSL algorithm [13], the ILS (Iterated Local Search) algorithm [14], the LHL (Learning via Hyper-graph Lifting) algorithm [15].

These algorithms attempt to learn a set of clauses that models data, and that can be used to predict the truth value of predicates given evidence. However, in many learning problems, there is a specific target predicate that must be inferred given evidence data and discriminative learning is preferred. Concerning discriminative structure learning, to the best of our knowledge, there exists only two systems. The first one uses *ALEPH* system to buld a large set of potential clauses, then learns the weights and prunes useless clauses [16]. The second method, called Iterated Local Search - Discriminative Structure Learning (ILS-DSL), chooses the structure by maximizing the CLL and sets the parameters by the L-BFGS algorithm maximizing the PLL [11].

In this paper, we propose a discriminative approach in order to learn the structure of a MLN. This approach consists of three main steps. First, we apply a technique to transform the dataset into an approximative representation and store it into boolean tables, which contains information related to the learning predicate. Then, starting from these boolean tables, we compose the set of candidate clauses. Finally, candidate clauses are used to learn the final MLN.

In the following, our method is presented in Section 2, whereas Section 3 is devoted to experiments and Section 4 is the conclusion of this paper.

2 Discriminative MLN Structure Learning Algorithm

Let us recall some basic notions of first order logic and make precise the task at hand. We consider a function-free first order language composed of a set $\mathcal{P}$ of predicate symbols, a set C of constants and a set of variables. An *atom* is an expression $p(t_1, \ldots, t_k)$, where p is a predicate and t_i are either variables or constants. A *literal* is either a positive or a negative atom; it is called a *ground literal* when it contains no variable and a *variable literal* when it contains only variables. A *clause* is a disjunction of literals; a *Horn clause* contains at most a positive literal. Two ground atoms are *connected* if they share at least a constant (or argument). A clause (resp. a ground clause) is *connected* when there is an ordering of its literals $L_1 \vee \ldots \vee L_p$, such that for each L_j, $j = 2 \ldots p$, there exists a variable (resp. a constant) occurring both in L_j and L_i, $i < j$. A *variabilization* of a ground clause e, denoted by $var(e)$, is obtained by assigning a new variable to each constant and replacing all its occurrences accordingly.

We have as inputs a query predicate QP and a database, called DB in the following, defining positive/negative examples. A set of clauses defining background knowledge may also be given. We aim at learning a MLN that correctly discriminates between true and false groundings of the query predicate QP.

In a MLN, if two variable literals L_i and L_j occur in a clause, then they are dependent and by construction, L_i is in the Markov blanket of L_j (i.e. L_i is a neighbor of L_j), and vice versa. As a basis of our algorithm, a MLN is built from the training dataset by first forming a set of possible variable literals, and then finding links among them (in the discriminative fashion, it means finding neighbors of each variable literal of the query predicate QP). We define a *template clause* as a disjunction of positive variable literals. They are generated from each query variable literal and its neighbors. Candidate clauses are then extracted from template clauses. It is obvious that the set of possible variable literals should be as small as possible to save time when constructing candidate clauses. However, this set must also be large enough to be able to describe relations of ground atoms in the dataset as well as to compose interesting clauses.

We describe the global structure of our method in Algorithm 1. For each query predicate QP, the algorithm first forms a set SL of possible variable literals. It then builds a boolean table $BT(L_{QP})$ (an approximation of the dataset),

Algorithm 1. DMSP(DB, MLN, QP, minWeight)

```
Initialization of the set of template clauses: $STC = \emptyset$
for each query predicate $QP$ do
    Form a set SL of possible variable literals from $DB, QP$
    for each variable literal $L_{QP} \in SL$ do
        Build a boolean table $BT(L_{QP})$; Create Template Clauses → STC
    end for
end for
Using $STC$ to learn the final $MLN$
$Return(MLN)$
```

corresponding to each variable literal L_{QP} of the query predicate QP, in order to apply a statistical test (i.e. χ^2 test) to check the dependence between the literals (the variable literal L_{QP} and the other ones). The set of variable literals dependent to L_{QP} forms the set of neighbors (the Markov blanket) of L_{QP}, and template clauses are then composed of L_{QP} and subsets of its neighbors. This set of template clauses STC is then used to learn the final MLN. In the following, we present techniques to form the set of variable literals (Subsection 2.1), to build the boolean table and create the set of template clauses (Subsection 2.2) and finally to learn the final MLN (Subsection 2.3).

We must emphasize that the method used to build the boolean tables in our approach can be viewed as a propositionalization approach in Inductive Logic Programming (ILP). As it is shown in [17], propositionalization is a kind of incomplete reduction, hence the quality of boolean tables affects the results of the next steps of our approach. Each boolean table in our method tries to catch as much information related to the target variable literal as possible. We call our method DMSP, which stands for Discriminative MLN Structure learning based on Propositionalization. It must also be noted that the outline of DMSP is, at a first glance, somewhat similar to the principle underlying BUSL [13] generative MLN structure learning algorithm. Although, they are deeply different in the ways propositionalization is performed, the set of candidate clauses is composed and the final MLN is learned from the set of candidate clauses. In Subsection 2.4 we will discuss these differences in more detail.

2.1 Generating a Set of Variable Literals

Let us define the concepts of *g-chain*, *v-chain* and *link*, which will be used in this subsection. Let g and s be two ground literals. A link of g and s, denoted by *link(g,s)*, is an ordered list composed of the name of the predicates of g and s followed by the positions of the shared arguments. If there is no shared argument between g and s then *link(g,s)* is empty. A *g-chain* of ground literals starting from a ground literal g_1 is an ordered list of ground literals $<g_1, ..., g_k, ...>$ such that for $i > 1$, $link(g_{i-1}, g_i)$ is not empty and every shared argument is not shared by $g_{j-1}, g_j, 1 < j < i$. It is denoted by $g\text{-}chain(g_1) =<g_1, ..., g_k, ...>$. The length of a g-chain is the number of ground atoms in it. A link of a g-chain $gc =<g_1, ..., g_k, ...>$ is an ordered list of $link(g_i, g_{i+1}), i \geq 1$, denoted by $g\text{-}link(gc) =<link(g_1, g_2)/.../link(g_i, g_{i+1})/...>$. We can see that if there exists a variabilization of a g-chain $gc =<g_1, ..., g_k>$ such that $var(gc) = vc =<v_1, ...,v_k>$ then *g-link(gc)* is similar to *g-link(vc)*. A g-link $gc =<g_1, ..., g_k >$ is said to be a prefix of a g-link $gs =<s_1, ..., s_n>$, $k \leq n$ if $link(g_i, g_{i+1})$ is similar to $link(s_i, s_{i+1})$, $\forall i, 0 \leq i < k$. Similarly, we define a *v-chain* as a *chain* of variable litterals, and a *v-link* as a *link* of a *v-chain*. We can see that if there exists a variabilization vc of a g-chain $gc =<g_1, ..., g_k>$ such that $var(gc) = vc =<v_1, ..., v_k>$ then *g-link(gc)* is similar to *v-link(vc)*.

The definitions of *g-chain* and *v-chain* ensure that a *g-chain* or a *v-chain* are connected clauses. These notions are related to relational pathfinding [18] and relational cliché [19].

Algorithm 2. Generating literals (*DB, QP*)

```
$maxVar = 0$; $mapVar[c_i] = 0, 1 \leq i \leq mc$, where mc is the number of constants.
for each true ground atom $tga$ of $QP$ do
  Find every g-chain(tga)
  if LinkOf(g-chain(tga), SOGL) then
    $SOGL = SOGL \cup g\text{-}link(g\text{-}chain(tga))$
    $SL = SL \cup Variabilize(g\text{-}chain(tga), maxVar, mapVar)$
  end if
end for
$Return(SL)$
```

In this subsection we present a method to build a set SL of variable literals given the dataset DB and the query predicate QP. This is a difficult task, since the dataset is only a set of ground atoms, with no predefined templates. For each true ground atom e of QP in DB, we build the set of *g-chains* starting from e. Using it, we then build the set SL assuring that for each *g-chain(e)*, there exists a variablilization such that $var(g\text{-}chain(e)) \subseteq SL$. The key insight is based on the observation that relational data usually contains regularities. As a consequence, it is reasonable to expect that many g-chains (starting from several ground atoms) are similar, and could thus be variabilized by a single v-chain vc (i.e. v-link(vc) similar to every g-link of some g-chain). In this case, only a set of variable literals appearing in this vc has to be stored into the set SL. Moreover, if a g-link of a *g-chain(e)* is a prefix of another one already processed, there exists at least a v-chain vc, its variable literals in SL, such that $g\text{-}link(g\text{-}chain(e))$ is a prefix of vc. This *g-chain(e)* is thus no longer to be considered for variabilizing. The task now is to variabilize such sets of similar g-chains to get a set of v-chains from which to achieve a set SL of variable literals.

As we have mentioned above, a g-chain is also a connected clause, hence each g-chain(e) could be variabilized to become a candidate clause. Unfortunately, there are a lot of candidate clauses like that, and learning the final MLN would be very complex. The set of v-chains achieved by variabilizing sets of similar g-chains could also be used as the set of candidate clauses, but it is also very large and a v-chain may not be a good candidate, mainly if its variable literals are not statistically dependent. In order to generate less candidate clauses, we aim at using the variabilization process to create a minimum set SL of variable literals, and then building candidate clauses with dependent variable literals. During the process of variabilization when forming v-chains, we try to reuse variables and variable literals that have been previously introduced in order to reduce the number of variable literals, and thus to reduce the search space for the next steps.

Algorithm 2 sketches our idea to build the set *SL* of variable literals given the learning dataset *DB* and the query predicate *QP*. The algorithm considers each true ground atom *tga* of the query predicate *QP* and builds every *g-chain(tga)*. Function *LinkOf(g-chain(tga))* performs two operations. It first creates *gl* = *g-link(g-chain(tga))* then checks whether *gl* is already in the set *SOGL* containing *g-links* already built (also the set of v-links after variabilizing). Variabilization will occur only if *gl* does not appear in the set *SOGL*. By storing the set *SOGL* of g-links instead of the set of g-chains we can reduce the memory need (a lot of g-chains have a similar g-link). By checking whether *gl* is in *SOGL*, we can remove a lot of g-chains (with g-link was already kept) and thus accelerate the process of finding g-chains. Regarding the variabilization problem, the replacement of constants by variables can be done using various strategies such as *simple variabilization*, *complete variabilization*, etc. [20]. Here, we use the *simple variabilization strategy* to variabilize each *g-chain(tga)* ensuring that different constants in this *g-chain* are replaced by different variables. In more details, to variabilize a *g-chain(tga)*, the algorithm uses the same variable literal for the starting true ground atom *tga*, and for the remaining ones a new variable is only assigned to the constants that have not previously been assigned a variable.

Lemma 1. *The set SL of variable literals created by Algorithm 2 is the minimum set such that for each ground atom* e *of the query predicate* QP*, for each* g-chain(e)*, there always exists at least a variabilization: var*(*g-chain*(*e*)) $\subseteq SL$*.*

Proof. Assume that the set *SL* of variable literals created by Algorithm 2 is not the minimum set. This means that there is a variable literal $vl \in SL$ such that: for each true ground atom *e*, for each *g-chain*$_k$(*e*), there always exists at least a variabilization *var*(*g-chain*(*e*)) $\subseteq SL \setminus vl$. Following the process of variabilization in Algorithm 2, there exists at least some *g-chain(e)* such that it is variabilized and $vl \in$ *var*(*g-chain*(*e*)). The positions of variable literals appearing in *var(g-chain(e))* are fixed. Besides, different variables in *var(g-chain(e))* map to different constants in *g-chain(e)*, therefore *vl* can not be replaced by the other element in the set *SL*, so that we cannot remove the variable literal *vl* from the set *SL*. Hence, the set *SL* is the minimum set.

2.2 Building a Set of Template Clauses

The second step in our method concentrates on finding links amongst variable literals in the set SL. In more detail, we aim at finding a set of neighbors (Markov blanket) of each variable literal L_{QP} of the query predicate QP. In a MLN, a variable literal is a neighbor of the other if they are dependent, thus we have to find a set of variable literals in SL, each of them and L_{QP} are dependent. To determine dependence between two variables X and Y from data we use Pearson's conditional independence chi-square (χ^2) test (see [21] for details of its calculation). The χ^2 test returns a *p-value*, denoted as *p*, which is the probability of the error of assuming that the two variables are dependent when in fact they are not. Independence is decided if and only if *1-p* is greater than a certain confidence threshold α. We use the standard value of $\alpha = 0.95$ in all our experiments. Data for the χ^2 test is often expressed in a contingency table. Therefore, for each variable literal L_{QP}, we need to transform the training dataset into a boolean table, from which to calculate contingency tables. Focusing on discriminative learning, this boolean table has to catch as much information (in DB) related to the variable literal L_{QP} as possible. We build the boolean table, called *Matrix*, organized

Algorithm 3. Build propositional task(DB, SL, L_{QP})

```
$Matrix = \emptyset$; Find the set of v-links $SVL = \{v\text{-}link(v\text{-}chain(L_{QP}))\}$;
for each true/false ground atom $qga$ of $QP$ do
    fillchar($OneRow$, 0); Find the set of g-links $SGL = \{g\text{-}link(g\text{-}chain(qga))\}$;
    for each g-link $gl \in SGL$ do
        if $\exists vl \in SVL$ s.t. $gl$ is similar to $vl$ then
            Fill $OneRow[L] = 1, \forall L$, $L$ is a variable literal appearing in $vl$
    end for
    $Matrix.append(OneRow)$
end for
$Return(Matrix)$
```

as follows: each column corresponds to a variable literal; each row corresponds to a true/false ground atom of the query predicate. Matrix[r][c] true means that there exists at least a v-chain *vc* containing a variable literal at column c, a g-chain *gc* starting from the ground atom at row r, and a variabilization of *gc* such that $var(gc) \subseteq vc$.

Algorithm 3 sketches the steps to fill values of table *Matrix*. For each variable literal L_{QP} of the learning predicate QP, DMSP finds the set SVL of v-links of v-chains starting from L_{QP} as follows. For each true/false ground atom *qga* of QP, it finds every g-link *gl* of g-chains starting from *qga*. If there exists some $vl \in SVL$ such that *vl* and *gl* are similar, then for the row of the *Matrix* corresponding to the ground atom *qga*, value at column L is set to 1 (true) for all variable literals L occurring in *vl*. We also use here the same technique as in the previous subsection by first finding the set SVL (from the set SL) then finding g-chains following each v-chain in SVL to accelerate this process.

The χ^2 test is then applied on *Matrix* to find a set *SDL* of variable literals, each of them being dependent on L_{QP}. A template clause c is then built from L_{QP} and a subset $S \subseteq 2^{SDL}$. If c is also a connected clause, it is added to the set STC of template clauses, which will be used to learn the final MLN.

2.3 Learning the MLN

For each template clause, we flip the sign of its variable literals to get candidate clauses. For each candidate clause a temporary MLN is formed that consists of this clause and the original ones. Weights are then learned by applying the L-BFGS algorithm to this temporary MLN. Because all the candidate clauses generated from a given template clause create a similar clique of the graph, DMSP keeps at most one Horn clause, which is the one with the highest CLL among those having a weight higher than a given *minWeight*. The final candidate clauses are sorted by increasing number of literals. Candidate clauses having the same number of literals are sorted by decreasing CLL. DMSP then considers candidate clauses in turn. For each candidate clause c, it learns the weights for a MLN composed of the initial MLN plus the clauses kept at the previous iterations and c. If the CLL of the MLN improves, c is kept. If c is not accepted and if there exists a clause *pc* in the current structure such that there exists a variable renaming $\theta, pc\theta \subseteq c$. DMSP checks if replacing *pc* by *c* improves the CLL. If it does, *pc* is replaced by *c*. Finally, as adding a clause into a MLN might drop down the weight of clauses added before, once all the clauses have been considered, DMSP tries to prune some clauses of the MLN, as was done in [12].

2.4 Comparing to BUSL

The outline of our method, at a first glance, is similar to the generative MLN structure learning algorithm BUSL [13]. Nevertheless, it differs deeply in all three steps: the way propositionalization is performed, the way to compose the set of candidate clauses and the way to learn the final MLN:

• **Propositionalization:** Approximative boolean tables (resp. constructed by BUSL and DMSP) are different in the meaning of columns, hence in the meaning of values of entries. Each column in the table *MP* (BUSL) is a *TNode* which can be either a single literal or a conjunction of several literals, while each column in the table *Matrix* (DMSP) is a variable literal. For instance, starting from the ground atom *stu(a)*, knowing *advBy(b,a)* and then *pub(t,b)*, BUSL would produce three *TNodes t1* = {*stu(A)*}, *t2* = {*advBy(B,A)*} and *t3* = {*AdvBy(C,A), Pub(D,C)*}, while DMSP would produce three distinct variable literals *l1* = *stu(A)*, *l2* = *AdvBy(B,A)* and *l3* = *Pub(T,B)*. The number of *TNodes* can be very high, depending on the number of atoms allowed per *TNode*, the size of the database and the links existing between ground atoms. On the contrary, DMSP produces just a minimum set of variable literals. For the *r-th* ground atom of learning predicate, *MP[r][t]* = *true* if and only if the conjunction of the set of literals in *t* is true, while *Matrix[r][l]* = *true* if there exists at least a *v-chain* starting from the *r-th* ground atom and containing *l*. These differences influence the performance when applying the χ^2-test and the GSMN.

• **Composing the set of candidate clauses:** BUSL uses GSMN algorithm to determine edges amongst *TNodes* and composes candidate clauses from cliques of *TNodes*. DMSP uses just the χ^2 test in order to get more links amongst variable literals. Moreover, candidate clauses in BUSL must contain all the literals appearing in a *TNode*. This might not be flexible enough as it might occur that a relevant clause contains only one of these two literals.

• **Learning the final MLN:** Since it has been designed for generative learning, BUSL uses a generative criteria for both setting parameters and choosing clauses, this can lead to sub-optimal results when used for prediction tasks. DMSP also sets the parameters by maximizing the PLL but chooses clauses by maximizing the CLL of the query predicates. We can also note a difference in the order clauses are taken into account. BUSL uses a decreasing order for sorting clauses while DMSP uses two criteria: first an increasing number of literals per clause and then a decreasing CLL. The different orders lead to different structures.

3 Experiment

3.1 Datasets

We use three publicly-available datasets[1]: Imdb, Uw-cse and Cora, respectively in order of increasing number of constants and of number of true ground atoms. Imdb describes a movie domain (1540 ground atoms of 10 predicates and 316 constants). We predict the probability of pairs of person occurring in the relation *WorkedUnder*. Uw-cse describes an academic department (2673 ground atoms of 15 predicates and 1323 constants). We have chosen the discriminative task of predicting who is *advisor* of who. Cora dataset is a collection of citations of computer science papers (70367 true/false ground atoms of 10 predicates and 3079 constants). We learn four discriminative MLNs, respectively according to four predicates: *sameBib, sameTitle, sameAuthor, sameVenue.*

3.2 Systems and Methodology

DMSP is implemented over the Alchemy[1] package. We ourself perform experiments to answer the following questions:

1. Does *DMSP* outperform the state-of-the-art discriminative systems?
2. Can we compare *DMSP* to the state-of-the-art generative systems?
3. Which boolean tables respectively created by BUSL and DMSP are better?

To answer question 1, we compare DMSP to the state-of-the-art discriminative system ISL-DSL [11]. To answer question 2, we choose to run the state-of-the-art generative system ILS [14] and also refer to the results of LHL published in [15]. For question 3, we implemented DMSP using the L-BFGS to set weights and the WPLL measure to choose clauses, called DMSP-W. To compare to BUSL, DMSP-W also uses the GSMN algorithm to find neighbors of each query variable literal, creates template clauses from every clique and considers all possible clauses of a template clause. BUSL and DMSP-W do only differ only in the step of building boolean tables, therefore allowing to assess the quality of boolean tables respectively created by them.

For all domains, we performed *5-fold cross-validation.* We measured the CLL and the area under the precision-recall curve *(AUC).* The CLL of a query predicate is the average log-probability over all its groundings given evidence. The precision-recall curve is computed by varying the threshold above which a ground atom is predicted to be true. Parameters for ILS-DSL, ILS and BUSL were set as in [11], [14] and [13]. For all systems we set the maximum number of literals per clause to 5 as in [11]. We used the package provided in [22] to compute the AUC. We ran our tests on a Dual-core AMD 2.4 GHz CPU - 4GB RAM machine.

3.3 Results

We performed inference on the learned *MLN* for each dataset and for each test fold, using *Lazy-MC-SAT* algorithm. It computes the probability of every

[1] http://alchemy.cs.washington.edu

Table 1. CLL, AUC measures

Algorithms →		DMSP		DMSP-W		ISL-DSL		ISL		BUSL	
DB	Pre	CLL	AUC	CLL	AUC	CLL	AUC	CLL	AUC	CLL	AUC
Imdb	WU	-0.022±0.007	0.382	-0.032±0.009	0.315	-0.029±0.007	0.311	-0.036±0.006	0.312	-0.225±0.011	0.129
Uw-cse	AB	-0.016±0.006	0.264	-0.027±0.008	0.216	-0.028±0.006	0.194	-0.031±0.005	0.187	-0.044±0.006	0.204
Cora	SB	-0.136±0.006	0.540	-0.151±0.009	0.400	-0.141±0.009	0.461	-0.173±0.005	0.346	-0.325±0.009	0.229
	ST	-0.085±0.009	0.624	-0.121±0.007	0.421	-0.134±0.010	0.427	-0.144±0.009	0.415	-0.284±0.009	0.418
	SA	-0.132±0.008	0.619	-0.170±0.006	0.410	-0.188±0.008	0.560	-0.234±0.007	0.369	-0.356±0.008	0.347
	SV	-0.109±0.007	0.475	-0.121±0.007	0.328	-0.132±0.009	0.297	-0.145±0.006	0.427	-0.383±0.010	0.276

Pre:Predicates WU:WorkedUnder AB:AdvisedBy SB:SameBib ST:SameTitle SA:SameAuthor SV:SameVenue

grounding of the learning predicate on the test fold, and these probabilities are used to compute the CLL over all the groundings and the relative AUC.

Table 1 presents the average CLL and AUC measures for the learning predicates over the different test folds, obtained for all the considered algorithms on the three datasets. The learning predicates considered are: *WorkedUnder* (Imdb), *AdvisedBy* (Uw-cse) and *SameBib*, *SameTitle*, *SameAuthor*, *SameVenue* (Cora). It must be noted that, although we used the same parameter setting, our results do slightly differ from the ones in [11]. We suppose this comes from the fact that we conducted inference using *Lazy-MC-SAT* instead of *MC-SAT*, and that in the training process ILS-DSL only uses one of the training folds for computing the CLL [11]. Table 2 gives the average runtimes over train folds for the datasets Imdb and Uw-cse, over four learning predicates for the dataset Cora.

First, comparing DMSP and ILS-DSL, we can notice that DMSP performs better both in terms of CLL and AUC for all predicates and for all datasets. Since CLL measures the quality of the probability predictions output by the algorithm, our algorithm outperforms this state-of-the-art discriminative algorithm from the point of view of predicting correctly the query predicates given evidences.

Let us consider now DMSP and ILS. DMSP gets better values in both CLL and AUC for all predicates and all datasets. Referring to results of LHL [15], DMSP gets better CLL values and slightly worse AUC values. Obtaining better results than ILS and the better results for CLL values than LHL tends to show the interest of DMSP compared to the state-of-the-art generative structure learning for MLNs. This is the answer to question 2.

Third, let us compare DMSP-W with BUSL. DMSP-W highly improves CLL values and always gets better AUC values. Because the main difference between DMSP-W and BUSL is the boolean tables, we can answer to question 3 that the boolean tables created by DMSP seems more suitable for the task of discriminative MLN structure learning than the ones created by BUSL.

Let us finally consider the algorithms all together. For all three datasets, DMSP obtains the best CLL and AUC values. It must be noted that this result holds not only on average but also on every test fold of all datasets. As a counterpart, DMSP runs somewhat slower than ILS-DSL and really slower than ILS. The runtime (of each system) includes the time for finding candidate clauses, the time for learning weights for each candidate clause and the time for choosing

Table 2. Runtimes(hour)

Algorithms →	DMSP	DMSP-W	ILS-DSL	ILS	BUSL
IMDB	0.40	0.40	0.49	0.34	0.38
UW-CSE	5.76	6.74	4.30	2.28	8.05
CORA	31.05	33.37	30.41	28.83	34.52

clauses for the final MLN. In practice we notice that the time spent for finding candidate clauses is much less important than the time for learning weights and the time spent by inference for computing the measure (i.e. CLL). To set weights for clauses, all the systems use the L-BFGS algorithm, and thus the runtime depends on the performance of this weights learning algorithm. BUSL and DMSP change completely the MLN at each step, thus learning weights by L-BFGS is very expensive. This problem is not encountered in ILS and ILS-DSL because at each step L-BFGS is initialized with the current weights (and with a weight equal to 0 for a new clause) and it converges within a few iterations [14], [11]. ILS-DSL also uses some heuristics to ignore a candidate clause when it needs too much time for inference [11]. We plan to improve the performance of our systems as it has been done in ILS-DSL, for instance, by finding a more efficient way of filtering candidate clauses.

4 Conclusion and Future Works

In this paper we introduce a new algorithm for discriminative learning the structure of a MLN based on a propositionalization method. This algorithm performs a bottom-up approach, which learns a MLN automatically and directly from a training dataset by first building an approximation table from which candidate clauses are built. Comparative results on some classical databases tends to show that the proposed algorithm performs better than the state-of-the-art *MLN* structure learning algorithms. More experiments should still be performed. We are currenlty working on a faster version of our algorithm, as well as on a discriminative learning variant. Two new structure learning algorithms have bean recently proposed by Kok and Domingos [15,23,24], the code of which has been freshly released. We plan to compare our approach to these two algorithms.

Acknowledgments. We would like to thank Marenglen Biba for his assistance on ILS and ILS-DSL. We also thank the anonymous reviewers for their comments which helped us to improve this paper considerably.

References

1. Friedman, N., Getoor, N., Koller, D., Pfeffer, A.: Learning Probabilistic Relational Models. In: IJCAI 1999, pp. 1300-1309. Springer, Heidelberg (1999)
2. Kersting, K., De Raedt, L.: Adaptive Bayesian Logic Programs. In: Rouveirol, C., Sebag, M. (eds.) ILP 2001. LNCS (LNAI), vol. 2157, pp. 104–117. Springer, Heidelberg (2001)

3. Neville, J., Jensen, D.: Dependency Networks for Relational Data. In: Perner, P. (ed.) ICDM 2004. LNCS (LNAI), vol. 3275, pp. 170–177. Springer, Heidelberg (2004)
4. Richardson, M., Domingos, P.: Markov Logic Networks. Mach. Learn. 62, 107–136 (2006)
5. Bishop, M.C.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2007)
6. Della Pietra, S., Della Pietra, V., Lafferty, J.: Inducing Features of Random Fields. Carnegie Mellon University (1995)
7. Sha, F., Pereira, F.: Shallow Parsing with CRFs. In: HLT-NAACL 2003, pp. 213–220. ACL (2003)
8. Singla, P., Domingos, P.: Discriminative Training of MLNs. In: AAAI 2005, pp. 868–873. MIT Press, Cambridge (2005)
9. Lowd, D., Domingos, P.: Efficient Weight Learning for MLNs. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 200–211. Springer, Heidelberg (2007)
10. Huynh, T.N., Mooney, R.J.: Max-Margin Weight Learning for MLNs. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5781, pp. 564–579. Springer, Heidelberg (2009)
11. Biba, M., Ferilli, S., Esposito, F.: Discriminative Structure Learning of Markov Logic Networks. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 59–76. Springer, Heidelberg (2008)
12. Kok, S., Domingos, P.: Learning the Structure of MLNs. In: ICML 2005, pp. 441–448. ACM, New York (2005)
13. Mihalkova, L., Mooney, R.J.: Bottom-up Learning of MLN Structure. In: ICML 2007, pp. 625–632. ACM, New York (2007)
14. Biba, M., Ferilli, S., Esposito, F.: Structure Learning of Markov Logic Networks through Iterated Local Search. In: ECAI 2008, pp. 361–365. IOS Press, Amsterdam (2008)
15. Kok, S., Domingos, P.: Learning Markov Logic Network Structure via Hypergraph Lifting. In: ICML 2009, pp. 505–512. ACM, New York (2009)
16. Huynh, N.T., Mooney, R.J.: Discriminative Structure and Parameter Learning for Markov Logic Networks. In: ICML 2008, pp. 416–423. ACM, New York (2008)
17. De Raedt, L.: Logical and Relational Learning. Springer, Heidelberg (2008)
18. Richards, B.L., Mooney, R.J.: Learning Relations by Pathfinding. In: AAAI 1992, pp. 50–55 (1992)
19. Silverstein, G., Pazzani, M.: Relational Clichés: Constraining Constructive Induction during Relational Learning. In: ML 1991, pp. 203–207 (1991)
20. Jorge, A.: Iterative Induction of Logic Programs - An Approach to Logic Program Synthesis from Incomplete Specifications. PhD thesis. Univ. of Porto (1998)
21. Agresti, A.: Categorical Data Analysis. John Wiley and Sons, Chichester (2002)
22. Davis, J., Goadrich, M.: The Relationship between Precision-Recall and ROC Curves. In: ICML 2006, pp. 233–240. ACM, New York (2006)
23. Kok, S., Domingos, P.: Learning Markov Logic Networks Using Structural Motifs. In: ICML 2010, pp. 551–558. OmniPress (2010)
24. Davis, J., Domingos, D.: Bottom-Up Learning of Markov Network Structure. In: ICML 2010, pp. 271–278. OmniPress (2010)

EWGen: Automatic Generation of Item Weights for Weighted Association Rule Mining

Russel Pears[1], Yun Sing Koh[2], and Gillian Dobbie[2]

[1] School of Computing and Mathematical Sciences, AUT University, New Zealand
rpears@aut.ac.nz

[2] Department of Computer Science, University of Auckland, New Zealand
{ykoh,gill}@cs.auckland.ac.nz

Abstract. Association Rule Mining is an important data mining technique that has been widely used as an automatic rule generation method. While having outstanding success in many different application domains, it also has the potential to generate a vast number of rules, many of which are of little interest to the user. Weighted Association Rule Mining (WARM) overcomes this problem by assigning weights to items thus enabling interesting rules to be ranked ahead of less interesting ones and making it easier for the user to determine which rules are the most useful. Past research on WARM assumes that users have the necessary knowledge to supply item weights. In this research we relax this assumption by deriving item weights based on interactions between items. Our experimentation shows that the rule bases produced by our scheme produces more compact rule bases with a higher information content than standard rule generation methods.

Keywords: Weighted Association Rule Mining, Gini Index, Eigen Vectors.

1 Introduction

The field of Association Rule Mining has long been dominated by algorithms that generate rules based on the identification of frequent patterns [1]. Items that co-occur together frequently enough signals that there is a potentially strong relationship between them and hence such items are candidates for the generation of rules. Hence commonly co-occurring items such as bread and milk give rise to rules, provided they meet user defined thresholds on minimum support and minimum confidence. While methods based on frequent patterns are useful, the danger is that the rules produced tend to capture what is already known to the user and thus is of little value to the decision maker [2,13].

On the other hand it would be useful to capture patterns between items that occur less frequently on their own but when they do occur they tend to predict the occurrence of other items with a high level of accuracy. For example champagne is an item that happens to be bought relatively infrequently on its own but when it is bought there is high likelihood of caviar also being bought. Patterns

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 36–47, 2010.

such as these are hard to detect with standard association rule mining algorithms as they require that the minimum support threshold be lowered drastically in order for such patterns to be detected. The lowering of support beyond a certain threshold leads to an exponential rise in the number of candidate patterns generated by standard association rule miners. This growth causes two major problems: one is a steep rise in execution time and secondly, a huge increase in the number of rules generated, most of which are of little or no interest to the decision maker. Clearly this is undesirable, and a different approach altogether is needed to solve this problem.

Weighted Association Rule Mining (WARM) was proposed as a solution to this problem. The WARM approach takes as input a set of item weights and identifies patterns that have a weighted support greater than a user defined threshold. Assigning higher weights to rare but high profit items such as champagne and caviar enable them to compete on more equal terms with frequent items such as bread and milk, without the need to artificially reduce the support threshold to enable the rare patterns to be discovered. Furthermore, the weights can be used to effectively rank the rules in order of importance thus reducing the cognitive load on the end user in identifying rules that are of value.

The crucial issue then becomes the weight assignment mechanism. Previous research has relied on end users supplying weights, based on their specialized knowledge of the domain area. For example in a market basket application, item profit is commonly used as a measure of item weight [4]. Likewise, in a web application page dwelling time is used as an indication of page importance and is used to rank or assign weights to the different pages [12]. A number of different algorithms have been proposed in the past which use explicit domain specific information. The difference between the various algorithms is the manner in which individual item weights are combined together into weights for patterns. Regardless of the mechanism used for weight combination the quality of the rules produced depends most crucially on the quality of the weight assignment to individual items.

While many domains exist where explicit domain information can be used to determine weights it is equally true that in many cases information is either not available or sufficient knowledge does not exist to convert raw domain information into useable item weights. For example in the medical application domain it may be possible to rank diseases based on some scale that measures how life threatening that disease may be, but it is much harder to rank symptoms in order of importance. While the propensity of a given symptom to cause a particular disease could be assessed the problem is that a given symptom may be associated with a number of different diseases, thus making it very difficult to assign a single cohesive weight to the symptom. Even in domains where user input on weights are readily available data volatility can give rise to changes in patterns, thus reducing the effectiveness of the weights that the user originally supplied. While it may be possible for the user to continuously monitor the state of the data and provide revisions to the weights such a process is not only tedious but error prone if the number of items involved is high. Finally, even when domain information is readily available in a useable form, there is still the danger that the patterns generated are shaped by the weights supplied and thus the knowledge generated only encapsulates known patterns, thus excluding the discovery of unexpected but nonetheless important rules. Taking into account all of the above issues we believe that an automatic weight generation mechanism based on the properties of the dataset is an important tool in driving the weighted association rule mining process. Our weight inference mechanism is based on Principal Components Analysis (PCA) [7] which is a mathematical technique that has been widely used to find relationships between different items in a given dataset.

The rest of the paper is organized as follows. In the next section we give a formal definition of the Weighted Association Rule Mining problem. In Section 3 we present an overview of past research that has been conducted in the WARM area. In Section 4 we present our methodology for deriving weights using our PCA based method. Our experimentation is presented in Section 5 while we conclude the paper in Section 6 with a summary of research achievements and outline some direction for future work in this area.

2 Weighted Association Rule Mining Fundamentals

Given a set of items, $I = \{i_1, i_2, \ldots, i_m\}$, a transaction may be defined as a subset of I and a dataset as a set D of transactions. A set X of items is called an itemset. The support of X, $\text{sup}(X)$, is the proportion of transactions containing X in the dataset. An *association rule* is an implication of the form $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ has coverage of s in the transaction set D, if $s = \text{sup}(XY)$, where $\text{sup}(XY)$ is the joint support of X and Y. The rule $X \rightarrow Y$ holds in the transaction set D with *confidence* c where $c = \text{conf}(X \rightarrow Y) = \text{sup}(XY)/\text{sup}(X)$. Given a transaction database D, a support threshold minsup and a confidence threshold minconf, the task of association rule mining is to generate all association rules that have support and confidence above the user-specified thresholds. In weighted association rule mining a weight w_i is assigned to each item i, where $0 \leq w_i \leq 1$ reflecting the relative importance of an item over other items that it is associated with. The weighted support of an item i is: $w_i \times \text{sup}(i)$. Similar to traditional association rule mining, a weighted support threshold and a confidence threshold is assigned to measure the strength of the association rules produced. The weight of an itemset X consisting of $|X|$ items is given by:

$$\frac{\sum_{i \in X} w_i}{|X|} \times \text{sup}(X) \ . \tag{1}$$

Here an itemset X is considered to be a frequent itemset if the weighted support of this itemset is greater than a user-defined minimum weighted support (wminsup) threshold.

$$\frac{\sum_{i \in X} w_i \times \text{sup}(X)}{|X|} \geq \text{wminsup} \ . \tag{2}$$

The weighted support of a rule $X \rightarrow Y$ is then the average weights of the itemsets X and Y multiplied by their joint support, $\sup(XY)$:

$$\frac{w_X + w_Y}{2} \times \sup(XY) \ , \tag{3}$$

where w_X , w_Y are the weights of itemsets X and Y respectively.

An association rule $X \rightarrow Y$ is called an interesting rule if its weighted support is greater than *wminsup* and the confidence of the rule is greater than or equal to a minimum confidence threshold, *minconf*.

3 Related Work on Weighted Association Rule Mining

Numerous algorithms have been proposed to implement weighted association rule mining. Sanjay et al. [8] introduced weighted support to association rule mining by assigning weights to both items and transactions. In their approach rules whose weighted support is larger than a given threshold are kept for candidate generation, much like in traditional Apriori [1]. A similar approach was adopted by Cai et al. [5], but they applied weights to items and did not weigh transactions. They also proposed two different ways to calculate the weight of an itemset, either as the sum of all the constituent items' weights or as the average of the weights. However, both of these approaches invalidated the downward closure property [5].

This led Tao et al. [11] to propose a weighted downward closure property. In their approach, two types of weights were assigned, item weight and itemset weight. The goal of using weighted support is to make use of the weight in the mining process and prioritize the selection of targeted itemsets according to their perceived significance in the dataset, rather than by their frequency alone.

Yan and Li [12] working in the domain area of Web mining proposed that weights be assigned on the basis of the time taken by a user to view a web page. Unlike the previous approaches that assumed a fixed weight for each item, Yan and Li allowed their weights to vary according to the dynamics of the system, as pages became more popular (or less popular) the weights would increase (or decrease), as the case may be.

Recently Jian and Ming [6] introduced a system for incorporating weights for mining association rules in communication networks. They made use of a method based on a subjective judgments matrix to set weights for individual items. Inputs to the matrix were supplied by domain specialists in the area of communications networks.

Sun and Bai [9] introduced the concept of w-support which assigns weights to items based on the properties of transactions in a given dataset thus removing the need for domain specific input. The dataset is first converted into a bipartite graph, with one set of nodes representing the items and the other set of nodes representing the transactions. The w-support was then calculated by counting the links between items and the transactions that the items appeared in. The method was shown to work well on sparse datasets but its performance on dense datasets was very similar to that of Apriori as the w-support measure and raw support values converged for dense datasets.

Thus it can be seen that there is a need for a weight assignment method that does not rely on domain specific information but automatically assigns weights and performs well on a wide variety of datasets with varying degrees of data density. As our results in Section 5 shows our proposed scheme performs well on both sparse and dense datasets. Furthermore, our objective is not just to produce interesting rules but rules that also have concise antecedents. Such rules have a greater range of applicability as they require fewer conditions to be satisfied for rule firing. None of the methods for proposed for WARM have been specifically designed to realize this objective.

4 Rule Generation Methodology

Our methodology for rule generation relies on finding a suitable method for assigning weights to items. In the absence of explicit user input on the relative importance of items we rely on the strength of relationships between the items in a given dataset.

4.1 Item Weighting Scheme

Before presenting our basis for deriving item weights we first present a motivating example. Consider the following example on a simplified version of the Zoo dataset taken from the UCI Machine Learning repository [3]. The Zoo dataset contains 101 instances of different animals; each animal is described by 17 different attributes. In our simplified version we have just 6 attributes, as given in Table 1 below.

Table 1. Rule Predictive Accuracy and Coverage of different sets of items

Item Combination	Support
aquatic=0, feathers=0, venomous=0, tail=1, type=1	30
aquatic=0, feathers=0, venomous=0, tail=0, type=1	6
aquatic=1, feathers=0, venomous=0, tail=1, type=1	5
milk=1, type=1	41
type=1	41

Based on the above statistics we can compute the Coverage and Confidence of the two rules (taken over all 101 instances of the Zoo dataset):

R1: aquatic=0 feathers=0 venomous=0 tail=1 → type=1
Coverage (R1) = 30/101 = 0.30, Confidence (R1) = 30/41=0.73

Rule R1 is produced by the standard Apriori algorithm which treats every item with the same weight.

R2: milk=1 → type=1
Coverage (R2) = 41/101=0.41, Confidence (R2) = 41/41=1.0

Rule R2 was produced by our weighted association rule generator with items assigned weights according to the scheme that we describe below. Rule R2 is clearly preferable to R1 as R2 not only has higher coverage but its antecedent is much more concise, containing 3 terms less than for rule R2. Rules containing concise antecedents are highly valued as they require a lesser number of conditions to be true for rule firing. In a medical diagnosis scenario, a rule linking a symptom (or two) with high discriminatory power with respect to a certain disease could be sufficient for a physician to make an initial prognosis which can be later refined into a diagnosis through the use of confirmatory tests. Likewise, a discriminatory rule defined on a web click stream environment involving a short sequence of clicks which are discriminatory with respect to a subsequent destination can be used to recommend that destination to the user, with a high level of confidence that the user will find that destination to be of use.

The key question is: How do we find items such as *milk=1* that on their own have higher predictive power and coverage than a given set of items? More generally, given a rule consequent C (consisting of one or more items), we seek the minimal set of items A in the rule antecedent such that the Rule: $A \rightarrow C$ has the highest possible confidence and coverage.

By referencing the full Zoo dataset, we observe that the item *milk=1* fully captures the variation that occurs in the Zoo dataset with respect to the *type* attribute. When *milk* takes the value 1, *type* takes the value 1; when *milk* takes the value 0, *type* takes values other than 1, in the range [2..7]. Furthermore, the attribute *milk* not just captures the variation on the *type* attribute; it also discriminates on a number of other attributes such as hair, eggs, toothed and so on. This suggests that an item should be ranked based on its ability to capture variation across the other items in the dataset.

Given a dataset D, the covariance of every pair of items can be expressed in terms of its covariance matrix M. The matrix P of all possible Eigen vectors can then be derived from:

$$P^{-1}MP = Q \ , \tag{4}$$

where Q is the diagonal matrix of Eigen values of M. The first Eigen vector $E_1 = (v_1, v_2, \ldots, v_n)$ will then contain a set of numeric values v_i for each item i in the range [0,1]; n is the number of items. For a given item i, the value v_i indicates the extent to which the item i captures the variation in the dataset D with respect to its first principal component. Thus principal components analysis provides us with a rigorous method of capturing the ranking of the items based on the values expressed in the principal components. We decided to rank items based on the first principal component as it captures the greatest amount of variation across a given dataset. Since we are interested in the magnitude of the variation, we compute vector $V = (|v_1|, |v_2|, \ldots, |v_n|)$. We then normalize each element of V to lie in the range [0..1] to yield another vector V_1. Our weight vector W for the set of items is then given by $W = V_1$.

Other weighting schemes that are based on a combination of the first m Eigen vectors are also possible and we will investigate the effectiveness of such schemes in future research. Returning back to our example, the weight vector W produced by taking the first Eigen vector of the dataset is:

W (feathers=0, milk=1, aquatic=0, venomous=0, tail=1, type=1) = (.34, .56, .25, .14, .08, .56)

The dominance of the *milk=1* item is clear amongst the predictors for the *type=1* item. The closest candidate to *milk=1* is *feathers=0* which on its own predicts *type=1* with the same coverage as *milk=1* but with only around half the confidence, i.e. 0.51 as opposed to 1.0.

We also note that the items *milk=1* and *type=1* have perfect correlation with each other and thus produce an identical weight value of 0.56, which in turn means that they act as perfect predictors for each other. However, we stress that pair-wise correlations by themselves are no substitute for the Eigen vector based weighting scheme that ranks the amount of variation captured by an item in relation to all other items across a given dataset.

Henceforth we refer to our rule generator as **E**igen **W**eight Based **Gen**erator or EWGen, for short. EWGen starts off by calculating for each item X the weighted support wsup(X), which is the product of its weight w with its support. All items whose weighted support are below a user defined minimum weighted support threshold (wminsup) are removed and the surviving items are then extended into itemsets in exactly the same manner as Apriori combines frequent items together, except that the criterion for itemset extension is not raw support but weighted support. Thus the main difference between EWGen and classical Apriori is the criterion that is applied for item generation and extension.

4.2 Rule Evaluation

We evaluate the rules produced by our weighted association rule generator using the standard interest measures such as Coverage, Confidence and Lift. In addition to the standard metrics we also decided to use the Gini (G) Index which is an interest measure that has also been widely applied in both classification and pattern mining to measure the amount of information gained by splitting a dataset according to a given criterion [10].

In assessing the effectiveness of our weighting scheme we first run standard Apriori at a given minimum support threshold, *minsup*. Since Apriori effectively assigns a constant weight of 1 for each item it is necessary to take the difference in the weight scales to determine the threshold for EWGen. Given *minsup* for Apriori, we set a threshold t for EWGen which is given by:

$$t = \text{minsup} \times \frac{\text{avg item weights for EWGen}}{\text{avg item weights for Apriori}} \ , \qquad (5)$$

$$t = \frac{\text{minsup} \times \sum_{i \in W} w_i}{n}, \text{since average item weight for Apriori is 1} \ , \qquad (6)$$

where i is a given item, n is the total number of items and W is the weight vector for the dataset D.

5 Experimental Results

In this section we report on the results of running EWGen on six real-world datasets taken from the UCI Machine Learning repository [3]. The six datasets that we selected were Zoo, Soybean, Mushroom, Flare, Hepatitis and Dermatology. These datasets were selected as they have varying levels of density and correlation between items. The Soybean and Dermatology datasets are highly dense, while the Flare dataset is relatively sparse. The Zoo dataset has a high degree of correlation between items, while Dermatology exhibits a low level of correlation between items.

We conducted three different sets of experiments. Our objective in the first set of experiments was to test whether EWGen performed better than Apriori on the key interest measures such as Confidence, Rule Coverage, Gini G measure and Lift. In the second set of experiments we investigated our premise that the introduction of a weighting scheme based on Principal Components analysis produces more concise rules. In our final set of experiments we conducted a Top 10 analysis of the top 10 rules generated by the two rule generators and compared the size of the rule antecedents.

5.1 Performance of EWGen on Selected Interest Measures

In order to keep the Apriori rule base within manageable size we restricted the rules produced to have a minimum confidence of 0.9 and a minimum lift of 1.0. In order for the comparison to be fair we imposed the same restrictions on EWGen. We ran both Apriori and EWGen on the six datasets using a minsup threshold value 0.1 for Apriori and a threshold t given by eq(1) above. Table 2 displays the results obtained. In each column of Table 2 we list a pair of values for each interest measure, with the EWGen value listed first followed by the corresponding value for Apriori. Table 2 shows that EWGen outperformed Apriori on all of the interest measures we tracked, except for rule coverage. We also note that the EWGen rule base was much smaller than that of Apriori for all of the datasets. A large number of rules produced by Apriori were filtered by EWGen as they contained items with low weights and were thus unable to meet the threshold that was set for rule generation.

As expected, EWGen significantly improves the information content of a rule with the G value for Zoo, Soybean and Flare (the three most correlated datasets) showing the greatest differences between the two methods. On the other hand,

Table 2. Performance on Rule Interest Measures for EWGen and Apriori

Dataset	# Rules	Confidence	G Value	Lift	Coverage
Zoo	6132, 644890	0.97, 0.97	0.36, 0.15	2.2, 1.4	0.38, 0.46
Mushroom	1023, 61515	0.95, 0.96	0.10, 0.06	1.6, 1.2	0.29, 0.42
Flare	2350, 200489	0.96, 0.95	0.07, 0.007	2.2, 1.1	0.21, 0.33
Soybean	7858, 179669	0.96, 0.95	0.27, 0.14	1.9, 1.2	0.38, 0.64
Hepatitis	724, 720633	0.98, 0.94	0.07, 0.05	1.5, 1.2	0.26, 0.43
Dermatology	2244, 375115	0.99, 0.99	0.26, 0.24	1.2, 1.2	0.75, 0.79

Dermatology, having the least amount of correlation between items showed the smallest improvement in G value. Along with G value we note that EWGen also generated rules with comparable Confidence to Apriori while improving on Lift. Apriori did generate rules with higher Coverage but on the whole the rules themselves had lower information content and had larger antecedents, as our results in Sections 5.2 and 5.3 will show.

5.2 Rule Antecedent Size

We next turned our attention to the size of the rule antecedents produced by the two rule generators. Table 3 displays the comparative results for EWGen and Apriori. We tracked the distribution of size of the rule antecedent, across the size range from 1 to 4 and beyond. Each cell of the table contains the value for EWGen followed by the corresponding value for Apriori. Table 3 clearly shows that EWGen produced more concise rules than Apriori as the percentage of rules having either 1 or 2 terms in its antecedent was consistently more than that of Apriori. On the other hand, Apriori had a consistently greater percentage of rules having 4 or more terms in its antecedent. We also note that the average size of the antecedent was always smaller for EWGen.

These results confirm the effectiveness of our weighting scheme based on the use of Eigen vectors. This is due to the fact that EWGen ranks rules by rule weight, which is the average weight taken across all items in the rule. Thus for any given threshold t, EWGen prefers rules that have the highest possible weight, which in turn means that it prefers rules with items that capture the greatest possible degree of variation in the data. As stated in Section 4 such items on their own or in combination with one or two other high weight items are able to predict the rule consequent with 90% or more accuracy (as we set our minimum confidence threshold to 90%). Since Apriori gives equal importance to all items, it requires, on the average a greater number of items in the rule antecedent to reach the 90% accuracy threshold that was set for rule generation.

Table 3. Antecedent Rule Size for EWGen and Apriori

Dataset	%=1	%=2	%=3	%$\geq$ 4	Average size
Zoo	8.8, 0.5	28.2, 6.9	33.4, 20.4	30.0, 72.2	2.6, 4.2
Mushroom	3.3, 1.2	16.7, 7.1	36.4, 17.4	43.6, 74.4	2.9, 4.9
Flare	4.8, 1.8	19.9, 10.4	33.5, 24.9	41.9, 63.0	2.7, 3.8
Soybean	2.2, 0.6	14.1, 6.3	29.1, 19.6	54.6. 73.5	1.8, 4.8
Hepatitis	3.2, 0.02	17.0, 0.42	33.8, 2.6	46.0, 97.0	3.0, 5.9
Dermatology	19.7, 0.4	31.5, 4.2	28.1, 17.0	20.8, 70.6	2.0, 4.4

5.3 Top 10 Rule Analysis

In our final set of experiments we investigated how the two methods behaved with respect to the topmost rules produced by the respective rule generators. We ranked the rule bases by G value and then compared the top 10 rules produced by the two methods. Our motivation in this analysis was to track whether the rules having the most information content according to the G index differed substantially from the rest of the rule base. From an end user point of view the rules that are most likely to be of interest to the user are those rules with the highest information content.

In the first part of our analysis we compared the average size of the rule antecedents for EWGen and Apriori. We then went on to investigate the actual rules produced by the two rule generators. Table 4 shows that the top rules produced by EWGen were much more concise than Apriori in their rule antecedents. In contrast to the analysis in Section 5.2 which was conducted across entire rule bases, we see that the difference between the two rule generators is now much greater. As shown by the Reduction Factor column, the ratio between the size of EWGen's rule antecedent to that of Apriori's is much greater, indicating that EWGen is more selective in its use of items with high weight and high degree of variation at the top of the G index value range.

We next investigated the actual rules produced by the two rule generators on the Zoo dataset. We took a sample of 4 rules produced by EWGen and Apriori from their respective Top 10 lists that were ranked by G index value. We see from Table 5 that EWGen's rules express different subclasses of mammals (*Type 1* animals). Animals that produce milk define a sub class of mammals who breathe and do not lay eggs (Rule 3). Alternatively, animals that produce milk correspond to mammals that breathe, have a backbone and do not have feathers (Rule 1). We also have another sub class of mammals that are defined by animals that do not lay eggs; such animals have teeth and produce milk (Rule 4). We also have Rule 2 which is a variation on Rule 1 that moves *type=1* to the antecedent while shifting *milk=1* to the consequent. Apriori's rules are quite different; not only do they have larger antecedents, but the nature of the rules is also different. While its rules express relationships between items that correspond to mammals,

Table 4. Antecedent Rule Size for EWGen and Apriori on the Top 10 rules

Dataset	EWGen Size	Apriori Size	Reduction Factor
Zoo	1	5.5	5.5
Mushroom	2.1	2.6	1.2
Hepatitis	2.1	4.3	2.0
Soybean	1.6	2.5	1.6
Flare	1.3	2.8	2.2
Dermatology	2.0	2.1	1.1

Table 5. Samples of Top 10 Rules for EWGen and Apriori

Sample of EWGen's Top 10 rules
milk = 1 → backbone = 1 breathes = 1 feathers = 0 type = 1
type = 1 → backbone = 1 breathes = 1 feathers = 0 milk = 1
milk = 1 → breathes = 1 eggs = 0 type = 1
eggs = 0 → milk = 1 toothed = 1 type = 1
Sample of Apriori's Top 10 Rules
backbone = 1 eggs = 0 feathers = 0 hair = 1 legs = 4 → airbourne = 0 fins = 0 type = 1 venomous = 0
backbone = 1 eggs = 0 feathers = 0 legs = 4 milk = 1 → airbourne = 0 fins = 0 type = 1 venomous = 0
backbone = 1 eggs = 0 feathers = 0 legs = 4 toothed = 1 → airbourne = 0 fins = 0 type = 1 venomous = 0
backbone = 1 eggs = 0 hair = 1 legs = 4 toothed = 1 → airbourne = 0 fins = 0 type = 1 venomous = 0

just like with EWGen, it differs in the degree of diversity with respect to its rule consequents. The EWGen rule generator was able to capture the conditions that define 3 different sub classes of mammals in its rule consequents, however Apriori was only able to capture a single sub class of mammal that does not fly, has no fins and is non venomous. We also note that Apriori had not used the item *milk =1* in any of its Top 10 rules, this is due to its emphasis on high support items at the expense of items having a high degree of variation.

6 Conclusion

This research has revealed that it is possible to produce concise rules with high information content on the basis of weights assigned to items from the Eigen vectors derived from a given dataset. The effect of our item weighting scheme was to assign higher priority to items that are able to capture a higher proportion of variance, which resulted in compact rule antecedents. As we noted early in the paper there is a real need for such automated schemes as users may be unable to supply item weights due to a variety of different reasons. We thus believe that this research represents an important step in the direction towards the automation of item weighting for association rule mining.

As future work we plan to investigate the effect of using a combination of the first m Eigen vectors in determining the weight of a given item. This will require a method of aggregating the variation captured by a given item across the set of m Eigen vectors. We also plan to run our weighting scheme across synthetic data that will allow us to control key data parameters such as data density and the degree of correlation between items.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pp. 207–216 (1993)
2. Ashrafi, M.Z., Taniar, D., Smith, K.: Redundant association rules reduction techniques. Int. J. Bus. Intell. Data Min. 2(1), 29–63 (2007)
3. Asuncion, A., Newman, D.: UCI machine learning repository (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
4. Barber, B., Hamilton, H.J.: Extracting share frequent itemsets with infrequent subsets. Data Min. Knowl. Discov. 7(2), 153–185 (2003)
5. Cai, C.H., Fu, A.W.C., Cheng, C.H., Kwong, W.W.: Mining association rules with weighted items. In: IDEAS 1998: Proceedings of the 1998 International Symposium on Database Engineering & Applications, p. 68. IEEE Computer Society, Washington (1998)
6. Jian, W., Ming, L.X.: An effective mining algorithm for weighted association rules in communication networks. Journal of Computers 3(4) (2008)
7. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer, Heidelberg (October 2002)

8. Sanjay, R., Ranka, S., Tsur, S.: Weighted association rules: Model and algorithm (1997), http://citeseer.ist.psu.edu/185924.html
9. Sun, K., Bai, F.: Mining weighted association rules without preassigned weights. IEEE Trans. on Knowl. and Data Eng. 20(4), 489–495 (2008)
10. Tan, P.N., Kumar, V., Srivastava, J.: Selecting the right interestingness measure for association patterns. In: KDD 2002: Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Alberta, Canada (2002)
11. Tao, F., Murtagh, F., Farid, M.: Weighted association rule mining using weighted support and significance framework. In: KDD 2003: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Dscovery and Data Mining, pp. 661–666. ACM, New York (2003)
12. Yan, L., Li, C.: Incorporating pageview weight into an association-rule-based web recommendation system. In: Australian Conference on Artificial Intelligence, pp. 577–586 (2006)
13. Zaki, M.J.: Generating non-redundant association rules. In: KDD 2000: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 34–43. ACM, New York (2000)

Best Clustering Configuration Metrics: Towards Multiagent Based Clustering

Santhana Chaimontree, Katie Atkinson, and Frans Coenen

Department of Computer Science
University of Liverpool, UK
{S.Chaimontree,katie,coenen}@liverpool.ac.uk

Abstract. Multi-Agent Clustering (MAC) requires a mechanism for identifying the most appropriate cluster configuration. This paper reports on experiments conducted with respect to a number of validation metrics to identify the most effective metric with respect to this context. This paper also describes a process whereby such metrics can be used to determine the optimum parameters typically required by clustering algorithms, and a process for incorporating this into a MAC framework to generate best cluster configurations with minimum input from end users.

Keywords: Cluster Validity Metrics, Multi-Agent Clustering.

1 Introduction

Clustering is a core data mining task. It is the process whereby a set of objects, defined in terms of a global set of features, are categorised into a set of groups *(clusters)* according to some similarity measure or measures. Most clustering algorithms require user-supplied parameters, such as the desired number of clusters or a minimum cluster size. Identification of the most appropriate parameters to produce a "best" cluster configuration is difficult and is normally achieved through a "trial and error" process conducted by the user. The identification of the most appropriate parameters is further hampered by difficulties in defining what we mean by a cluster configuration that "best" fits the underlying data.

The need to be able to automatically identify best parameters with respect to a notion of a "best" cluster configuration is of particular relevance in the context of Multi Agent Clustering (MAC) as suggested in [6,7]. The view of clustering presented in [6] is that of a "anarchic" collection of agents: some equipped with algorithms to conduct clustering operations or to validate the output from clustering algorithms, some holding data, and others performing house keeping and management tasks. The aim of this Multi-Agent System (MAS), although not fully realised, is to produce MAC solutions to clustering problems that require minimal input from the user and output a most appropriate cluster configuration. This objective is currently hampered by the lack of a clear understanding of what is meant by a best cluster configuration, and how this might be measured. Further, assuming we can define a best cluster configuration, how can

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 48–59, 2010.

appropriate parameters be derived so that the desired best configuration can be realised?

In this paper a “best” cluster configuration, in the context of MAC, is defined in terms of some validity metric that must be optimised. The desired configuration is generated using a sequence of parameter values (the nature of which is dependent on the clustering algorithm adopted) to produce a collection of cluster configurations from which the most appropriate can be selected according to the adopted validity metric. Much depends on the accuracy of this metric. In this paper three such metrics are considered: the Silhouette coefficient, the Davies-Bouldin (DB) index and WGAD-BGD (this last derived by the authors).

The rest of the report is organised as follows: Section 2 surveys some relevant previous work. An overview of the metrics used to identify “best” cluster configurations, is given in Section 3. Section 4 describes the generation propose. Some evaluation and comparison is then presented in Section 5. Some conclusions are presented in Section 6.

2 Previous Work

This section provides a review of a number of established metrics used to evalualte clustering results, and current work on MAC within the context of the cluster configuration validity issue identified above. Different clustering algorithms provide different clustering results depending on the characteristics of the input data set and the input parameters used to define the nature of the desired clusters.

Cluster validity techniques are used to evaluate and assess the result of clustering algorithms, and may be used to select the best cluster configuration that best fits the underlying data. The available techniques can be typically classified into: (i) external criteria and (ii) internal criteria [10]. *External criteria* techniques use prelabelled datasets with “known” cluster configurations and measure how well clustering techniques perform with respect to these known clusters. *Internal criteria* techniques are used to evaluate the “goodness” of a cluster configuration without any priory knowledge of the nature of the clusters. This technique uses only the quantities and features inherent in the data set. Techniques based on external criteria require a pre-labelled *training* data set. Techniques based on internal criteria tend to be founded on statistical methods, A major drawback of which is that this often incurs high computational complexity.

A survey of well established cluster validity techniques, with respect to the above, can be found in [10]. Well known techniques include: Dunn indexing [9], Davies-Bouldin indexing[8], BR-index [20], SD [12], S_Dbw [11], the Silhouette coefficient [21], and SSB and SSE [23]. Reviews of a number of these techniques are presented in [16] and [20]. They can be categorised as measuring either: (i) the degree of intra-cluser cohesion, or (ii) the degree of inter-cluster separation; or (iii) both cohesion and separation (i.e. hybrid methods). Four of these techniques (Davies-Bouldin, Silhouette and SSB and SSE) are used in the study described in this paper and are considered in further detail in Section 3. The four techniques were selected as they represent a mixture of approaches; SSB is used to measure inter-cluster separation and SSE is used to measure intra-cluster cohesion, while the Davies-Bouldin index and the Silhouette coefficient represent hybrid methods.

An essential feature of MAC is that the agents should determine the most appropriate number of clusters for a given data set, and the associated parameters required for cluster generation, without end user intervention. As far as the authors are aware there has been no reported work on this element of MAC. There are a number of proposed tools that present alternatives to end-users for selection. The presentation is usually in some graphical format, hence these tools may be labelled as "visual" cluster validation tools. One such tool is CVAP (Cluster Validity Analysis Platform) [24]. CVAP operates by applying a number of clustering algorithms, with a sequence of parameters, to a given data set. Each result is assessed using a "validity" index which is plotted on a graph which may then be inspected and a selection made. Another such tool is described in [17].

There has been some previous work on multi-agent clustering. The earliest reported systems are PADMA [13] and PAPYRUS [4]. The aim of these systems was to achieve the integration of knowledge discovered from different sites with a minimum amount of network communication and a maximum amount of local computation. PADMA is used to generate hierarchical clusters in the context of document categorisation. PADMA agents are employed for local data accessing and analysis. A facilitator (or coordinator) agent is responsible for interaction between the mining agents. All *local clusters* are collected at the central site to generate the *global clusters*. PADMA is therefore based on a centralised architecture, whereas PAPYRUS adopts a Peer-To-Peer model where both data and results can be moved between agents according to given MAS strategies. Another MAC based on the Peer-To-Peer model is proposed in [22] where a distributed density-based clustering algorithm, called KDEC [15], is used. Density estimation samples are transmitted, instead of data values, outside the site of origin in order to preserve data privacy. Reed et al. [19] proposed a MAS for distributed clustering for text documents which assigns new, incoming documents to clusters. The objective here was to improve the accuracy and the relevancy of information retrieval processes. Kiselev et al. [14] proposed a MAC dealing with data streams in distributed and dynamic environments whereby input data sets and decision criteria can be changed at runtime. Clustering results are available at anytime and are continuously revised to achieve the global clustering. (There are also some reported agent based systems for supervised learning, such as that reported in [3,5,1] from which some parallels may be drawn with respect to MAC.) What the above reported systems have in common, unlike the generic vision espoused in this paper, is that they are founded on a specific clustering algorithm that feature preset parameters.

3 Quality Measures

In this section four of the quality measures introduced in the previous section (Davies-Bouldin, Silhouette, SSB and SSE) are considered in further detail.

Table 1. Basic notation

notation	**Description**
K	The number of clusters
N	The number of objects (records) in a data set
$\{C_1, ..., C_K\}$	Set of K clusters
$D = d(x_i, x_j)$	Matrix of "distance" among objects
$x_1, ..., x_N$	Set of objects to be clustered
$minPts$	The minimum number of possible objects in a cluster
$maxPts$	The maximum number of possible objects in a cluster
$\|C_i\|$	The number of objects in a cluster i (i.e. the size of a cluster)

Recall that to determine the effectiveness of a cluster configuration we can adopt three approaches: (i) we can measure the cohesion of the objects within clusters, (ii) the separation between clusters, or (iii) a combination of the two. Cohesion can be measured using the Sum Squared Error (SSE) measure and separation the Sum of Squares Between groups (SSB) measure. While the Davies-Bouldin index and the Silhouette coefficient combine the two. The basic notation used through out the discussion is given in Table 1. For the work described in this paper the maximum number of clusters, $maxPts$, is calculated as the square root of the number of objects (records), N, thus $maxPts = \lceil\sqrt{N}\rceil$. The intuition here is that the maximum number of clusters that the records in a data set can be categorised into is proportional to N (the number of records). The minimum number of clusters ($minPts$) is typically set to 2.

SSE [23] is the sum of the squared intra-cluster distance of each cluster centroid, c_i, to each point (object) x_j in that cluster. More formaly the total SSE of a given cluster configuration is defined as:

$$Total\ SSE = \sum_{i=1}^{i=K} \sum_{j=1}^{j=|C_i|} dist(x_j, c_i)^2 \ . \tag{1}$$

The lower the total SSE value the greater the intra-cluster cohesion associated with the given cluster configuration.

SSB [23], in turn, is the sum of the squared distance of each cluster centroid (c_i) to the overall centroid of the cluster configuration (c) multiplied (in each case) by the size of the cluster. Formaly the total SSB of a given cluster configuration is defined as:

$$Total\ SSB = \sum_{i=1}^{i=K} |C_i|(dist(c_i, c))^2 \ . \tag{2}$$

The higher the total SSB value of a cluster configuration the greater the degree of separation. Note that the cluster centriod is used to represent all the points in a cluster, sometimes referred to in the literature as the *cluster prototype*, so as to reduce the overall computational complexity of the calculation (otherwise distance from every point to every other point would have to be calculated).

Using the above the calculated SSE and SSB values tend to be large numbers because of the squared operation. Therefore, in the context of the work described here, the authors use variations of SSE and SSB metrics. We refer to these metrics as the total Within Group Average Distance (WGAD) [18] and the total Between Group Distance (BGD). WGAD and BGD are derermined as follows:

$$Total\ WGAD = \sum_{i=1}^{i=K} \frac{\sum_{j=1}^{j=|C_i|} \ dist(x_j, c_i)}{|C_i|} \ . \tag{3}$$

$$Total\ BGD = \sum_{i=1}^{i=K} dist(c_i, c) \ . \tag{4}$$

Experiments conducted by the authors (not reported here) suggested that both separation and cohesion are significant and thus we find the difference between a pair of WGAD and BGD values to express the overall validity of a given cluster configuration. We refer to this measure as the WGAD-BGD measure. To obtain a best cluster configuration we must minimise the WGAD-BGD measure.

The Overall Silhouette Coefficient (*OverallSil*) of a cluster configuration is measure of both the cohesiveness and separation of the configuration [21]. It is determined by first calculating the *silhouette* (*Sil*) of each individual point x_j within the configuration as follows:

$$Sil(x_j) = \frac{b(x_j) - a(x_j)}{max(a(x_j), b(x_j))} \ . \tag{5}$$

where $a(x_j)$ is the average intra-cluster distance of the point x_j to all other points within its cluster, and $b(x_j)$ is the minimum of the average inter-cluster distances of x_j to all points in each other cluster (see Figure 1 for further clarification). The overall silhouette coefficient (*OverallSil*) is then calculated as follows:

$$OverallSil = \frac{\sum_{i=1}^{i=K} \frac{\sum_{j=1}^{j=|C_i|} sil(x_j)}{|C_i|}}{K} \ . \tag{6}$$

The resulting overall silhouette coefficient is then a real number between −1.0 and 1.0. If the silhouette coefficient is close to −1, it means the cluster is undesirable because the average distance to points in the cluster, is greater than the minimum average distance to points in the other cluster(s). The overall silhouette coefficient can be used to measure the goodness of a cluster configuration. The larger the coefficient the better the cluster configuration.

The Davies-Bouldin (DB) *validity index* is the sum of the maximum ratios of the intra-cluster distances to the inter-cluster distances for each cluster i:

$$DB = \frac{1}{K} \sum_{i=1}^{i=K} R_i \ . \tag{7}$$

Where R_i is the maximum of the ratios between cluster i and each other cluster j (where $1 \leq j \leq K$ and $j \neq i$). The lower the DB value the better the associated cluster configuration. The individual ratio of the intra-cluster distances to the inter-cluster distances for cluster i with respect to cluster j is given by:

$$R_{ij} = \frac{S_i - S_j}{d_{ij}} \ . \tag{8}$$

where d_{ij} is a distance between the centroid of cluster i and the centroid of cluster j, and S_i (S_j) is the average distance between the points within cluster i (j):

$$S_i = \frac{1}{|C_i|} \sum_{n=1}^{n=|C_i|} d(x_n, c_i) \ . \tag{9}$$

where c_i is the centroid of cluster i, and x is a point (object) within the cluster i. The derivation of DB is illustrated in Figure 2.

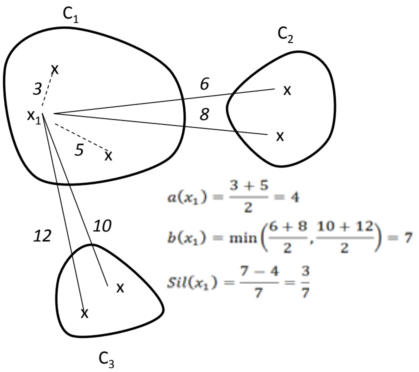

C1
C2
C3
x
3
x1
5
6
8
12
10
x
x
x
x
x
$a(x_1) = \frac{3+5}{2} = 4$
$b(x_1) = \min\left(\frac{6+8}{2}, \frac{10+12}{2}\right) = 7$
$Sil(x_1) = \frac{7-4}{7} = \frac{3}{7}$

Fig. 1. Derivation of the Overall Silhouette Coefficient (*OverallSil*)

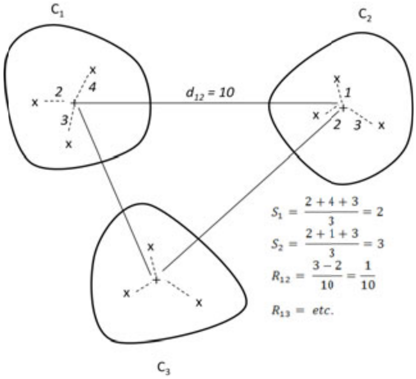

C1
C2
C3
$d_{12} = 10$
2
4
3
1
2
3
x
$S_1 = \frac{2+4+3}{3} = 2$
$S_2 = \frac{2+1+3}{3} = 3$
$R_{12} = \frac{3-2}{10} = \frac{1}{10}$
R_{13} = etc.

Fig. 2. Derivation of the Davies-Bouldin (DB) *validity index*

4 Parameter Identification for Clustering Algorithms

Most clustering algorithms require user-supplied parameters. For example: K-means requires the number of classes, K, as a parameter; and KNN requires a threshold (t) to identify the "nearest neighbour". To obtain a best cluster configuration in terms of the metrics described above the most appropriate parameters are required. To act as a focus for this part of the research reported in this paper the well known K-means and KNN algorithms were adopted, although other clustering algorithms could equally well have been adopted.

The most obvious mechanism for indentifying appropriate parameters is to adopt some kind of *generate and test* loop whereby a cluster configuration is "generated" using a particular parameter value which is then "tested" (evaluated) using a validity metric (such as those discussed in the foregoing section) as a result of which the parameter value is adjusted. However, this did not prove successful. Figure 3 shows the validity measures obtained using the silhouette coefficient (Sil. Coef.) and the Davies-Bouldin index (DB) and with a range of t values using the KNN clustering algorithm when applied to the Iris data set taken from the UCI data repository [2] (similar results were obtained using other data sets). From the figure it can be seen that there are local maxima and minima which means that a generate and test procedure is unlikely to prove successful. Figure 4 shows the WGAD-BGD measures obtained using the KNN algorithm and the Iris data set. Recall that, we wish to minimise this measure. From the figure it can be seen that a range of "best" t values are produced. Thus a generate and test process would not find the most appropriate parameters. Instead, the process advocated here is to generate a sequence of cluster configurations for a range of parameter values.

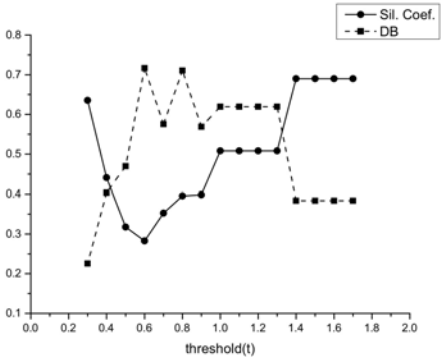


Fig. 3. Validity values using Sil.Coef. and DB index for KNN algorithm using the Iris data set

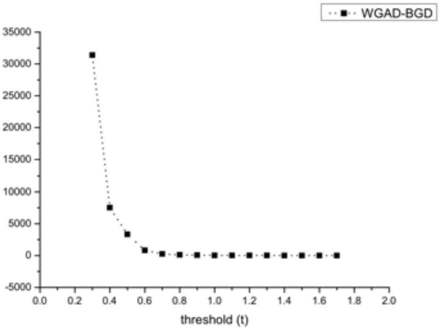


Fig. 4. Validity values using WGAD-BGD for KNN algorithm using the Iris data set

Thus, for K-means, to identify the most appropriate number of clusters the algorithm is run multiple times with a sequence of values for K ranging from 2 to $\lceil\sqrt{N}\rceil$, where N is the number of records in the given data set. The "goodness" of each generated cluster configuration was tested using the identified validity metrics. The generation algorithm is presented in Table 2.

In the case of KNN the algorithm is run multiple times with a range of different values for the nearest neighbour threshold (t). Note that any t value that does not generate a number of clusters of between 2 and $\lceil\sqrt{N}\rceil$ is ignored. The algorithm is presented in Table 3.

The process was incorporated into a MAC framework founded on earlier work by the authors and reported in [6,7]. An issue with K-means is that the initial points (records/objects) used to define the initial centroids of the clusters are randomly selected. Experiments indicated that the selection of start points can greatly influence the operation of K-means, to the extent that different best values of K can be produced depending on where in the data set the algorithm starts. Therefore use of K-means to identify a "proper" number of clusters does

Table 2. Algorithm to identify the most appropriat number of clusters using K-means

```
Algorithm: KmeansIdentifiesK

Input: $x_1, ..., x_N$
Output: $K$, the overall validity value

1. For $K = 2$ to $K = \lceil\sqrt{N}\rceil$ do
2.       Do K-means clustering
3.       Evaluate the clustering result (a set of clusters) by using a chosen metric
4.       Keep $K$ and the overall cluster validity
5. Select $K$ which provide the good cluster configuration
```

Table 3. Algorithm to identify the most appropriate number of clusters using KNN

```
Algorithm KNNIdentifiesK

Input: $x_1, ..., x_N$
Output: $t$, $K$, the overall validity value

1. Calculate distance between objects in a data set
2. Ascending order all of objects in the data set
3. Choose one point in which its position is at the middle
4. Choose distance between the object from step 3 and other objects in the data set
5. Set $t = distance$
6. Ascending order distances and select distinct distances
7. For each $t$ do
8.       Do KNN clustering
9.       If the result generated from step (8) is different from the last result
10.            Evaluate a clustering result.
11.      Keep $t$, $K$ and the overall validity value
12. Select $t$ providing the "best" cluster configuration
```

not represent a consistent approach. However, the KNN parameter selection process can be used to indirectly determine the most appropriate value for K to be used in the K-means approach.

5 Experimental Evaluation

In Section 3 a number of metrics for identifying a best cluster configuration were identified. These in turn were incorporated into the parameter identifictaion mechanism identified in the foregoing section. The evaluation of the proposed approach is presented in this section. The evaluation was conducted using ten data sets taken from the UCI repository [2]. Table 4 gives some statistical information concerning these data sets. Note that the data sets display a variety of features.

Table 4. Statistical information for the datasets used in the evaluation

No.	Data Set	Num Records (N)	Num Attr	Num Classes	Attribute Description
1	Iris	150	4	3	4 Numeric
2	Zoo	101	16	7	15 Boolean, 1 Numeric
3	Wine	178	13	3	13 Numeric
4	Heart	270	13	2	6 Real, 1 Ordered, 3 Binary, 3 Nominal
5	Ecoli	336	7	8	7 Real
6	Blood Transfusion	748	4	2	4 Integer
7	Pima Indians	768	8	2	8 Numeric
8	Yeast	1484	8	10	8 Numeric
9	Red wine quality	1599	11	6	11 Numeric
10	Breast Cancer	569	21	2	21 Real

Table 5. Results using K-means to generate a best set of clusters

No.	Data Set	Known K	K	Sil. Coef.	K	DB Index	K	WGAD-BGD
1	Iris	3	2	0.68	4	0.31	2	3.93
2	Zoo	7	4	0.47	8	0.68	2	3.66
3	Wine	3	2	0.66	12	0.21	2	583.02
4	Heart	2	**2**	0.38	12	0.67	**2**	81.75
5	Ecoli	8	4	0.43	12	0.67	2	0.57
6	Blood Transfusion	2	**2**	0.70	15	0.20	**2**	3044.25
7	Pima Indians	2	**2**	0.57	3	0.51	**2**	223.53
8	Yeast	10	3	0.27	26	0.78	2	0.29
9	Red Wine	6	2	0.60	34	0.87	2	3.00
10	Breast cancer	2	**2**	0.70	20	0.29	**2**	1331.33

Table 5 shows a comparison of the operation of the above process using: K-means; and the identified cluster validity techniques, namely silhouette coefficient (Sil. Coef.), Davies-Bouldin index (DB index), and the combination between WGAD and BGD (WGAD-BGD); when applied to the data sets listed in Table 4. The comparison is conducted by considering the number of clusters (K) associated with the best cluster configuration and the known value for K. Table 6 presents a similar comparison using the KNN algorithm.

In both Table 5 and Table 6 the values in bold indicate where the identified number of clusters (K) exactly matches the "known" value of K. (In Table 5 it can also be argued that the K value of 8 produced using the DB index metric is significantly close to the "required" value of 7.)

The results are summarised in Table 7 (again values in bold indicate best results). From Table 7 it can be observed that the DB-index does not perform well, particularly when used in conjunction with K-means. In this latter case the use of DB-index over specified the number of clusters that define a best

Table 6. Results using KNN to generate a best set of clusters

No.	Data Set	Known	*t*	*K*	Sil.	*t*	*K*	DB	*t*	*K*	WGAD-
		K			Coef.			Index			BGD
1	Iris	3	1.49	2	0.69	1.49	2	0.38	1.49	2	3.97
2	Zoo	7	2.83	2	0.40	2.65	4	0.71	2.83	2	3.94
3	Wine	3	195.07	**3**	0.61	75.78	14	0.33	206.37	2	810.82
4	Heart	2	81.63	**2**	0.76	81.63	**2**	0.16	81.63	**2**	316.67
5	Ecoli	8	0.53	2	0.39	0.28	10	0.39	0.53	2	0.79
6	Blood Transfusion	2	2000.02	**2**	0,83	250.27	18	0.12	2000.02	**2**	7404.08
7	Pima Indians	2	193.36	**2**	0.79	193.36	**2**	0.23	193.36	**2**	682.60
8	Yeast	10	0.60	2	0.74	0.26	21	0.31	0.60	2	0.74
9	Red Wine	6	1.42	3	0.17	1.42	3	2.21	1.42	3	8.47
10	Breast cancer	2	992.39	**2**	0.80	992.39	**2**	0.13	992.39	**2**	3888.91

Table 7. Summary of results

			KNN			K-Means		
No.	Data Set	*K*	*K*	*K*	*K*	*K*	*K*	*K*
		UCI	Sil.	DB	WGAD-	Sil.	DB	WGAD-
			Coef.	index	BGD	Coef.	index	BGD
1	Iris Plants	3	2	2	2	2	4	2
2	Zoo	7	2	4	2	4	8	2
3	Red Wine	3	**3**	14	2	2	12	2
4	Heart	2	**2**	**2**	**2**	**2**	12	**2**
5	Ecoli	8	2	10	2	4	12	2
6	Blood Transfusion	2	**2**	18	**2**	**2**	15	**2**
7	Pima Indians	2	**2**	**2**	**2**	**2**	3	**2**
8	Yeast	10	2	21	2	3	26	2
9	Car	6	3	3	3	2	34	2
10	Breast Cancer	2	**2**	**2**	**2**	**2**	20	**2**
	Totals		5	3	4	4	0	4

cluster configuration, in all cases. Both the Silhouette Coefficient and WGAD-RGD produced better results, with the best result generated using Silhouette in conjunction with KNN. Further experiments using K-means demonstrated that the results obtained were very inconsistent in that they were very dependent on the nature of the selected start locations (centroids). In this respect KNN produced more consistent results. It is also worth nothing that using some data sets (Heart, Pima Indians and Breast Cancer) consistent results were obtained, with respect to other data sets (Iris, Zoo, Ecoli, Yeast and Car) poorer results were obtained. Inspection of the nature of these data sets (Table 4) indicats that the proposed technique operates best where data sets feature a small number of clusters (classes).

6 Conclusions

In this paper a set of experiments directed at identifying the most appropriate metrics to determine the validity of a cluster configulation have been reported. Three validity metrics were considered, the silhouette coeficient, the DB index and WGAD-BGD. The last being a combination of the SSB and SSE metrics. Experiments were conducted using K-means and KNN, and a collection of data sets taken from the UCI repository. The context of the work described was Multi-Agent Clustering (MAC) directed at generating a cluster configuration that best fits the input data using a variety of clustering mechanisms. This in turn required a mechanism for identifying best cluster configurations. The main findings of the work are as follows: (i) the overall best cluster configuration validation technique is the Silhouette coefficient, (ii) KNN is a much more reliable technique to find the best value for K than K-means (but can be used to discover the K value required by K-means), and (iii) a *generate and test* process does not necessarily achieve the desired result.

These findings are currently being incorporated into a MAS framework [6,7]. The techniques investigated sofar, and reported here, do not serve to find the best results in all cases and further investigation is therefore required, however the authors are greatly encouraged by the result reported in this paper.

References

1. Albashiri, K.A., Coenen, F.: Agent-enriched data mining using an extendable framework. In: Cao, L., Gorodetsky, V., Liu, J., Weiss, G., Yu, P.S. (eds.) Agents and Data Mining Interaction. LNCS (LNAI), vol. 5680, pp. 53–68. Springer, Heidelberg (2009)
2. Asuncion, A., Newman, D.: UCI machine learning repository (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
3. Baik, S., Bala, J., Cho, J.: Agent based distributed data mining. In: Liew, K.-M., Shen, H., See, S., Cai, W. (eds.) PDCAT 2004. LNCS, vol. 3320, pp. 185–199. Springer, Heidelberg (2004)
4. Bailey, S., Grossman, R., Sivakumar, H., Turinsky, A.: Papyrus: A system for data mining over local and wide area clusters and super-clusters. IEEE Supercomputing (1999)
5. Canuto, A.M.P., Campos, A.M.C., Bezerra, V.M.S., Abreu, M.C.d.C.: Investigating the use of a multi-agent system for knowledge discovery in databases. International Journal of Hybrid Intelligent Systems 4(1), 27–38 (2007)
6. Chaimontree, S., Atkinson, K., Coenen, F.: Clustering in a multi-agent data mining environment. In: Cao, L., Bazzan, A.L.C., Gorodetsky, V., Mitkas, P.A., Weiss, G., Yu, P.S. (eds.) ADMI 2010. LNCS (LNAI), vol. 5980, pp. 103–114. Springer, Heidelberg (2010)
7. Chaimontree, S., Atkinson, K., Coenen, F.: Multi-agent based clustering: Towards generic multi-agent data mining. In: Perner, P. (ed.) ICDM 2010. LNCS (LNAI), vol. 6171, pp. 115–127. Springer, Heidelberg (2010)
8. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1(2), 224–227 (1979)

9. Dunn, J.C.: Well separated clusters and optimal fuzzy-partitions. Journal of Cybernetics 4, 95–104 (1974)
10. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: Cluster validity methods: part I. SIGMOD Record 31(2), 40–45 (2002)
11. Halkidi, M., Vazirgiannis, M.: Clustering validity assessment: Finding the optimal partitioning of a data set. In: ICDM 2001: Proceedings of the 2001 IEEE International Conference on Data Mining, pp. 187–194. IEEE Computer Society, Washington (2001)
12. Halkidi, M., Vazirgiannis, M., Batistakis, Y.: Quality scheme assessment in the clustering process. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 265–276. Springer, Heidelberg (2000)
13. Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining using an agent based architecture. In: Proceedings the Third International Conference on the Knowledge Discovery and Data Mining, pp. 211–214. AAAI Press, Menlo Park (1997)
14. Kiselev, I., Alhajj, R.: A self-organizing multi-agent system for online unsupervised learning in complex dynamic environments. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, pp. 1808–1809. AAAI Press, Menlo Park (2008)
15. Klusch, M., Lodi, S., Moro, G.: Agent-based distributed data mining: The KDEC scheme. In: Klusch, M., Bergamaschi, S., Edwards, P., Petta, P. (eds.) Intelligent Information Agents. LNCS (LNAI), vol. 2586, pp. 104–122. Springer, Heidelberg (2003)
16. Legány, C., Juhász, S., Babos, A.: Cluster validity measurement techniques. In: AIKED 2006: Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases, pp. 388–393. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point (2006)
17. Qiao, H., Edwards, B.: A data clustering tool with cluster validity indices. In: ICC 2009 - International Conference of Computing in Engineering, Science and Information, pp. 303–309 (2009)
18. Rao, M.: Clustering analysis and mathematical programming. Journal of the American statistical association 66(345), 622–626 (1971)
19. Reed, J.W., Potok, T.E., Patton, R.M.: A multi-agent system for distributed cluster analysis. In: Proceedings of Third International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS 2004) W16L Workshop - 26th International Conference on Software Engineering. pp. 152–155. IEE, Edinburgh (2004)
20. Ristevski, B., Loshkovska, S., Dzeroski, S., Slavkov, I.: A comparison of validation indices for evaluation of clustering results of DNA microarray data. In: 2nd International Conference on Bioinformatics and Biomedical Engineering, iCBBE 2008, pp. 587–591 (2008)
21. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20(1), 53–65 (1987)
22. da Silva, J., Klusch, M., Lodi, S., Moro, G.: Privacy-preserving agent-based distributed data clustering. Web Intelligence and Agent Systems 4(2), 221–238 (2006)
23. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Reading (2005)
24. Wang, K., Wang, B., Peng, L.: CVAP: Validation for cluster analyses. Data Science Journal 8, 88–93 (2009)

On Probabilistic Models for Uncertain Sequential Pattern Mining

Muhammad Muzammal and Rajeev Raman

Department of Computer Science, University of Leicester, UK
{mm386,r.raman}@mcs.le.ac.uk

Abstract. We study uncertainty models in *sequential pattern mining*. We consider situations where there is uncertainty either about a *source* or an *event*. We show that both these types of uncertainties could be modelled using probabilistic databases, and give possible-worlds semantics for both. We then describe "interestingness" criteria based on two notions of frequentness (previously studied for frequent itemset mining) namely *expected support* [C. Aggarwal et al. KDD'09;Chui et al., PAKDD'07,'08] and *probabilistic frequentness* [Bernecker et al., KDD'09]. We study the interestingness criteria from a complexity-theoretic perspective, and show that in case of source-level uncertainty, evaluating probabilistic frequentness is #P-complete, and thus no polynomial time algorithms are likely to exist, but evaluate the interestingness predicate in polynomial time in the remaining cases.

Keywords: Mining Uncertain Data, Sequential Pattern Mining, Probabilistic Databases, Novel Algorithms for Mining, Theoretical Foundations of Data Mining.

1 Introduction

The problem of *sequential pattern mining (SPM)*, or finding frequent sequences of events in data with a temporal component, was originally proposed by Agrawal and Srikant for analyzing retail data [16], [3]. Since then, it has gained a great deal of attention [23], [15], [4] and has found applications in other domains, including discovering web access patterns and motif extraction [9], [20].The classical SPM problem assumes that the data is *deterministic*, or entirely determined. However, it is recognized that data obtained from a wide range of data sources is inherently uncertain [1], [17]. An approach to modelling uncertainty that has seen increased interest recently is that of *probabilistic* databases [17], [1]; even more recently data mining and ranking problems have been studied in the context of probabilistic databases, including top-k [24], [7] and frequent itemset mining (FIM) [2], [5], [6]. This paper considers the SPM problem in probabilistic databases.

Although SPM is superficially similar to FIM, we observe that generalizing SPM to probabilistic databases gives rise to some subtle issues. As we show, different ways of formalizing the problem can lead to very different outcomes in

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 60–72, 2010.

terms of computational complexity. We now outline the main issues, for which we give a high-level overview of SPM (a formal definition is in Section 2). In SPM, *events* (retail transactions, observations of objects/persons) are associated with *sources* (customers, sensors/cameras). Events have a time-stamp, so the event database is easily re-organized into a collection of *source sequences*, each of which is a sequence of events (ordered by time stamp) associated with a given source. The SPM problem is to find sequential patterns – patterns of events that have a temporal order – that occur in a significant number of source sequences. Unlike FIM, uncertainty in SPM can occur in three different aspects: the source, the event and the time may all be uncertain. Uncertainty in time seems not well-suited to the probabilistic database approach, and we focus on uncertainty in (any one of) the source and the event.

Source-Level Uncertainty (SLU). In the following situations, an event is recorded deterministically, but the source is not readily identifiable:

(a) a customer (source) purchases some items (event) from a store, and provides identity information (loyalty programme, filling a form etc.). However, multiple matches may emerge in the customer database: the customer's details may be incomplete or incorrect, or the customer database itself may be probabilistic as a result of "deduplication" or cleaning [11].

(b) a vehicle/person is identified by a sensor/camera using methods such as biometrics or automatic number plate recognition (ANPR), which are inherently noisy. To mine patterns such as "10% of cars pass camera X, then camera Y and later camera Z", we consider each car as a source and each sighting as an event.

In such scenarios, it is certain that an event occurred (e.g. a customer bought some items, a vehicle/person entered an area) but the source associated with that event is uncertain. The software would typically assign confidence values to the various alternative matches. Whilst large databases of type (a) are readily available in commercial organizations, large databases of type (b) are now becoming more common, a notable example being the UK's police ANPR database [21]. We model the above scenarios by assuming that each event is associated with a probability distribution over possible sources that could have resulted in the event. This formulation thus shows *attribute-level* uncertainty [17], as the "source" attribute of each event is uncertain.

Event-Level Uncertainty (ELU). In some scenarios, the source of the data is known, but the events are uncertain. Such scenarios could be modelled as a sequence of events output by a source, but with each event only having a certain probability of having truly occurred (i.e. *tuple-level* uncertainty [17]), or as above, using attribute-level uncertainty. For example:

(a) Employees movements are tracked in a building using RFID sensors [13]. The stream of tags read by various sensors is stored in a relation `SIGHTING(t, tID, aID)`, which denotes that the RFID tag `tID` was detected by antenna `aID` at time `t`. Since an RFID antenna has only a certain probability of reading a tag within its range, the PEEX system [13] processes the

SIGHTING relation to output an *uncertain* higher-level event relation such as `MEETING(time, person1, person2, room, prob)`. An example tuple in MEETING could be (103, 'Alice', 'Bob', 435, 0.4), which means that at time 103, PEEX believes that Alice and Bob are having a meeting (event) with probability 0.4 in room 435 (source) [13]; since antennae are at fixed locations, the source is certain but the event is uncertain.[1]

(b) A logged-in user (source) enters terms into a search engine (events), which are disambiguated in many different ways e.g. a search term "Tiger" could be disambiguated as $\langle(\text{Animal}, 0.5), (\text{Sports Personality}, 0.3), (\text{Airplane}, 0.1), \ldots\rangle$.

Our Contributions. The problem of SPM in probabilistic databases was introduced in [14]. Our main contributions are as follows.

(1) We formalize the notion of event-level uncertainty, provide a possible-worlds semantics, and contrast it with source-level uncertainty. From the possible-worlds semantics, we obtain definitions for frequent sequences under two measures: *expected support* and *probabilistic frequentness*. These measures were used earlier for FIM in probabilistic data [5], [6]. For SPM [14] addressed source-level uncertainty with the expected support; the other three combinations are new.

(2) We discuss the computational complexity of the fundamental question:

Given a sequence s and a probabilistic database, is s frequent?

In the framework described in [10], which includes not only SPM but also FIM and a host of other database optimization and machine learning problems, the above question is the *interestingness* or *quality* predicate. As noted in [10], given such a predicate as a "black box", one can embed it into a variety of candidate generate-and-test frameworks for finding not only frequent sequential patterns, but also maximal frequent sequential patterns. Many popular classical SPM algorithms such as GSP [16], SPADE [23] or SPAM [4] all fit into this framework, and algorithms such as PrefixSpan [15], which do not explicitly generate candidates, also implicitly evaluate the interestingness predicate.

Whilst the interestingness predicate can often be readily evaluated in polynomial time in the deterministic setting, this is not so clear in the probabilistic setting, as a naive approach would involve enumerating exponentially many possible worlds. We show a significant difference from a complexity-theoretic viewpoint between the four variants of the problem (two models and two measures); whilst for three variants, it takes polynomial time to check if a sequence is frequent, for the fourth, this task is #P-complete, and thus no polynomial-time algorithms are likely to exist.

Related Work. Classical SPM has been studied extensively [16], [23], [15], [4]. Modelling uncertain data as *probabilistic* databases [17], [1] has led to several ranking/mining problems being studied in this context. The top-k problem (a ranking problem) was intensively studied recently (see [12], [24], [7] and references therein). In particular [7] highlights subtle issues in the semantics of

[1] Some formulations of this and similar problems may even exhibit SLU.

the term "top-k" when applied to probabilistic databases. FIM in probabilistic databases was studied under the *expected* support measure in [2], [6]; and under the *probabilistic frequentness* measure in [5]. A dichotomy result, showing that certain queries on probabilistic databases are either #P-complete or in PTIME was given in [8], but this applies to "tuple-level independent" probabilistic structures, which do not model SPM (upper bounds) or source-level uncertainty for SPM (for hardness). To the best of our knowledge, apart from [14], the SPM problem in probabilistic databases has not been studied. However, uncertainty in the time-stamp attribute was considered in [18] – we do not consider time to be uncertain. Also [22] studies SPM in "noisy" sequences, but the model proposed there is very different to ours and does not fit in the probabilistic database framework.

2 Problem Statement

2.1 Models of Uncertainty

In this section, we give formal definitions of the source-level and event-level uncertain models. We begin by formally defining the classical SPM problem, then the event-level uncertain model. When defining the event-level uncertain model, for simplicity we consider only the tuple-level uncertain case (example in Table 1). We then recap the source-level uncertain case from [14].

Deterministic SPM. We begin by recapitulating the definition of standard SPM. Let $\mathcal{I} = \{i_1, i_2, \ldots, i_q\}$ be a set of *items* and $\mathcal{S} = \{\sigma_1, \ldots, \sigma_m\}$ be a set of *sources*. An *event* $e \subseteq \mathcal{I}$ is a collection of items. A *database* $D = \langle r_1, r_2, \ldots, r_n \rangle$ is an ordered list of *records* such that each $r_i \in D$ is of the form (eid_i, e_i, σ_i), where eid_i is an event-id, e_i is an event and σ_i is a source. A *sequence* $s = \langle s_1, s_2, \ldots, s_a \rangle$ is an ordered list of events. Let $s = \langle s_1, s_2, \ldots, s_q \rangle$ and $t = \langle t_1, t_2, \ldots, t_r \rangle$ be two sequences. We say that s is a *subsequence* of t, denoted $s \preceq t$, if there exist integers $1 \leq i_1 < i_2 < \cdots < i_q \leq r$ such that $s_k \subseteq t_{i_j}$, for $k = 1, \ldots, q$. The *source sequence* corresponding to a source i, denoted by D_i, is just the multiset $\{e | (eid, e, i) \in D\}$, ordered by *eid*. For a sequence s and source i, let $X_i(s, D)$ be an indicator variable, whose value is 1 if $s \preceq D_i$, and 0 otherwise. The objective is to find all sequences s whose *support* (*Supp*) is at least some user-defined threshold $\theta, 1 \leq \theta \leq m$, where:

$$Supp(s, D) = \sum_{i=1}^{m} X_i(s, D). \tag{1}$$

Event-Level Uncertainty. A *probabilistic database* D^p is a collection of *p-sequences* $D_1^p, \ldots, D_m^p$, where D_i^p is the p-sequence of source $i \in \mathcal{S}$. A p-sequence is the natural analogue of the source sequence in classical SPM, and each p-sequence is of the form $\langle (e_1, c_1) \ldots (e_k, c_k) \rangle$, where the e_j's are events (ordered by *eid*) and c_j is the confidence that e_j actually occurred. In examples, we write a p-sequence $\langle (\{a, d\}, 0.4), (\{a, b\}, 0.2) \rangle$ as $(a, d : 0.4)(a, b : 0.2)$. An example of an event-level uncertain database is in Table 1(L). The possible worlds semantics

Table 1. An event-level uncertain database (L), and all possible worlds of D^p_Y along with their probabilities (R)

	p-sequence
D^p_X	$(a, d : 0.6)(a, b : 0.3)(b, c : 0.7)$
D^p_Y	$(a, d : 0.4)(a, b : 0.2)$
D^p_Z	$(a : 1.0)(a, b : 0.5)(b, c : 0.3)$

$\langle\rangle$	$(1 - 0.4) \times (1 - 0.2) = 0.48$
$\{(a, d)\}$	$(0.4) \times (1 - 0.2) = 0.32$
$\{(a, b)\}$	$(1 - 0.4) \times (0.2) = 0.12$
$\{(a, d)(a, b)\}$	$(0.4) \times (0.2) = 0.08$

Table 2. The set of possible worlds for every p-sequence in Table 1 (L). In all, there are $8 \times 4 \times 4 = 128$ possible worlds of D^p, one such possible world is shown (R).

$PW(D^p_X)$	$\{\langle\rangle = 0.084\}; \{(a, d) = 0.126\}; \{(a, b) = 0.036\}; \{(b, c) = 0.196\};$ $\{(a, d)(a, b) = 0.054\}; \{(a, d)(b, c) = 0.294\};$ $\{(a, b)(b, c) = 0.084\}; \{(a, d)(a, b)(b, c) = 0.126\}$
$PW(D^p_Y)$	$\{\langle\rangle = 0.48\}; \{(a, d) = 0.32\}; \{(a, b) = 0.12\}; \{(a, d)(a, b) = 0.08\}$
$PW(D^p_Z)$	$\{(a) = 0.35\}; \{(a)(a, b) = 0.35\}; \{(a)(b, c) = 0.15\};$ $\{(a)(a, b)(b, c) = 0.15\}$

D^*_X	$\{(a, d)$ (b,c)$\}$	0.294
D^*_Y	$\{(a, d)\}$	0.32
D^*_Z	$\{(a)$ $(a, b)\}$	0.35

of D^p is as follows. For each event e_j in a p-sequence D^p_i there are two kinds of worlds; one in which e_j occurs and the other where it does not. Let *occur* $= \{x_1, \ldots, x_l\}$, where $1 \leq x_1 < \ldots < x_l \leq k$, be the indices of events that occur in D^*_i. Then $D^*_i = \langle e_{x_1}, \ldots, e_{x_l}\rangle$, and $\Pr(D^*_i) = \prod_{j \in occur} c_j * \prod_{j \notin occur}(1 - c_j)$. In other words, we assume that the events in a p-sequence are stochastically independent. For example, all possible worlds of D^p_Y are shown in Table 1(R) along with detailed probabilities. The set of possible worlds of D^p_i, denoted by $PW(D^p_i)$, is obtained by taking all possible 2^l alternatives for *occur*, and we say $PW(D^p) = PW(D^p_1) \times \ldots \times PW(D^p_m)$. The full set of possible worlds for each source is shown in Table 2(L). For any $D^* \in PW(D^p)$ such that $D^* = (D^*_1, \ldots, D^*_m)$, the probability of D^* is given by $\Pr[D^*] = \prod_{i=1}^{m} \Pr(D^*_i)$, i.e. we assume that the p-sequences of all sources are mutually independent. An example possible world D^* is shown in Table 2(R), which is obtained by taking one possible world each from the worlds of every p-sequences in Table 2(L). The probability of D^* is the product of probabilities of all the worlds in it, $\Pr[D^*]$ $= 0.29 \times 0.32 \times 0.35 = 0.03$.

Source-Level Uncertainty. A *probabilistic database* D^p is an ordered list of records $\langle r_1, \ldots, r_n\rangle$ of the form (eid, e, W) where eid is an event-id, e is an event and W is a probability distribution over $\mathcal{S}$. The distribution W contains pairs of the form (σ, c), where $\sigma \in \mathcal{S}$ and $0 < c \leq 1$ is the confidence that the event e is associated with source σ; we assume $\sum_{(\sigma,c) \in W} c = 1$. An example of a source-level uncertain database is in Table 3(L). A *possible world* D^* of D^p is generated by taking each event e_i in turn, and assigning it to one of the possible sources $\sigma_i \in W_i$, where $\sigma_i \in \mathcal{S}$. Thus every record $r_i = (eid_i, e_i, W_i) \in D^p$ takes the form $r'_i = (eid_i, e_i, \sigma_i)$, for some $\sigma_i \in \mathcal{S}$ in D^*. By enumerating all such possible combinations, we get the complete set of possible worlds. Assuming that the distributions associated with each record r_i in D^p are stochastically independent, the probability of a possible world D^* is $\Pr[D^*] = \prod_{i=1}^{n} \Pr_{W_i}[\sigma_i]$. For example,

Table 3. A source-level uncertain database (L) transformed to p-sequences (R). Note that the p-sequence representation is the same as the event-level uncertain database of Table 1. However, events like e_1 (marked with † on (R)) can only be associated with one of the sources X and Y in any possible world.

eid	event	W
e_1	(a, d)	$(X : 0.6)(Y : 0.4)$
e_2	(a)	$(Z : 1.0)$
e_3	(a, b)	$(X : 0.3)(Y : 0.2)(Z : 0.5)$
e_4	(b, c)	$(X : 0.7)(Z : 0.3)$

	p-sequence
D^p_X	$(a, d : 0.6)^{\dagger}(a, b : 0.3)(b, c : 0.7)$
D^p_Y	$(a, d : 0.4)^{\dagger}(a, b : 0.2)$
D^p_Z	$(a : 1.0)(a, b : 0.5)(b, c : 0.3)$

Table 4. All possible worlds for the database of Table 3 along with their probabilities. The right-most column shows the support of the sequence (a)(b) in each possible world.

D^*	X	Y	Z	$\Pr(D^*)$	
D^*_1	$(a, d : 0.6)(a, b : 0.3)(b, c : 0.7)$	$\langle\rangle$	$(a : 1.0)$	0.126	1
D^*_2	$(a, d : 0.6)(a, b : 0.3)$	$\langle\rangle$	$(a : 1.0)(b, c : 0.3)$	0.054	2
D^*_3	$(a, d : 0.6)(b, c : 0.7)$	$(a, b : 0.2)$	$(a : 1.0)$	0.084	1
D^*_4	$(a, d : 0.6)$	$(a, b : 0.2)$	$(a : 1.0)(b, c : 0.3)$	0.036	1
D^*_5	$(a, d : 0.6)(b, c : 0.7)$	$\langle\rangle$	$(a : 1.0)(a, b : 0.5)$	0.210	2
D^*_6	$(a, d : 0.6)$	$\langle\rangle$	$(a : 1.0)(a, b : 0.5)(b, c : 0.3)$	0.090	1
D^*_7	$(a, b : 0.3)(b, c : 0.7)$	$(a, d : 0.4)$	$(a : 1.0)$	0.084	1
D^*_8	$(a, b : 0.3)$	$(a, d : 0.4)$	$(a : 1.0)(b, c : 0.3)$	0.036	1
D^*_9	$(b, c : 0.7)$	$(a, d : 0.4)(a, b : 0.2)$	$(a : 1.0)$	0.056	1
D^*_{10}	$\langle\rangle$	$(a, d : 0.4)(a, b : 0.2)$	$(a : 1.0)(b, c : 0.3)$	0.024	2
D^*_{11}	$(b, c : 0.7)$	$(a, d : 0.4)$	$(a : 1.0)(a, b : 0.5)$	0.140	1
D^*_{12}	$\langle\rangle$	$(a, d : 0.4)$	$(a : 1.0)(a, b : 0.5)(b, c : 0.3)$	0.060	1

a possible world D^* for the database of Table 3 can be generated by assigning events e_1, e_3 and e_4 to X with probabilities $0.6, 0.3$ and 0.7 respectively, and e_2 to Z with probability 1.0, and $\Pr[D^*] = 0.6 \times 1.0 \times 0.3 \times 0.7 = 0.126$. The complete set of possible worlds for database of Table 3 is in Table 4.

Of course, a source-level uncertain database can be transformed into a collection of p-sequences, and it will be useful to do so. Specifically, for $i = 1, \ldots, m$, let D^p_i be a list of those events in D^p that have non-zero confidence of being associated with source i, ordered by *eid*, together with the associated confidence. However, the resulting p-sequences are *not* independent; as shown in Table 3. Thus, one may view a source-level uncertain database as a collection of p-sequences with dependencies in the form of x-tuples [7].

2.2 Notions of Frequentness

We now use the possible-worlds semantics to give two definitions of frequentness.

Expected Support. As every possible world D^* is a (deterministic) database, $Supp(s, D^*)$ is defined as in Eq. 1. We then define the *expected support* of a sequence s in D^p as follows:

$$ESupp(s, D^p) = \sum_{D^* \in PW(D^p)} \Pr[D^*] * Supp(s, D^*). \tag{2}$$

Table 5. The SPD for the probabilistic DBs of Table 1 (left) and Table 3 (right)

No. of sources	0	1	2	3
support probability	0.142	0.456	0.372	0.029

No. of sources	0	1	2	3
support probability	0.0	0.712	0.288	0.0

We illustrate these concepts by computing the expected support of a sequence $s = (a)(b)$ and the event-level uncertain database of Table 1. Since there are too many possible worlds, we do not use Eq. 2 and compute $ESupp(s)$ as follows. We first compute, for every source, the probability that it supports s. E.g. from Table 2, all worlds in $PW(D^p_Z)$ support s except the first one, so the probability that Z supports s is $(0.35 + 0.15 + 0.15) = 0.65$, and the probability that it does not is 0.35. Similarly, the probabilities that X and Y support s are 0.558 and 0.08. Now for $i = 0, 1, 2, 3$, we use the independence of p-sequences to compute the probability that exactly i sources support s as shown in Table 5(L), e.g. the probability that s is supported by all three sources is $(0.558 \times 0.08 \times 0.65)$ $= 0.029$. Then we get $ESupp(s) = (0 \times 0.142 + \ldots + 3 \times 0.029) = 1.288$.

To compute $ESupp(s)$ for the source-level uncertain case (Table 3), we directly use Eq. 2 together with Table 4 (and get that $ESupp(s) = (1\times0.126+2\times0.054+$ $\ldots+1\times0.060) = 1.288$. Note that the event- and source-level uncertain databases of Tables 1 and 3, which have the same p-sequence form, have the same expected support, even though the possible worlds are very different.

We now formalize the computational task of finding all frequent sequences in a probabilistic database D^p [14]:

Definition 1. *Given a probabilistic database D^p, determine all sequences s such that $ESupp(s, D^p) \geq \theta$, for some user-specified threshold θ.*

Probabilistic Frequent Sequences. A criticism of expected support [5] is that the expectation of a random variable is only one of its measures, and it does not provide confidence bounds that the support of a sequence is high. Following [5], we define the notion of *probabilistic frequent sequences.*

Given a probabilistic database D^p and its set of possible worlds $PW(D^p)$, the *support probability* for a sequence s and support value k is denoted by:

$$\Pr_k(s) = \textstyle\sum_{D^* \in PW(D^p),(Supp(s,D^*)=k)} \Pr(D^*), \tag{3}$$

where $Supp(s, D^*)$ is the support of s in D^*. In other words, $\Pr_k(s)$ is the probability that the support of s is exactly k. Next define the *support probability distribution* (SPD) as the vector $\langle \Pr_0(s), \ldots, \Pr_m(s)\rangle$. The SPDs for the databases of Table 1 and Table 3 are shown in Table 5 and are very different, even though the p-sequences are the same, e.g. for the event-level DB in Table 1, $\Pr_3(s) = 0.029$, but in the source-level DB of Table 3, $\Pr_3(s) = 0$, as no such world exists where all three sources support s (see Table 4). Finally, denote by $\Pr_{\geq\theta}(s) = \sum_{k=\theta}^{m} \Pr_k(s)$ the probability that the support of s is at least θ. We now define:

Definition 2. *Given a probabilistic database D^p and two user-specified thresholds, a* support *threshold θ, $1 \leq \theta \leq m$ and a* confidence *threshold $\tau \in (0, 1]$, find all* probabilistic frequent sequences (PFSes)*, which are sequences s s.t. $\Pr_{\geq\theta}(s) \geq \tau$ (i.e. s is a PFS if it has probability $\geq \tau$ of having support $\geq \theta$).*

Observe that the SPD gives far more detailed information than expected support; from the SPD of a sequence one can easily compute not only the expected support, but also the variance and higher moments.

3 Support Computation

We now discuss the computational complexity of computing frequent sequences based on the definitions in the previous section. As indicated earlier, we focus on the interestingness predicate, which when specialized to our problem and definitions of frequentness, yield the following questions. Given a probabilistic database D^p (either source-level or event-level uncertain):

- For a given sequence s and a threshold θ, is $ESupp(s, D^p) \geq \theta$?
- For a given sequence s and thresholds θ and τ, is $\Pr(Supp(s, D^p) \geq \theta) \geq \tau$?

We assume that the database, whether source-level or event-level uncertain, is given as a list of p-sequences. We denote by m the number of sources (as before), by N_i the number of events in the p-sequence of the i-th source, by $N = \sum_{i=1}^{m} N_i$ the total size of all p-sequences and by k the number of events in s.[2]

3.1 Source Support Probability

The first task is to determine the *source support probability*, namely the probability that a given source σ_i supports a sequence s, or $\Pr(s \preceq D_i^p)$. An important issue is that s may be a subsequence of D_i^p in many different ways. For example, if $s = \langle \underbrace{(a)(a)\ldots(a)}_{k \text{ times}} \rangle$ and $D_i^p = \langle (a : c_1), (a, c_2), \ldots, (a, c_{N_i}) \rangle$, then any subset of k positions from D_i^p could be the (sole) basis for the i-th source to support s. In [14] a dynamic programming recurrence is given that computes this probability in polynomial time:

Theorem 1 ([14]). *Given s and D_i^p, we can calculate $\Pr(s \preceq D_i^p)$ in $O(k \cdot N_i)$ time.*

3.2 Expected Support

For both event-level and source-level uncertainty, it was shown in [14] that:

$$ESupp(s, D^p) = \sum_{i=1}^{m} \Pr(s \preceq D_i^p). \tag{4}$$

[2] We assume that an event consists of at most a constant number of items.

This holds even though p-sequences are not independent in the source-level uncertain case, due to the principle of linearity of expectation. To calculate $ESupp(s, D^p)$ using Eq. 4, we can apply Theorem 1 to each source in turn, which takes $O(k \cdot \sum_{i=1}^{m} N_i) = O(kN)$ time, and obtain:

Theorem 2. *Given s and D^p, we can calculate $ESupp(s, D^p)$, and hence evaluate the interestingness predicate, in $O(kN)$ time, for both event-level and source-level uncertainty.*

3.3 Probabilistic Frequentness

Event-Level Uncertainty. As in [5], we compute the entire support probability distribution (SPD), namely the vector $\langle \Pr_0(s), \ldots, \Pr_m(s) \rangle$, where $\Pr_k(s)$ is the probability that the support of s is exactly k. Given the SPD, we then compute the cumulative probabilities $\Pr_{\geq \theta}(s) = \sum_{k=\theta}^{m} \Pr_k(s)$ in $O(m)$ time and thereby answer the interestingness predicate.

In the case of event-level uncertainty, the p-sequences are independent. This allows the SPD to be calculated as a dynamic programming recurrence as follows. We first compute $\Pr(s \preceq D_i^p)$ for all sources i in $O(kN)$ time as in Thm. 2. Next, we define $\Pr_{i,j}(s)$, for $0 \leq i, j \leq m$, as the probability that exactly i of the first j sources support s. We then use the formula:

$$\Pr_{i,j}(s) = \Pr_{i-1,j-1}(s) \cdot \Pr(s \preceq D_i^p) + \Pr_{i,j-1}(s) \cdot (1 - \Pr(s \preceq D_i^p)), \qquad (5)$$

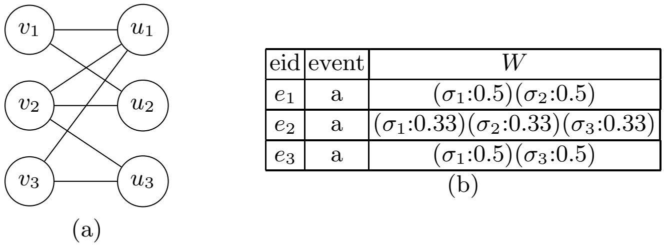
where $\Pr_{0,j}(s) = 1,\ 0 \leq j \leq m$ and $\Pr_{i,j}(s) = 0, \forall\ i > j$, to compute all the values $\Pr_{i,j}$ in $O(m^2)$ time. Since $\Pr_{i,m}(s) = \Pr_i(s)$, we get the full SPD and can use this to determine if s is a PFS; the overall time is $O(kN + m^2)$. In summary:

Theorem 3. *Given s and D^p, if D^p is an event-level uncertain database, we can calculate the support probability distribution, and hence the interestingness predicate in $O(kN + m^2)$ time.*

Remark 1. When computing $\Pr_{i,j}(s)$ using Eq. 5, we consider two cases: either σ_i supports s, and exactly $j-1$ of $\sigma_1, \ldots, \sigma_{i-1}$ support s (the first term) or σ_i does not support s and exactly j of $\sigma_1, \ldots, \sigma_{i-1}$ support s. The correctness of Eq. 5 depends crucially on the fact that we assume independence among p-sequences, so $\Pr(s \preceq D_i^p)$ (resp. $\Pr(s \not\preceq D_i^p)$) is unchanged even when conditioned on knowing that exactly $j-1$ (resp. j) of the first $i-1$ sources support s.

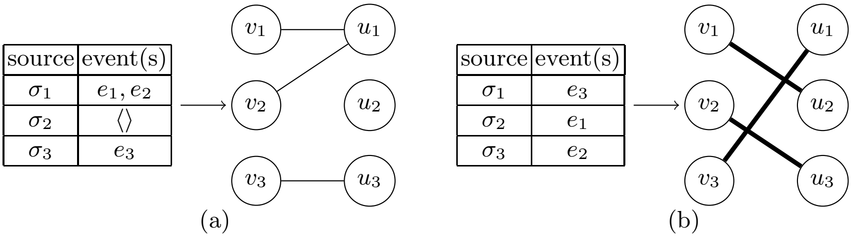
Remark 2. Since N/m is the average length of a p-sequence, $N/m \ll m$ and (since k is not very large as well) the m^2 term will often dominate the kN term. This means the interestingness predicate for probabilistic frequentness will often be computationally more expensive than for expected support.

Source-Level Uncertainty. In source-level uncertainty, we cannot use Eq. 5 to efficiently compute the SPD, since the p-sequences are not independent (see Remark 1). Consider the very simple probabilistic database which consists of just the event $\{a\}$, associated with source σ_1 and σ_2 with probabilities 0.5 each.



eid	event	W
e_1	a	$(\sigma_1{:}0.5)(\sigma_2{:}0.5)$
e_2	a	$(\sigma_1{:}0.33)(\sigma_2{:}0.33)(\sigma_3{:}0.33)$
e_3	a	$(\sigma_1{:}0.5)(\sigma_3{:}0.5)$

Fig. 1. A sample bipartite graph (a) transformed to a probabilistic database (b)



source	event(s)
σ_1	e_1, e_2
σ_2	$\langle\rangle$
σ_3	e_3

source	event(s)
σ_1	e_3
σ_2	e_1
σ_3	e_2

Fig. 2. Two possible worlds in $PW(D^p)$ and their bipartite graph representations. A perfect matching (b), when every vertex in $U \cup V$ is adjacent to a single edge.

There are only two possible worlds, the first with the event $\{a\}$ associated with σ_1 and nothing with σ_2, and the second the other way around. Clearly, if s is the sequence $\langle\{a\}\rangle$, then $\Pr(s \preceq D_1^p) = \Pr(s \preceq D_2^p) = 0.5$. However, applying Eq. 5 gives that $\Pr_{1,1} = 0.5$ (correct) and $\Pr_{2,2} = 0.25$ (incorrect). The probability that both sources support s is zero, not 0.25 – there is no possible world in which both sources support s. To see what goes wrong, consider the computation of $\Pr_{2,2}(s)$ as $\Pr_{1,1}(s) \cdot 0.5 + \Pr_{2,1}(s) \cdot 0.5 = 0.5 \cdot 0.5 + 0 \cdot 0.5 = 0.25$. Looking at the first term, if σ_1 supports s, then conditioned on this knowledge, the probability that σ_2 supports s is zero. Thus, the correct computation for $\Pr_{2,2}(s)$ would be as $0.5 \cdot 0 + 0 \cdot 0.5 = 0$. Unfortunately, the difficulty is not with one particular approach but is intrinsic to the problem: we now show that computing even a single entry of the SPD is provably hard. Define the following problem:

Definition 3 (Exact-k Support problem). *The input is a source-level uncertain probabilistic database D^p, a sequence s and a number k, $0 \leq k \leq m$. The output is $\Pr_k(s)$, i.e. the probability that exactly k sources support s.*

Theorem 4. Exact-k Support *is #P-complete.*

Proof. We reduce the problem of computing the number of perfect matchings in a bipartite graph, a known #P-complete problem [19] to Exact-k Support.

Let $G(U, V, E)$ be an undirected bipartite graph, where U and V are disjoint sets of *vertices* and E is the set of *edges* between them, $E \subseteq U \times V$. We assume that $|U| = |V| = n$. A *perfect matching* M is a subset of edges in E such that each vertex in $U \cup V$ is adjacent to exactly a single edge in M. Given a bipartite graph G, the problem of counting how many perfect matchings there are in G is #P-complete [19].

Given a bipartite graph G, to compute the number of matchings in G, we create an instance of Exact-k Support (a probabilistic database D^p, a sequence s and a number k) in polynomial time such that solving the latter instance gives the number of perfect matchings in G. Given $G = (U, V, E)$ where $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$, we create a set of sources $\mathcal{S} = \{\sigma_1, \ldots, \sigma_n\}$ such that $\sigma_i \in \mathcal{S}$ represents $u_i \in U$. The probabilistic database D^p is a set of records $r_i = (e_i, e, W_i)$, where e_i is a event id, e is an event and W_i is a distribution. The record r_i represents v_i in V together with all the edges from v_i to the *neighborhood* of v_i, i.e. the set of vertices in U adjacent to v_i. In what follows, we denote the neighborhood of v_i as $N(v_i)$. The event contained in *every* record is a set containing just the singleton element $\{a\}$. In the i-th record r_i, the distribution W_i contains only the sources σ_j that represent vertices $u_j \in N(v_i)$. All sources in W_i have the same probability, i.e. $(1/|N(v_i)|)$. An example of such a transformation is shown in Fig. 1. Finally, we choose $k = 0$ and the sequence $s = (a)(a)$, and ask to compute $\Pr_0(s)$, i.e. the probability that no sources support s. This completes creation of the instance of Exact-k Support, and the transformation is clearly polynomial-time. Clearly, every possible world $D^* \in PW(D^p)$ is equally likely, and the probability of a possible world D^* is $\phi = (1/|PW(D^p)|)$, where $|PW(D^p)| = \prod_{i=1}^{n} |N(v_i)|$. For example, there are 12 possible worlds for the database in Fig. 1(b), and the probability of each world is (1/12). In each possible world, each record r_i is associated with some source σ_j, so a possible world can be viewed as a sub-graph of G where every vertex in V has degree exactly one. Two possible worlds and their corresponding graphs are shown in Fig. 2. Those possible worlds where each source is associated with exactly one record corresponds to a perfect matching (Fig. 2(b)); in such possible worlds, the support of the sequence $s = (a)(a)$ will clearly be zero. Thus, we see that $\Pr_0(s) = \phi \cdot (\#\text{ matchings in } G)$, and once we are given $\Pr_0(s)$, we obtain the number of matchings in G by multiplying by the total number of possible worlds. For example, in database in 1(b), there are only three possible worlds where each source is associated with exactly one event, which are $(\sigma_1 : e_1,\ \sigma_2 : e_2,\ \sigma_3 : e_3)$, $(\sigma_1 : e_2,\ \sigma_2 : e_1,\ \sigma_3 : e_3)$ and $(\sigma_1 : e_3,\ \sigma_2 : e_1,\ \sigma_3 : e_2)$. Hence, $Supp(s, D^*) = 0)$ in three worlds and therefore, $Pr_0(s) = 3 \times (1/12) = 0.25$. We multiply the answer by the number of possible worlds, 12, to get 3, the number of perfect matchings in G.

Hence, we have shown that if the value $\Pr_k(s)$ can be computed for $s = (a)(a)$ and $k = 0$ in D^p, we can also find number of perfect matchings in G, thus reducing the problem of counting perfect matchings in a bipartite graph to Exact-k Support, and showing that Exact-k Support is #P-complete. □

4 Conclusions and Future Work

We studied uncertainty models for sequential pattern mining and defined the notions of frequentness under the *expected suppot* and *probabilistic frequentness* measures. We elaborated on these measures from complexity-theoretic viewpoint, and discussed the computational cost of evaluating these measure for our considered models. Thus we were able to show, that whilst dynamic programming could be used to find frequent sequences efficiently, computing probabilistic frequentness for *source-level uncertainty* is #-P complete. An empirical evaluation and comparison of our considered measures in terms of computational cost and quality of the solution should be an interesting direction to explore.

References

1. Aggarwal, C.C. (ed.): Managing and Mining Uncertain Data. Springer, Heidelberg (2009)
2. Aggarwal, C.C., Li, Y., Wang, J., Wang, J.: Frequent pattern mining with uncertain data. In: KDD, pp. 29–38 (2009)
3. Agrawal, R., Srikant, R.: Mining sequential patterns. In: ICDE, pp. 3–14 (1995)
4. Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: KDD, pp. 429–435 (2002)
5. Bernecker, T., Kriegel, H.-P., Renz, M., Verhein, F., Züfle, A.: Probabilistic frequent itemset mining in uncertain databases. In: KDD, pp. 119–128 (2009)
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. Cormode, G., Li, F., Yi, K.: Semantics of ranking queries for probabilistic data and expected ranks. In: ICDE, pp. 305–316 (2009)
8. Dalvi, N.N., Suciu, D.: The dichotomy of conjunctive queries on probabilistic structures. In: PODS, pp. 293–302 (2007)
9. El-Ramly, M., Stroulia, E., Sorenson, P.G.: From run-time behavior to usage scenarios: an interaction-pattern mining approach. In: KDD, pp. 315–324 (2002)
10. Gunopulos, D., Khardon, R., Mannila, H., Saluja, S., Toivonen, H., Sharm, R.S.: Discovering all most specific sentences. ACM ToDS 28(2), 140–174 (2003)
11. Hassanzadeh, O., Miller, R.J.: Creating probabilistic databases from duplicated data. The VLDB Journal 18(5), 1141–1166 (2009)
12. Hua, M., Pei, J., Zhang, W., Lin, X.: Ranking queries on uncertain data: a probabilistic threshold approach. In: SIGMOD Conference, pp. 673–686 (2008)
13. Khoussainova, N., Balazinska, M., Suciu, D.: Probabilistic event extraction from rfid data. In: ICDE, pp. 1480–1482 (2008)
14. Muzammal, M., Raman, R.: Mining sequential patterns from probabilistic databases. Tech. Rep. CS-10-002, Dept. of Computer Science, Univ. of Leicester (2010), http://www.cs.le.ac.uk/people/mm386/pSPM.pdf
15. 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 Trans. Knowl. Data Eng. 16(11), 1424–1440 (2004)
16. 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)
17. Suciu, D., Dalvi, N.N.: Foundations of probabilistic answers to queries. In: SIGMOD Conference, p. 963 (2005)
18. Sun, X., Orlowska, M.E., Li, X.: Introducing uncertainty into pattern discovery in temporal event sequences. In: ICDM, pp. 299–306 (2003)

19. Valiant, L.G.: The complexity of computing the permanent. Theor. Comput. Sci. 8, 189–201 (1979)
20. Wang, K., Xu, Y., Yu, J.X.: Scalable sequential pattern mining for biological sequences. In: CIKM, pp. 178–187 (2004)
21. Wikipedia: http://en.wikipedia.org/wiki/anpr — Wikipedia, the free encyclopedia (2010), http://en.wikipedia.org/wiki/ANPR (Online; accessed April 30, 2010)
22. Yang, J., Wang, W., Yu, P.S., Han, J.: Mining long sequential patterns in a noisy environment. In: SIGMOD Conference, pp. 406–417 (2002)
23. Zaki, M.J.: Spade: An efficient algorithm for mining frequent sequences. Machine Learning 42(1/2), 31–60 (2001)
24. Zhang, Q., Li, F., Yi, K.: Finding frequent items in probabilistic data. In: SIGMOD Conference, pp. 819–832 (2008)

Cube Based Summaries of Large Association Rule Sets

Marie Ndiaye[1,2], Cheikh T. Diop[2], Arnaud Giacometti[1],
Patrick Marcel[1], and Arnaud Soulet[1]

[1] Laboratoire d'Informatique, Université François Rabelais Tours,
Antenne Universitaire de Blois, 3 place Jean Jaurès, 41000 Blois (France)
{marie.ndiaye,arnaud.giacometti,
patick.marcel,arnaud.soulet}@univ-tours.fr

[2] Laboratoire d'Analyse Numérique et d'Informatique,
Université Gaston Berger de Saint-Louis, BP 234 Saint-Louis (Senegal)
cheikh-talibouya.diop@ugb.edu.sn

Abstract. A major problem when dealing with association rules post-processing is the huge amount of extracted rules. Several approaches have been implemented to summarize them. However, the obtained summaries are generally difficult to analyse because they suffer from the lack of navigational tools. In this paper, we propose a novel method for summarizing large sets of association rules. Our approach enables to obtain from a rule set, several summaries called Cube Based Summaries (CBSs). We show that the CBSs can be represented as cubes and we give an overview of OLAP [1] navigational operations that can be used to explore them. Moreover, we define a new quality measure called homogeneity, to evaluate the interestingness of CBSs. Finally, we propose an algorithm that generates a relevant CBS w.r.t. a quality measure, to initialize the exploration. The evaluation of our algorithm on benchmarks proves the effectiveness of our approach.

Keywords: Association rules, summary, cubes.

Classical mining algorithms generally produce a huge number of association rules [1] making it difficult to efficiently analyze the discovered rules. Methods that generate generic bases are then proposed to reduce the number of the mined rules [2,3,4]. Unfortunately, this quantity often remains too important. Several methods of summarization and navigation have been proposed to facilitate the exploration of association rule sets. Summarization is a common method for representing huge amounts of patterns [5,6,7,8,9,10]. However, summaries generated by the existing methods are generally difficult to explore. Indeed, they are usually displayed in the form of pattern lists with no further organization. In addition, they suffer from the lack of navigational tools. For the purpose of exploring association rules, these latter are represented with cubes [11,12,13]. In particular, the approach exposed in [13] treats class association rules [14].

[1] On Line Analytical Processing.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 73–85, 2010.

However, the cubes proposed by those methods don't represent summaries. Indeed, they only depict a portion of the rules which doesn't provide a global and complete view of the whole set of association rules. In this paper, we propose a new approach to explore large sets of association rules.

Our first contribution is the proposal of cube based summaries (CBSs) to summarize large sets of association rules. Theses CBSs can be represented with cubes which enable to depict the rules according to multiple levels of detail and different analytical axes. Our motivation is based on the fact that, by representing rule sets with cubes, we can exploit OLAP navigational operations [15,16] in order to facilitate the analysis of the rules.

Our second contribution is the proposal of a greedy algorithm which generates a relevant CBS not exceeding a user-specified size for a given rule set. Such a CBS can be used to initialize the exploration of the latter. The algorithm is based on a quality measure that evaluates the interestingness of CBSs. We propose a new measure to evaluate the quality of CBSs. It is called homogeneity and is based on the Shannon conditional entropy.

Finally, empirical tests on generic bases show that an interesting CBS can be computed within a reasonable time, even if the size of the initial set of rules is very large. They also show that the quality of a summary generated by our algorithm is close to that of the optimal solution.

The remainder of the paper is organized as follows. Some preliminary definitions and notations are presented in Section 1. In Section 2, we describe CBSs and we give an overview of possible utilizations of OLAP navigational operations. In Section 3, we propose a new quality measure before detailing our summarization algorithm. In Section 4, we present the results of empirical evaluations of our algorithm performed on generic bases of association rules. A state of the art on existing summarization methods is presented in Section 5. We eventually conclude in Section 6 and we expose prospects for future research.

1 Context and Motivation

Summary is practical for representing huge amounts of patterns. It provides a global view of the patterns. It also enables to analyse the extracted patterns in a broader context which highlights relations between them. After recalling the definition of association rule, we propose a summary framework which can be applied not only to association rules, but also to other patterns. Finally, we formulate the problem that we address in the remainder of this paper.

1.1 Association Rules

Let $\mathcal{A}$ be a finite set of attributes such that each attribute $A \in \mathcal{A}$ takes its values in a set $dom(A)$, called the domain of A. In the following, we assume that the domains of the attributes in $\mathcal{A}$ are pairwise disjoint.

Given an attribute $A \in \mathcal{A}$, an *item* defined on A is a value of $dom(A)$.

An *itemset* defined on $\mathcal{A}$ is a set of items $X = \{a_1, ..., a_K\}$ such that $a_k \in dom(A_k)$ for all $k \in \{1, ..., K\}$ and $\{A_1, ..., A_K\} \subseteq \mathcal{A}$. The attribute set $sch(X) = \{A_1, ..., A_K\}$ denotes the schema of X.

Table 1. Association rules

$r_1 : \{auto\} \Rightarrow \{stab\}$	$r_4 : \{stab\} \Rightarrow \{yes\}$	$r_7 : \{yes\} \Rightarrow \{stab\}$
$r_2 : \{auto\} \Rightarrow \{stab, yes\}$	$r_5 : \{stab\} \Rightarrow \{auto\}$	$r_8 : \{yes\} \Rightarrow \{auto, stab\}$
$r_3 : \{auto\} \Rightarrow \{yes\}$	$r_6 : \{stab\} \Rightarrow \{auto, yes\}$	$r_9 : \{yes\} \Rightarrow \{auto\}$

An *association rule* defined on $\mathcal{A}$ is a relation $X \Rightarrow Y$ where X and Y are itemsets defined on $\mathcal{A}$ such that $X \cap Y = \emptyset$. X and Y are the body and the head of the rule, respectively. Thereafter, $\mathcal{R}$ denotes the language of all possible association rules.

Let us consider the attribute set $\mathcal{A} = \{\textsc{control}, \textsc{stability}, \textsc{visibility}\}$ that describes data about spacecraft landing where $dom(\textsc{control}) = \{auto, noauto\}$, $dom(\textsc{stability}) = \{stab, xstab\}$ and $dom(\textsc{visibility}) = \{yes, no\}$. Table 1 shows an excerpt from association rules defined on $\mathcal{A}$. As aforementioned, analysis of rule sets which are often huge in practice requires smaller representations.

1.2 Summary Framework

Even if this paper focuses on summaries of association rules, we present in this section a framework for any language of patterns. Indeed, Definitions 1 and 2 are generalizations of definitions proposed in [6] for itemsets. The notion of summary relies on coverage relation:

Definition 1 (Cover). *Let $(\mathcal{P}, \preceq_{\mathcal{P}})$ and $(\mathcal{S}, \preceq_{\mathcal{S}})$ be two partially ordered pattern languages. A coverage relation over $\mathcal{P} \times \mathcal{S}$, denoted $\lhd$, is a binary relation over $\mathcal{P} \times \mathcal{S}$ such that for all $p \in \mathcal{P}$ and $s \in \mathcal{S}$:*
(i) for all $p' \in \mathcal{P}$, if $p \preceq_{\mathcal{P}} p'$ and $s \lhd p$, then $s \lhd p'$,
(ii) for all $s' \in \mathcal{S}$, if $s' \preceq_{\mathcal{S}} s$ and $s \lhd p$, then $s' \lhd p$.

In our approach, $\preceq_{\mathcal{P}}$ and $\preceq_{\mathcal{S}}$ are specialization relations. The notation $s' \preceq_{\mathcal{S}} s$ means that s' is more general than s and s is more specific than s'. If $\mathcal{P} = \mathcal{S}$ then the specialization becomes a coverage relation. In the following, we consider the specialization relation $\preceq_{\mathcal{R}}$ for association rules described hereafter. Given two association rules $r_1 : X_1 \Rightarrow Y_1$ and $r_2 : X_2 \Rightarrow Y_2$, r_2 is more specific than r_1 ($r_1 \preceq_{\mathcal{R}} r_2$) if $X_1 \subseteq X_2$ and $Y_1 \subseteq Y_2$. For example, $r_2 : \{auto\} \Rightarrow \{stab, yes\}$ is more specific than $r_1 : \{auto\} \Rightarrow \{stab\}$. The specialization relation $\preceq_{\mathcal{R}}$ can also be used as a coverage relation over $\mathcal{R} \times \mathcal{R}$ because the conditions (i) and (ii) of Definition 1 are satisfied. Given a set of patterns $P \subseteq \mathcal{P}$ and a pattern $s \in \mathcal{S}$, $cover(s, P)$ denotes the subset of patterns in P covered by s. Now, let us formalize the notion of summary.

Definition 2 (Summary). *Let $\lhd$ be a coverage relation over $\mathcal{P} \times \mathcal{S}$ and P be a subset of $\mathcal{P}$. A summary of P w.r.t. $\lhd$ is a subset S of $\mathcal{S}$ such that:*
(i) for all pattern p in P, there exists a pattern s of S such that $s \lhd p$,
(ii) each pattern of S covers at least one pattern of P,
(iii) $|S| \leq |P|$.

Subsequently, $\mathcal{S}$ is called a language of summary patterns. Note that most of the existing methods use the same language for P and S [5,6,7,8]. For instance, given the rule set R composed of the rules detailed in Table 1 and the coverage relation $\preceq_{\mathcal{R}}$, the rule set $S = \{r_1 : \{auto\} \Rightarrow \{stab\}, r_{10} : \{\} \Rightarrow \{auto\}, r_{11} : \{\} \Rightarrow \{stab\}, r_{12} : \{\} \Rightarrow \{yes\}\}$ is a summary of R w.r.t. $\preceq_{\mathcal{R}}$. Indeed, conditions (i) and (ii) of Definition 2 are satisfied since $cover(r_1, R) = \{r_1, r_2\}$, $cover(r_{10}, R) = \{r_5, r_6, r_8, r_9\}$, $cover(r_{11}, R) = \{r_1, r_2, r_7, r_8\}$ and $cover(r_{12}, R) = \{r_2, r_3, r_4, r_6\}$. Condition (iii) is also satisfied because S which contains 4 rules is smaller than R with 9 rules. Moreover, if r_1 is removed from S, $S' = \{r_{10}, r_{11}, r_{12}\}$ remains a summary of R because all the rules of R are still covered. But, if one of the other rules is removed, the result is no longer a summary. S' is called a minimal summary of R. This concept is defined below:

Definition 3 (Minimal Summary). *Let $S \subseteq \mathcal{S}$ be a summary of $P \subseteq \mathcal{P}$ w.r.t. $\lhd$. S is minimal if there is no set $S' \subset S$ such that S' is a summary of P.*

1.3 Problem Statement

A pattern set can have several minimal summaries in a given language of summary patterns. For example, the set of rules $\{r_1, r_3, r_4, r_7, r_9\}$ is also a minimal summary of the previous pattern set R. This summary is larger than S given above but it is also relevant because it provides more details about the rules of R. Thus, we would like to navigate between the different summaries of a rule set.

In this paper, we are particularly interested in summaries of association rule sets and we address two problems which are formulated as follows:

1. How should we define a language of summary patterns and a coverage relation that enable to build minimal summaries and to explore effectively large sets of association rules?
2. Which minimal summaries are the most interesting?

Sections 2 and 3 address these problems, respectively.

2 Summarizing Large Sets of Association Rules

To solve the first problem, we propose summaries named cube based summaries (CBSs) that we represent with cubes. Then, we present operations that can be performed on a CBS.

2.1 Cube Based Summary

We introduce in this section CBSs which give smaller representations of large sets of association rules. Fig. 1 depicts a CBS for the set R that contains the association rules presented in Table 1. It is composed of three dimensions which constitute its schema $\langle Body.\textsc{control}, Body.\textsc{visibility}, Head.\textsc{control}\rangle$. Each dimension corresponds to an attribute. The attributes prefixed with *Head* appear in the head of the rules and those with the prefix *Body* appear in their body. On each dimension, values that belong to the domain of the corresponding attribute

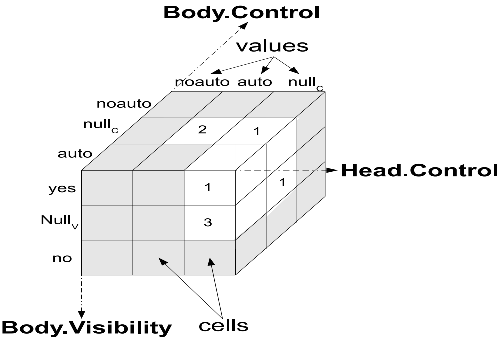


Fig. 1. Representation of a CBS

		Head.CONTROL	
		auto	$null_C$
Body.VISIBILITY	yes	2	1
	$null_V$	2	4

Fig. 2. A roll-up operation

are displayed. The null values $null_C$ and $null_V$ are added on the dimensions to express the absence of value. Each cell of the cube is referenced with a tuple $\langle b_1, b_2, h_1\rangle$ which belongs to $[dom(\text{CONTROL}) \cup \{null_C\}] \times [dom(\text{CONTROL}) \cup \{null_C\}] \times [dom(\text{CONTROL}) \cup \{null_C\}]$. These tuples are called the references defined on the schema $\langle Body.\text{CONTROL}, Body.\text{VISIBILITY}, Head.\text{CONTROL}\rangle$. Thereafter, $\mathcal{S}$ denotes the language of references of all the possible schemas. A reference can cover one or more rules of R and its cell contains the number of rules it covers. The cells filled in grey correspond to the references that cover no rule of R. The values *noauto* of *Body*.CONTROL and *Head*.CONTROL and *no* of *Body*.VISIBILITY can be removed since all the cells associated with them are empty. Our coverage relation is defined over $\mathcal{R} \times \mathcal{S}$. It is based on the specialization relation $\preceq_{\mathcal{R}}$ for association rules (see Section 1) and the specialization relation $\preceq_{\mathcal{S}}$ for references defined hereafter. Given two references $s =< b_1, \ldots, b_I, h_1, \ldots, h_J >$ and $s' =< b'_1, \ldots, b'_K, h'_1, \ldots, h'_L >$, s is more general than s' ($s \preceq_{\mathcal{S}} s'$) if $\{b_1, \ldots, b_I\} \subseteq \{b'_1, \ldots, b'_K\}$ and $\{h_1, \ldots, h_J\} \subseteq \{h'_1, \ldots, h'_L\}$. Considering those specialization relations, we define the following coverage relation that is used in Fig. 1.

Definition 4 (Rule Cover). *Given the language of association rules $\mathcal{R}$ and the language of references $\mathcal{S}$, a reference $s =< b_1, \ldots, b_I, h_1, \ldots, h_J > \in \mathcal{S}$ defined on the schema $\langle Body.B_1, \ldots, Body.B_I, Head.H_1, \ldots, Head.H_J\rangle$ covers a rule $r : X \Rightarrow Y \in \mathcal{R}$ ($s \lhd_{\mathcal{S},\mathcal{R}} r$) if for all $v = b_i$, $i \in \{1, \ldots, I\}$ (resp. $v = h_j$, $j \in \{1, \ldots, J\}$):*

- *when v does not equal to the null value of $Body.B_i$ (resp. $Head.H_j$), X (resp. Y) contains v;*
- *otherwise, X (resp. Y) does not contain an item defined on B_i (resp. H_j).*

For example, the reference $\langle null_C, yes, auto\rangle$ defined on $\langle Body.\text{CONTROL}, Body.\text{VISIBILITY}, Head.\text{CONTROL}\rangle$ covers the rules $r_8 : \{yes\} \Rightarrow \{auto, stab\}$ and $r_9 : \{yes\} \Rightarrow \{auto\}$ because their body and their head contain *yes* and *auto*, respectively. However, $\langle null_C, null_V, auto\rangle$ does not cover $r_9 : \{yes\} \Rightarrow \{auto\}$ because the latter contains in its body the item *yes* which is defined on VISIBILITY.

It can straightforwardly be proved that $\lhd_{\mathcal{S},\mathcal{R}}$ is a coverage relation. Indeed we intuitively observe that every reference that covers a rule $r \in \mathcal{R}$ also covers the specializations of r. And conversely, every rule covered by a reference s is also covered by the generalizations of s. Now, we formalize the notion of CBS.

Definition 5 (Cube Based Summary). *Given a rule set R, the cube based summary of schema $C = \langle Body.B_1, \ldots, Body.B_I, Head.H_1, \ldots, Head.H_J \rangle$ of R, denoted $S_{C,R}$, is the set of all the references defined on C which cover at least one rule of R, w.r.t. $\lhd_{\mathcal{S},\mathcal{R}}$.*

Any CBS $S_{C,R}$ of a rule set R is a summary of R w.r.t. $\lhd_{\mathcal{S},\mathcal{R}}$ because $S_{C,R}$ satisfies the conditions of Definition 2.

The CBS of schema $\langle Body.\text{CONTROL}, Body.\text{VISIBILITY}, Head.\text{CONTROL} \rangle$ depicted in Fig. 1 is composed of the references: $\langle auto, null_V, null_C \rangle$, $\langle null_C, yes, auto \rangle$, $\langle null_C, yes, null_C \rangle$, $\langle null_C, null_V, auto \rangle$ and $\langle null_C, null_V, null_C \rangle$. Note that if at least one of the references detailed above is removed from the CBS, the rule set R is no more covered entirely. The property below generalizes this observation:

Property 1 (Minimality). If $S_{C,R}$ is a CBS of a rule set R, then it is a minimal summary of R w.r.t. $\lhd_{\mathcal{S},\mathcal{R}}$.

Property 1 enables us to ensure that a CBS is not only a summary, but it is also minimal. Therefore, the amount of information presented to the user is reduced as much as possible. In order to respect the requirements regarding the number of pages, the proof of the above property is not exposed.

After proposing the language of references $\mathcal{S}$ and the coverage relation $\lhd_{\mathcal{S},\mathcal{R}}$ that we have used to define CBSs, we show in the next section how CBSs can be explored with OLAP navigational operations.

2.2 Navigational Operations

Classical OLAP navigational operations can be used to explore CBSs, particularly granularity operations, i.e. roll-up and drill-down, which enable to modify the granularity level of the represented data. Roll-up consists in moving from a detailed to a more aggregated level. In our context, it corresponds to the removal of attributes from the schema of a CBS. Drill-down is the reverse of roll-up, it consists in adding attributes to the schema. Fig. 2 shows a result of roll-up performed on the CBS represented in Fig. 1 by removing $Body.\text{CONTROL}$.

Given two CBSs $S_{C_1,R}$ and $S_{C_2,R}$ of a rule set R, if $S_{C_1,R}$ can be obtained by performing a sequence of roll-up from $S_{C_2,R}$, then $S_{C_1,R}$ is more general than $S_{C_2,R}$ and $S_{C_2,R}$ is more specific than $S_{C_1,R}$. The CBS whose schema is empty is the most general and CBS with the schema $\langle Body.B_1, ..., Body.B_I, Head.H_1, ..., Head.H_J \rangle$ where $\{B_1, ..., B_I\} = \{H_1, ..., H_J\} = \mathcal{A}$ is the most specific. Furthermore, the set of the possible CBSs is closed under the granularity operations. Property 2 shows that any CBS can be reached with those operations.

Property 2 (Reachability). Given two distinct CBSs $S_{C_1,R}$ and $S_{C_2,R}$ of a rule set R, there exists a finite sequence of granularity operations $\langle O_1, ..., O_N \rangle$, i.e. roll-up and drill-down operations, such that $S_{C_2,R} = O_1 \circ .... \circ O_N(S_{C_1,R})$.

In conclusion, OLAP navigational operations are well-adapted to explore CBSs. In the next section, we show that some summaries are more suitable than others to perform an effective analysis.

3 Generating an Interesting Cube Based Summary

In this section, we initially focus on the definition of a specific quality measure to evaluate the interestingness of the CBS. Then, we propose an algorithm which finds an approximate solution of the most interesting CBS w.r.t. a quality measure, in order to initialize the exploration of association rules.

3.1 Quality Measure

Given several possible CBSs, the user can hardly identify the most relevant one. Therefore, he needs a measure to evaluate the quality of the CBSs. A quality measure of a summary is a function ϕ that associates with every pair composed of a rule set and a summary, a value in $\mathbb{R}$. In our approach, a CBS is more interesting than its generalizations since it provides more precision about the rules of the set it summarizes. Thus, for assessing effectively the interestingness of CBSs, a quality measure must satisfy the following property:

Property 3 (Monotony). Let ϕ be a quality measure. ϕ is monotone if, for any CBSs $S_{C_1,R}$ and $S_{C_2,R}$ of a ruleset R, if $S_{C_1,R}$ is more specific than $S_{C_2,R}$ then $\phi(R, S_{C_1,R}) \geq \phi(R, S_{C_2,R})$.

Intuitively, if ϕ is monotone then the more specific the CBS is, the higher the measure ϕ. Thus, it is easy to see that the quality of the most general CBS is the smallest. Conversely, the quality of the most specific one is the highest.

Now, we propose a measure called homogeneity that can be used to evaluate the quality of CBSs. Intuitively, the homogeneity reflects the similarity of the rules covered by the same reference. The rules are similar if they have roughly the same items in their head and their body.

Given a ruleset R whose rules are defined on $\mathcal{A}$, let s_j be a reference of a CBS $S_{C,R}$. The similarity between the rules covered by s_j is evaluated by $\sum_{a_i \in \mathcal{I}} p(i,j) \ln[p_j(i)]$ where $\mathcal{I}$ is the set of all the items defined on an attribute of $\mathcal{A}$, $p(i,j) = |\{X \Rightarrow Y \mid (X \Rightarrow Y \in cov(s_j, R)) \wedge (a_i \in X \cup Y)\}|/|R|$ is the joint probability that a rule of R contains a_i and is covered by s_j and $p_j(i) = |\{X \Rightarrow Y \mid (X \Rightarrow Y \in cov(s_j, R)) \wedge (a_i \in X \cup Y)\}|/|cov(s_j, R)|$ is the conditional probability that a rule covered by s_j contains a_i. The homogeneity of $S_{C,R}$ w.r.t. R is given by (1).

$$H(R, S_{C,R}) = \frac{1}{|\mathcal{A}|} \sum_{s_j \in S_{C,R}} \left[\sum_{a_i \in \mathcal{I}} p(i,j) \ln[p_j(i)] \right] \tag{1}$$

$H(R, S_{C,R})$ is a Shannon conditional entropy weighted with the size of $\mathcal{A}$. More details on conditional entropy properties can be found in [17]. Furthermore, the homogeneity satisfies Property 3. Its value is negative or null and it corresponds to 0 for the most specific CBS.

For example, let us consider the item $a_i = yes$ and the reference $s_j = \langle null_C, yes, auto\rangle$ of the CBS $S_{C_1,R}$ that summarizes the rule set of Fig. 1 where $C_1 = \langle \mathit{Body}.\textsc{control}, \mathit{Body}.\textsc{visibility}, \mathit{Head}.\textsc{control}\rangle$. Knowing that $cov(s_j, R) = \{r_8, r_9\}$ with $r_8 : \{yes\} \Rightarrow \{auto, stab\}$ and $r_9 : \{yes\} \Rightarrow \{auto\}$, we have $p(i,j) = \frac{|\{r_8,r_9\}|}{|R|} = \frac{2}{9}$ and $p_j(i) = \frac{|\{r_8,r_9\}|}{|\{r_8,r_9\}|} = \frac{2}{2}$. When we measure the homogeneity of the CBSs $S_{C_1,R}$ and $S_{C_2,R}$ represented in Fig. 1 and Fig. 2, respectively, we obtain $H(R, S_{C_1,R}) = -0,24$ and $H(R, S_{C_2,R}) = -0,3$. We observe that Property 3 is satisfied since $S_{C_1,R}$ is more specific than $S_{C_2,R}$ and $H(R, S_{C_1,R}) > H(R, S_{C_2,R})$.

3.2 Algorithm for Finding an Interesting Cube Based Summary

This section aims at initializing navigation by automatically searching in the space of the possible CBSs, an interesting one from which the user can begin the exploration. The problem can be formulated as follows: given a set of rules R extracted from a dataset of schema $\mathcal{A}$ and a quality measure ϕ, find the most interesting CBS $S_{C^*,R}$ (w.r.t. ϕ) whose size is smaller than a fixed threshold N. This problem is NP-complete. Its NP-completeness can be shown by using the knapsack problem. Hence, we propose an approximate solution computed with a greedy algorithm. We begin from the most general CBS $S_{C_0,R}$ where $C_0 = \langle\rangle$ and we use granularity navigational operations to explore the space of the possible CBSs. Recall that this space is closed under the granularity operations. More precisely, we use the two following drill-down operations:

Algorithm 1. greedy_CBS

```
Input: $R$ {Set of rules}, $\mathcal{A}$ {The set of attributes}
   and $N$ {The maximal size of the summary}
Output: {A cube based summary}
$C_0 = \langle\rangle$, $i = 0$ {Initializations}
repeat
  $i = i + 1$
  $C_i = C_{i-1}$
  for all $\mathcal{O} \in \{AddToHead, AddToBody\}$ do
    for all $A \in \mathcal{A}$ do
      $C = \mathcal{O}(S_{C_{i-1},R}, A)$
      if $\phi(R, S_{C,R}) > \phi(R, S_{C_i,R})$ and $|C_i| \leq N$ then
        $C_i = C$
      end if
    end for
  end for
until $\phi(R, S_{C_{i-1},R}) = \phi(R, S_{C_i,R})$
return  $S_{C_i,R}$
```

AddToHead and *AddToBody*. Given an attribute A, $AddToHead(S_{C,R}, A)$ and $AddToBody(S_{C,R}, A)$ consist in adding $Head.A$ and $Body.A$ in C, respectively. The CBSs whose schema result from either *AddToHead* or *AddToBody* operations are more specific than $S_{C,R}$. Therefore, their quality is better than H(R, $S_{C,R}$). In the worst case, they have the same quality as $S_{C,R}$.

Algorithm 1 seeks an approached solution of $S_{C^*,R}$. It consists in choosing at each step i an attribute from $\mathcal{A}$ which will be added to $S_{C_{i-1},R}$ to obtain $S_{C_i,R}$. The attribute retained at step i is selected such that the CBS $S_{C_i,R}$ has the greatest quality compared to the summaries obtained by adding one of the other attributes of $\mathcal{A}$ (line 7 to 9). The addition of attributes stops when the quality of the CBS obtained in the previous step cannot be improved.

4 Experimental Analysis

In this section, we study the performance of our proposed approach on benchmarks. We use discretized datasets[2] resulting from the UCI Machine Learning repository[3]. We perform our experimentations on generic bases of association rules extracted with the CHARM[4] [18] algorithm. Our summarization algorithm is implemented in Java. All the experiments are executed on a computer Intel dual core 2GHz with 2GB-memory and running Windows Vista. We evaluate the summarization performance in terms of computation time and homogeneity.

4.1 Runtime Performance

We conducted our first experiment using the data sets shown in the table in Fig. 3. For each data set, we generate several rule sets by varying the support level (*minsup*) where the confidence level (*minconf*) is fixed to 50%.

Fig. 3a reports the computation time of the CBSs generated with the maximal size $N = 50$ w.r.t. the number of rules contained in the generic bases. We first notice that the runtime does not exceed 120 seconds even if the rule set contains more than $30,000$ rules. Moreover, the slope of the curves depends on the number of attributes in $\mathcal{A}$. The larger the number of attributes, the higher the slope of the curve. For instance, the curve of *mushroom* has the greatest slope because this dataset contains more attributes (23 attributes) than the others.

Fig. 3b plots the computation time of CBSs when their maximal size N varies between 10 and 100. For this experiment, the rule sets are generated with $minconf = 50\%$ and $minsup = 25\%$ for *mushroom* and *zoo* and $minconf = 50\%$ and $minsup = 15\%$ for *australian* and *vehicle*. We observe that for all the datasets, the computation time sublinearly increases with N. Indeed, in the algorithm implementation, each attribute added by using an operation is not tested again with this operation.

[2] users.info.unicaen.fr/~frioult/uci

[3] mlearn.ics.uci.edu/MLRepository.html

[4] www.cs.rpi.edu/~zaki/software/

			$minsup$										
		$\|\mathcal{A}\|$	5%	8%	10%	15%	20%	25%	30%	35%	40%	45%	50%
Dataset	mushroom	23	-	38406	29441	11629	6681	2915	1732	838	390	227	110
	vehicle	19		31873	13890	3899	1066	339	52	4	0	0	0
	australian	15	39060	-	7573	2437	1019	486	247	124	62	23	9
	zoo	17	31053	-	20583	13253	8446	5382	3283	1864	957	569	300

Number of rules generated from datasets according to $minsup$ where $minconf = 50\%$

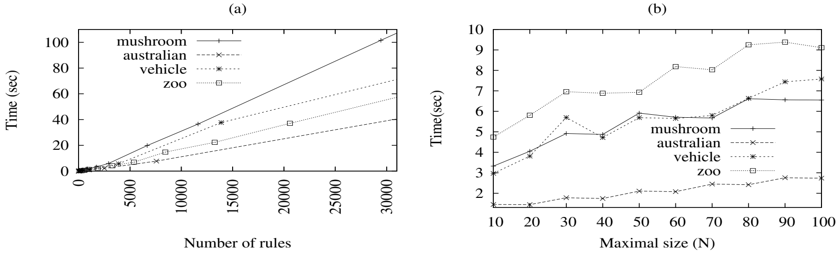


Fig. 3. Execution time

4.2 Quality of the Approximate Solution

This section aims at evaluating the quality of our approximate solution. Thereby, given a rule set R and a maximal size N, we compare the homogeneity of CBSs resulting from 3 approaches:

- **Greedy solution (`greedy_CBS`):** Our algorithm produces an approximate solution.
- **Optimal solution (`exact`):** A naive algorithm enumerates all the CBSs in order to return the optimal solution $S_{C^*,R}$.
- **Average solution (`average`):** A naive algorithm enumerates all the summaries in order to compute the average homogeneity (and the standard deviation) of the most specific summaries (not exceeding N references).

Of course, the algorithm that computes the optimal solution is very costly and fails on datasets containing a large number of attributes. For this purpose, we perform experiments on the sets of association rules mined stemming from small datasets: *cmc*, *glass* and *tic-tac-toe*. The table in Fig. 4 details the number of rules for each generic base according to the dataset, the minimal support and confidence thresholds.

Fig. 4 plots the homogeneity of the three approaches detailed above (i.e. `greedy_CBS`, `exact` and `average`) for each generic base w.r.t. the maximal size of the CBSs. Let us note that we also report the confidence interval of the `average` curve. As expected, we observe that homogeneity logarithmically increases with the maximal size N for the three approaches. This is because the more N increases, the more the generated CBS is specific. Furthermore, we observe that the homogeneity of our summaries is close to that of the optimal summaries for all the experiments. Even if the threshold N increases, the distance between the approximate solution and the optimal one remains moderate. In Fig. 4a,

	Dataset		
	cmc	glass	tic-tac-toe
$\lvert\mathcal{A}\rvert$	10	10	10
minsup	8%	6%	5%
minconf	80%	50%	50%
Number of rules	1116	1210	1299

Data description

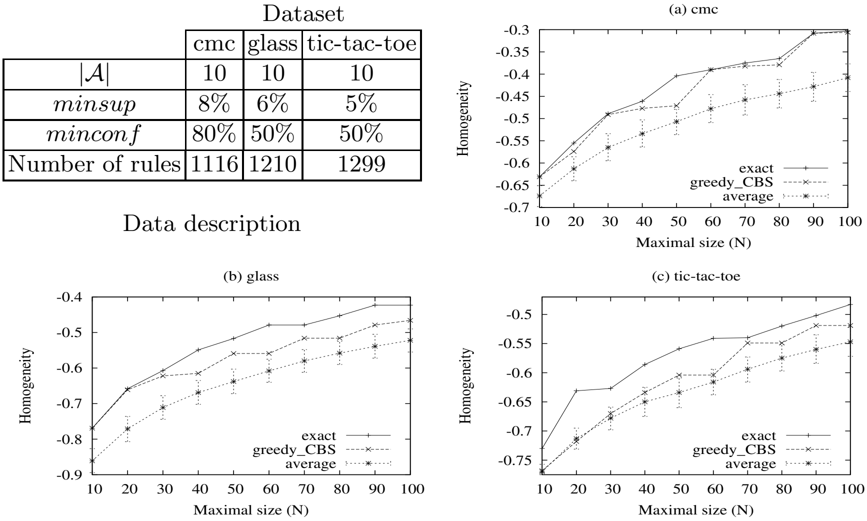


Fig. 4. Homogeneity of CBSs

we notice that for several values of N (e.g., 60 or 90), the homogeneity of our summaries is even optimal. We also notice that the homogeneity of summaries generated by `greedy_CBS` is always greater than that of the summaries generated by `average`. More interestingly, our algorithm finds really relevant CBSs (w.r.t. the homogeneity) because the homogeneity of `greedy_CBS` is often above the confidence interval.

5 Related Work

In the past few years, summarization of pattern sets has been addressed following several approaches. Table 2 highlights some characteristics of the approaches closest to ours [10,6,5,8,7,9] . We comment the table below.

Language of patterns and language of summary patterns: The considered patterns in practically all the cited papers are itemsets, except in [9] where the authors address the problem of summarizing association rules. Furthermore, the patterns of the summaries generally belong to the same language as those of the initial set. However, in [10] authors use pattern profiles which describe the itemsets and approximate their support. In [9], the patterns of the summaries are class association rules [14], i.e. association rules whose head are restricted to a class item.

Coverage: The methods proposed in [10,6,7,9] produce summaries that cover the entire set, i.e. each pattern is covered by at least a pattern of the summary, contrary to other methods which provide summaries that approximately cover the set of association rules [5,8]. The Summaries of the former methods are summaries in the sense of definition 2.

Table 2. Methods for summarizing sets of patterns

Reference	[9]	[6]	[5]	[10]	[7]	[8]	Our method
$\mathcal{P}$	rules	itemsets	itemsets	itemsets	itemsets	itemsets	rules
$\mathcal{S}$	class rules	itemsets	itemsets	profiles	itemsets	itemsets	references
Regeneration	no	no	patterns	frequency	frequency	frequency	no
Coverage	entirely	entirely	partially	entirely	entirely	partially	entirely
Measure ($\star$)	no	CG, IL	NPC	RE	RE	IL	H
Representation	no	no	no	no	no	no	yes
Navigation	no	no	no	no	no	no	yes

$\mathcal{P}$: Language of patterns $\mathcal{S}$: Language of summary patterns
($\star$) CG: compaction gain, RE: restoration error, IL: information loss, H: homogeneity
NPC: number of patterns covered

Regeneration: The main objective of some approaches is to build summaries so that they can be used to regenerate the original itemsets [5] or their frequency [8,7,10] whereas the others are interested in the representation of the patterns for future exploration [6,9]. Our approach is in the latter category.

Measure: Measures of restoration error [10,7] and information loss [8] are used to evaluate the error generated when the frequencies of the patterns can be approximated from a summary. The measure used in [5] calculates the size of the subset of patterns which are really covered by at least one pattern of the summary. In [6], the authors apply a measure called compaction gain to evaluate the compression ratio of a pattern set w.r.t. its summary. They also use an information loss measure to assess the quantity of information contained in the patterns and absent from the patterns of the summary which cover them. The measures which evaluate the regeneration error and the quantity of covered patterns are not adapted to our approach. Indeed, the CBSs are not intended to regenerate information and they always cover the entire pattern set.

Representation and navigation: Our approach enables to explore summaries represented with cubes by using OLAP operations. Contrary to us, the approaches cited above don't propose a representation for the summaries they build. Furthermore, those approaches suffer from the lack of exploration techniques.

6 Conclusion

We have proposed in this paper a new framework to summarize large sets of association rules. Our summaries, called Cube Based Summaries (CBSs) can be represented with cubes and explored using OLAP navigational operations. An algorithm is presented to generate an interesting summary w.r.t. a quality measure of summary in order to initialize the rules exploration. Finally, our experiments prove the feasibility of our approach.

We project to propose a generalization of our approach by using our summary framework, in order to summarize other types of patterns. Furthermore, the space of CBSs is not closed under the OLAP selection operations because they produce portions of summaries. We aim to define a selection operation that provides CBSs like granularity operations.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: SIGMOD, pp. 207–216. ACM, New York (1993)
2. Liu, B., Hsu, W., Ma, Y.: Pruning and summarizing the discovered associations. In: KDD 1999, pp. 125–134 (1999)
3. Srikant, R., Vu, Q., Agrawal, R.: Mining association rules with item constraints. In: KDD 1997, pp. 67–73 (1997)
4. Zaki, M.J.: Generating non-redundant association rules. In: KDD 2000, pp. 34–43 (2000)
5. Afrati, F., Gionis, A., Mannila, H.: Approximating a collection of frequent sets. In: KDD 2004, pp. 12–19 (2004)
6. Chandola, V., Kumar, V.: Summarization — compressing data into an informative representation. In: ICDM 2005, pp. 98–105 (2005)
7. Jin, R., Abu-Ata, M., Xiang, Y., Ruan, N.: Effective and efficient itemset pattern summarization: regression-based approaches. In: KDD 2008, pp. 399–407 (2008)
8. Mielikäinen, T., Mannila, H.: The pattern ordering problem. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) PKDD 2003. LNCS (LNAI), vol. 2838, pp. 327–338. Springer, Heidelberg (2003)
9. Ordonez, C., Ezquerra, N., Santana, C.: Constraining and summarizing association rules in medical data. Knowledge and Information Systems 9, 259–283 (2006)
10. Yan, X., Cheng, H., Han, J., Xin, D.: Summarizing itemset patterns: a profile-based approach. In: KDD 2005, pp. 314–323 (2005)
11. Blumenstock, A., Schweiggert, F., Müller, M., Lanquillon, C.: Rule cubes for causal investigations. Knowledge and Information Systems 18, 109–132 (2009)
12. Boulicaut, J.F., Marcel, P., Rigotti, C.: Query driven knowledge discovery in multidimensional data. In: DOLAP 1999, pp. 87–93 (1999)
13. Liu, B., Zhao, K., Benkler, J., Xiao, W.: Rule interestingness analysis using olap operations. In: KDD 2006, pp. 297–306 (2006)
14. Hu, K., Lu, Y., Zhou, L., Shi, C.: Integrating classification and association rule mining: A concept lattice framework. In: Zhong, N., Skowron, A., Ohsuga, S. (eds.) RSFDGrC 1999. LNCS (LNAI), vol. 1711, pp. 443–447. Springer, Heidelberg (1999)
15. Chaudhuri, S., Dayal, U.: An overview of data warehousing and olap technology. SIGMOD Rec. 26(1), 65–74 (1997)
16. Romero, O., Abelló, A.: On the need of a reference algebra for olap. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2007. LNCS, vol. 4654, pp. 99–110. Springer, Heidelberg (2007)
17. Shannon, C.E.: A mathematical theory of communication. SIGMOBILE Mob. Comput. Commun. Rev. 5(1), 3–55 (2001)
18. Zaki, M.J., Hsiao, C.J.: Charm: An efficient algorithm for closed itemset mining. In: SDM, pp. 457–473 (2002)

A Perceptron-Like Linear Supervised Algorithm for Text Classification

Anestis Gkanogiannis and Theodore Kalamboukis

Department of Informatics
Athens University of Economics and Business, Athens, Greece
{utumno,tzk}@aueb.gr

Abstract. A fast and accurate linear supervised algorithm is presented which compares favorably to other state of the art algorithms over several real data collections on the problem of text categorization. Although it has been already presented in [6], no proof of its convergence is given. From the geometric intuition of the algorithm it is evident that it is not a Perceptron or a gradient descent algorithm thus an algebraic proof of its convergence is provided in the case of linearly separable classes. Additionally we present experimental results on many standard text classification datasets and artificially generated linearly separable datasets. The proposed algorithm is very simple to use and easy to implement and it can be used in any domain without any modification on the data or parameter estimation.

Keywords: Perceprton, Machine Learning, Classification.

1 Introduction

The increasing availability of text in digital form demands fast and accurate methods for classifying it. Classification is the process of assigning elements into a set of predefined classes. Over the last years the problem has found several important applications. For example, in text classification it is used for text filtering [21,22,5,3], in social networks to automatically apply tags (annotation) to new posts for spamming detection, in medicine and many others.

Binary classification is a special case, where instances are assigned into two classes namely, the positive class and its complement or negative class. A multi-labeled classification problem can always be transformed to a set of binary classification problems. To solve these binary classification problems, several Machine Learning techniques and algorithms have been developed. For example, there are Naive Bayesian techniques, Decision Trees, Neural Networks, Linear Classifiers [15], Instance Based algorithms, Support Vector Machines [2], etc. These methods train a model by using a training set and then use this model to classify new unseen instances (Supervised Learning techniques).

Linear classifiers is a family of classifiers whose trained model is a linear combination of features. In another perspective linear classifiers train a model which defines a hyperplane in a high dimensional feature space. The goal of a linear

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 86–97, 2010.

classifier is to find a perfect hyperplane that splits the space into two subspaces: one that contains all the positive examples and another that contains the negative ones. There are a lot perfect hyperplanes for a given binary classification problem.

Perceptron is a flavor of Linear Classifiers. It starts with an initial model and iteratively refines this model using the classifications errors during training. It is the elementary particle of neural networks and it has been investigated and studied since the 1950s [19]. It has been shown that when trained on a linearly separable set of instances, it converges to a separating hyperplane in a finite number of steps [17]. The convergence rate depends on the geometric characteristics of the instances on their feature space.

In this paper we briefly present a method, which has been already presented by [6] on the dataset of the ECML/PKDD 2009 Discovery Challenge. Although at a first glance looks very similar to Perceptron method, it uses a similar main iterative form of model refinement but exploits the geometric characteristics of the train instances in order to find faster a separating hyperplane. In the case of a non separable set it acts in a best-effort manner, preserving the best hyperplane, after applying a max predefined number of steps. In this paper the proposed algorithm is compared with the original perceptron, on various datasets used primarily in the text classification field and also on several artificially generated linearly separable datasets. Experimental results show that the algorithm converges much faster than perceptron. As far as the performance is concerned there are no major differences in the case of the artificial data, since both methods eventually find a separating hyperplane. However there are significant differences in favor of the proposed algorithm in the case of real text datasets (these datasets in general are not linearly separable). Finally the method is compared to the state-of-the-art method in the Machine Learning Classification field, the SVM method [9,10], and the results show that it behaves comparably if not better.

The next Section 2 describes briefly the Linear Classifiers and Perceptron. In Section 3 we briefly present the proposed algorithm and a proof of its convergence. In Section 4 we describe the datasets used for comparison and evaluation and present the experimental results. Finally Section 5 concludes with our findings and future extensions.

2 Linear Classifiers

Assuming that the feature space is of n dimensions, an instance x_i will be represented as a n dimensional vector

$$\overrightarrow{x}_i = (w_{i1}, w_{i2}, \cdots, w_{in}) \ , \tag{1}$$

where w_{ik} denotes the weight of the k^{th} feature in the TF*IDF weighting scheme [20]. Each instance in the training set is accompanied with the class label, a value y_i defined by:

$$y_i = \begin{cases} 1 & \text{if } x_i \in C_+ \\ -1 & \text{if } x_i \in C_- \end{cases} \ , \tag{2}$$

were C_+ denotes the positive class and C_- the negative one. The training set Tr would be a set of tuples

$$Tr = \{(\overrightarrow{x}_1, y_1), (\overrightarrow{x}_2, y_2), \cdots, (\overrightarrow{x}_m, y_m)\} \ . \tag{3}$$

A linear classifier then is defined by a model $\left\langle \overrightarrow{W}, b \right\rangle$ where $\overrightarrow{W}$ is the weight vector in the n-dimensional space and b is a scalar bias value. This model defines a hyperplane h

$$h : \overrightarrow{W} \cdot x + b = 0 \ , \tag{4}$$

and a decision for an instance is taken by function

$$f(\overrightarrow{x}) = \begin{cases} 1 & \text{if } \overrightarrow{W} \cdot \overrightarrow{x} + b > 0 \\ -1 & \quad \text{otherwise} \end{cases} \ . \tag{5}$$

The distinction of the different linear classifiers in the literature lie on the approach of defining the weight vector $\overrightarrow{W}$ and the bias b. Thus there are several algorithms ranging from very simple ones, like the centroid classifier, to more sophisticated ones, like the Perceptron, winnow, Widrow-Hoff, algorithms that try to minimize a cost function, like steepest descent algorithm, to the state of the art algorithm of SVMs that constructs the best separating hyperplane by solving a quadratic optimization problem.

2.1 Centroid Classifier

A Centroid classifier is a simple linear classifier, that will help us to understand the notion behind the proposed algorithm presented in the next Section. We first construct the centroid vectors of the two classes, C_+ and C_-,

$$\overrightarrow{C}_+ = \frac{1}{|C_+|} \sum_{\overrightarrow{x_i} \in C_+} \overrightarrow{x_i} \quad , \quad \overrightarrow{C}_- = \frac{1}{|C_-|} \sum_{\overrightarrow{x_i} \in C_-} \overrightarrow{x_i} \ , \tag{6}$$

and then the classifier is defined by vector $\overrightarrow{W}$:

$$\overrightarrow{W} = \overrightarrow{C}_+ - \overrightarrow{C}_- \ , \tag{7}$$

which is the normal vector of the separating hyperplane, and a bias value b which defines its displacement from the origin.

Figure 1 illustrates a simple case of a centroid classifier in a 2-dimensional space. We note that in this simple example, it is possible to find a value for the bias b such that $< \overrightarrow{W}, b >$ is a perfect separating hyperplane. The bias takes a value

$$b_i = \overrightarrow{W} \cdot \overrightarrow{x}_i, \forall \overrightarrow{x}_i \in Tr \ , \tag{8}$$

which maximizes the performance measure F_1 of the classifier $< \overrightarrow{W}, b_i >$. This value is referred in the literature as the Scut [23] value. In Figure 1 this instance is marked by the point $\overrightarrow{x}_{Scut}$.

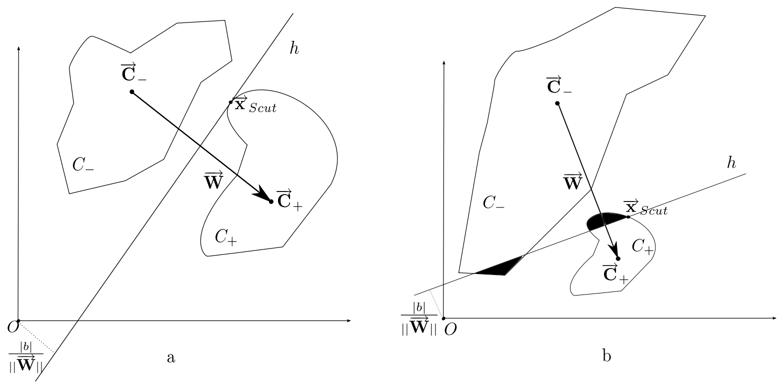

h
$\overrightarrow{\mathbf{C}}_-$
$\overrightarrow{\mathbf{x}}_{Scut}$
$\overrightarrow{\mathbf{W}}$
$\overrightarrow{\mathbf{C}}_+$
C_-
C_+
O
$\frac{|b|}{||\overrightarrow{\mathbf{W}}||}$
a
h
$\overrightarrow{\mathbf{C}}_-$
$\overrightarrow{\mathbf{W}}$
C_-
$\overrightarrow{\mathbf{x}}_{Scut}$
C_+
$\overrightarrow{\mathbf{C}}_+$
$\frac{|b|}{||\overrightarrow{\mathbf{W}}||}$
O
b

Fig. 1. A simple Centroid Classifier in 2 dimensions for two linearly separable datasets

This simple algorithm has been investigated and several methods have been proposed for altering the initial centroids in order to achieve a better classifier [12,7,1].

2.2 Batch Perceptron

Perceptron is a simple algorithm that has been extensively used in the literature. The algorithm appears in two modes: as a single-sample (online algorithm) and as batch mode with a fixed or a variable learning rate. In this section we briefly describe the batch version with variable learning rate that resembles a similarity with the proposed algorithm.

The algorithm starts with an arbitrary vector $\overrightarrow{W}_0$ which is updated at each iteration step based on the misclassified examples. If we define at iteration step k the set:

$$Err_k = \{(\overrightarrow{x}_i, y_i)\}, f_k\left(\overrightarrow{x}_i\right) \neq y_i\} \ , \tag{9}$$

of the examples misclassified by the classifier, then $\overrightarrow{W}_k$ is updated as:

$$\overrightarrow{W}_{k+1} = \overrightarrow{W}_k + \alpha_k \sum_{\left(\overrightarrow{x}_i, y_i\right) \in Err_k} y_i \overrightarrow{x}_i \ . \tag{10}$$

Similarly the bias is updated by the relation:

$$b_{k+1} = b_k + \alpha_k \sum_{\left(\overrightarrow{x}_i, y_i\right) \in Err_k} y_i \ . \tag{11}$$

The learning rate denoted with α_k should form a decreasing sequence in order for the method to converge [4]. Such a value is for example $\alpha_k = c/k$, where c is a constant.

3 The Proposed Algorithm

Figure 1a shows an ideal case where the centroid classifier defines a perfect separating hyperplane. This is not however in general the case as it is shown

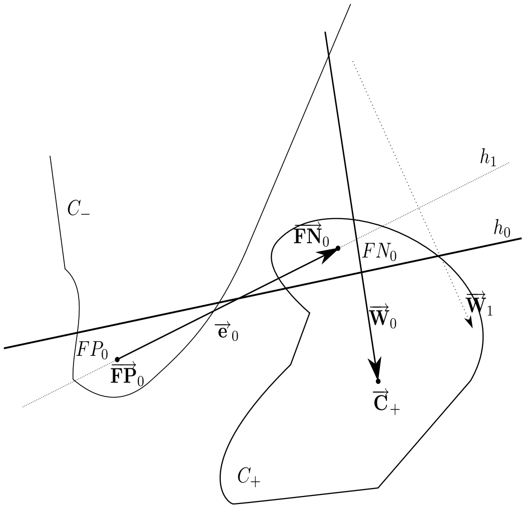

h_1
C_-
h_0
$\overrightarrow{\mathbf{FN}}_0$
FN_0
$\overrightarrow{\mathbf{W}}_1$
$\overrightarrow{\mathbf{W}}_0$
$\overrightarrow{\mathbf{e}}_0$
FP_0
$\overrightarrow{\mathbf{FP}}_0$
$\overrightarrow{\mathbf{C}}_+$
C_+

Fig. 2. The new hyperplane passes through the centroids of the misclassified sets, FN and FP

in Figure 1b. In such a case a Perceptron type algorithm may use the centroid classifier as a starting vector to find a separating hyperplane following the process described in the previous section.

A weak point of Perceptron algorithm is determining the value of the learning rate. A too large choice of α_k causes the hyperplane to swing wildly moving away the optimum, while a too small value slows down the convergence rate which makes the algorithm not scalable to large problems.

The proposed algorithm starts with an initial weight vector and bias as defined by the centroid classifier, relations 7 and 8. Then the algorithm proceeds by iteratively modifying the weight vector $\overrightarrow{W}_k$ by an error vector

$$\overrightarrow{e}_k = \overrightarrow{FN}_k - \overrightarrow{FP}_k \ , \tag{12}$$

defined by the difference of the centroid vectors of the misclassified regions FP (False Positive examples) and FN (False Negative) (see figure 2). We define these misclassified centroids at each iteration as

$$\overrightarrow{FP}_k = \frac{1}{|FP_k|} \sum_{\overrightarrow{x_i} \in FP_k} \overrightarrow{x_i} \ , \tag{13}$$

$$\overrightarrow{FN}_k = \frac{1}{|FN_k|} \sum_{\overrightarrow{x_i} \in FN_k} \overrightarrow{x_i} \ . \tag{14}$$

The update rule of the algorithm thus becomes:

$$\overrightarrow{W}_{k+1} = \overrightarrow{W}_k + \alpha_k \overrightarrow{e}_k \ . \tag{15}$$

The bias b_{k+1} is estimated by forcing the resulting hyperplane to pass through the centroids of the misclassified regions, that is

$$b_{k+1} = -\overrightarrow{W}_{k+1} \cdot \overrightarrow{FP}_k = -\overrightarrow{W}_{k+1} \cdot \overrightarrow{FN}_k \ . \tag{16}$$

This enforcement means that $\overrightarrow{W}_{k+1} \perp \overrightarrow{e}_k$ or $\overrightarrow{W}_{k+1} \cdot \overrightarrow{e}_k = 0$.

From the last relation and the update rule 15 follows that:

$$\alpha_k = -\frac{\overrightarrow{W}_k \cdot \overrightarrow{e}_k}{||\overrightarrow{e}_k||^2} \ . \tag{17}$$

From the description of the algorithm it is evident that although similar is not a Perceptron algorithm. Thus a convergence proof is necessary.

3.1 Convergence Proof of the Proposed Algorithm

Theorem. Given a set of linearly separable training examples x_i if $\overrightarrow{W}^*$ denotes a perfect solution hyperplane such that: $R^2 = \min ||e_k||^2$ and $\gamma = \min W^* e_k$ over all possible nonempty subsets of S, then the update rule in Eq. (15) converges to W^* if $\alpha_i > 0$ and

$$\lim_{k \to \infty} \sum_{i=1}^{k} \alpha_i = \infty \ . \tag{18}$$

Proof. We shall show that each update brings the weight vector closer to the solution region. In particular we prove that $||W_{k+1} - \lambda W^*|| \leq ||W_k - \lambda W^*||$ for W^* sufficiently long ($\lambda > 0$).

$$\begin{aligned} ||\overrightarrow{W}_{k+1} - \lambda \overrightarrow{W}^*||^2 &= ||\overrightarrow{W}_k - \lambda \overrightarrow{W}^*||^2 + \alpha_k^2 ||\overrightarrow{e}_k||^2 + 2\alpha_k \overrightarrow{W}_k \cdot \overrightarrow{e}_k - 2\lambda\alpha_k \overrightarrow{W}^* \cdot \overrightarrow{e}_k \\ &\leq ||\overrightarrow{W}_k - \lambda \overrightarrow{W}^*||^2 + \alpha_k^2 ||\overrightarrow{e}_k||^2 + 2\alpha_k \overrightarrow{W}_k \cdot \overrightarrow{e}_k \end{aligned}$$

The last inequality holds since $\overrightarrow{W}^* \cdot \overrightarrow{e}_k > 0$. The term $\alpha_k^2 ||\overrightarrow{e}_k||^2 + 2\alpha_k \overrightarrow{W}_k \cdot \overrightarrow{e}_k \leq 0$ when $\alpha_k = -\frac{2\overrightarrow{W}_k \cdot \overrightarrow{e}_k}{||\overrightarrow{e}_k||^2} \geq 0$. Therefore the optimal value of the learning rate α_k is: (derivative equal to zero)

$$\alpha_k = -\frac{\overrightarrow{W}_k \cdot \overrightarrow{e}_k}{||\overrightarrow{e}_k||^2} \geq 0 \ . \tag{19}$$

By substituting the optimal value of α_k we get:

$$\begin{aligned} ||\overrightarrow{W}_{k+1} - \lambda \overrightarrow{W}^*||^2 &= ||\overrightarrow{W}_k - \lambda \overrightarrow{W}^*||^2 - \alpha_k^2 ||e_k||^2 - 2\lambda\alpha_k \overrightarrow{W}^* \cdot \overrightarrow{e}_k \leq \\ &\leq ||\overrightarrow{W}_k - \lambda \overrightarrow{W}^*||^2 - \alpha_k^2 R^2 - 2\lambda\alpha_k\gamma \end{aligned}$$

where R^2 and γ are defined over all the misclassified subsets of the training set. After k updates we have:

$$||\overrightarrow{W}_{k+1} - \lambda \overrightarrow{W}^*||^2 \leq ||\overrightarrow{W}_0 - \lambda \overrightarrow{W}^*||^2 - R^2 \sum_{i=1}^{k} \alpha_i^2 - 2\lambda\gamma \sum_{i=1}^{k} \alpha_i \ . \tag{20}$$

From (20) follows that there is a k_0 such that $\forall k \geq k_0$ the right hand side is negative if relation (18) holds. Thus for $k \geq k_0$, $||\overrightarrow{W}_{k+1} - \lambda \overrightarrow{W}^*||^2 = 0$ or $\overrightarrow{W}_{k+1} \to \lambda \overrightarrow{W}^*$.

4 Experimental Results

In this Section we describe the datasets and present experimental results of text classification from several real datasets as well as from artificially generated linearly separable datasets.

4.1 Text Classification Datasets

For evaluation of the performance, scalability and robustness of the algorithms, five main text corpuses were used; namely the Reuters-21578 corpus, the Reuters-RCV1, the OHSUMED, the TREC-AP corpus and the 20 Newsgroup corpus.

From these corpuses, we have constructed various subsets in order to explore the behavior of the algorithms under different synthesis and size of both the train and the testing set. Training and testing instances are represented by vectors with tf*idf weights. Stemming was applied using Porter's stemmer [18] and stop words were removed. In the following paragraphs the text datasets are briefly described. Some statistics of the datasets are presented in Table 1.

Table 1. Statistics of text datasets used for evaluation

Dataset Name	Train Set	Test Set	Train Set	Test Set
	Documents	Documents	Unique Terms	Unique Terms
Reuters-10	9,603	3,299	19,970	11,673
Reuters-90	9,603	3,299	19,970	11,673
Reuters-10-x	6,490	2,545	16,442	9,916
Reuters-90-x	7,770	3,019	18,029	11,129
Reuters-RCV1-103	23,149	781,265	45,860	275,268
Ohsumed-23	6,286	7,643	20,338	22,566
Ohsumed-96	183,229	50,216	98,893	57,497
Ohsumed-49	183,229	50,216	98,893	57,497
Ohsumed-28	183,229	50,216	98,893	57,497
Ohsumed-96-x	12,822	3,760	17,323	10,842
Ohsumed-49-x	12,445	3,632	17,031	10,625
Ohsumed-28-x	953	274	5,201	3,153

Reuters-21578. Reuters-21578[1] corpus. In the Mod-Apte split [14,15], which it is used, only 90 categories have at least one document in the training and one in the testing set. This dataset is called, Reuters-90. Reuters-10 dataset contains only the 10 largest classes. The subsets, Reuters-90-x and Reuters-10-x, contain documents exclusively from the corresponding, 90 or 10, categories.

[1] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

OHSUMED. OHSUMED[2] corpus was compiled by [8]. Ohsumed-23 is a subset of the original collection, with documents classified to one out of 23 classes from the Heart Diseases subtree of the MeSH classification scheme. Ohsumed-96 contains 96 classes, those having at least one document in the training and one in the testing set. Ohsumed-96 was further divided into two subsets. Ohsumed-49 that contains the largest classes, those with at least 75 documents in the training set and ohsumed-28 contains 28 classes with 15 to 74 documents each in the training set. The subsets with the suffix -x contain documents only from the corresponding categories.

Reuters-RCV1. Reuters-RCV1[3] corpus [16] is a new generation Reuters corpus with multi-labeled that contains 103 classes.

TREC-AP. TREC-AP[4] corpus contains 209,783 documents classified into 20 classes. (142,791 are used for training and 66,992 for testing).

Single-label Datasets. We have also experimented with 3 more single-label datasets. NG-20, with 20 classes of Newsgroups[5] [13], with 9840 documents for training and 6871 for testing. Two more single-label dataset were derived from the Reuters-21578 corpus. Reuters-8 is the subset of Reuters-10 with single label documents only, and similarly Reuters-52 is the subset of Reuters-90.

Artificially Generated Datasets. Several linearly separable datasets were generated using MATLAB scripts for various dimensions. For each dimension-value datasets of 100, 1000 and 10000 instances were created. In total there are 18 artificially generated linear separable datasets. In our notation, 2-dim-1000-inst stand for a 2-dimensional space and 1000 instances.

4.2 Evaluation Procedure

Experiments were run on the same machine with the same initial conditions, i.e. the same initial weight vector $\overrightarrow{W}^{(0)}$ (Eq. 7) and the same bias value b (Scut value). For Perceptron a fixed learning rate was used ($\alpha = 1$). Our main goal was to compare the cost for the training phase (in terms of training iterations) for the Perceptron and the proposed algorithm. In the case where the datasets are not linearly separable, training is stopped after a predefined number of iterations and the best hyperplane (in terms of the F_1 evaluation measure) is retained. The artificially generated datasets, are shortage from a testing set.

Finally a comparison of the efficiency (as measured by AUC) is applied between the proposed algorithm and the SVM. We used the SVMPerf [11].

Evaluation Measures. In both training and testing phase the microaveraged F_1 and the AUC (Area Under the ROC Curve) were used.

[2] ftp://medir.ohsu.edu/pub/ohsumed/

[3] http://www.daviddlewis.com/resources/testcollections/rcv1/

[4] http://www.daviddlewis.com/resources/testcollections/trecap/

[5] http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html

5 Discussion of Experimental Results

Table 2 illustrates the average number of iterations (over all binary problems of each dataset) and the micro averaged F_1 measure of all classes of each dataset). In same Table is also presented the performance of the classifiers on the testing phase. The performance is measured by the AUC score macro-averaged over all classes in each dataset. In the last column of 2 results from an SVM classifier on the same datasets are presented.

Table 2. Performance of the Perceptron and Proposed algorithm of the training an testing phase

	Training Phase				Testing Phase		
Dataset Name	Avg Iterations		microF_1 Score		macro AUC Score		
	Perceptron	Proposed	Perceptron	Proposed	Perceptron	Proposed	SVM
Reuters-10	462	136	0.9933	0.9976	0.9721	0.9719	0.9940
Reuters-90	318	28	0.9843	0.9979	0.9793	0.9845	0.9011
Reuters-10-x	434	150	0.9968	0.9984	0.9745	0.9744	0.9970
Reuters-90-x	300	32	0.9899	0.9983	0.9834	0.9842	0.9138
Reuters-RCV1	455	37	0.8845	0.9918	0.9385	0.9418	0.9491
Ohsumed-23	340	14	0.9423	1.0	0.9347	0.9326	0.9196
Ohsumed-96	362	19	0.8122	1.0	0.9688	0.9907	0.7218
Ohsumed-49	454	30	0.8235	1.0	0.9769	0.9846	0.9289
Ohsumed-28	228	11	0.6983	1.0	0.9411	0.9985	0.4810
Ohsumed-96-x	309	12	0.9101	1.0	0.9681	0.9747	0.9267
Ohsumed-49-x	317	18	0.9155	1.0	0.9516	0.9594	0.9436
Ohsumed-28-x	57	5	1.0	1.0	0.9792	0.9786	0.9685
Trec-20	483	59	0.8963	0.9997	0.9847	0.9885	0.9922
NG-20	395	89	0.9989	0.9996	0.9792	0.9802	0.9766
Reuters-8	339	85	0.9984	1.0	0.9846	0.9846	0.9980
Reuters-52	251	27	0.9961	0.9993	0.9855	0.9855	0.9402

The rate of convergence on the artificially generated datasets is illustrated by the number of iterations each method takes to converge. In all the experiments the initial conditions were used. Figure 3 summarizes the results from these datasets. The y-axis indicates the number of iterations for convergence and the x-axis represents the 18 artificial data sets.

Considering the results of Table 2, three main conclusions can be derived.

First, both the perceptron and the proposed algorithm, perform at least as good (if not better) as the state-of-the-art method SVM, on new unseen data.Measurements of the AUC score on testing datasets show that they perform extremely well, at least on those text datasets we evaluated them on.Out of the 16 text datasets we tested, SVM outperforms on 5 of them, perceptron on 3 of them and the proposed algorithm on 9 of them. However, their differences

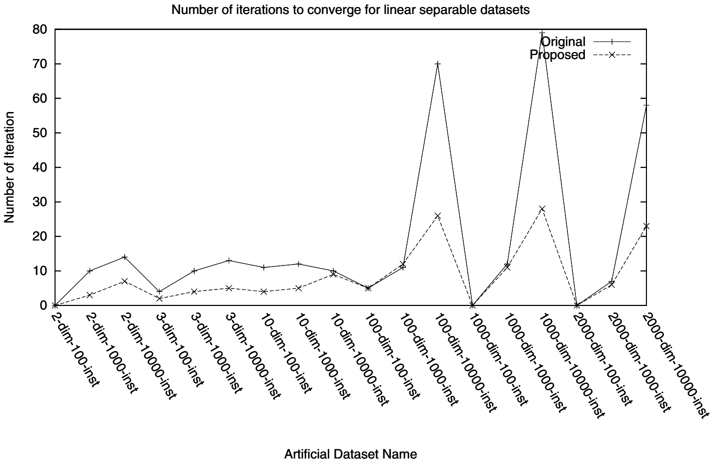


Fig. 3. Convergence results of perceptron and proposed algorithm on artificially generated datasets

on the AUC score are minor except on the Ohsumed dataset were the proposed algorithm significantly outperforms both of the other methods.

Second, the proposed algorithm finds a perfect or a better separating hyperplane more often than the original fixed increment batch perceptron. During the training phase of the text datasets, the proposed method achieved better micro-averaged F_1 score on all 16 datasets. Indeed on 7 of the text datasets the algorithm found a perfect separating hyperplane (that is why micro$F_1 = 1.0$).

Third, the proposed method converges much faster than the original perceptron. This is shown by the average number of iteration each method needs to converge.

As far for the linearly separable artificial datasets ($F_1 = 1$ at the training phase), see Figure 3, we may conclude that: On spaces of high dimensionality (thousands dimensions) and large datasets the proposed method needs significantly less iterations than the original perceptron. On spaces of low dimensionality (less than 100 dimensions), and small datasets both methods require about the same number of iteration to converge. However in latter case and with large datasets then the proposed method needs slightly less iterations. This is an interest case for further investigation using kernel functions to accelerate convergence.

6 Conclusions

In this paper, an algorithm,introduced in [6], was presented for training a classifier, without the need of using an arbitrary value or prior estimated learning rate parameter. A proof of convergence was given. Results from training text classifiers and from artificially generated linear separable datasets, show that the method is both fast and accurate. In concluding we summarize that:

– The proposed method is very simple and easy to implement and operate.
– It is computationally cheap and experimental results show that it is very fast,
– It can be used in any domain and any data without any modification or parameter estimation.
– It converges after a finite number of steps to a separating hyperplane in the case of linearly separable data.

Finally, we note that there is a lot of space for further research of the proposed algorithm, which is currently under investigation, in order to find the best classifier, the one with maximal margins from the datasets as well as to improve convergence for low-dimension data using kernel functions.

References

1. Buckley, C., Salton, G.: Optimization of relevance feedback weights. In: SIGIR 1995: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 351–357. ACM, New York (1995)
2. Cristianini, N., Shawe-Taylor, J.: An Introduction To Support Vector Machines (and other kernel-based learning methods). Cambridge University Press, Cambridge (2000)
3. Dagan, I., Karov, Y., Roth, D.: Mistake-driven learning in text categorization. In: 2nd Conference on Empirical Methods in Natural Language Processing, EMNLP 1997, pp. 55–63 (1997)
4. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley Interscience, Hoboken (November 2000)
5. Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization (1998)
6. Gkanogiannis, A., Kalampoukis, T.: A modified and fast perceptron learning rule and its use for tag recommendations in social bookmarking systems. In: ECML PKDD Discovery Challenge 2009 - DC 2009 (2009)
7. Harman, D.: Relevance feedback and other query modification techniques, pp. 241–263 (1992)
8. Hersh, W., Buckley, C., Leone, T., Hickman, D.: Ohsumed: an interactive retrieval evaluation and new large test collection for research (1994)
9. Joachims, T.: Text categorization with support vector machines: learning with many relevant features (1998)
10. Joachims, T.: Making large-scale support vector machine learning practical. In: Schölkopf, B., Burges, C.J.C., Smola, A.J. (eds.) Advances in Kernel Methods: Support Vector Learning, pp. 169–184. MIT Press, Cambridge (1999), http://portal.acm.org/citation.cfm?id=299104
11. Joachims, T.: Training linear svms in linear time. In: KDD 2006: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 217–226. ACM, New York (2006), http://dx.doi.org/10.1145/1150402.1150429

12. Karypis, G., Shankar, S.: Weight adjustment schemes for a centroid based classifier (2000)
13. Lang, K.: Newsweeder: learning to filter netnews (1995)
14. Lewis, D.D.: Evaluating text categorization. In: Workshop on Speech and Natural Language HLT 1991, pp. 312–318 (1991)
15. Lewis, D.D., Schapire, E.R., Callan, P.J., Papka, R.: Training algorithms for linear text classifiers. In: 19th ACM International Conference on Research and Development in Information Retrieval SIGIR 1996, pp. 298–306 (1996)
16. Lewis, D.D., Yang, Y., Rose, T., Li, F.: Rcv1: A new benchmark collection for text categorization research. Journal of Machine Learning Research 5, 361–397 (2004)
17. Novikoff, A.B.: On convergence proofs for perceptrons. In: Proceedings of the Symposium on the Mathematical Theory of Automata, vol. 12, pp. 615–622 (1963), http://citeseer.comp.nus.edu.sg/context/494822/0
18. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
19. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychological Review 65(6), 386–408 (1958)
20. Salton, G.: Automatic Text Processing – The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, Reading (1989)
21. Schapire, R.E., Singer, Y., Singhal, A.: Boosting and rocchio applied to text filtering. In: SIGIR 1998: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 215–223. ACM, New York (1998)
22. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
23. Yang, Y.: A study on thresholding strategies for text categorization (2001)

Research on Time Series Forecasting Model Based on Moore Automata

Yixiong Chen[1,*], Zhongfu Wu[1], Zhiguo Li[2], and Yixing Zhang[3]

[1] Chongqing University, School of Computer Science, 400044 Chongqing, China
{chenyx,wzf}@cqu.edu.cn
[2] Shanghai Baosight Software Corporation, 400039 Chongqing, China
lizhiguo@cqu.edu.cn
[3] Fudan University, School of Mathematical Sciences, 200433 ShangHai, China
07300180062@fudan.edu.cn

Abstract. For time series data mining (TSDM), the problem of time series forecasting has attracted wide attention as solving it actually paves a way to extrapolate past behavior into the future. Researchers have long been interested in modeling the problem by linear regression, neural network, chaos, support vector machines, etc. In this paper, we explore the use of Moore automata for time series forecast modeling and demonstrate how the Moore automata can be converted to solve the problem with regression methods. The effectiveness of the proposed approach has been verified by experiments.

Keywords: Time Series Forecasting, Data Mining, Automata.

1 Introduction

A time-series is a sequence of data which is associated with time, such as daily temperature measurement. Forecasting time-series means that we extend the historical values into the future where the measurements are not available yet [1].

Recently, various technologies have been proposed for time-series forecasting [2], [3], [6], [10]. However, there is room for improvement as many of the technologies show their weaknesses in one way or another: for neural network methods, overtraining or insufficient-training that will lead to a local optimum is hard to avoid without a reliable model; for statistical methods, modeling complex time series forecasting problems is a hard job. Therefore, the discovery of some novel modeling methods is recognized as an important issue for time series forecasting.

Motivated by this background, we explore the use of Moore automata for time series forecast modeling in this paper. By way of contrast, our method has wider range of application than classical statistical methods (e.g. ARIMA) as it can analyze relationships between two time-series while statistical methods normally can only deal with one. In addition, our method has no leaning phrase which is time consuming for most neural network methods, so it works more efficiently during the process of regression especially for short term forecasting.

[*] Corresponding author.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 98–105, 2010.

2 Time Series Forecasting Model

2.1 Time Series Generating

We use automata to reflect the process of time series generating. Here, we assume that the time series data is generated by a discrete dynamic system with feedbacks [7].

Table 1. List of terminology

Symbols	Description
R^p	A p-dimension space of real number
$x_1, x_2 \dots x_k$	Input time series
$y_1, y_2 \dots y_k$	Output time series
$g()$	State transition function
$h()$	Output Function
$s_0, s_1 \dots s_k$	Internal state sequence
m	Order of differential equation
$f()$	Forecasting function

As can be seen from the table, let x, s, y be the input vector, internal state, and output vector respectively. The system state and output at time k can be defined as:

$$s_k = g(s_{k-1}, x_k), \; y_k = h(s_k, x_k) \; . \tag{1}$$

The formula can be also depicted as a Mealy automaton [4] (Fig.1a). When state s is taken from a finite state space instead of R^r, the Mealy automaton can be seen as a finite state automaton. Furthermore, if y_k is only determined by current states, the established automaton is also a Moore automaton.

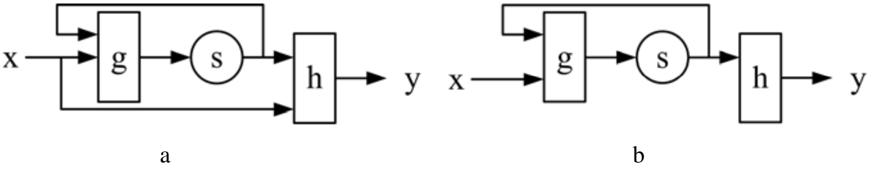


Fig. 1. Mealy automaton (a) and Moore automaton (b) for time series generating

Although we can implement a Mealy automaton for dynamic behaviors with fewer states and dimensions, Moore automata are more convenient for analysis and design. Since the equivalence between Mealy automata and Moore automata has been proved [5], we adopt a Moore automaton (Fig.1b) which is given by:

$$s_k = g(s_{k-1}, x_k), \; y_k = h(s_k) \; . \tag{2}$$

Now, time series generating can be seen as outputs of the Moore automaton.

2.2 Approximation of the Automata

The purpose of forecasting is to predict an unknown output $\{y_{n+1},\ldots,y_{n+q}\}$ for input sequence $\{x_{n+1},\ldots,x_{n+q}\}$ according to the observed sequence $\{x_1,\ldots,x_n\}$ and $\{y_1,\ldots,y_n\}$. During the process, function *g* and *h* need to be determined to find a predicted output series which approximates the given data series as closely as possible.

If we define *g* and *h* with quaternion (x_k, s_{k-1}, s_k, y_k), then the problem can be converted as a regression problem. Although the internal states are unknown, they can be estimated since s_k is determined by all its previous state $s_0,\ldots,s_{k-1}$, and $s_1,\ldots,s_{k-1}$ are influenced by $x_1,\ldots,x_{k-1}$ through *g* and generate $y_1,\ldots,y_{k-1}$ through *h*. So s_k is actually determined by $x_1,\ldots,x_k$ and s_0. While s_0 is also unknown, we can use historical data $x_1,\ldots,x_k$ and $y_1,\ldots,y_{k-1}$ to estimate s_0 only if it is distinguishable with $y_1,\ldots,y_{k-1}$.

$$y_k = f_k(y_1,\ldots,y_{k-1},x_1,\ldots,x_{k-1},x_k), k = 2,\ldots,n \ . \tag{3}$$

Thus, we can eliminate s_0 and define y_k as formula (3). It is actually a non-state model which can be seen as an approximation of the Moore automaton.

2.3 Backward Differential Equation with m-Order

Normally, it is impossible to estimate y_k from s_0 because *n* is always a huge number in data mining. In addition, $f_2,\ldots,f_n$ can not be generated through regression methods as there is only one data group available at each time *k*. To solve this problem, we may start from state s_{k-m-1}, calculate y_k through $x_{k-m},\ldots,x_k$. Although s_{k-m-1} is unknown, s_{k-m-1} can be estimated as long as we have sufficient data from y_{k-m} to y_{k-1}, thus:

$$y_k = f(y_{k-m},\ldots,y_{k-1},x_{k-m},\ldots,x_{k-1},x_k), k = m+1,\ldots,n \ . \tag{4}$$

This is an *m*-order backward differential equation. We can use the latest *m* values, instead of all previous inputs and outputs, to estimate the current state.

2.4 Model Analysis

Here we use the following instance to illustrate how the model is constructed. Let *n*=18, *m*=7, then we have *n-m*=11 data groups as follows:

$$\begin{aligned} & y_8 = f(y_7,y_6,y_5,y_4,y_3,y_2,y_1,x_8,x_7,x_6,x_5,x_4,x_3,x_2,x_1) \\ & \quad \ldots\ldots \\ & y_{18} = f(y_{17},y_{16},y_{15},y_{14},y_{13},y_{12},y_{11},x_{18},x_{17},x_{16},x_{15},x_{14},x_{13},x_{12},x_{11}) \end{aligned} \ . \tag{5}$$

The most common used data are $x_7, x_8,\ldots, x_{11}, y_8, y_9,\ldots, y_{11}$ in the above example. This means that these data are crucial for determining *f* with regression method. To get better estimations of unknown states, *m* should not be too small when compared with *n*. Conversely, it is difficult to keep enough data groups for regression methods if *m* is too large. To acquire an ideal value of *m*, heuristics, such as ACO, Tabu Search, can be adopted. Due to the space limitation, we have tried a simple Golden section method in this paper:

$$m = n*(1-0.618) \quad or \quad m = n*(1+0.382) \ . \tag{6}$$

Formula (6) can be repeatedly adopted to adjust m tentatively based on the effect of approximation.

3 Algorithm Design

3.1 General Framework

Using methods in section 2.2, the proposed automaton can be approximately converted as a non-state backward m-order differential equation. To solve it, the general framework of the algorithm is:

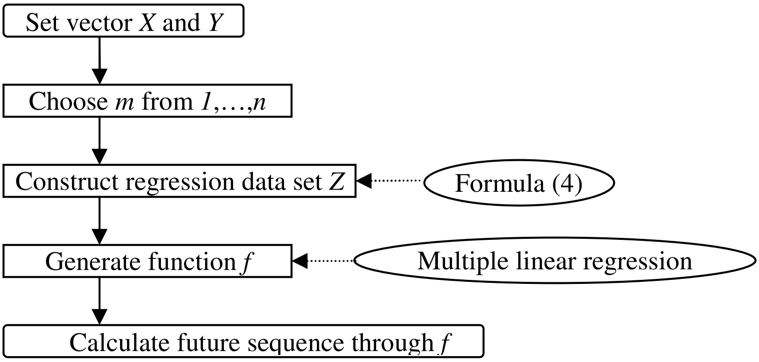

Set vector X and Y
Choose m from 1,…,n
Construct regression data set Z
Formula (4)
Generate function f
Multiple linear regression
Calculate future sequence through f

Fig. 2. General steps of the algorithm

3.2 Regression Method

Firstly, we set m with formula (6) or other heuristic methods mentioned in section 2. Similar to the expansion of Moore automata (Fig.3), we convert the original time series data to:

$$\begin{aligned} & y_{m+1} = f(y_m, y_{m-1}, \dots, y_1, x_{m+1}, x_m, \dots, x_1) \\ & y_{m+2} = f(y_{m+1}, y_m, \dots, y_2, x_{m+2}, x_{m+1}, \dots, x_2) \\ & \dots \\ & y_n = f(y_{n-1}, y_{n-2}, \dots, y_{n-m}, x_n, x_{n-1}, \dots, x_{n-m}) \end{aligned} . \tag{7}$$

Matrix Z with dimension *(n-m)*2(m+1)* is constructed as:

$$Z = \begin{bmatrix} y_{m+1} & y_m & y_{m-1} & \cdots & y_1 & x_{m+1} & x_m & \cdots & x_1 \\ y_{m+2} & y_{m+1} & y_m & \cdots & y_2 & x_{m+2} & x_{m+1} & \cdots & x_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ y_n & y_{n-1} & y_{n-2} & \cdots & y_{n-m} & x_n & x_{n-1} & \cdots & x_{n-m} \end{bmatrix} . \tag{8}$$

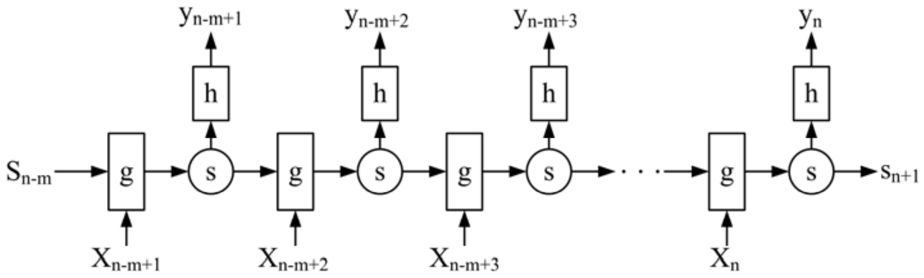


Fig. 3. Expansion of Moore Automata

Here, $z^{(i)}$ stands for the i-th column in Z. The linear relationship between $z^{(i)}$ and other vectors $z^{(j_1)}$,..., $z^{(j_p)}$ needs to be determined by multiple linear regression method [7], [9], and parameter a and b are estimated by approximation of function f:

$$z_k^{(i)} \approx a + \sum_{l=1}^{p} b_l (z_k^{(j_l)} - \bar{z}^{(j_l)}) \cdot \tag{9}$$

Let q be dimension of column vector $z^{(i)}$, the average square deviation is given by:

$$E = \frac{1}{q} \sum_{k=1}^{q} (z_k^{(i)} - a - \sum_{l=1}^{p} b_l (z_k^{(j_l)} - \bar{z}^{(j_l)}))^2 \cdot \tag{10}$$

The conditions to obtain local minimum are:

$$\frac{\partial E}{\partial a} = -\frac{2}{q} \sum_{k=1}^{q} (z_k^{(i)} - a - \sum_{l=1}^{p} b_l (z_k^{(j_l)} - \bar{z}^{(j_l)})) = 0 \cdot \tag{11}$$

$$\frac{\partial E}{\partial b_l} = -\frac{2}{q} \sum_{k=1}^{q} (z_k^{(j_l)} - \bar{z}^{(j_l)})(z_k^{(i)} - a - \sum_{\lambda=1}^{p} b_\lambda (z_k^{(j_{\lambda l})} - \bar{z}^{(j_\lambda)})) = 0 \cdot \tag{12}$$

For all $l=1,\dots,p$, we have $a = \bar{z}^{(i)}$. As for parameter b_l ($l=1,\dots,p$), they can be calculated by solving equations:

$$\sum_{\lambda=1}^{p} b_\lambda \sum_{k=1}^{q} (z_k^{(j_\lambda)} - \bar{x}^{(j_\lambda)})(z_k^{(j_l)} - \bar{z}^{(j_l)}) = \sum_{k=1}^{q} (z_k^{(i)} - \bar{z}^{(i)})(z_k^{(j_l)} - \bar{z}^{(j_l)}) \Leftrightarrow \sum_{\lambda=1}^{p} c_{j_\lambda j_l} b_\lambda = c_{i j_l} \cdot \tag{13}$$

Now f can be generated and applied to predict future output data sequence y_{n+1}, $y_{n+2},\dots, y_{n+i}$ based on input sequence $x_{n+1}, x_{n+2},\dots, x_{n+i}$.

4 Experiments and Case Study

4.1 Data Preparation

We adopted MatlabR2007 to implement the approach and conduct tests on time-series data from [8]. Through the test, our method is expected to analysis the relationship or influence between time series X and Y, then predict the trend of Y based on X.

To further verify our method, we have also applied our method to real world data. Let *X* as the daily opening price (2004.10~2010.7) of Shenzhen Stock Market and *Y* as the daily opening price of Shanghai market, we have adopted our approach to forecast the trend of Shanghai Index according to Shenzhen Index (see Fig.4b).

Table 2. Prepared Input and Output Data

Sequence No.	Input *X*	Output *Y*	Matrix(*Z*) Element for Regression							
1	3	7								
2	-2	-1								
3	1	2								
4	-1	3	3	2	-1	7	-1	1	-2	3
5	2	5	5	3	2	-1	2	-1	1	-2
6	-3	7	7	5	3	2	-3	2	-1	1
7	4	9	9	7	5	3	4	-3	2	-1
8	-4	11	11	9	7	5	-4	4	-3	2

4.2 Results and Analysis

According to our method, when we define m=3, the corresponding parameter b_1, b_2, b_3, b_4, b_5, b_6, b_7 can be determined by:

$$\begin{bmatrix} 6.56 & 6.76 & 0.76 & -1.72 & 1.68 & -0.68 & 0.08 \\ 6.76 & 7.36 & -0.84 & -1.12 & 1.28 & -0.08 & -0.92 \\ 0.76 & -0.84 & 7.36 & -3.32 & 2.68 & -3.28 & 4.48 \\ -1.72 & -1.12 & -3.32 & 9.04 & -7.16 & 5.16 & -4.16 \\ 1.68 & 1.28 & 2.68 & -7.16 & 5.84 & -4.24 & 3.24 \\ -0.68 & -0.08 & -3.28 & 5.16 & -4.24 & 3.44 & -3.04 \\ 0.08 & -0.92 & 4.48 & -4.16 & 3.24 & -3.04 & 3.44 \end{bmatrix} \cdot \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} 7.2 \\ 7.6 \\ 0 \\ -1.6 \\ 1.6 \\ -0.4 \\ -0.4 \end{bmatrix} \tag{14}$$

Thus, all the parameters needed by the final regression equation can be determined by solving the equation (14) and listed as follows:

Table 3. Test Results of Parameters

Parameters	Value	Parameters	Value	Parameters	Value
m	3	b_2	0.7093	b_5	1.2188
a	7	b_3	-6.39e-14	b_6	1.3869
b_1	0.3140	b_4	0.4746	b_7	0.7178

So the regression equation is: y_k=7+0.314(y_{k-1}-5.2)+0.7093(y_{k-2}-3.2)-6.39×10^{-14}(y_{k-3}-3.2)+0.4746(x_k+0.4)+1.2188(x_{k-1}-0.6)+1.3869(x_{k-2}+0.6)+0.7178(x_{k-3}-0.6).

Now we can calculate a data sequence y_4,…, y_8, its deviation can be measured by:

$$E=\frac{1}{5}\sqrt{\sum_{k=4}^{8}(y_k-7-\sum_{l=1}^{7}b_l(x_k^{(j_l)}-\bar{x}^{(j_l)}))^2}=2.5661\times10^{-5}\ \cdot \tag{15}$$

It shows that our method works effectively to generate the expected regression function. With this function, we can predict the future trend of time series Y based on the time series X (i.e. if x_9=5 then y_9=13.6872, etc.).

The test result can be depicted by Fig.4a. Although it is conducted on a short time series to illustrate the calculating process, our method is also verified by applying to real world data from Chinese stock market and the result is shown in Fig.4b.

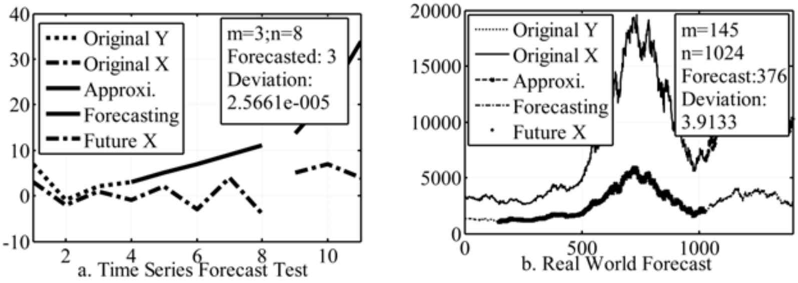


Fig. 4. Forecasting Experiment Results

From the experiments, we have noticed that the predicted results can be trusted especially when applying to short term forecasting. But for long term forecasting, the predicted values need to be verified with measured value or expertise. We have also found that our method has a wide range of adaptability for time series forecasting except some datasets with strong nonlinear relation.

5 Conclusions

In this paper, we have proposed a Moore automata-based time series forecasting approach. The similarity between automata and regression methods when generating data sequences inspires us exploring the use of automata for time series forecasting.

The advantage of our methods lies in: a) It can analysis relationships between two time series while most traditional statistical methods are often designed for a single time series; b) When predicting short term trend of a time-series, our method works more efficiently than some other methods during the process of regression.

In our future works, the characteristics of original data series and its influences on the effect of prediction need further discussion. The way of determining the crucial parameter m also needs further verification and is expected to be improved.

Acknowledgments. This work is supported by the Fundamental Research Funds for Central Universities (No.CDJZR10180006), Natural Science Foundation Project of CQ CSTC (No.2008BB2183), National Social Sciences Fund Special Projects in Educational Science (No.ACA07004-08), Key Projects in National Science & Technology Pillar Program (No.2008BAH37B04), “211 Project” phrase Ⅲ (No.S-10218) and National Natural Science Foundation Key Program (No.90818028).

References

1. Xinning, S., Jianglin, Y., et al.: Data Mining Theory and Technology. Science Technology Press, Beijing (2003)
2. Pengtao, J., Huacan, H., Li, L., Tao, S.: Overview of Time Series Data Mining. J. Application Research of Computer 24, 15–18 (2007)
3. Wenlong, Q., Haiyan, L., Wei, Y., et al.: Research on Multi-scale Prediction of Time Series based on Wavelet and Support Vector Machines. J. Computer Engineering and Applications 43, 182–185 (2007)
4. Edward, F.M.: Mind-Experiments on Sequential Machines. J. Automata Studies, 129–153 (1956)
5. Istvn, B.: Equivalence of Mealy and Moore automata. J. Acta Cybernetica 14, 541–552 (2001)
6. Fusheng, T.: Modeling Dynamical Systems by Recurrent Neural Networks. Technical report, University of California, San Diego (1994)
7. George, E.P.B., Gwilym, M.J., Gregory, C.R.: Time Series Analysis Forecasting and Control. Prentice-Hall, New Jersey (1994)
8. Programmers United Develop Net, `http://www.pudn.com`
9. Leona, S.A., Stephen, G.W.: Multiple Regression: Testing and Interpreting Interactions. Sage, USA (1991)
10. Xingli, B., Chengjian, Z.: Research on Time Series Forecasting Model Based on Support Vector Machines. In: 2010 International Conference on Measuring Technology and Mechatronics Automation, pp. 227–230. IEEE Press, Changsha (2010)

A Clustering Algorithm FCM-ACO for Supplier Base Management

Weining Liu and Lei Jiang

School of computer science, Chongqing University, Chongqing, China 400030
verajianglei@126.com

Abstract. Supplier selection is one of the critical components of supply chain management and has played a very important role to improve the competitiveness of the entire supply chain. Successful supplier selection may have significant business implications, while how to determine the suitable suppliers is a complex decision making problem which includes both qualitative and quantitative factors. Therefore, this paper proposes a novel approach that combines the fuzzy c-means (FCM) algorithm with ant colony optimization (ACO) algorithm to cluster suppliers into manageable smaller groups with similar characteristics. The simulation results show that the proposed method improves the performance of FCM algorithm for it is less sensitive to local extreme.

Keywords: fuzzy clustering, ACO, supplier base management, clustering supplier.

1 Introduction

With the developing of economic globalization and integration, the competition of market subject is no longer in single enterprise, but largely depends on the supply chain which is influenced by the suppliers. [1]

Meanwhile, international business development and popularization of the Internet has expanded the range of manufacturers' procurement options, so the supplier management has become a challenging problem in business and academic research.

Supplier management is a dynamic process which contains several different stages. Prahinski and Benton believed that the manage process of supplier should consist of 4 parts: the supplier selection, supplier classification, supplier establishment and the cooperation between enterprise and supplier [2]. However, from the view of supplier relationship, Wagner and Johnson thought there are 5 stages: the supplier classification, supplier evaluation, the eligible review for supplier, the qualified evaluation for supplier, the grading for supplier and supplier development. [3] It is thus clear that supplier management process will involve the analysis of suppliers and classification.

Classification means clustering all suppliers and dividing them into smaller sets, in which each set has a greater degree of similar characteristics. [4] Therefore, there is a need to develop an effective methodology to help clustering suppliers.

Recently various clustering approaches have been developed for this proposes. These algorithms can be classified into three main categories: heuristic, hierarchical

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 106–113, 2010.

and partition clustering methods. [5] Fuzzy clustering algorithms optimize an objective function that evaluates a given fuzzy assignment of objects to clusters. One of the widely used objective function clustering models is fuzzy c-means (FCM) [6] and a popular method to minimize the FCM objective function is alternating optimization (AO). But AO finds saddle points or local extrema of the objective function that might not be global extrema. [7] At the same time, the convergence rate of algorithm may be greatly limited by the initial value, especially large clustering number. Therefore, several evolutionary methods are used (proposed) for clustering problems. Bezdek and Hathaway optimized the hard c-means model (HCM) with genetic algorithm, [8] which is extended by Klawonn and Keller to the fuzzy c-means model (FCM). [9] Currently, swarm algorithms are increasingly considered for clustering. Particle swarm optimization (PSO) was first introduced by Eberhart and Kennedy [10] in order to optimize various continuous nonlinear functions.

Ant colony optimization (ACO) has also been successfully applied in clustering. ACO [11] is a class of algorithms, whose first member, called Ant System, was initially proposed by Colorni, Dorigo and Maniezzo. [11, 12] Handlet, Knowles and Dorigo [13] introduced a heuristic ACO algorithm for clustering, but this method does not consider any underlying cluster model. Runkler introduced an ACO algorithm that explicitly solves the HCM and FCM cluster models. [14]

2 Fuzzy c-Means Model (FCM)

The fuzzy c-means model can be described as follows [6]: when the original point set $X = \{x_1,\ldots,x_n\} \subset R^p$ is given, the goal of fuzzy clustering is to cluster the data points into $c \in \{2,\ldots,n-1\}$ subsets (clusters). The cluster assignments can be specified by a partition matrix $U \in \{0,1\}^{cxn}$ with all row sums > 0 and all column sums = 1. The objective function of the FCM is defined as

$$J_{fcm}(U,V;X) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} d_{ik}^{2} \ . \tag{1}$$

With $V = \{v_1,\ldots,v_c\} \subset R^P$ and $U \in M_{fcm}$ where

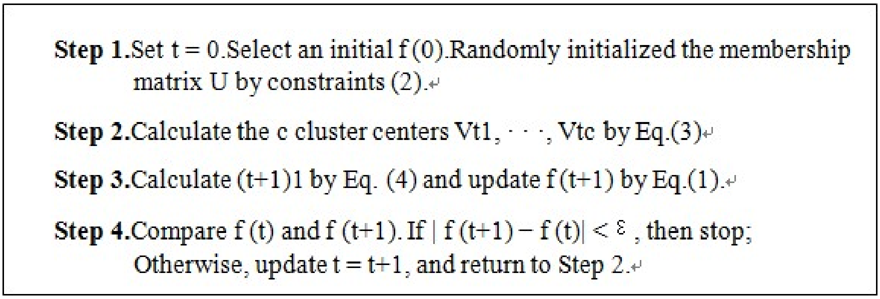
$$M_{fcm} = \{U \in [0,1]^{cXn} \mid \sum_{i=1}^{c} u_{ik} = 1, k = 1,\ldots n, \sum_{k=1}^{n} u_{ik} > 0, i = 1,\ldots c\} \ . \tag{2}$$

The necessary conditions for minimizing J_{fcm} are the following equations:

$$v_i = \frac{\sum_{k=1}^{n} u_{ik}^{m} x_k}{\sum_{k=1}^{n} u_{ik}^{m}}, i = 1,2,\ldots,c \ . \tag{3}$$

$$u_i = [\sum_{j=1}^{c} (\frac{d_{ik}}{d_{jk}})^{\frac{2}{m-1}}]^{-1}, i = 1,2,\ldots,c; k = 1,2,\ldots,n \ . \tag{4}$$

The fuzziness parameter m (m >1) is always selected according to the specific problem. The partition becomes fuzzier by increasing m and there is currently no theoretical basis for an optimal choice of m. [15] Parameters of the algorithm include the desired number of clusters c, real number m, and stop conditions should be given before the algorithm implementation. The algorithm steps can be described as Fig 1.



Step 1. Set t = 0.Select an initial f (0).Randomly initialized the membership matrix U by constraints (2).↵

Step 2. Calculate the c cluster centers Vt1, · · ·, Vtc by Eq.(3)↵

Step 3. Calculate (t+1)1 by Eq. (4) and update f (t+1) by Eq.(1).↵

Step 4. Compare f (t) and f (t+1). If | f (t+1) − f (t)| < ε , then stop; Otherwise, update t = t+1, and return to Step 2.↵

Fig. 1. Fuzzy c-means model steps

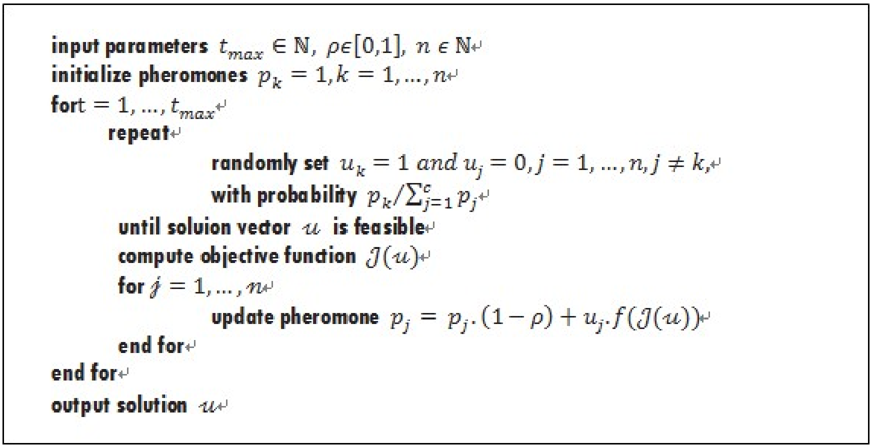
However we can randomly initialize V at the first step of the algorithm instead of U as shown above, and then alternatingly compute U and V, which is known as FCM-AO, meanwhile the above steps is called FCM-AO′. Both of FCM-AO and FCM-AO′ had been used in fuzzy clustering problems widely and successfully.

3 Ant Colony Optimization (ACO)

Ant colony optimization (ACO) was originally introduced by Dorigo and Maniezzo. [11] After that, many other applications of ACO were proposed where discrete optimization problems with large scale can be solved efficiently. The main underlying idea of ACO is inspired by the behavior of real ants in search of food. Despite the relatively poor perception of ants, the whole ant colony is able to efficiently find the shortest path between nest and food sources by building pheromone trails along the shortest paths.

In the ACO model, each ant moves randomly with a preference for higher pheromone concentrations; each ant leaves corresponding amount of pheromones according to the trail length; and the pheromones gradually evaporate over time, so the paths that were preferred earlier can be abandoned if the environment changes.

The ACO algorithm imitates these mechanisms by choosing solutions based on pheromones and updating pheromones based on the solution quality, evaporation also need to consider. In the original ant system there is a feature so-called visibility of path. It introduces a tabu list to store the solution candidates that violating any acceptability condition. This is useful in the traveling salesman problem. But in clustering (as well as in many other applications) all possible solutions are acceptable, so tabu list is unnecessary. This has discussed by Runkler. [14] In this paper, we want to apply ACO to solve the fuzzy clustering problem, so no tabu list is needed. We reference the tailored



```
input parameters $t_{max} \in \mathbb{N}$, $\rho \epsilon [0,1]$, $n \epsilon \mathbb{N}$
initialize pheromones $p_k = 1, k = 1, \ldots, n$
fort $= 1, \ldots, t_{max}$
      repeat
            randomly set $u_k = 1$ and $u_j = 0, j = 1, \ldots, n, j \neq k,$
            with probability $p_k / \sum_{j=1}^{c} p_j$
      until soluion vector $u$ is feasible
      compute objective fundion $\mathcal{J}(u)$
      for $j = 1, \ldots, n$
            update pheromone $p_j = p_j . (1 - \rho) + u_j . f(\mathcal{J}(u))$
      end for
end for
output solution $u$
```

Fig. 2. Tailored ACO algorithm steps

ACO algorithm described by Runkler and the Pseudo-code of the algorithm is described in Fig 2. For more details of this algorithm can be found in [14].

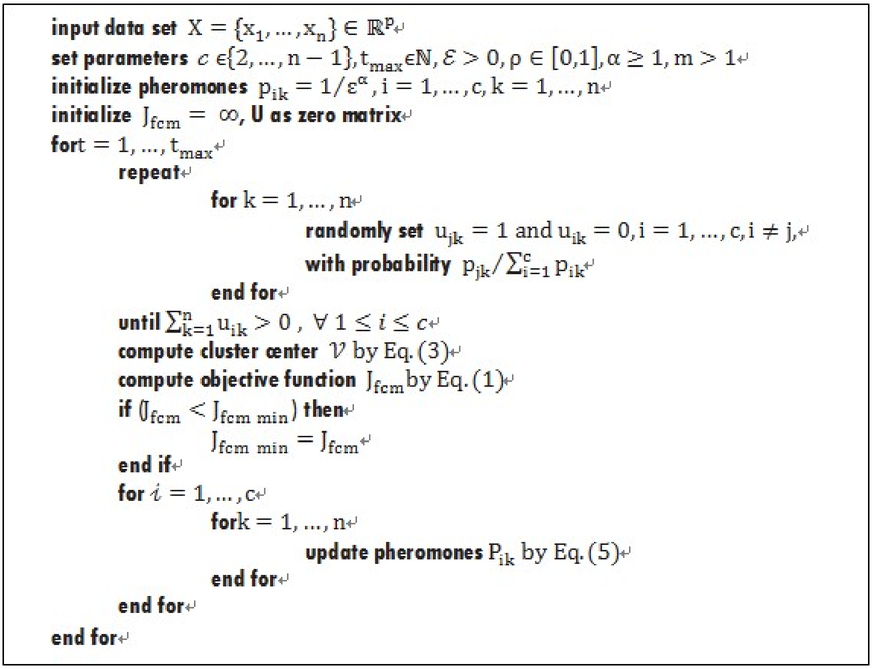
4 Proposed Algorithm FCM-ACO

Since FCM algorithm can get trapped in local minima, the aim of this paper is to find an effective way to improve the widely used FCM algorithm, making for better and efficient clustering results. Here, we focus on swarm-based stochastic optimization so-called Ant colony optimization which is described in section 3.

The FCM model's optimization is a mixture of two optimization problems. Finding the partition matrix u is a discrete optimization problem, and finding the cluster prototypes v is a continuous optimization problem. Since the ACO is particularly suited to solve discrete optimization problems, we will apply it to find the partition matrix and use the local optimization (part of FCM) to find the cluster prototypes. In our algorithm, each ant represents one data point and assigns it to one of the clusters. This assignment is based on a pheromone matrix $P \in R^{c \times n}$, so that data point x_k is assigned to cluster i with probability $p_{ik} \big/ \sum_{j=1}^{c} p_{jk}, i = 1,..., c, k = 1,..., n$. After a solution (U, V) is found, the value of the FCM objective function (using (1)) is computed, and the pheromones are updated by

$$p_{ik} = p_{ik} \cdot (1 - \rho) + u_{ik} \big/ (J - j_{\min} + \varepsilon)^{\alpha}, i = 1,...,c; k = 1,...,n \ . \qquad (5)$$

Where $j_{\min}$ is the minimum value of the objective function achieved yet, ε>0 is a technical parameter to avoid divisions by zero (when current objective function result J equals $j_{\min}$), $\rho \in [0,1]$ is the evaporation rate of pheromone, and α ≥ 0 is an exponent that affects the convergence speed. The pseudo-code of the whole method is shown below as Fig 3.



```
input data set $X = \{x_1, \dots, x_n\} \in \mathbb{R}^p$
set parameters $c \in \{2, \dots, n-1\}, t_{max} \in N, \mathcal{E} > 0, \rho \in [0,1], \alpha \geq 1, m > 1$
initialize pheromones $p_{ik} = 1/\varepsilon^{\alpha}, i = 1, \dots, c, k = 1, \dots, n$
initialize $J_{fcm} = \infty$, U as zero matrix
fort $= 1, \dots, t_{max}$
    repeat
        for $k = 1, \dots, n$
            randomly set $u_{jk} = 1$ and $u_{ik} = 0, i = 1, \dots, c, i \neq j,$
            with probability $p_{jk}/\sum_{i=1}^{c} p_{ik}$
        end for
    until $\sum_{k=1}^{n} u_{ik} > 0$ , $\forall\, 1 \leq i \leq c$
    compute cluster center $V$ by Eq. (3)
    compute objective function $J_{fcm}$by Eq. (1)
    if ($J_{fcm} < J_{fcm\ min}$) then
        $J_{fcm\ min} = J_{fcm}$
    end if
    for $i = 1, \dots, c$
        fork $= 1, \dots, n$
            update pheromones $P_{ik}$ by Eq. (5)
        end for
    end for
end for
```

Fig. 3. Proposed FCM-ACO algorithm steps

5 Experimental Results

In this section, we first compare the performance of our proposed FCM-ACO with that of classical FCM-AO based on the test data from the five well-known real-world data sets [16]. Then, take the clustering test for a set of discrete supplier data to illustrate its application. The coding of all methods in this paper is under help of the fuzzy tools available in Matlab2008. In order to get the best performance of both FCM and FCM-ACO, several tests have been performed and best values for their parameters are selected for the comparison. Based on this, the following parameters settings are used: $m = 2, \varepsilon = 0.01, \rho = 0.005, \alpha = 1$ and the upper limit of iteration is 1000.

The five well-known real-world data sets used in our experiments test include:

- ➢ The Iris Data Set, which consists of three different species of iris flower. For each species, 50 samples with four features were collected;
- ➢ The Wine data set, which has 178 objects and 13 features, each object is classified to three classes;
- ➢ The Glass Identification Data Set, which consists of 214 objects and 6 different types of glasses. Each type has 9 features;
- ➢ The Breast Cancer Wisconsin (Diagnostic) Data Set, which consists of 683objects and 2 categories characterized by 9 features;
- ➢ The Contraceptive Method Choice(CMC) Data Set, which consists of 1473 objects and 3 different types characterized by 9features;

These data sets cover examples of data of low, medium and high dimensions. We take 120 independent runs for the test of FCM-AO and 100 for FCM-ACO respectively. The experimental results are summarized in Table 1. All the numerical values in this table are the results of objective function (Eq.1). As shown in the table, the hybrid FCM-ACO obtained superior results than FCM-AO in most of data sets and it can escape from local optima efficiently.

Above experiments take tests of comparison for the basic clustering performance of FCM-ACO, and the following shows its use for supplier base management. While taking clustering experiments, we need a group of supplier information data that after discretization. Here, we use the supplier dataset from D. Parmar et al [4] as our test

Table 1. Results of FCM and FCM-ACO methods

Data set(n,c,d)	FCM			FCM-ACO		
	worst	average	best	worst	average	best
Iris(150,3,4)	73.32	70.13	68.82	71.96	68.65	66.93
Wine(178,3,13)	11933.6	11896.2	1168.9	12218.1	11603.9	11096.7
Glass(214,6,9)	73.35	72.64	71.83	74.31	72. 26	72.03
Cancer(683,2,9)	2246.7	2192.3	2126.8	2231.7	2190.6	2084.5
CMC(1473,3,9)	3568.2	3526.1	3502.9	3551.3	3508.2	3486.8

Table 2. Discretized data set

Obj	**QMP**	**SA**	**PMC**	**MGT**	**DD**	**C**	**Q**	**P**	**D**	**CRP**	**Other**	**Effi**
1	3	2	3	3	4	2	1	1	1	1	1	I
2	2	2	1	1	1	2	2	1	1	1	1	E
3	1	1	1	1	2	1	3	1	4	1	3	E
4	5	2	2	2	3	4	5	2	4	1	4	E
5	5	2	3	3	4	4	3	3	3	1	2	I
6	3	2	2	3	3	3	3	3	4	2	3	E
7	2	1	3	2	4	3	4	2	2	2	3	E
8	5	2	3	3	3	3	2	2	2	1	2	I
9	5	2	3	3	4	4	1	1	1	1	1	I
10	1	2	1	1	1	1	5	1	4	1	3	E
11	2	1	1	2	3	2	1	1	3	1	2	I
12	1	1	3	3	3	3	3	3	2	1	2	E
13	5	2	3	3	4	4	1	2	3	1	3	I
14	4	2	3	3	4	4	2	1	2	1	2	I
15	4	2	2	3	1	1	4	3	4	2	3	E
16	3	2	3	3	2	2	3	1	1	1	1	I
17	5	2	3	3	2	2	3	1	1	1	1	I
18	5	2	3	3	4	4	4	1	1	1	1	I
19	4	2	3	1	4	3	4	1	4	1	3	I
20	4	2	3	3	1	4	2	2	4	1	4	E
21	5	2	3	2	4	4	2	1	3	1	2	I
22	5	2	2	3	3	4	5	3	4	2	4	E
23	4	2	3	3	2	3	4	3	3	2	4	E

data. This data set includs 23 suppliers with ten attributes (shown in Table 2).These attributes include quality management practices and systems(QMP), documentation and self-audit(SA), process/manufacturing capability(PMC), management of firm(MGT), design and development capabilities(DD), cost(C), quality(Q), price(P), delivery(D), cost reduction performance(CRP) and others. The efficiency of each supplier is shown in the last column of Table 2. All suppliers with 'E' equal to one are considered efficient and 'I' means inefficient. Then, we will apply the FCM-ACO algorithm to clustering the dataset and the efficiency is used to measure the purity of the clusters later.

We apply the FCM-ACO on this dataset with the number of cluster set from 2 to 5.The results confirm that the purity of each cluster consistently increases with the number of clusters increases. We list the results of tests based on cluster number of 3 and 4 in Table3.

Table 3. Results of FCM-ACO on discretized data set

3 Clusters			**4 Clusters**		
Cluster number	Object number	Effi	Cluster number	Object number	Effi
1	3	E	1	6	E
1	6	E	1	7	E
1	7	E	1	11	I
1	11	I	1	12	E
1	12	E	2	4	E
2	4	E	2	15	E
2	15	E	2	20	E
2	22	E	2	22	E
2	23	E	2	23	E
3	1	I	3	1	I
3	2	E	3	2	E
3	5	I	3	3	E
3	8	I	3	10	E
3	9	I	4	5	I
3	10	E	4	8	I
3	13	I	4	9	I
3	14	I	4	13	I
3	16	I	4	14	I
3	17	I	4	16	I
3	18	I	4	17	I
3	19	I	4	18	I
3	20	E	4	19	I
3	21	I	4	21	I

For three clusters, the purity of Cluster 1 is 80% (4/5), Cluster 2 is 100% (4/4) and Cluster 3 is 78.6% (11/14). And for four clusters, the purity of Clusters 1, 2, 3, and 4 is 75% (3/4), 100 %(5/5), 75 %(3/4), and 100% (10/10), respectively. The performance tests show that the FCM-ACO algorithm is effective in clustering suppliers during the supplier base management.

6 Conclusion

Nowadays, managing a massive supplier base is a challenging task. It is necessary to cluster suppliers into manageable smaller subsets. In this paper, in order to overcome the shortcomings of the fuzzy c-means we present an algorithm named FCM-ACO which combined Ant Colony Optimization with FCM. Experimental results based on five well known data sets show that the proposed hybrid method is efficient and can reveal very encouraging results that the new method can escape from local optima. Therefore, this method is proved to be feasible and effective. So far, we did not have access to real-world huge amounts of suppliers' data. We are currently working on improving the algorithm and developing the corresponding tools. After that we will negotiate with different companies to apply our clustering tools on real data.

References

1. Esmaeil, M.: A fuzzy clustering PSO algorithm for supplier base management. International Journal of Management Science and Engineering Management 4, 311–320 (2009)
2. Prahinski, C., Benton, W.C.: Supplier evaluations: communication strategies to improve supplier performance. Journal of Operations Management 22(1), 39–62 (2004)
3. Wagner, S.M., Johnson, J.L.: Configuring and managing strategic supplier portfolios. Industrial Marketing Management 33(8), 717–730 (2004)
4. Parmar, D., Wu, T., et al.: A clustering algorithm for supplier base management, `http://citeseerx.ist.psu.edu/viewdoc/download;?doi=10.1.1.93.4585&rep=rep1&type=pdf`
5. Keinprasit, R., Chongstitvatana, P.: High-level synthesis by dynamic ant. International Journal of Intelligent Systems 19, 25–38 (2004)
6. Bezdek, J.C.: Pattern recognition with fuzzy objective function algorithms. Kluwer Academic Publishers, Norwell (1981)
7. Thomas, A., Runkler, T.: Wasp Swarm Optimization of the c-Means Clustering Model. International Journal of Intelligent Systems 23, 269–285 (2008)
8. Bezdek, J., Hathaway, R.: Optimization of fuzzy clustering criteria using genetic algorithms. In: Proceedings of the IEEE Conference on Evolutionary Computation, vol. 2, pp. 589–594 (1994)
9. Klawonn, F., Keller, A.: Fuzzy clustering with evolutionary algorithms. International Journal of Intelligent Systems 13, 975–991 (1998)
10. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. of the IEEE Int. Joint Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
11. Dorigo, M., Di Caro, G., Gambardella, L.M.: Ant Algorithms for Discrete Optimization. Artificial Life 5(2), 137–172 (1999)
12. Colorni, A., Dorigo, M., Maniezzo, V.: Distributed optimization by ant colonies. In: Proceedings of the First European Conference on Artificial Life, pp. 134–142. Elsevier Publishing, Amsterdam (1991)
13. Handl, J., Knowles, J., Dorigo, M.: Strategies for the increased robustness of ant-based clustering. In: Di Marzo Serugendo, G., Karageorgos, A., Rana, O.F., Zambonelli, F. (eds.) ESOA 2003. LNCS (LNAI), vol. 2977, pp. 90–104. Springer, Heidelberg (2004)
14. Runkler, T.: Ant colony optimization of clustering models. International Journal of Intelligent Systems 20, 1233–1261 (2005)
15. Klir, G., Yuan, B.: Fuzzy sets and Fuzzy logic, theory and applications. Prentice-Hall, Englewood Cliffs (2003)
16. The UCI Website, `http://archive.ics.uci.edu/ml/datasets.html`

Nearest Neighbour Distance Matrix Classification

Mohd Shamrie Sainin[1,2] and Rayner Alfred[1]

[1] School of Engineering and Information Technology
Universiti Malaysia Sabah, Locked Bag 2073,
Kota Kinabalu, Sabah, Malaysia
[2] On Study Leave from Universiti Utara Malaysia
Department of Computer Science,
College of Arts and Sciences
Sintok, Kedah, Malaysia
shamrie@uum.edu.my, ralfred@ums.edu.my

Abstract. A distance based classification is one of the popular methods for classifying instances using a point-to-point distance based on the nearest neighbour or *k*-NEAREST NEIGHBOUR (*k*-NN). The representation of distance measure can be one of the various measures available (e.g. Euclidean distance, Manhattan distance, Mahalanobis distance or other specific distance measures). In this paper, we propose a modified nearest neighbour method called Nearest Neighbour Distance Matrix (NNDM) for classification based on unsupervised and supervised distance matrix. In the proposed NNDM method, an Euclidean distance method coupled with a distance loss function is used to create a distance matrix. In our approach, distances of each instance to the rest of the training instances data will be used to create the training distance matrix (TADM). Then, the TADM will be used to classify a new instance. In supervised NNDM, two instances that belong to different classes will be pushed apart from each other. This is to ensure that the instances that are located next to each other belong to the same class. Based on the experimental results, we found that the trained distance matrix yields reasonable performance in classification.

Keywords: data mining; machine learning; nearest neighbour; distance matrix; classification.

1 Introduction

The nearest neighbour method in instance-based learning is a traditional and simplest method for nonparametric pattern classification [1]. It simply stores all examples as training data and classification is postponed until a query is presented. All instances in the examples will be examined and a set of similar instances are presented to classify the query. Therefore this traditional pattern classification has no model for approximation but many local approximations depending on the target function for each new query.

The idea of instance-based learning is further explored for its possibility of creating new variation of learning. The *k*-NN rule which classifies an instance based on its

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 114–124, 2010.

nearest or majority instance open the new idea of utilizing the distances of all instances in the training example as a matrix of distances. The distance measure such as Euclidean distance is still performing well despite ignoring the statistical regularities. Therefore, by applying this distance calculation combined with the loss function as proposed by Weinberger and Saul [2], one may increase the performance accuracy of a classification task. Our implementation of the loss function simply pushes instances with different label in the search space in order to group instances that belong to the same target class.

This paper is organized as follows. Section 2 discusses the general distance metric with *k*-NN classification and reviews some of the related works. Section 3 describes the proposed method, called the Nearest Neighbour Distance Matrix (NNDM) method, for classification algorithm formulation. The experimental results for the algorithm are discussed in section 4 and finally Section 5 concludes this paper.

2 Background

The *k*-NN is the most basic example of nearest neighbour method where all instances are viewed as an array of *n*-dimensional space and their distances are defined using a standard measure such as an Euclidean distance. Each feature that describes the instances will be considered as part of the elements used for calculating the distances and thus used for classifying the new query. This means that *k*-NN is heavily dependent on how the distance is measured and represented. Furthermore, *k*-NN classification works by taking the majority voting or labels of its *k*-nearest neighbours in the examples set. Although this method is simple and yet has high classification cost, some researches reported that this method can produce good results when combined with prior knowledge [3-5].

Although the standard Euclidean distance computation ignores any estimation of statistical regularities from the training set, researchers were working to improve the *k*-NN classification by using the distance metric from labelled examples [2, 6-9]. This shows that Euclidean distance still plays an important role in the distance based classification tasks.

The Mahalanobis distance was first introduced by Mahalanobis in 1939 [10]. It is a distance measurement based on correlation of variables in random vectors in non spherical symmetric distribution. It takes into account of the mean, standard deviation, and covariance of each group in the sample set as defined in the following squared distance metric:

$$D_M(\vec{x}_i, \vec{y}_j) = \sqrt{(\vec{x}_i - \vec{y}_j)\, S^{-1} (\vec{x}_i - \vec{y}_j)^T} \;. \qquad (1)$$

where $\vec{x}$ and $\vec{y}$ are random vectors of the same distribution with S^{-1} is the covariance matrix of the group in the distribution.

Meanwhile, apart from the similarity matrix of $w_{ij}=sim(x_i,x_j)$ which is also known as an affinity matrix, the approach introduced by Bai implements two important contributions to their method, a probabilistic transition matrix from w_{ij} (Eq. 2) and a recursive procedure (Eq. 3) for learning a new similarity function as $sim_t(x_1, x_i) = f_t(x_i)$.

$$P_{ij} = \frac{w_{ij}}{\sum_{k=1}^{n} w_{ik}} . \tag{2}$$

$$f_{t+1}(x_i) = \sum_{j=1}^{n} P_{ij} f_t(x_j) \text{ for } i=2,..,n \text{ and } f_{t+1}(x_1) = 1 . \tag{3}$$

The Mahalanobis distance is a statistical estimation derived from the training set, while the implementation of the distance, proposed by Bai, depends on the problems (e.g. shortest paths inside the shapes [11]) and it is converted into a similarity measure using Gaussian Kernel for constructing the affinity matrix.

Inspired by these works, we formulate our distance-based classification method from *k*-NN as the base method which is called as the Nearest Neighbour Distance Matrix (NNDM). Our approach is largely driven by the conceptual of Mahalonobis distance [2] and graph transduction distance learning [12]. Although our proposed method shares a similar direction of the works by the two approaches, however the proposed method differs in term of the distance measure complexity where we exploit the standard Euclidean distance without applying the probabilistic transition matrix and recursive similarity function learning to create the distance matrix. We drop the probabilistic transition matrix (Eq. 2) and the recursive $f_t(x_i)$ (Eq. 3) to minimize the computation process during the classification phase. We implement a push function (during training phase) to modify the similarity matrix w_{ij}, in order to separate instances with target label from the instances with the non-target label. Our proposed method computes less complex distance measure and yet, the proposed method is able to produce reasonable results.

3 Nearest Neighbour Distance Matrix (NNDM)

The nearest neighbour distance matrix (NNDM) is a matrix of distances for each training data in the dataset. The NNDM method starts by computing the distances between every pair of items in the dataset and stores them in an *n*x*n* matrix, called Training Distance Matrix (TADM). This distance matrix is created prior to classifying a new query to speed up the process during the classification task. The proposed NNDM method is similar to the *k*-NN, where distance measure is used to classify the query instance. However, in the NNDM method, the classification of the new query instance is performed based on the distance matrix by first calculating the distance of the query instance from the training instances (creating testing distance matrix or TEDM) and the classification will be voted by the distance matrix obtained from training (TADM). The *k* in NNDM refers to the *k*-nearest instances obtained from TADM.

Based on the similarity function defined as $sim:(xi,xj) \rightarrow d(x_i,x_j)$, the Training Distance Matrix (TADM) is created during training phase. The TADM is then sorted in ascending order in which the smallest distance measure is listed first. In testing phase, the process begins by loading the TADM along with the testing and training data. Then, the distances between the query instances in the testing data with the training instances will be calculated and sorted in an ascending order as Testing Distance Matrix (TEDM). The first item in TEDM is the smallest distance of the query with certain item in the training data.

In order to classify a query or set of queries, the first item in TEDM will be referenced to TADM for classification based on 1NN or *k*-NN. This method of classification will depend on TADM instead of TEDM, where the nearest neighbour in TADM has more influence to the classification compared to the nearest neighbour of the query in TEDM. At this point, our algorithm for classification using NNDM is still not considered efficient. In term of algorithm complexity, we have four important or significant processes in NNDM classification which are calculate distances of *xq* to each training data, sorting, indexing TEDM and classification. Therefore, for examples with *n* examples, *d* features and *k* nearest neighbour, the overall classification time complexity is at least $O(dn^2 \log n)$ for creating the *nxn* distance matrix, $O(n \log n)$ for sorting the distance matrix, $O(n^2)$ for indexing the distance matrix, and $O(k)$ for classification using *k* nearest neighbour from TADM.

The flow diagram of basic training and classification phases in NNDM is shown in Fig.1. This technique can be further divided into unsupervised NNDM and supervised NNDM.

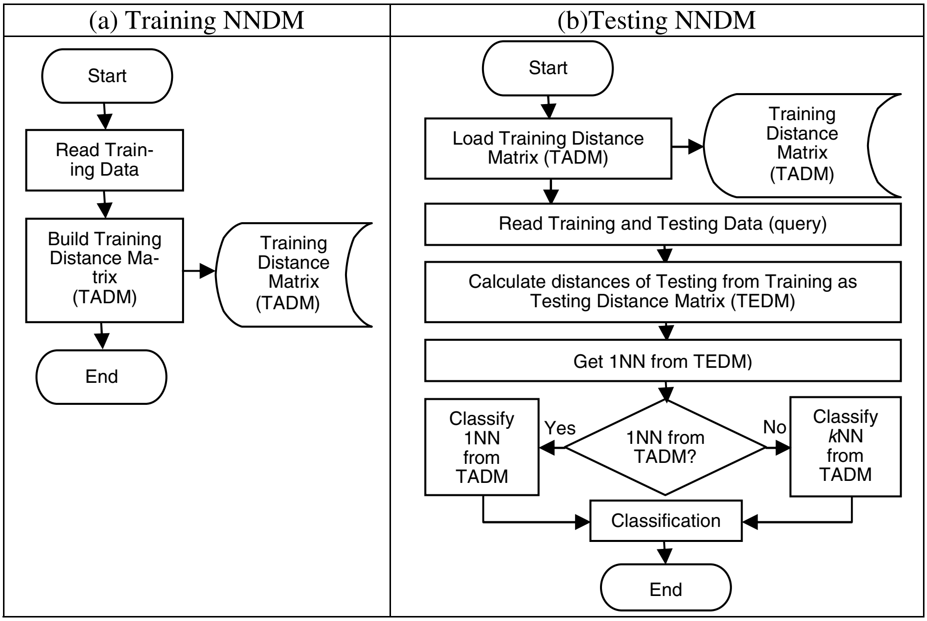

(a) Training NNDM
(b)Testing NNDM
Start
Read Training Data
Build Training Distance Matrix (TADM)
Training Distance Matrix (TADM)
End
Start
Load Training Distance Matrix (TADM)
Training Distance Matrix (TADM)
Read Training and Testing Data (query)
Calculate distances of Testing from Training as Testing Distance Matrix (TEDM)
Get 1NN from TEDM)
Classify 1NN from TADM
Yes
1NN from TADM?
No
Classify kNN from TADM
Classification
End

Fig. 1. Basic process of NNDM. (a) Training phase to create Training Distance Matrix (TADM). (b) Testing phase to classify a query or queries in testing data using 1NN or *k*-NN from TADM.

3.1 Unsupervised NNDM

In the unsupervised NNDM method, the class label will be considered in constructing the distance matrix prior to the classification task. The distance values are purely calculated based on the standard Euclidean Distance of two instances. The pseudo-code for this method is shown in Fig. 2.

```
Training
      Define training NNDM array, $M$
      Iterate Training, $x_i$, $x_j$
            Build matrix $M = d(x_i, x_j)$
      End
      Sort $M$
      Index $M$ as INDEX(TEDM)
End
```

Fig. 2. The pseudo-code for the unsupervised NNDM

3.2 Supervised NNDM

On the other hand, in the supervised NNDM method, the class label of the data will be taken into consideration in constructing the distance matrix. The distance matrix is used to illustrate the distance between the query's instance and the existing instances in the dataset. Besides having the training instances, the supervised NNDM method will also group similar instances together and a distance loss function will be applied. A penalty (distance loss function) is imposed when a training instance, x_j, belongs to a different class label with another instance, x_i, where a push function, $\varepsilon_{push}(d)$ shown in equation 5, will push this particular instance x_j far from the instance x_i. Therefore, instances are located nearer to each other when they belong to the same class, while non class instances will be placed far away in the NNDM space. The loss function terminology in this method is similar to the idea presented by Weinberger and Saul [2]. However, we remove the pull term function in our approach. The process starts by finding the maximum distance, $d_{max}(f(x_i))$, between two instances that belong to the same class as shown in equation 4.

$$d_{\max}(f(x_i)) = \max(d(x_i, x_j)) \ . \tag{4}$$

$$\varepsilon_{push}(d) = d_{\max}(f(x_i)) + d(x_i, x_j) \ . \tag{5}$$

Equation 5 illustrates the push function where this function is applied to the current distance between two instances, x_i and x_j, that belong to different classes. The aim is to separate non-class members from the query instance in the nearest neighbour distance matrix. The resulting matrix will consist of short distances for all instances that belong to the same target class. The illustration of the distance loss function applied in the supervised NNDM is depicted in Fig. 3.

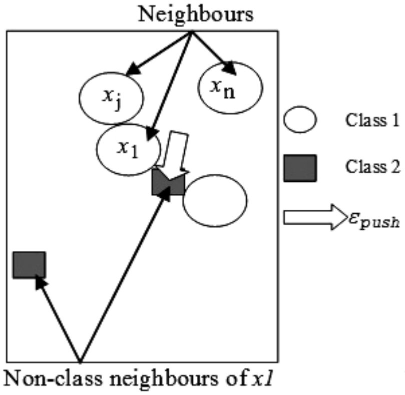


Fig. 3. Illustrations of push function before training (left) and after training (right). The non-member of the current class will be pushed using Equation 5, if the distance is less than the *dmax(f(xi))*.

The pseudo-code for the supervised NNDM is shown in Fig. 4, where the class labels of both training x_i and x_j will be examined. If both instances have different classes membership, then the push function will be applied, otherwise the push function will not be applied in computing the distances. The distances of each training instance against all other instances will produce a distance matrix *M*.

```
Function Training
  Define training NNDM array, $M$
  Iterate Training, $x_i$, $x_j$
      If $c(x_i \ != x_j)$ $D = max(f(x_i))+d(x_i,x_j)$
      Else $D = d(x_i,x_j)$
      Build matrix $M = D$
  End
  Sort and Index $M$ as INDEX(TEDM)
End
```

Fig. 4. The pseudo-code for the supervised NNDM

3.3 Classification Process

Given a new instance, x_q, to be classified, the distances between the new instance x_q to the other training instances will be calculated and stored in a Testing Distance Matrix (TEDM).

Then, by sorting the values of *sim*(x_q, x_i) in increasing order. The first instance listed in TEDM is the instance with the minimum distance to the new instance and this instance is considered as having the most similar characteristics when compared to the new instance. Using the 1-Nearest Neighbour (1NN) obtained TEDM, the classification will point to the nearest instance from the training NNDM. For *k*-NN, *k* nearest instances from TADM will be considered for classification.

The main difference between the proposed NNDM and the learning distances proposed by Bai [12] is that there is no probabilistic transition matrix and the recursive distance learning applied in the proposed NNDM, in which the time taken to classify an instance can be reduced, especially if a large dataset with a large number of attributes are presented to be computed. The pseudo code for classification method using the NNDM is depicted in Fig. 5

```
Function Classification
  Load TADM, Define k
  Iterate Testing instances, xq
  For xi to $x_n$ in Training, xq
    Calculate sim($xq_i$,$x_i$)
  End
  Sort sim(xq,$x_i$),Index sim as INDEX(TEDM)
  Get first item on INDEX(TEDM)
  For i=1,..n of INDEX(TADM)
    Get n most similar instances from INDEX(TADM)
  End
  If k=1
    If train.class = xq.class correct++;
  Else if k>1
    Iterate k
       Count train.class = xq.class
       Specify majority class
  End
  Calculate Rate
End
```

Fig. 5. The pseudo-code for classification phase in NNDM

4 Experimental Results

This section demonstrates the performance of our approach for various datasets obtained from UCI Machine Learning Repository, UCI Time Series and other sources of dataset. The datasets for our experiments consist of Wisconsin Breast Cancer Database, Image Segmentation, Pen-based Recognition of Handwritten Digits, Wine recognition, Diabetes and Pima Indians Diabetes Database from UCI Repository of Machine Learning Databases and Domain Theories [13], Swedish Leaf and Face (All) Database from UCR Time Series Classification/Clustering Page [14], and Leaf Image Data [15]. Table 1 shows the description of these datasets.

The comparison of performance for the proposed NNDM with standard Mahalanobis distance (using Matlab function), the method applied in Bai and Weka kNN are investigated. In order to compare the NNDM and Bai's method, we obtained the sample Matlab file provided by Bai and reprogrammed them into Matlab and Java format.

Table 1. Dataset description

Dataset	Num. of Attribute (inc.Class)	Training Size	Testing Size	Num. of Class	Instance per Class
Swedish	129	375	750	15	25
Breast Cancer	31	273	410	2	136
Face All	132	560	1690	14	40
Letter	17	676	1046	26	26
Segmentation	20	924	1386	7	132
Pen Digit	17	300	700	10	30
Wine	14	71	107	3	23.67
Diabetes	9	576	192	2	288
Leaf Data	142	111	50	14	7.9

Table 2 indicates that the overall performance of the proposed unsupervised NNDM with 1NNDM does not produce better results compared to the *k*-NN algorithm implemented in the Weka System (using 1NN). The proposed supervised NNDM based on the 1NNDM and *k*-NNDM however has improved the classification performance especially for datasets with many attributes. The performance of the supervised NNDM for the datasets with less attributes remains lower than the results produced by Weka 1NN (except for the dataset Letter and Pen Digits).

Table 2. Comparison of Unsupervised(U) 1NNDM, Supervised(S) 1NNDM, Unsupervised *k*-NNDM, Supervised *k*-NNDM, Weka 1NN, Matlab Mahalanobis function (*mahald(Y, X)*) and Bai's distance learning (Bai) as unsupervised 1NN

Data	Num Att	U 1NNDM	S 1NNDM	U *k*-NNDM (k)	*S* *k*-NNDM (k)	Weka	mahald	Bai
Swedish	129	85.47	92.67	85.47 (1)	92.67 (1)	75.87	*Na*	92.67
Breast Cancer	10	95.37	97.32	98.29 (3)	97.32 (1)	97.32	86.59	97.32
Face All	132	75.03	78.28	75.03 (1)	78.28 (1)	68.76	*na*	78.28
Letter	17	61.57	63.57	61.76 (8)	63.57 (1)	46.46	*na*	63.57
Segmentation	20	91.13	95.17	91.13 (1)	95.02 (1)	96.32	*na*	95.02
Pen Digits	17	85.71	89.57	86.71 (3)	89.71 (1)	89.57	76.43	89.71
Wine	14	61.68	69.16	73.83 (28)	71.02 (37)	97.20	94.39	69.16
Diabetes	9	66.14	65.10	76.56 (34)	65.10 (1)	68.23	70.31	65.10
DataLeaf	142	52.00	74.00	52.00 (1)	74.00 (1)	28.00	*na*	74.00

There are three findings obtained from the results. First, it can be seen that the proposed method is somehow good for data with more attributes as listed in the Swedish, Face (All) and Data Leaf datasets. The datasets with less attributes show that 1NNDM may not be good. Next, despite the low performance of the proposed method for datasets with less attributes, unsupervised *k*-NNDM slightly improves the classification accuracy in the Breast Cancer, Pen Digits, Wine and Diabetes datasets.

The results of the supervised 1NNDM and *k*NNDM are comparable due to the application of the loss function in the supervised *k*NNDM that causes similar neighbours to be grouped together so that it is feasible to use any values for *k* up to the number of training instances per class. Fig. 6 shows the classification rates (Swedish dataset) for the unsupervised *k*NNDM and the supervised *k*NNDM with *k* (1 to 50). It shows that the supervised *k*NNDM retrieval rates are quite consistent as the value of *k* neighbours increases, while the performance accuracy of the unsupervised *k*NNDM drops as the number of *k* neighbours increases. This indicates that the proposed NNDM method can be used to retrieve any *k* number of similar instances up to the number of sample per class, when the loss function is applied. The results of the experiment which are obtained by applying the `mahal(Y,X)`function in Matlab show lower performance accuracy than the expected values, except for the diabetes dataset. We believe that the lower performance accuracy results obtained are due to the fact that the testing size is more than the training size for most of the data. Our own Mahalanobis distance program (Java) also produces similar results as Matlab, where *na* indicates the error of "*row must be more than number of column (features)*".

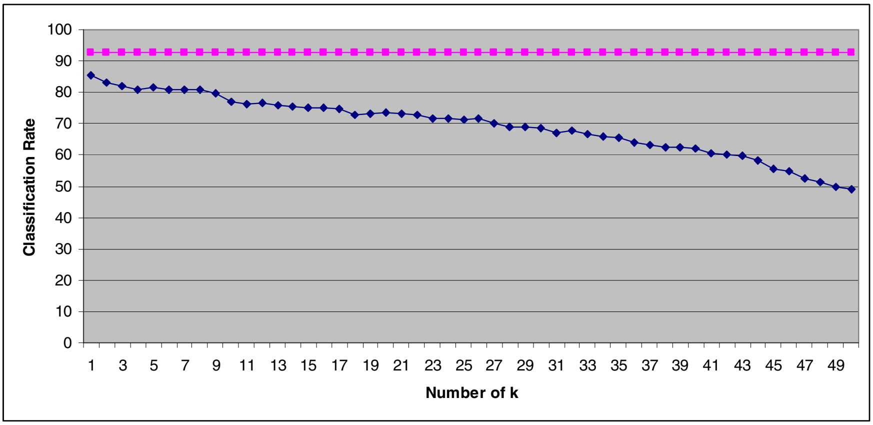


Fig. 6. Retrieval accuracy (Swedish dataset) of unsupervised *k*NNDM(diamond dots) and supervised *k*NNDM (square dots)

Finally, we have also followed exactly the implementation of the distance learning proposed by Bai and rewrite the Matlab code to suit our dataset and the classification task. The distance measure is simply using the Euclidean distance and tested with unsupervised NNDM method. We found that the method produces better results than the proposed unsupervised NNDM but similar to our supervised NNDM. Equation 3 proves that the class or label prediction is propagated along the *t* iteration.

Although we have implemented the exact method proposed by Bai [12], we get lower retrieval rates compared to the reported results for the Swedish and Face All datasets. This is due to different experimental settings used in the experiments. The Swedish leaf dataset which is our main concern in this paper shows that with our supervised NNDM, we manage to achieve a promising result (92.67%). The time required to train (produce the affinity matrix/NNDM) and testing is also much faster than

Table 3. Training and classification time of NNDM for each data set (using 1NN)

Dataset	NNDM	Bai's Method
Swedish	4s	81s
Breast Cancer	0s	42s
Face All	16s	188s
Letter	6s	112s
Segmentation	16s	154s
Pen Digit	1s	73s
Wine	0s	11s
Diabetes	2s	21s
Leaf Data	0s	5s

Bai's method. Table 3 shows the overall time taken (in second) to train and classify the data (implemented in Java, using Core2 1.8GHz and 3GB of RAM Computer).

5 Conclusion

The supervised and unsupervised NNDMs are considered as new variation of the distance based classification algorithms that utilize the nearest neighbour algorithm to build a classifier model (nearest neighbour distance matrix) before classifying new unseen instances. The result of the proposed NNDM shows that this approach can perform well in some datasets but may not perform well in other datasets. This paper introduces the distance based classification using unsupervised or supervised Nearest Neighbour Distance Matrix (NNDM). Our initial objective was to develop and test the simplified distance based classifier that utilizes distance matrix prior to classifying new instances. The experiment results show that the performance of the proposed method is comparable to the more complex methods discussed earlier.

Based on the NNDM method, it can be applied for various distance-based classification tasks which require similar retrieval for any *k*-nearest neighbour. However, if the data has high probability of noise and outliers, our method may not perform. Our feature work will focus on investigating this problem and application to large data sets because our current method is not designed to work seamlessly with very large data sets. Thus, to be able to work with large data sets and many attributes, feature selection optimization is another one of the interesting challenges for our method to improve the classification.

References

1. Cover, T.M., Hart, P.E.: Nearest Neighbor Pattern Classification. IEEE Transactions in Information Theory IT-13, 21–27 (1967)
2. Weinberger, K.Q., Saul, L.K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification. Journal of Machine Learning Research 10, 207–244 (2009)
3. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape context. IEEE Trans. PAMI 24, 705–722 (2002)

4. Simard, P.Y., LeChun, Y., Decker, J.: Efficient pattern Recognition using a new transformation distance. In: Hanson, S., Cowan, J., Giles, L. (eds.) Advances in Neural Information Processing System, vol. 6, pp. 50–58. Morgan Kaufman, San Mateo (1993)
5. Zhang, B., Srihari, S.N.: Fast k-Nearest Neighbor Classification Using Cluster-Based Trees. IEEE Transaction on Pattern Analysis and Machine Intelligence 26, 525–528 (2004)
6. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005), pp. 349–356 (2005)
7. Goldberger, J., Roweis, S., Hinton, G., Salakhutdinov, R.: Neighbourhood components analysis. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems 17, pp. 513–520. MIT Press, Cambridge (2005)
8. Shalev-Shwartz, S., Singer, Y., Ng, A.Y.: Online and batch learning of pseudo-metrics. In: Proceedings of the Twenty First International Conference on Machine Learning (ICML 2004), pp. 94–101 (2004)
9. Shental, N., Hertz, T., Weinshall, D., Pavel, M.: Adjustment learning and relevant component analysis. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 776–790. Springer, Heidelberg (2002)
10. Mahalanobis, P.C.: On the generalised distance in statistics. Proceedings of the National Institute of Sciences of India 2, 49–55 (1939)
11. Ling, H., Jacobs, D.W.: Shape Classification Using the Inner-Distance. IEEE Transactions on Pattern Analysis and Machine Intelligence 29, 286–299 (2007)
12. Bai, X., Yang, X., Latecki, L.J., Liu, W., Tu, Z.: Learning Context Sensitive Shape Similarity by Graph Transduction. IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI) 31 (2009)
13. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
14. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR Time Series Classification/Clustering Homepage (2006), `http://www.cs.ucr.edu/~eamonn/time_series_data/`
15. Agarwal, G., Ling, H., Jacobs, D., Shirdhonkar, S., Kress, W.J., Russell, R., Belhumeur, P., Dixit, A., Feiner, S., Mahajan, D., Sunkavalli, K., Ramamoorthi, R., White, S.: First steps toward an electronic field guide for plants. Taxon 55, 597–610 (2006)

Classification Inductive Rule Learning with Negated Features

Stephanie Chua, Frans Coenen, and Grant Malcolm

Department of Computer Science,
University of Liverpool, Ashton Building,
Ashton Street, L69 3BX Liverpool, UK
{s.chua,coenen,grant}@liverpool.ac.uk

Abstract. This paper reports on an investigation to compare a number of strategies to include negated features within the process of Inductive Rule Learning (IRL). The emphasis is on generating the negation of features while rules are being "learnt"; rather than including (or deriving) the negation of all features as part of the input. Eight different strategies are considered based on the manipulation of three feature sub-spaces. Comparisons are also made with Associative Rule Learning (ARL) in the context of multi-class text classification. The results indicate that the option to include negated features within the IRL process produces more effective classifiers.

Keywords: Rule Learning, Negation, Multi-class Text Classification.

1 Introduction

The generation of rule-based classifiers is a popular data mining method which is applicable to many data formats. Its popularity is largely due to its *transparency*, the ease with which classifications can be explained, and its relative simplicity (compared to other types of classifiers such as support vector machines or neural networks).

Rules in rule-based classifiers are typically represented in the form $X \Rightarrow k$, where X is some subset of the global set of features (A) present in the input data, and k is a class label. For example, we might have a rule $a \wedge b \Rightarrow k$ (where a and b are two features taken from A). The rule is interpreted as, "if a and b exist in a record, then classify the record as belonging to class k". Examples of rule-based classifiers include Associative Rule Learning (ARL) systems such as Classification based on Multiple class-Association Rules (CMAR) [12], Classification based on Predictive Association Rules (CPAR) [17], or Total From Partial Classification (TFPC) [5] and Inductive Rule Learning (IRL) systems such as First Order Inductive Learner (FOIL) [13] and Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [6].

Many rule-based classifiers such as CPAR, CMAR, TFPC and FOIL do not provide the option to include rules with negated features of the form: $a \wedge b \wedge \neg c \Rightarrow k$. Note that such a rule would be interpreted as, "if a and b exist in a record and

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 125–136, 2010.

c does not, then classify the record as belonging to class k". There are only a few examples of rule-based classifier generation systems which generate rules with negation. One such system is the Olex system [14]. However, this system uses a very restrictive rule template (one positive feature and zero or more negative features) for rule generation and that significantly restricts the nature of the rules that can be generated.

Of course, negation can be included explicitly in the input data by including a negated feature for every feature that exists (for example, a and $\neg a$). It is also possible to generate negated versions for all features automatically as part of the classifier generation process. For some applications, this is a realistic option. However, for many applications, this is not realistic because it doubles the number of features to be considered. One example is in the field of text classification where documents are encoded using the *bag-of-words* representation. Using this representation, the feature set comprises several thousand words and thus, doubling the number of features would introduce an additional computational overhead (the computational complexity of rule-based classifier generators increases exponentially with the number of features).

The approach advocated in this paper is to include negated features within the rule generation process without explicitly encoding such negations in the input or generating all possible negations. In this case, there are two issues to be addressed: (i) the process of identifying features whose negation and inclusion in a rule under consideration during the rule generation process can benefit the final result (the accuracy of the classifier), and (ii) the rule refinement strategies to dictate whether a positive feature or a negative feature should be included in a rule. In this paper, a number of strategies are proposed and evaluated for addressing both these issues. The focus of this work is on multi-class text classification using the bag-of-words representation. However, the reported investigation is equally applicable to other forms of classification.

The rest of this paper is organized as follows. Section 2 describes some previous reported work concerning rule learning with and without negation. Section 3 discusses the adopted rule-based text classification model, followed by the proposed inductive rule learning algorithm in Section 4. The experimental setup is presented in Section 5. The results and an analysis of the experiments are discussed in Section 6. Section 7 concludes this paper with a brief discussion of future work.

2 Previous Work

Many machine learning methods for text classification have been proposed. Inductive Rule Learning (IRL) methods tend to be based on the covering algorithm. In the covering algorithm, rules are learned one at a time based on the training examples given. The examples "covered" by a rule are then removed and the process is repeated until a stopping condition is met. Examples of IRL systems based on the covering algorithm include: (i) Incremental Reduced Error Pruning (IREP) [7], (ii) Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [6] and (iii) Swap-1 [16,3]. An example of the use of negation in IRL can be found in the Olex system [14] which uses positive and negative features to generate rules using a template of one positive feature and none or more negative feature(s). However, the use of this template is very restrictive in that no two positive features can co-occur together in a rule.

ARL algorithms are based on the concept of Association Rule Mining (ARM) as first proposed by [1]. ARM operates by generating the set of relationships between features that exist in a given data set. These relations are expressed as probabilistic rules, called Association Rules (ARs). ARs are of the form $X \Rightarrow Y$ (where X and Y are disjoint subsets of the the global set of features A). ARs are interpreted as "if X exists in a record then it is likely that Y will also exist". The set of classifcation rules that exist in a data set are a subset of the set of ARs. A number of ARL systems were identified in Section 1. Negation has been used in previous work in ARL. For example, Antonie and Zaïane [2] proposed an algorithm that discovers negative ARs (one example being, "if X exists in a record, then it is likely that Y will NOT exist". Baralis and Garza [4] constructed an associative classifier that used rules with negated words. They reported that the use of negated words improved the quality of their classifiers. A criticism of the ARL approach is that a great many rules are generated, typically many more than in the case of IRL systems such as the approach proposed in this paper. However, to evaluate the IRL approach proposed in this paper, comparisons are made with the ARL technique using the TFPC algorithm [5] because we are interested in interpretable rules as a classifier and also because our experiments here focus on multi-class classification instead of binary classification. Therefore, we will not compare with the support vector machine (SVM) [9] method (reported to be one of the best method for text classification).

3 Rule-Based Text Classification Model

A general model for a rule-based text classification consists of a number of processes (Figure 1). These processes include preprocessing, text representation and rule learning. To evaluate the generated classifier, the input data (document base) is usually split into a training set and a test set. The first step is to preprocess the data. There are many sub-processes that may be applied at this stage, such as stop word removal, stemming and feature selection. After preprocessing has been completed, the documents in their preprocessed form are translated into some appropriate representation suitable for the application of text classification algorithms. A popular method for text representation is the *bag-of-words* representation. In this representation, a document is represented in an N-dimensional vector space, where N is the number of features. The value of each feature in the vector space can take either a Boolean value to indicate the presence or absence of a word in a document or a numeric value to indicate its frequency in a document. Note that when using the bag-of-words representation, information regarding the order and position of the words is lost.

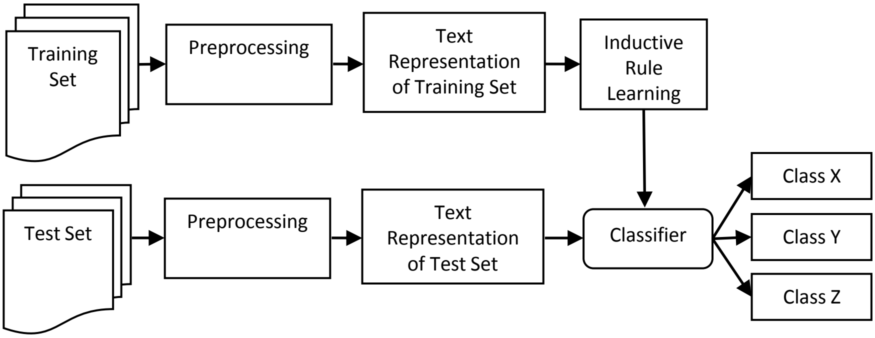


Fig. 1. Rule-based text classification model

Once the appropriate representation has been generated, the IRL process can be applied. During this process, the desired set of classifcation rules are learned, resulting in a classifier which may be applied to "unseen" data.

4 Inductive Rule Learning with Negation

The algorithm for our inductive rule learner is presented in Table 1. Our inductive rule learner is founded on the sequential covering algorithm. For a given class k, rules are learned sequentially one at a time based on training examples using the *LearnOneRule* method (Table 2). The examples "covered" by a rule learnt are then removed and the process is repeated until some stopping condition is met. Stopping conditions are: (i) when there are no more uncovered documents or (ii) when there are no more unused features in the feature set. Each rule generated is the ruleset *so far*. Post-processing is done on the final ruleset using the *PostProcess* method to remove rules with a rule accuracy, defined in our case using the Laplace estimation accuracy, lower than a user pre-defined threshold.

Rules are generated using the *specialization* approach, whereby a rule is made more specific by adding features to its *condition*. In the *LearnOneRule* method, rule learning starts with an empty *condition* and a class c in the *conclusion*. The top most unused feature from the $Feature\_set$ is added to the empty rule and the rule is checked using the *CheckRule* method (Table 3). In the *CheckRule* method, the rule "coverage" is checked, i.e. the number of positive and negative documents that it covers. If a rule covers negative document(s) and it can be further refined, then the *RefineRule* method is called. There are a number of different strategies for refining rules presented in this paper. These strategies for refinement are what make our IRL system different from previously proposed systems. While previous methods used pruning (e.g.[7,6]) to improve their rule quality, our approach uses refinement strategies that are applied whilst the rule is being generated. Rules with and without negation are generated based on the usage of search space division, as described in Section 4.1.

Table 1. Algorithm for inductive rule learner (adapted from [8])

```
Algorithm: Learn a set of IF-THEN rules for classification.

Input:
D, a dataset of class-labelled documents;
Feature_set, the set of features for class c;

Output: A set of IF-THEN rules.

Method:
Rule_set = { }; //initial set of rules learned is empty

for each class c do
        repeat
                Rule =LearnOneRule(Feature_set, D, c);
                remove documents covered by Rule from D;
                Rule_set = Rule_set + Rule; //add new rule to ruleset
        until stopping condition;
endfor
PostProcess(Rule_set);
return Rule_set;
```

Table 2. Algorithm for LearnOneRule method

```
Algorithm: LearnOneRule.

Input:
D, a dataset of class-labelled documents;
Feature_set, the set of features for class c;
c, class label;

Output: Rule

Method:
Create a new empty rule, Rule
Set Rule condition to c
Add top most unused feature from Feature_set
Rule = CheckRule(Rule, D)
return Rule
```

The model was implemented with a view to multi-class classification, i.e. the generation of a rule-based classifier designed to classify unseen cases into one of N classes. Other classifiers include binary classifiers, which assign an unseen case as belonging to a class or not and classifiers that can assign more than one class to an unseen case. A sequence of binary classifiers is usually used to achieve multi-class classification in most previous work. In binary classification, the classifier has only rules from one class and these rules are used to classify documents as belonging to the particular class or not. In contrast, in multi-class classification, the classifier consists of rules from all the classes in the dataset. In order to determine which rule is to fire when classifying a document, the rules have to be ordered. The higher order rules will be fired before lower ones. In our experiments (see Section 5), the rules were first ordered according to descending rule length (the number of features in the rule *condition*) so that more specific rules would be fired first; and then, if the rules were of the same length, then the descending rule weight (the number of documents the rule covered in the learning phase) will be used for ordering. This meant that more specific rules with higher coverage would rank higher than less specific and lower coverage rules.

Table 3. Algorithm for CheckRule method

```
Algorithm: CheckRule

Input:
D, a dataset of class-labelled documents;
Rule, the rule to check;

Output: Rule

Method:
Check rule coverage


If Rule covers negative document(s)
        If Rule can be further refined
                Rule = RefineRule(Rule)
return Rule
```

4.1 Rule Refinement

During the proposed IRL process, the construction of each rule commences with an initial “start” rule comprising a single positive feature in its *condition* and an associated class in the *conclusion*. This single positive feature is selected from the top most unused feature in the feature set, ranked in descending order of the chi-square value of the features. If the rule *so far* covers both positive and negative documents, then the rule has to be refined in order to learn a rule that can separate the positive and negative documents. If the rule covers only positive documents, then it is added to the ruleset and the process continues with the generation of the next rule. During the refinement process, an appropriate feature is selected from the search space to add to the rule. The search space contains features from both the positive and negative documents that are covered by the rule. The search space can thus be divided into three sub-spaces that contain different kinds of feature:

- *Unique postive* (UP) features which are only found in positive documents.
- *Unique negative* (UN) features which are only found in negative documents.
- *Overlap* (Ov) features which are found in both positive and negative documents.

Such a division allows for effective and efficient identification of both positive features and features that can advantageously be negated. It should be noted that the UP, UN and Ov sub-spaces may be empty, as the existence of these features is dependent upon the content of the documents covered by a rule. When refining a rule, a feature from either the UP, UN and Ov feature sub-spaces can be selected to be added to the rule *so far*. If a rule is refined with a UP or Ov feature; then a new rule *so far*, with no negation, is generated. If a rule is refined with a UN feature, then the rule *so far* will include a negated feature. The Ov feature is added as a positive feature despite appearing in both positive and negative documents because negating an Ov feature in a rule will generate a rule that rejects positive documents.

Eight different rule refinement strategies were devised with respect to the three identified sub-spaces. These strategies are presented in Table 4. Note that strategies UP, Ov and BestPosRule generate rules without any negated features and have been included for comparison purposes. Strategies UN and UN-UP-Ov will always generate rules with negation provided the UN sub-space is not empty. Note that the UN strategy will result in rules comprising a positive feature and one or more negative features; thus a generating rules with a template identical to the Olex system [14] described in Section 2. Strategy UP-UN-Ov will generate rules without negation provided UP is not empty. The BestStrategy strategy will choose the best rule from the rules generated using the UP, UN, Ov, UP-UN-Ov and UN-UP-Ov strategies and thus may generate rules with negation. The BestRule strategy may also generate rules with negation, provided that when adding a UN feature, a better rule is generated than when adding a UP or Ov feature.

5 Experimental Setup

The data sets used were the 20 Newsgroups data set [10] and the Reuters-21578 data set [11]. The 20 Newsgroups data set consists of 20 classes and 19,997 documents, while the Reuters-21578 data set consists of 135 classes and 21,578 documents. In our preparation of the data sets, we adopted the approach of

Table 4. Rule refinement strategies

Strategy	Sub-space used	Description
UP	UP	Add a UP feature to refine a rule
UN	UN	Add a UN feature to refine a rule
Ov	Ov	Add an Ov feature to refine a rule
UP-UN-Ov	Either a UP, UN or Ov feature in every round of refinement	If UP is not empty, add a UP feature to refine a rule; Else If UN is not empty, add a UN feature to refine a rule; Else If Ov is not empty, add an Ov feature to refine a rule.
UN-UP-Ov	Either a UP, UN or Ov feature in every round of refinement	If UN is not empty, add a UN feature to refine a rule; Else If UP is not empty, add a UP feature to refine a rule; Else If Ov is not empty, add an Ov feature to refine a rule.
BestStrategy	All sub-spaces	Choose the best rule from the five rules generated by each UP, UN, Ov, UP-UN-Ov and UN-UP-Ov.
BestPosRule	UP and Ov in every round of refinement	Generate two versions of rule; one refined with a UP feature and the other refined with an Ov feature. Choose the best between the two versions.
BestRule	UP, UN and Ov in every round of refinement	Generate three versions of rule; one refined with a UP feature, one refined with a UN feature and the other refined with an Ov feature. Choose the best between the three versions.

Wang [15]. In his work, he split the 20 Newsgroups data set into two non-overlapping data sets (*20NG-A* and *20NG-B*), each containing 10 classes. Each class contained 1,000 documents with the exception of one class in *20NG-B* that had 997 documents. In the Reuters-21578 data set, the top ten most populous classes were first selected and multi-labelled and empty documents (documents that did not include any text) were removed, leaving a data set with eight classes and 6,643 documents, hereafter referred to as *Reuters8*.

The rule-based text classification model described in Section 3 was applied to all the data sets. Preprocessing included stop words removal and feature selection. Chi-square was used in a local feature selection environment with a reduction factor of 0.9 (only 10% of the features from each class were used). The proposed rule refinement strategies were applied during the rule learning stage. Rules with a Laplace estimation accuracy lower than the threshold of 50% were removed from the ruleset. The performance of our inductive rule learner, with each of the different proposed rule refinement strategies, was compared with the TFPC associative rule learner of Coenen and Leng [5]. The evaluation metric used is the average accuracy produced using ten fold cross validation.

6 Results and Analysis

The experiments conducted compared our inductive rule learner, and its different rule refinement strategies, with an associative rule learner in a multi-class classification setting. Hereafter, our inductive rule learner is denoted as RL appended with the identifier of the different rule refinement strategies, and the associative rule learner is denoted as ARL. Table 5, 6 and 7 show the average accuracy obtained using ten fold cross validation, the average number of rules generated, the average number of rules with negation generated and the percentage of rules with negation, for the *20NG-A*, *20NG-B* and *Reuters8* data sets respectively. In each case, the highest accuracy is highlighted in bold.

Table 5. Results using the 20NG-A dataset

Method	Avg accuracy	Avg # rules generated	Avg # rules with negation generated	Avg % of rules with negation
RL + UP	78.3	218.0	0.0	0.0
RL + UN	73.3	132.0	46.0	34.8
RL + Ov	77.1	194.2	0.0	0.0
RL + UP-UN-Ov	78.3	218.0	0.0	0.0
RL + UN-UP-Ov	77.8	218.9	76.1	34.8
RL + BestStrategy	77.7	199.9	37.8	18.9
RL + BestPosRule	78.5	218.9	0.0	0.0
RL + BestRule	**79.8**	194.4	48.6	25.0
ARL	76.0	1582.9	0.0	0.0

With respect to classification accuracy, inspection of the results indicated that the best performing method in all cases was the RL + BestRule strategy. RL + UN performed worst in the *20NG-A* dataset, RL + Ov performed worst in the *20NG-B* dataset, while ARL and RL + UN performed equally worst in the *Reuters8* dataset.

With respect to the overall number of rules generated in each case, ARL produced many more rules than any of the RL strategies. This is because ARL approaches, such as TFPC produce many frequent *item sets* (patterns) from which classification rules are generated. The number of rules generated by ARL systems is a criticism frequently directed at this approach. With respect to the eight strategies, there is no significant variation with respect to the number of rules generated although the *Reuters8* data set seems to require fewer rules than the *20NG-A* and *20NG-B* data sets. Recall that RL + UP, RL + Ov and

Table 6. Results using the 20NG-B dataset

Method	Avg accuracy	Avg # rules generated	Avg # rules with negation generated	Avg % of rules with negation
RL + UP	79.3	216.8	0.0	0.0
RL + UN	79.9	110.3	47.2	42.8
RL + Ov	78.7	196.4	0.0	0.0
RL + UP-UN-Ov	79.3	216.8	0.0	0.0
RL + UN-UP-Ov	79.8	209.6	81.0	38.6
RL + BestStrategy	79.5	161.1	45.2	28.1
RL + BestPosRule	79.0	217.5	0.0	0.0
RL + BestRule	**81.2**	169.9	53.0	31.2
ARL	79.2	1546.1	0.0	0.0

Table 7. Results using the Reuters8 dataset

Method	Avg accuracy	Avg # rules generated	Avg # rules with negation generated	Avg % of rules with negation
RL + UP	85.0	133.0	0.0	0.0
RL + UN	75.2	25.6	25.6	100.0
RL + Ov	78.0	105.9	0.0	0.0
RL + UP-UN-Ov	85.0	132.9	0.0	0.0
RL + UN-UP-Ov	89.1	121.6	108.6	89.3
RL + BestStrategy	89.0	99.5	58.9	59.2
RL + BestPosRule	85.1	129.1	0.0	0.0
RL + BestRule	**90.3**	97.3	66.9	68.8
ARL	75.2	3235.6	0.0	0.0

RL + BestPosRule cannot generate rules with negative features. In addition, inspection of the results shows that RL + UP-UN-OV also does not generate rules with negative features, suggesting that the UP sub-space was never empty with respect to all three data sets and thus, produces the same classification results as the RL + UP strategy. The relatively poorer performance of the RL + UN method is attributed to the structure of the rules generated, in that rules with negation generated by this strategy take the structure of "one positive feature and one or more negative feature" (i.e. identical to the Olex system [14]). The disadvantage of this rule structure is that having only one positive feature in its rule condition (despite having one or more negative features) makes the rule overly general in that the single positive feature is prone to misclassifying documents which the negative features cannot overturn. Also, it is worth noting that RL + UN generates fewer rules than any of the other strategies. This is again attributed to the rule structure in that a single positive feature can generally cover a large portion of documents, and thus, less rules are being learnt. In direct comparison of RL + BestPosRule and RL + BestRule, where the latter is an extension of the former by allowing the inclusion of UN feature in refinement, RL + BestRule outperformed RL + BestPosRule in all three datasets. This strongly suggests that the use of negated features can produce more effective classifiers.

If we look at the number of negative rules generated in the case of strategies that permit negative features, a significant proportion are rules that include negated features. The proportion is higher with respect to the *Reuters8* data set than for the *20NG-A* and *20NG-B* data sets. Overall, the results indicate that the strategies that include negated features in classification rules generated using IRL give a higher accuracy than strategies that do not include negated features.

7 Conclusion and Future Work

In this paper, we have presented an evaluation of a number of strategies to include negated features in IRL systems. These different rule refinement strategies enabled the generation of rules with and without negation. The approach was applied to text classification using the bag-of-words representation. The reported comparison between the different rule refinement strategies demonstrated that RL + BestRule produced the best results in all cases. In general, rule refinement strategies that involved negated features performed well indicating the advantage that can be gained in the context of text classifcation. In addition, a comparison was also made with an Associative Rule Learner (ARL); the proposed variations of the IRL method outperformed ARL in nearly all cases. Moreover, all of the IRL strategies used were able to produce more compact (smaller) classifiers with less rules. In conclusion, the results reported in this paper demonstrate that rules with negation are more effective for the multi-class classification task than rules that rely on positive features only.

Our future work will further extend the current work by using phrases instead of just single keywords in the text classification task. Phrases contain semantic information which, as noted above, are lost in the bag-of-words representation. For example, “Bank of England” and “river bank” are two phrases with entirely different semantic content. We are motivated by the potential benefit of using this semantic information in phrases to provide an added advantage to the text classification task. More specifically, we will also look into the use of negated phrases within the context of the work described in this paper.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 207–216 (1993)
2. Antonie, M.-L., Zaïane, O.R.: An associative classifier based on positive and negative rules. In: Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, pp. 64–69 (2004)

3. Apté, C., Damerau, F.J., Weiss, S.M.: Automated learning of decision rules for text categorization. ACM Transactions on Information Systems 12, 233–251 (1994)
4. Baralis, E., Garza, P.: Associative text categorization exploiting negated words. In: Proceedings of the ACM Symposium on Applied Computing, pp. 530–535 (2006)
5. Coenen, F., Leng, P.: The Effect of Threshold Values on Association Rule Based Classification Accuracy. Journal of Data and Knowledge Engineering 60(2), 345–360 (2007)
6. Cohen, W.: Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning (ICML), pp. 115–123. Morgan Kaufmann, San Francisco (1995)
7. Fürnkranz, J., Widmer, G.: Incremental reduced error pruning. In: Proceedings of the 11th International Conference on Machine Learning (ICML). Morgan Kaufmann, San Francisco (1994)
8. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco (2006)
9. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
10. Lang, K.: Newsweeder: Learning to filter netnews. In: Proceedings of the 12th International Conference on Machine Learning, pp. 331–339 (1995)
11. Lewis, D.D.: Reuters-21578 text categorization test collection, Distribution 1.0, README file, v 1.3 (2004), http://www.daviddlewis.com/resources/testcollections/reuters21578/readme.txt
12. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient Classification based on Multiple class-Association Rules. In: Proceedings of the IEEE International Conference on Data Mining, pp. 369–376 (2001)
13. 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)
14. Rullo, P., Cumbo, C., Policicchio, V.L.: Learning rules with negation for text categorization. In: Proceedings of the 22nd ACM Symposium on Applied Computing, pp. 409–416. ACM, New York (2007)
15. Wang, Y.J.: Language-independent pre-processing of large documentbases for text classifcation. PhD thesis (2007)
16. Weiss, S.M., Indurkhya, N.: Optimized rule induction. IEEE Expert: Intelligent Systems and Their Applications 8, 61–69 (1993)
17. Yin, X., Han, J.: CPAR: Classification based on Predictive Association Rules. In: Proceedings of the SIAM International Conference on Data Mining, pp. 331–335 (2003)

Fast Retrieval of Time Series Using a Multi-resolution Filter with Multiple Reduced Spaces

Muhammad Marwan Muhammad Fuad and Pierre-François Marteau

VALORIA, Université de Bretagne Sud, Université Européenne de Bretagne
BP. 573, 56017 Vannes, France
{marwan.fuad,pierre-francois.marteau}@univ-ubs.fr

Abstract. Fast retrieval of time series that are similar to a given pattern in large databases is a problem which has received a lot of attention in the last decade. The high dimensionality and large size of time series databases make sequential scanning inefficient to handle the similarity search problem. Several dimensionality reduction techniques have been proposed to reduce the complexity of the similarity search. Multi-resolution techniques are methods that speed-up the similarity search problem by economizing distance computations. In this paper we revisit two of previously proposed methods and present an improved algorithm that combine the advantages of these two methods. We conduct extensive experiments that show the show the superior performance of the improved algorithm over the previously proposed techniques.

Keywords: Time Series Databases, Similarity Search, Dimensionality Reduction Techniques, Sequential Scanning, MIR, MIR_X, Tight-MIR.

1 Introduction

Time series similarity search is a problem that has many applications in computer science. Similarity between two time series can be depicted using a similarity distance, which is usually costly compared with other tasks such as CPU time or even I/O time.

Formally, range query can be defined as follows: given a time series database U of size n and a query (q, r), where r represents a *threshold.* The range query problem can be specified as retrieving all the time series $u \in U$ which satisfy: $d(q,u) \leq r$.

The trivial answer to this problem that sequential scanning offers, which is based on scanning U and returning the time series that satisfy the above condition, is slow because it requires n distance computations, which can be computationally very expensive in large databases with costly distances.

Time series dimensionality reduction techniques aim at reducing this high complexity by transforming the time series into a lower dimensional space, thus decreasing query-time distance evaluations, and processing the similarity search in this reduced space.

There have been different suggestions to represent time series in lower dimensional spaces, to mention a few: *Discrete Fourier Transform* (DFT) [1,2], *Discrete Wavelet*

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 137–148, 2010.

Transform (DWT) [4], *Singular Value Decomposition* (SVD) [10], *Adaptive Piecewise Constant Approximation* (APCA) [9], *Piecewise Aggregate Approximation* (PAA) [8,18], *Piecewise Linear Approximation* (PLA) [11], *Chebyshev Polynomials* (CP) [3].

Time series dimensionality reduction techniques are based on predefined schemes, in that the parameters which control their performance, and which have been chosen at indexing-time, may prove to be inappropriate at query-time. Multi-resolution techniques offer more control on the parameters that determine the effectiveness and efficiency of dimensionality reduction methods. In [12] and [13] two such multi-resolution techniques have been proposed. In this paper we propose an improved algorithm which the advantages of the two previously proposed techniques in terms of performance and autonomy.

The work presented in this paper is organized as follows: in section 2 we present related work and background. In section 3 of this paper we introduce the improved algorithm and we evaluate its performance in section 4. In section 5 we give concluding remarks.

2 Related Work and Background

2.1 Similarity Distances

Let D be a set of objects. A function $d : D \times D \rightarrow \Re^{+} \cup \{0\}$, is called a distance metric if the following holds:

(p1) $d(x, y) \geq 0$ (non-negativity)

(p2) $d(x, y) = d(y, x)$ (symmetry)

(p3) $x = y \Leftrightarrow d(x, y) = 0$ (identity)

(p4) $d(x, z) \leq d(x, y) + d(y, z)$ (triangular inequality)

$\forall x, y, z \in D$. We call (D, d) a metric space

2.2 Dimensionality Reduction Techniques

There are several dimensionality techniques in the literature, we present in the following two of them

Piecewise Aggregate Approximation (PAA): This method was proposed in [8] and [18], independently. Its basis is simple and straightforward, yet this method has been successfully used as a competitive method.

PAA reduces the dimensionality of a time series from n in the original space to N in the reduced space by segmenting the time series into equal-sized frames and representing each segment by the mean of the data points that lie within that frame.

The similarity distance used in the reduced space is:

$$d(X,Y) = \sqrt{\frac{n}{N}}\sqrt{\sum\nolimits_{i=1}^{N}(\overline{x_i} - \overline{y_i})^2} \tag{1}$$

Where n is the length of the time series, N is the number of frames, which should be a factor of n . The compression ratio m is n/N .

It is proven in [8] and [18] that the above similarity distance is lower bounding of the Euclidean distance applied in the original space of time series.

Since d^N is lower bounding of the Euclidean distance in the original space then, according to the GEMINI algorithm, which is a generic approach of indexing and retrieving time series [6], any time series u that satisfies:

$$d^N(q,u) > r \tag{2}$$

can not be answer to the query and should be excluded.

The Symbolic Aggregate Approximation (SAX): Symbolic representation of time series has attracted much attention recently, because by using this method we can not only reduce the dimensionality of time series, but also benefit from the numerous algorithms used in bioinformatics and text data mining. Of all the symbolic representation methods, the symbolic aggregate approximation method (SAX) [7] is one of the most powerful ones in time series data mining.

SAX is based on the fact that normalized time series have highly Gaussian distribution [7], so by determining the breakpoints that correspond to the chosen alphabet size, one can obtain equal sized areas under the Gaussian curve.

SAX is applied in the following steps: in the first step all the time series in the database are normalized. In the second step, the dimensionality of the time series is reduced by using PAA. In the third step, the PAA representation of the time series is discretized. This is achieved by determining the number and the locations of the breakpoints. This number is related to the chosen alphabet size (it is chosen by the user), i.e. *alphabet_size=number(breakpoints)+1*. The locations of the breakpoints are determined by statistical lookup tables so that these breakpoints produce equal-sized areas under the Gaussian curve. The interval between two successive breakpoints is assigned to a symbol of the alphabet, and each segment of the PAA that lies within that interval is discretized by that symbol.

The last step of SAX is using the following similarity distance:

$$MINDIST(\hat{S},\hat{T}) \equiv \sqrt{\frac{n}{N}}\sqrt{\sum_{i=1}^{N}(dist(\hat{s}_i,\ \hat{t}_i))^2} \tag{3}$$

Where n is the length of the original time series, N is the number of the frames, $\tilde{S}$ and $\tilde{T}$ are the symbolic representations of the two time series S and T, respectively, and where the function $dist(\ )$ is implemented by using the appropriate lookup table.

3 The Proposed Algorithm

A Multi-resolution Indexing and Retrieval scheme (MIR) has been proposed in [12]. This scheme is based on using several reduced spaces that correspond to different resolution levels. The principal of the algorithm is as follows: let U be the original, n-dimensional space where the time series are embedded. R is a $2m$-dimensional space, where $2m \leq n$. Each time series $u \in U$ is divided into m segments. Each segment is approximated by a function of low dimension. The functions used are polynomials of degree 1, 3, or 5, and where the approximation error, according to a given distance, between this segment and the approximating function is minimal, so this function is the best approximation of that segment. A function of the same type and the same degree is used to approximate all the segments of all the time series in the database. The image vector $\overline{u}$ is, by definition, an n-dimensional vector whose components are the images of all the points of all the segments of a time series on that approximating function. The images of the two end points of that segment on the approximating function are called the main image of that segment. So for a time series of m segments we have $2m$ main images. Those $2m$ main images are, by definition, the projection vector u^R of the time series on R. Two distances are defined on the database: the first is denoted by d , and is defined on a n-dimensional space, so it is the distance between two time series in the original space , i.e. $d(u_i,u_j)$, or the distance between the original time series and its image vector, i.e. $d(u_i,\overline{u}_i\ )$. The second distance is denoted by d^R , and is defined on a $2m$-dimensional space, so it is the distance between two projection vectors, i.e. $d^R(u_i^R,u_j^R)$. It is shown in [12] that d^R is a lower bounding of d when the Minkowski distance is used. The resolution level k is an integer related to the dimensionality of the reduced space R . So the above definitions of the projection vector and the image vector can be extended to further segmentation of the time series using different values $m \leq m_k$, The image vector and the projection vector at level k are denoted by $\overline{u}^{(k)}$ and $u^{R(k)}$, respectively.

Given a query (q,r) , let $\overline{u}$, $\overline{q}$ be the projection vectors of u , q , respectively, on their approximating functions, where u is a time series in the database. By applying the triangular inequality we get:

$$\left| d(u,\overline{u}) - d(q,\overline{q}) \right| > r \tag{4}$$

This relation represents a pruning condition which is called the first filter.

By applying the triangular inequality again we get :

$$d^R(u^{R(k)},q^{R(k)}) > r + d(q,\overline{q}^{(k)}) + d(u,\overline{u}^{(k)}) \tag{5}$$

This relation represents the second filter.

At indexing-time the application of the method starts by choosing the length of segments for each resolution level. The shortest length corresponds to the lowest level, and the longest corresponds to the highest level. Then the approximating function to be used with all the time series and for all resolution levels is chosen. The distances $d(u,\overline{u}^{(k)})\ \ \forall u \in U$ are then computed and stored. The segmenting scheme is simple and straightforward. The segments are connected and their length is a power of 2. So for a time series of 152 dimensions, for instance, the segmenting scheme for the first resolution level is: $[1,64],[64,128],[128,152]$. Notice that the three segments have three different lengths (63,64,14), respectively, and this difference of lengths will continue to appear at higher levels. Notice also that this segmentation contains the information of four points (1,64,128,152).

At query-time the query is divided into segments with the same length as that of the time series and for each resolution level. These segments are approximated using the same approximating function that was used to approximate the time series. The distances $d(q,\overline{q}^{(k)})$ are then computed. The algorithm tries to exclude the time series at each resolution level using the first filter which is less costly to apply than the second filter, because it does not include any distance evaluation at query-time. The second filter uses two distances that have been computed at indexing-time: $d(q,\overline{q}^{(k)}),d(u,\overline{u}^{(k)})$, so the only distance that is to be computed at query-time is $d^{R}(u^{R(k)},q^{R(k)})$. The second filter is less costly at lower levels than at higher levels. But at any level, the cost of applying the second filter is never as costly as the distance computations at the original space, because it is assumed that $2m \leq n$.

MIR starts with the lowest level and tries to exclude the first time series using (4). If this time series is pruned, the algorithm moves to the next time series, if not, the algorithm tries to prune this time series using relation (5). If all the time series in the database have been pruned the algorithm terminates, if not, the algorithm moves to a higher level. Finally, after all levels have been exploited, we get a candidate answer set, which is sequentially scanned to filter out all the false alarms and return the true answer set.

In [13] a multi-resolution scheme is presented. This scheme combines the first filter as described in (4) with the exclusion condition of one of the dimensionality reduction techniques known in the literature. This algorithm is called MIR_X, where X is the dimensionality reduction technique used. In MIR_X each segmented time series has two representations; one that is identical to that of MIR and the other is the representation proposed by the dimensionality reduction technique used. The algorithm used is similar to that of MIR. The experiments show that MIR_X improves the performance of dimensionality reduction techniques.

Both MIR and MIR_X have their drawbacks: The two distances $d(q,\overline{q}^{(k)}),d(u,\overline{u}^{(k)})$ used to apply the second filter (relation (5)) lower its pruning power. Besides, the segmenting scheme used is not optimal and can results in segments of completely different lengths when the length of the time series is not a power of 2. The direct consequence of this segmentation is that the approximation error is not harmonized which, in turn, lowers the pruning power of the first filter. MIR has one main advantage; it is a standalone method, unlike MIR_X.

MIR_X has the same segmenting scheme as that of MIR. However, using the exclusion condition of a dimensionality reduction technique which , in the case of the dimensionality reduction techniques used in [13], uses a lower bounding condition to the distance in the original space makes the second filter described in (5) redundant, because it is overwritten by the more powerful exclusion condition of the dimensionality reduction technique. Hence, the performance of MIR_X is better than that of MIR. Nevertheless, MIR_X has a few drawbacks: it is completely dependent on the dimensionality reduction technique used. If the dimensionality reduction technique uses a lower bounding condition, MIR_X can be applied, if not, MIR_X can never be applied. On the other hand, the application of MIR_X requires adopting a different concept of resolution level for each dimensionality reduction technique, which is not intuitive. Besides, some dimensionality reduction techniques have certain requirements (the length of the time series should be a power of 2 for DWT, N should be a factor of n for PAA and SAX). All these factors influence the application of MIR_X.

In this paper we present an improvement on the two previous versions. This improved algorithm, which we call *Tight-MIR,* has the advantages of both MIR and MIR_X in that it is a standalone method, yet it has the same competitive performance of MIR_X.

The redundancy of the second filter in MIR_X suggests that the application of MIR can use two separate filters. In fact the requirement that the approximating error of the approximation function be minimal is not exploited when constructing the second filter as described in [12].

In Tight-MIR instead of using the projection vector to construct the second filter, we access the raw data in the original space directly using a number of points that corresponds to the dimensionality of the reduced space at that resolution level. In other words, we use $2m$ raw points, instead of $2m$ main images, to compute d^R. There are several positive effects to this modification; the first is that the new d^R is obviously tighter than d^R as computed in [12]. The second is that when using a Minkowski distance d^R is lower bounding to the original distance in the original space. The direct consequence of this is that the two distances $d(q,\overline{q}^{(k)}), d(u,\overline{u}^{(k)})$ become redundant, so the second filter is overwritten by the usual lower bounding condition $d^R(u^{R(k)}, q^{R(k)}) > r$.

Notice that the complexity of the modified d^R is $O(2m)$ which is the same complexity as described in [12]. So this modification does not require any extra cost.

Tight-MIR also utilizes an improved segmenting scheme: at each level, the segments are equal and disconnected, so for a time series of 152 dimensions (the one presented earlier this section) the segmentation used for the first level is $[1,76]$, $[77,152]$, this segmentation has two advantages: the error is more harmonized. The second advantage is that although we are using two segments only (at the first resolution levels) we are representing the information of four points of four points (1,76,77,152), which is the same number of points presented in the original segmenting. So the modified algorithm needs one more resolution level less than the original segmenting, which is another advantage in terms of space complexity.

4 Performance Evaluation

The objective of our experiments is to show that the modified algorithm Tight-MIR has the advantages of both MIR and MIR_X together, so we have to show that it outperforms MIR, and that it has the same performance as that of MIR_X. We also conduct other experiments to compare Tight-MIR against other dimensionality reduction techniques, because Tight-MIR it is a standalone method (unlike MIR_X).

Although several papers present experiments based on wall clock time, but it is a poor choice and subject to bias [5,8], so we prefer to use the latency time concept presented in [15]. Latency time refers to the number of cycles the processor takes to perform different arithmetic operations (>,+ - ,*,abs, sqrt) when performing the similarity search. Then the number of each operation is multiplied by the latency time of that operation to get the total latency time of the similarity search. The latency time is 5 cycles for (>, + -), 1 cycle for (abs), 24 cycles for (*), and 209 cycles for (sqrt). This method of testing performance is the same one that was used in both [12] and [13].

We conducted extensive experiments using datasets of different sizes and dimensions and from different repositories [14, 16, 17, 19]. We also used different threshold values. We conducted experiments using approximating polynomials of different degrees, but because of space limitation we report here the results of approximating by first degree polynomials to facilitate the comparison.

We first show a comparison between MIR and Tight-MIR. The way d^R is computed in Tight-MIR enables us to modify the codes used to avoid the square root, which is a very costly operation, of course sequential scanning was also modified to avoid this operation. This modification is not possible with MIR. So the comparison we present here is made between the speed-up of MIR and Tight-MIR compared to

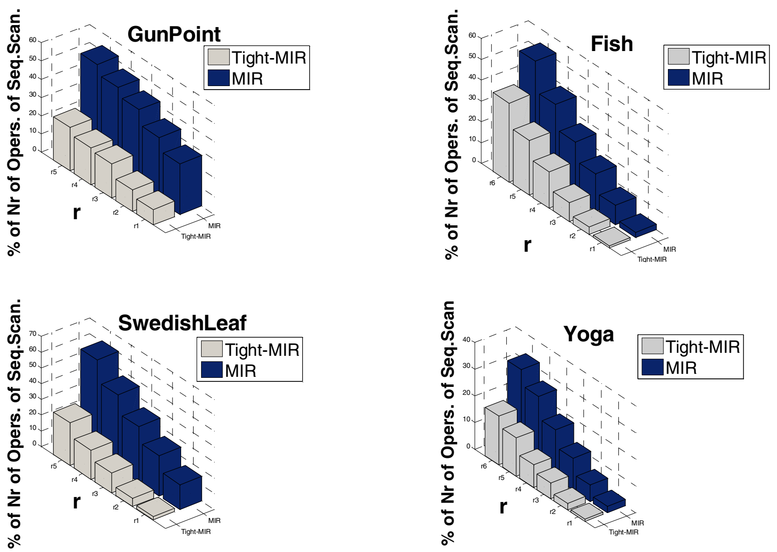


Fig. 1. Comparison of the speed-up of MIR and Tight-MIR on four datasets (Yoga), (Swedish-Leaf), (GunPoint) and (Fish) over sequential scanning

sequential scanning. In Figure 1 we present the results of four datasets. The results clearly show that Tight-Mir outperforms MIR for all the datasets and for all values of r . As in the experiments of [12] and [13], the values of r vary between those which return 1% and 10% of the time series in sequential scanning.

In the second series of experiments we compared MIR_X, where X is dimensionality reduction technique, with Tight-MIR to show that the two methods give similar results. Figure 2 shows some of the results we obtained comparing

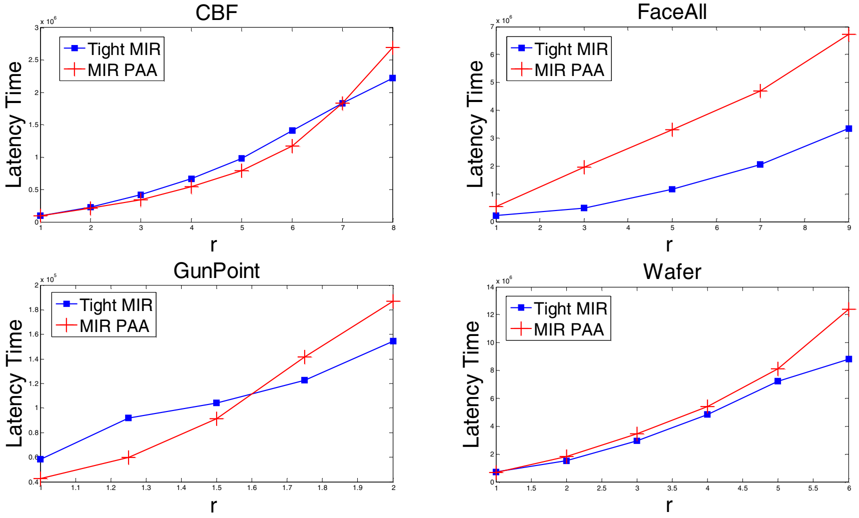


Fig. 2. Comparison between MIR_PAA and Tight-MIR on datasets (CBF), (FaceAll), (Wafer), and (GunPoint)

Tight-MIR with MIR_PAA. The results presented in Figure 2 show that MIR_PAA and Tight-MIR have the same performance. In fact, we can even say that the performance of Tight-MIR is even slightly better.

Comparing Tight-MIR with MIR_SAX also showed that the two methods have the same performance.

It is important to mention that both MIR_PAA and MIR-SAX require that N be a factor of n , the length of the time series (but Tight-MIR does not require that).

We also conducted other experiments to compare Tight-MIR with PAA, using different datasets and different values of r . We report in Figure 3 some of the results we obtained. The results show that Tight-MIR outperforms PAA for all the datasets.

We also conducted experiments comparing Tight-MIR with SAX, which is a very competitive and fast method. SAX appeared in two versions; in the first one the alphabet size varied in the interval (3:10), and in the second one the alphabet size varied in the interval (3:20).

We conducted experiments on different datasets, and for different values of the alphabet size. The codes we used in the experiments were optimized versions of the original codes of SAX, since the original codes written by the authors of SAX were not optimized for speed, so we optimized them to make a fair comparison.

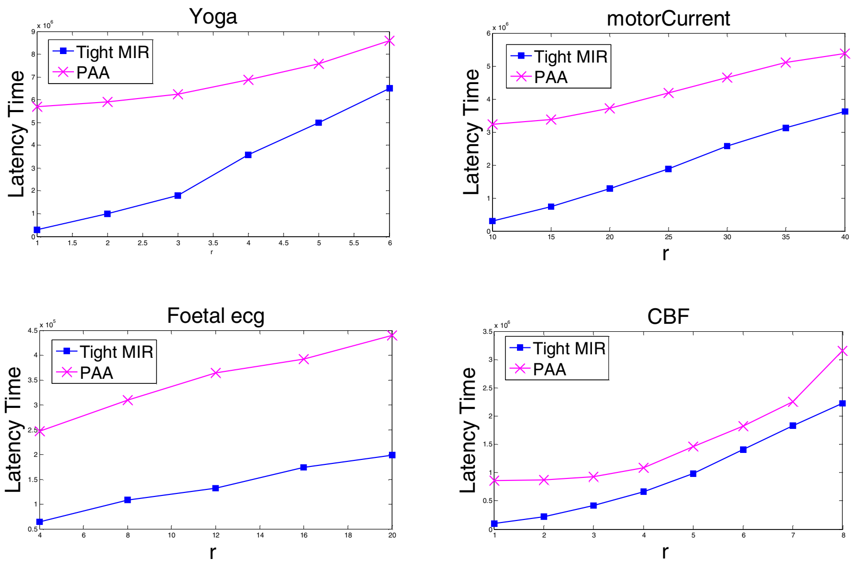


Fig. 3. Comparison between PAA and Tight-MIR on datasets (Yoga), (motorCurrent), (CBF), and (Foetal ecg)

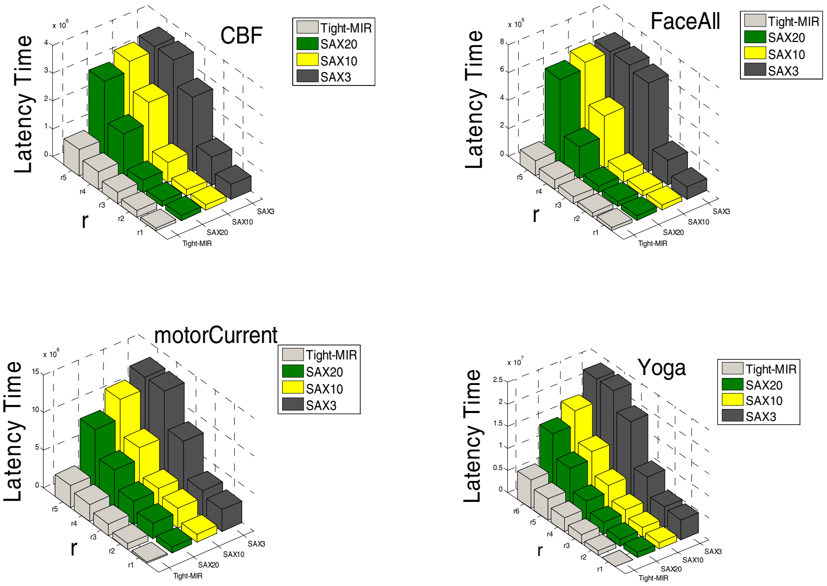


Fig. 4. Comparison of the latency time between Tight-MIR and SAX for alphabet size=3, 10, and 20 on datasets (FaceAll), (CBF),(motoCurrent), and (Yoga)

We report in Figure 4 the results of several datasets and for different values of r. The results shown here are for alphabet size 3 (the smallest alphabet size possible for SAX), 10 (the largest alphabet size in the first version of SAX), and 20 (the largest alphabet size in the second version of SAX).

We have to mention that the datasets used in these last experiments were normalized because SAX can only be applied to normalized time series.

The results obtained show that Tight-MIR clearly outperforms SAX for the different values of r and for the different values of the alphabet size.

It is important to mention that the results of SAX as show in the above Figure may give the fake impression that with some datasets the number of operations seems to be stable after a certain value of r. This phenomenon does not indicate stability of performance. It only indicates that the SAX exploited all the indexed time series using the lower bounding condition without being able to exclude any time series, so the search process moved to sequential scanning. So this phenomenon is the worst scenario possible because the number of operations exceeds even that of sequential scanning and reaches the maximum number possible of operations, i.e. the maximum number of distance evaluations.

In all the experiments we presented so far comparison was made based on speed as measured by the latency time, but comparisons between representation methods can also be made according to their pruning power. In Table 1 we present the pruning power of MIR-Tight and SAX with alphabet size=20 (which is the most effective version of SAX) on the datasets presented in Figure 4. The results show that the pruning power of Tight-MIR is much more stable than that of SAX20 as r gets larger.

Table 1. Comparison of the pruning power between Tight-MIR and SAX20 (alphabet size=20) for the smallest and largest values of r that were used in the experiments presented in Figure 4

	SAX20		**Tight-MIR**	
	$\mathbf{r}_{\mathbf{min}}$	$\mathbf{r}_{\mathbf{max}}$	$\mathbf{r}_{\mathbf{min}}$	$\mathbf{r}_{\mathbf{max}}$
CBF	99.89 %	14.88 %	99.89 %	98.69 %
FaceAll	99.94 %	7.51 %	99.94 %	98.89 %
motoCurrent	98.40 %	33.15 %	99.87 %	88.13 %
Yoga	99.53 %	37.61 %	99.63 %	85.84 %

In the final set of experiments we wanted to test if the performance of Tight-MIR is stable with different dimensions of time series. We conducted experiments using datasets of different dimensions and the results showed high stability of performance. We present in Figure 5 the results of applying Tight-MIR on dataset (Wind) whose dimension is 12, and dataset (motoCurrent) whose dimension is 1500, compared to sequential scanning which represents the baseline performance.

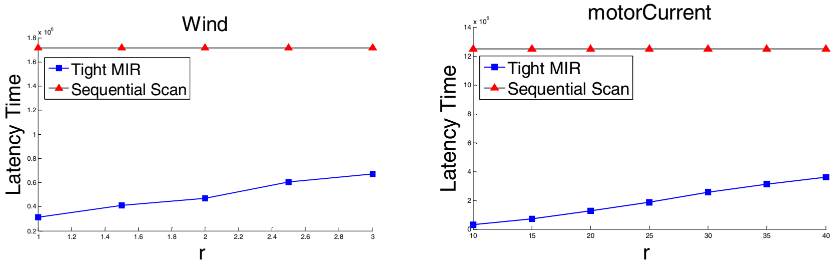


Fig. 5. Comparison of the latency time between Tight-MIR and Sequential Scanning on datasets (Wafer) and (motoCurrent)

5 Conclusion

In this work we presented an improved multi-resolution algorithm of time series retrieval. This new algorithm combines the advantages of two previously proposed algorithms. We conducted extensive experiments comparing the new algorithm with the previously proposed algorithms. The results of the experiments show the superiority of the improved algorithm over the two previous ones.

We also conducted other experiments which compare the performance of the new algorithm with other dimensionality reduction techniques. The results also show that the improved algorithm outperforms those tested dimensionality reduction techniques both in terms of speed and pruning power.

References

1. Agrawal, R., Faloutsos, C., Swami, A.: Efficient Similarity Search in Sequence Databases. In: Lomet, D.B. (ed.) FODO 1993. LNCS, vol. 730. Springer, Heidelberg (1993)
2. Agrawal, R., Lin, K.I., Sawhney, H.S., Shim, K.: Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-series Databases. In: Proceedings of the 21st Int'l Conference on Very Large Databases, Zurich, Switzerland, pp. 490–501 (1995)
3. Cai, Y., Ng, R.: Indexing Spatio-temporal Trajectories with Chebyshev Polynomials. In: SIGMOD (2004)
4. Chan, K., Fu, A.W.: Efficient Time Series Matching by Wavelets. In: Proc. of the 15th IEEE Int'l Conf. on Data Engineering, Sydney, Australia, March 23-26, pp. 126–133 (1999)
5. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures. In: Proc of the 34th VLDB (2008)
6. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast Subsequence Matching in Time-series Databases. In: Proc. ACM SIGMOD Conf., Minneapolis (1994)
7. Lin, J., Keogh, E.J., Lonardi, S., Chiu, B.Y.-c.: A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In: DMKD (2003)
8. Keogh, E.,, Chakrabarti, K.,, Pazzani, M., Mehrotra: Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. J. of Know. and Inform. Sys. (2000)

9. Keogh, E., Chakrabarti, K,. Pazzani, M., Mehrotra: Locally Adaptive Dimensionality Reduction for Similarity Search in Large Time Series Databases. In: SIGMOD (2001)
10. Korn, F., Jagadish, H., Faloutsos, C.: Efficiently Supporting ad hoc Queries in Large Datasets of Time Sequences. In: Proceedings of SIGMOD 1997, Tucson, AZ, pp. 289–300 (1997)
11. Morinaka, Y., Yoshikawa, M., Amagasa, T., Uemura, S.: The L-index: An indexing Structure for Efficient Subsequence Matching in Time Sequence Databases. In: Cheung, D., Williams, G.J., Li, Q. (eds.) PAKDD 2001. LNCS (LNAI), vol. 2035, pp. 51–60. Springer, Heidelberg (2001)
12. Muhammad Fuad, M.M., Marteau, P.F.: Multi-resolution Approach to Time Series Retrieval. In: Fourteenth International Database Engineering & Applications Symposium– IDEAS 2010, Montreal, QC, Canada (2010)
13. Muhammad Fuad, M.M. , Marteau P.F.: Speeding-up the Similarity Search in Time Series Databases by Coupling Dimensionality Reduction Techniques with a Fast-and-dirty Filter. In: Fourth IEEE International Conference on Semantic Computing– ICSC 2010.Carnegie Mellon University, Pittsburgh (2010)
14. `http://povinelli.eece.mu.edu/`
15. Schulte, M.J., Lindberg, M., Laxminarain, A.: Performance Evaluation of Decimal Floating-point Arithmetic in IBM Austin Center for Advanced Studies Conference (February 2005)
16. SISTA's Identification Database, `http://www.esat.kuleuven.ac.be/~tokka/daisydata.html`
17. StatLib - Datasets Archive, `http://lib.stat.cmu.edu/datasets/`
18. Yi, B.K., Faloutsos, C.: Fast Time Sequence Indexing for Arbitrary Lp norms. In: Proceedings of the 26st International Conference on Very Large Databases, Cairo, Egypt (2000)
19. UCR Time Series datasets, `http://www.cs.ucr.edu/~eamonn/time_series_data/`

DHPTID-HYBRID Algorithm: A Hybrid Algorithm for Association Rule Mining

Shilpa Sonawani[1] and Amrita Mishra[2]

JSS Academy of Technical Education,
Department of Computer Science & Engineering,
Noida, Uttar Pradesh, India

Abstract. Direct Hashing and Pruning algorithm of ARM performs well at initial passes by smaller candidate 2-itemset generation and turns out to be very powerful to facilitate initial itemset generation. Efficient pruning technique of AprioriTid algorithm is highly effective for frequent itemset generation in the later passes. Theoretical and practical studies reveals the underneath facts for the scope of improvement for both the algorithms. This paper describes a hybrid approach for association rule mining, based on coupling of best part of two well known ARM algorithms, AprioriTid and DHP with a binary approach which significantly improves the performance.

Keywords: Association rule mining algorithm, DHP, AprioriTid, binary approach, coupled algorithm.

1 Introduction

With a huge amount of data stored in databases and data warehousing, it is progressively more important to develop powerful tools for analysis of such data and mining useful knowledge from it. Data mining is the process of extracting useful information from the huge amount of data available. Association rule is an important technique of data mining to discover hidden rules, correlation relationships among items in the transactions. It studies the frequency of items occurring together in transactional database which is used for identifying and consequently defining "interesting" associations among data items. Association rules are then generated from the frequent itemsets. Rules generation majorly involve two steps.

1. Find all frequent itemsets: Each of frequent itemsets will occur at least as frequently as a pre-determined minimum support count.
2. Generate strong association rules from the frequent itemsets: These rules must satisfy minimum support and minimum confidence.

2 Problem Description

Association rule mining is popular task in data mining. It involves various techniques for discovery of rules. Some of the well known algorithms are Apriori algorithm,

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 149–160, 2010.

AprioriTid algorithm, Direct Hashing and Pruning algorithm, CHARM algorithm, FP-Growth etc. The above mentioned algorithms use different approaches & methodologies for mining association rules. But through various empirical studies it has been proved that the performance of association rule mining algorithm depends on wide range of factors such as size of database, size of transactions in a row (row size) and minimum support. Almost each ARM algorithms has some favorable environment that provides high performance & faster computation of frequent itemsets. These factors vary from algorithm to algorithm.

For example, Apriori algorithm performs well when the frequent itemsets generated are less[1], DHP algorithm performs well at initial passes for smaller candidate 2-itemset generation. AprioriTid algorithm performs well at later passes when more relevant set of itemsets are available[1]. Let's take into some detailed description of DHP algorithm and AprioriTid algorithm.

2.1 Direct Hashing and Pruning Algorithm (DHP Algorithm)

2.1.1 Positive Aspects of DHP Algorithm

The algorithm has two major features, one is generation of large itemsets & the other is reduction on transaction database size. The first feature generates smaller candidate 2-itemset that leads to the efficient pruning/reduction of transactional database. Let's take an example, as in each pass we use the set of large itemsets, L(k) to form the set of candidate large itemset C(k+1) by joining L(k) with L(k) on (k-1). We then scan the database and count the support of each itemset in C(k+1) so as to determine L(k+1). As a result, more the itemsets in C(k), the higher will be the processing cost of determining L(k). It is noted that given a large database, the initial extraction of useful information from the database is the most expensive, resource consuming & costly part. It has been already mentioned that the processing cost of first pass, that means the generation of L(1) and L(2) is in fact dominates the total processing cost. That's not all from DHP, by constructing a significantly smaller C(2), DHP can also generates the smaller D(3) to derive C(3). But one thing must be noted that without the smaller C(2), database cannot be pruned effectively. So, we can say that generation of smaller C(2) leads to the generation of smaller D(3). And after this, the size of L(k) decreases rapidly as "k" increases. A smaller L(k) leads to a smaller C(k+1) and thus a smaller corresponding processing cost. In view of the above, DHP is so designed that it will reduce the number of itemsets to be explored in C(k) in initial iterations notably. The corresponding processing cost to determine L(k) from C(k) is therefore reduced.

In the core, DHP uses the technique of hashing to filter out unnecessary itemsets for next candidate itemset generation. When the support of candidate k-itemsets is counted by scanning the database, DHP accumulates information about candidate (k+1)-itemsets in advance in such a way that all possible (k + 1)-itemsets of each transaction after some pruning are hashed to a hash table. Each bucket in the hash table consists of a number to represent how many itemsets have been hashed to this bucket so far. We note that based on the resulting hash table, a bit vector can be constructed, where the value of one bit is set to be one if the number in the corresponding entry of the hash table is greater than or equal to minimum support. As can be seen later, such a bit vector can be used to greatly reduce the number of itemsets in C(k). It is worth mentioning that DHP algorithm is particularly powerful to determine large itemsets in early stages, thus improving the performance bottleneck.

2.1.2 Negative Aspects of DHP Algorithm

There are some factors that can affect the performance of DHP, firstly, the hashing function, size of transaction items (row size) etc. Such as, the number of nodes in the hash tree of C(2) gets larger then the maintenance of each bucket, hash-count, bit vector, final bucket pruning and the searching cost goes relatively higher. Secondly, the DHP uses a formula known as hashing function, to generate hash-bucket & to hash the set of 2-itemset into the hashtable. As it is difficult to evaluate the performance of the hash function of DHP before starting the mining process, the collision rate may be very high as the number of transactions increases. As, the transaction size may affects the cardinality of hash bucket and candidate 2-itemset generation. Moreover, if the collision rate of the hashing function is too high, the size of C(2) may be close (or equal) to that of Apriori[2]. Therefore, in some cases, the performance of DHP may be worse than Apriori.

2.1.3 Corrective Measures for DHP Algorithm

DHP involves multiple pruning steps in hash bucket. Firstly, DHP prune hash bucket by using BIT-Vector concept. Secondly, as collision occurs in buckets, again we have to check itemset count in bucket and prune irrelevant data from the bucket to generate C2. After First-Step pruning done, Second pruning step always required. Second pruning step basically required to check collision case in bucket. It's better if we reduce the pruning step to a single step. This will obviously save computation time and memory used by DHP pruning process. In addition to above trick, we can implement only those buckets with at least one itemset in bucket. That will obviously help to improve the computation time and hence efficiency.

DHP scan database and H(t), set of 2-itemsets, and using H(t) & hashing function generate hashtable H(2). Here, efficiency of DHP may increase if the set of 2-itemsets and hashtable H (2) is created by using pruned database that has obviously smaller size and more relevant items compare to main dataset. For the generation of C(k+1) and L(k+1), for k=2, DHP uses Apriori algorithm based approach, that provides good performance but not good enough than the AprioriTid algorithm. So, if we implement kind of AprioriTid approach for later passes of DHP algorithm, which may provide some extra boost to the performance.

2.2 AprioriTid Algorithm

2.2.1 Positive Aspects of AprioriTid Algorithm

AprioriTid does not count the support of itemsets from the tuples of the database, instead, a specific data structure $\overline{C}$ (k+1) is maintained, whose content is the frequent k-itemsets contained by every tuple therefore, as k-increases, the tuples and the frequent k-itemsets contained by each tuple in $\overline{C}$ (k+1) decrease, so that the contents required by scanning for counting is reduced. The AprioriTid, for computing the supports for larger itemsets never revisit the original dataset but transforms the table as it goes along. Also, it has been proved during the development of AprioriHybrid algorithm that AprioriTid beats Apriori in later passes[1]. The raised in performance caused due to the fact that, AprioriTid progressively prune in-memory dataset using candidate itemset and in the later passes, the number of candidate itemsets reduces, dataset pruning process remove more irrelevant itemsets and reduce the database size. As a result, AprioriTid scan smaller dataset at each pass. So, at each phase, database reduces gradually and performance increases gradually.

2.2.2 Negative Aspects of AprioriTid Algorithm

It has been proved during the development of AprioriHybrid algorithm that Apriori does better than AprioriTid[1]. The reason for dropped in performance is as follows, Apriori & AprioriTid use the same candidate generation procedure and therefore count the same itemsets but at initial pass AprioriTid involved in the generation of in-memory temporary dataset that may consume I/O resources and may decrease the net execution time at initial phases. So, working with AprioriTid at initial passes can be expensive and performance can be dropped down below the Apriori algorithm. One more thing that AprioriTid uses very efficient database pruning method but the performance of pruning process somewhere more effective if applied on smaller number of candidate itemset at initial phases or with high minimum support, as higher minimum support triggered more pruning and hence reduced database, consequently the computation time of frequent itemset generation.

2.2.3 Corrective measures for AprioriTid Algorithm

AprioriTid algorithm can be effective if more relevant set of itemset is provided. and DHP's efficient hashing and pruning technique have the capability to generate smaller size candidate 2-itemset and hence can be applied further to prune database. The database pruning process not only prunes the irrelevant itemset but also trim irrelevant items from transactions. If this database is provided to AprioriTid, performance improvement can be established. AprioriTid algorithm can be effective if more relevant set of itemset provided.

3 DHPTID-HYBRID Algorithm

Here, presented a new algorithm called DHPTID-HYBRID ALGORITHM for discovering all significant association rules among items in a large database of transactions. The proposed algorithm is a result of coupling process between two well acknowledged algorithms AprioriTid algorithm and Direct Hashing & Pruning algorithm(DHP) to quickly discover the frequent itemsets from the huge dataset and is capable to determine association rules from the generated frequent itemset.

3.1 Proposed Algorithm

The DHPTID-HYBRID Algorithm stated below, mainly divided in phases to provide better understanding of the various operations/activities involved within the algorithm. List of terminology involved in DHPTID-Hybrid algorithm as follow:

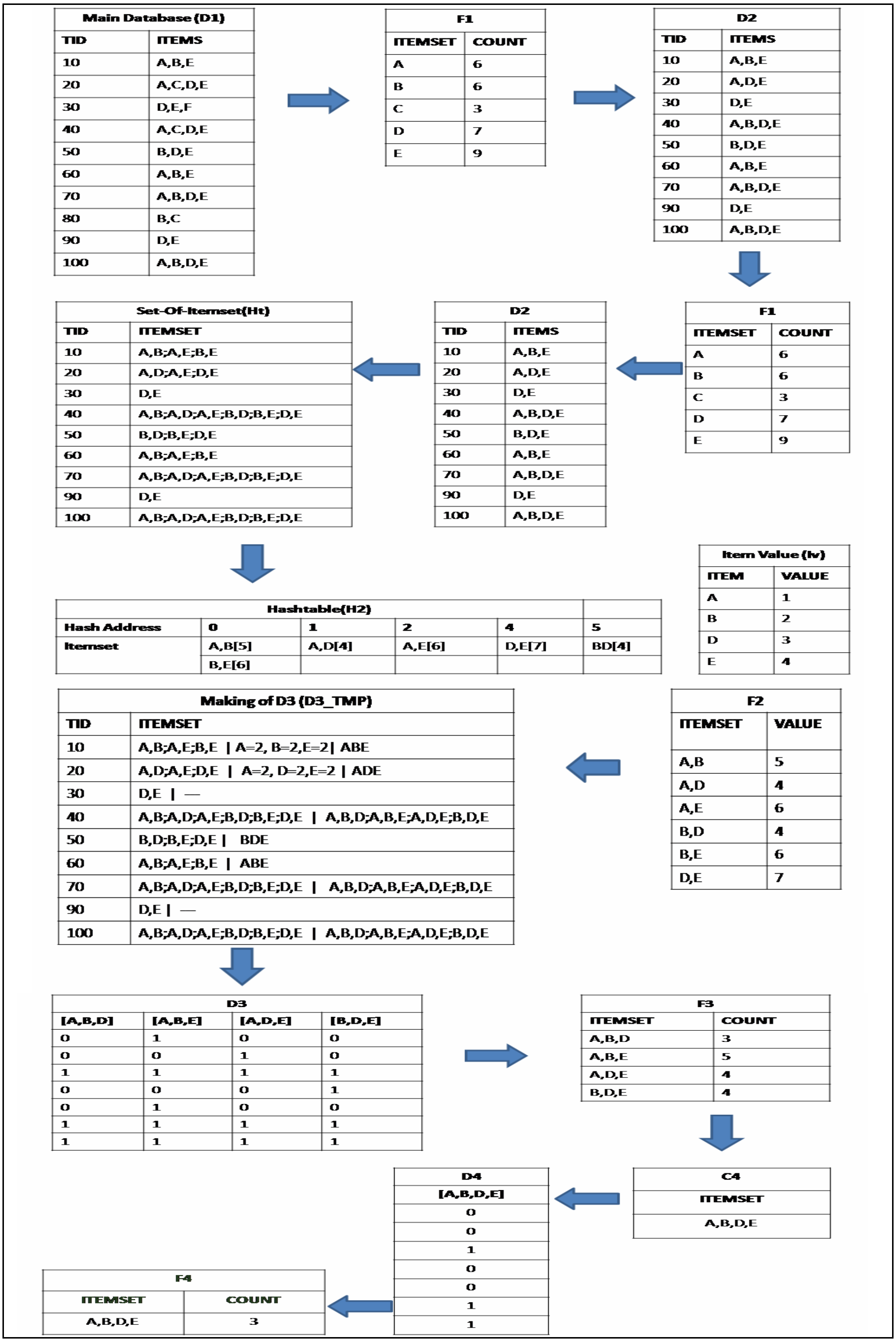
k = Implies the phases/passes/stages etc, k=1….n
D [1] = Main dataset, D [k+1] = In-memory pruned dataset
F [k] = Frequent k-Itemset, C [k] = Candidate k-Itemset
H [t] = Set-of-2-Itemset, H [2] = Hashtable
D[k] = Matrix table with binary inputs, for k=3 & above.
Phase-I, II is implemented using DHP algorithm with modifications & Phase-III, IV & above implemented using AprioriTid algorithm with modifications.

```
//PHASE-I: Generate F[k]:-
k = 1; D[1] = Main Database  ;
For each transaction t ∈ D[k] do begin
  For each item i ∈ t do begin //i=individual item in t
      i.count++;
  End
End
For each column i ∈ D[1] do begin
    If (i.count >= min_support) Then F[k] = F[k] U C;
End
//Prune D[k] & Generate H[t]:-
k = 2; D[k] = ∅; H[t] = Set-of-2Itemset;
For each transaction t ∈ D[k-1] do begin
   For each w ∈ t do begin    // w is [k-1] subset of t
      If (w ∈ F [k-1])  Then t1 = t1 U w;
   End
   D[k] = D[k] U t1;
   For each w ∈ t1 do begin //w=k-subset of t1 of D[k]
       H[t].Add (t1.TID,w);
   End
End
```

```
//PHASE-II : Generate Hashtable H[k]:-
For each transaction t in H[t] do begin
   For each subset w of t do begin
      x=w1, y=w2;
     H{xy}=[(order of x)*10+(order of y)]mod(itemcount+2)
      If (H{x,y} ∉ H[k] ) Then H[k] = {H{x,y} , xy};
      Else H[k]{x,y}++;
   End
End
//Generate F[k]:-
k=2; F[k] = ∅;
For each transaction j in H[k] do begin
    If (H[k].HasSupport(j))  Then F(k) = F(k) U j;
End
//Prune D[k-1] & Generate D[k]_TMP:-
k = 3; D[k]_Tmp = ∅ ; F[k] = ∅;
List[k] = ∅; //Unique itemset of D[k]_Tmp;
For each transaction t ∈ H(t) do begin
  For each subset w ∈ t do begin    // w = subset of t
      If (w ∈ F [k])  Then   t1 = t1 U w;
  End
  For each j ∈ t1 do begin //j=all items in t1
       j.count++;
  End
  t2= j ∈ t1 | j.count >= 2; //all valid items in t2
  Forall k subset n of t2 dobegin//n=k subset of items t2
    t3= t3 U n;
      If (n ∉ List[k])  Then List[k] = List[k] U n;
  End
    If (t3 ≠ ∅) Then D[k]_Tmp = D[k]_Tmp U t3;
End
//PHASE-III : Generate matrixtable with binary data:-
D(k)  = ∅;  //Pruned Matrix table
For each itemset I in List[k] do begin
    D[k].ColumnName (I)
End
For each transaction t ∈ D[k]_Tmp  do begin
 For all (k) subset w of t do begin
   If (D[k].ColumnName == w) Then D[k] Else D[k] = 0 ;
 End
End
//Generate F[k]:-
F(k) = ∅;
For each column C ∈ D[k] do begin
w = D[k].{Select C count(C) =1}If (w.count >=min_support)
Then F[k] = F[k] U C;Else D[k].Remove(C);
End
```

```
//PHASE-IV : Generate D[k] & F(k), for k >=4:-
For (k=4; F(k-1) ≠ ∅; k++) do begin
   D(k) = ∅; F(k) = ∅;
   C[k]= APRIORI-GEN (F(k-1)); //new candidates itemset
   For all  Items c ∈  C[k] do begin
     If (((k-1) subset s of c) ∈ D(k-1).ColumnName)
       Then List[k] = List[k] U  c;
   End
   D[k] = D[k]_MATRIXTABLE_GENERATION(List[k],D[k-1])
   For each column C ∈ D[k] do begin
     w = { Select C count (C) ==1 }
     If (w.count >= min_support) Then F[k] = F[k] U C ;
     Else D[k].Remove(C);
   End
End
Answer = F(k);
//D[K] MATRIXTABLE_GENERATION Function, for k>=4:-
D[k]_MATRIXTABLE_GENERATION(List[k],D[k-1])
{
   For each item I in  List[k] do begin
     D[k].ColumnName (I);
   End
   For each ColumnName C of D[k].ColumnName do begin
     For each row R of  D[k-1] do begin
      //s1, s2, s3…sn (k-1) subset of C,
      //R.s1 = value of s1 at at Rth row  at D[k-1]
      D[k][ColumnName] = R.s1 AND R.s2 AND R.s3 AND…R.sn
     End
   End
   Return D[k];
}
//APRIORI-GEN Function, for k>=4:-
APRIORI-GEN(F(k-1))
{//1.Join Step
   Insert Into C[k]
   Select p.item(1),p.item(2),…,p.item(k-1),q.item(k-1)
   From F(k-1)  p,  F(k-1)  p
   Where p.item(1)=q.item(1),…,p.item(k-2)=q.item (k-2)
   And p.item(k-1) < q.item(k-1) ;
   //2.Prune Step
   For all itemsets  c  ∈  C[k]do
      For all (k-1) subset s of c do
         If (s ∉ L(k-1) ) Then Delete c from C[k];
      End
   End
   Return C[k];
}
```

Let’s take a look on a simple example of DHPTID-HYBRID Algorithm.



Main Database (D1)

TID	ITEMS
10	A,B,E
20	A,C,D,E
30	D,E,F
40	A,C,D,E
50	B,D,E
60	A,B,E
70	A,B,D,E
80	B,C
90	D,E
100	A,B,D,E

F1

ITEMSET	COUNT
A	6
B	6
C	3
D	7
E	9

D2

TID	ITEMS
10	A,B,E
20	A,D,E
30	D,E
40	A,B,D,E
50	B,D,E
60	A,B,E
70	A,B,D,E
90	D,E
100	A,B,D,E

F1

ITEMSET	COUNT
A	6
B	6
C	3
D	7
E	9

D2

TID	ITEMS
10	A,B,E
20	A,D,E
30	D,E
40	A,B,D,E
50	B,D,E
60	A,B,E
70	A,B,D,E
90	D,E
100	A,B,D,E

Set-Of-Itemset(Ht)

TID	ITEMSET
10	A,B;A,E;B,E
20	A,D;A,E;D,E
30	D,E
40	A,B;A,D;A,E;B,D;B,E;D,E
50	B,D;B,E;D,E
60	A,B;A,E;B,E
70	A,B;A,D;A,E;B,D;B,E;D,E
90	D,E
100	A,B;A,D;A,E;B,D;B,E;D,E

Hashtable(H2)

Hash Address	0	1	2	4	5
Itemset	A,B[5]	A,D[4]	A,E[6]	D,E[7]	BD[4]
	B,E[6]				

Item Value (Iv)

ITEM	VALUE
A	1
B	2
D	3
E	4

F2

ITEMSET	VALUE
A,B	5
A,D	4
A,E	6
B,D	4
B,E	6
D,E	7

Making of D3 (D3_TMP)

TID	ITEMSET
10	A,B;A,E;B,E | A=2, B=2,E=2| ABE
20	A,D;A,E;D,E | A=2, D=2,E=2 | ADE
30	D,E | —
40	A,B;A,D;A,E;B,D;B,E;D,E | A,B,D;A,B,E;A,D,E;B,D,E
50	B,D;B,E;D,E | BDE
60	A,B;A,E;B,E | ABE
70	A,B;A,D;A,E;B,D;B,E;D,E | A,B,D;A,B,E;A,D,E;B,D,E
90	D,E | —
100	A,B;A,D;A,E;B,D;B,E;D,E | A,B,D;A,B,E;A,D,E;B,D,E

D3

[A,B,D]	[A,B,E]	[A,D,E]	[B,D,E]
0	1	0	0
0	0	1	0
1	1	1	1
0	0	0	1
0	1	0	0
1	1	1	1
1	1	1	1

F3

ITEMSET	COUNT
A,B,D	3
A,B,E	5
A,D,E	4
B,D,E	4

C4

ITEMSET
A,B,D,E

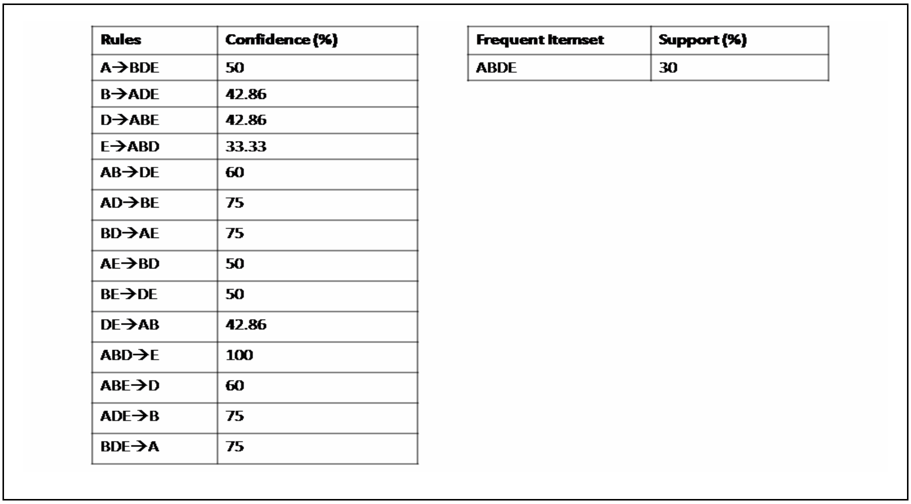
D4

[A,B,D,E]
0
0
1
0
0
1
1

F4

ITEMSET	COUNT
A,B,D,E	3

Fig. 1. DHPTID-Hybrid Algorithm: An example



Rules	Confidence (%)
A→BDE	50
B→ADE	42.86
D→ABE	42.86
E→ABD	33.33
AB→DE	60
AD→BE	75
BD→AE	75
AE→BD	50
BE→DE	50
DE→AB	42.86
ABD→E	100
ABE→D	60
ADE→B	75
BDE→A	75

Frequent Itemset	Support (%)
ABDE	30

Fig. 2. Association rules generated using frequent itemset A, B, D, E

3.2 Phases of DHPTID-Hybrid Algorithm

The DHPTID-Hybrid algorithm is the coupled algorithm, a resultant of coupling process between DHP and AprioriTid algorithm. Both the referenced algorithm has contributed to this proposed algorithm.

DHPTID-Hybrid = DHP (1) + AprioriTid (2)

The DHP algorithm is used as the first phase. It is basically used to generate frequent 1-itemset (F1), frequent 2-itemset (F2) and generate pruned in-memory dataset D(3). The reason for using DHP algorithm is due to the fact that DHP is effective for huge dataset. It is one of the best association rule algorithms that has capability o generate smaller candidate 2-itemset compare to other ARM algorithms. Its efficient hashing technique generates smaller candidate 2-itemset. With the help of smaller candidate 2-itemsets DHP efficient dataset pruning techniques prunes data effectively. These initial phases considered as the expensive and time taking process, but use of DHP at initial phase not only generates smaller candidate 2-itemset or prune dataset but it actually provides a performance hikes for later phases. As a result, the use of DHP provides performance hikes especially in expensive and lengthy initial phases. But at later phase, it work almost like Apriori algorithm. The experimental results already proved that just after the second pass i.e., after the creation of candidate 2-itemset, DHP gives almost the same performance like Apriori algorithm. So, using DHP for later phases is never going to be a smart choice[4].

Now, if we talked about the second phase, AprioriTid is implemented as the second phase algorithm. At later passes DHPTID-Hybrid algorithm uses AprioriTid algorithm for further computation of frequent itemsets. AprioriTid algorithm work upon the concept in-memory dataset and never revisit the main dataset again and again for creation of frequent itemset rather maintain an in-memory temporary dataset and pruned it after every pass. The pruning technique of AprioriTid is very efficient & effective as at later stages it gives marvelous results. AprioriTid is very effective with high minimum support or when more pruning is available. But the only problem is the expensive, lengthy initial phases. Experimental results already proved that AprioriTid algorithm has a performance issue at initial phases[1]. At, initial phases, AprioriTid perform lower than Apriori algorithm and the reason is large candidate itemset generation that results less or insignificant pruning. But at later passes it performs far better then Apriori algorithm[5].

From the above given facts about DHP and AprioriTid, it seems very clear that these two algorithm complements each other, means DHP is good at initial phase but average on later passes (similar to Apriori algorithm) , similarly, AprioriTid algorithm drooped at initial passes but raised at later passes. So, this way they both complement each other at initial and later phases and these facts are base of existence of DHPTID-Hybrid algorithm. With a binary approach in AprioriTid, smaller size matrix table generation with binary inputs for further computation of frequent itemsets can significantly improve the performance.

3.3 Characteristics of DHPTID-Hybrid Algorithm

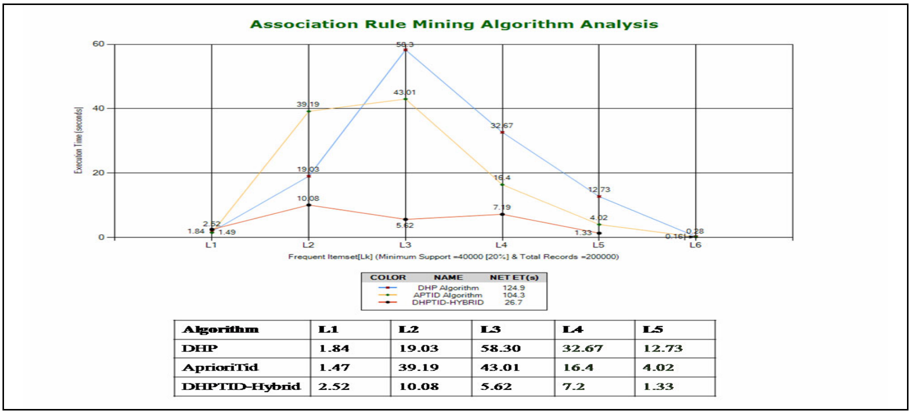
DHPTID-Hybrid algorithm has got some modification that ultimately improved overall performance of the algorithm. Here, explored the special five of DHPTID-Hybrid algorithm.

- Initial phase pruning for the generation of small sized hashtable
- Concise, faster & improved Hashing & Pruning Technique.
- Generation of smaller candidate 2-itemset.
- Smaller size matrix table generation with binary inputs
- Column vector/matrix addition using AND operation to prune database.

4 Experimental Results

Theoretical & Practical comparisons are performed using synthetic data. It has been previously analyzed that there are various factors that can influence the performance on an association rule mining algorithm. The performance gap varied with parameters such as total records size or database size, item size, row size or transaction size, minimum support count, etc. These factors can be considered as testing parameters and examined DHPTID-Hybrid Algorithm against the DHP & AprioriTid algorithm.

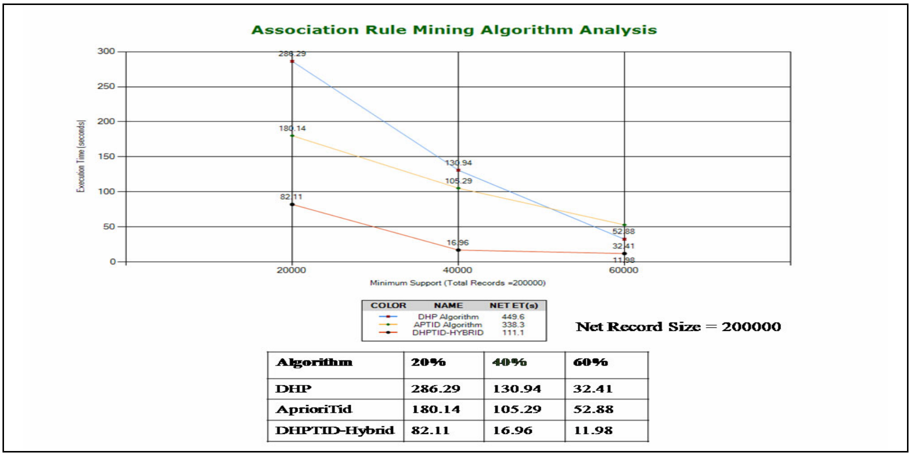
Phase-Wise Analysis- In this testing criterion, each phase of each algorithm is analyzed demonstrated in Fig.3. Minimum Support Analysis-Each algorithm is analyzed with fixed database size but on variety of minimum supports demonstrated in Fig.4. Database/Record Size Analysis-Each algorithm analyzed on fixed support but variety of database sizes demonstrated in Fig 5.



COLOR	NAME	NET ET(s)
—●—	DHP Algorithm	124.9
—●—	APTID Algorithm	104.3
—●—	DHPTID-HYBRID	26.7

Algorithm	L1	L2	L3	L4	L5
DHP	1.84	19.03	58.30	32.67	12.73
AprioriTid	1.47	39.19	43.01	16.4	4.02
DHPTID-Hybrid	2.52	10.08	5.62	7.2	1.33

Fig. 3. Phase wise execution time of all the algorithms

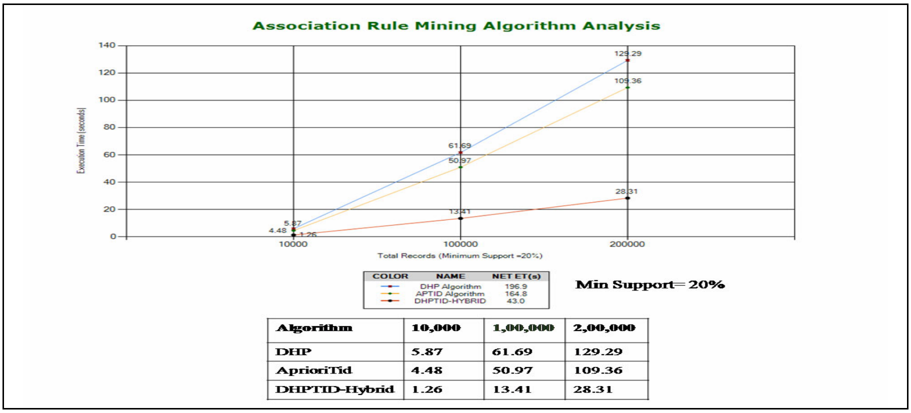


COLOR	NAME	NET ET(s)
—●—	DHP Algorithm	449.6
—●—	APTID Algorithm	338.3
—●—	DHPTID-HYBRID	111.1

Net Record Size = 200000

Algorithm	20%	40%	60%
DHP	286.29	130.94	32.41
AprioriTid	180.14	105.29	52.88
DHPTID-Hybrid	82.11	16.96	11.98

Fig. 4. Minimum Support Analysis



COLOR	NAME	NET ET(s)
—●—	DHP Algorithm	196.9
—●—	APTID Algorithm	164.8
—●—	DHPTID-HYBRID	43.0

Min Support= 20%

Algorithm	10,000	1,00,000	2,00,000
DHP	5.87	61.69	129.29
AprioriTid	4.48	50.97	109.36
DHPTID-Hybrid	1.26	13.41	28.31

Fig. 5. Record/Database Size Analysis

We have above all experimental results using synthetic data. This shows that the proposed algorithms always outperform AprioriTid and DHP algorithm.

5 Conclusions

The proposed coupled algorithm has the best performing features of two referenced algorithm. The new algorithm also got some modification such as Initial phase pruning, Modified & faster hashing technique, Matrix table generation with binary inputs, Column Matrix (or column vector) addition using logical AND operation on BITS. The proposed algorithm has various advantages such as, reduction in search space, faster computations of frequent itemset & most of all, overall performance improved by gradually reducing the computational cost at later stages.

Theoretical & practical evaluation and comparisons performed using synthetic data. Experimental results shows that the proposed algorithm always outperform.

References

1. Velu, C.M., Ramakrishnan, M., Somu, V., Loganathan, P., Vivekanandan, P.: Efficient Association Rule for Data Mining. Department of Mathematics. Anna University, Chennai (2007)
2. Tseng, J.C.R., Hwang, G.-J., Tsai, W.-F.: A Minimal Perfect Hashing Scheme to Mining Association Rules from Frequently Updated Data. Journal of the Chinese Institute of Engineers 29(3), 39–401 (2006)
3. Lam, S.N.: Discovering Association Rules in Data Mining. Department of Computer Science, University of Illinois at Urbana-Champaign (2006)
4. Chen, M.-S., Park, J.S., Yu, P.S.: An Effective Hash-Based Algorithm For mining Association Rules. IBM Thomas J.Watson Reasech Center (1995)
5. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. IBM Almaden Research Center, San Jose (1994)

An Improved Rough Clustering Using Discernibility Based Initial Seed Computation

Djoko Budiyanto Setyohadi, Azuraliza Abu Bakar, and Zulaiha Ali Othman

Center for Artificial Intelligence Technology University Kebangsaan Malaysia Bangi,
Selangor Darul Ehsan, 43000 Malaysia
djokobdy@gmail.com, {aab,zao}@ftsm.ukm

Abstract. In this paper, we present the discernibility approach for an initial seed computation of Rough K-Means (RKM). We propose the use of the discernibility initial seed computation (ISC) for RKM. Our proposed algorithm aims to improve the performance and to avoid the problem of an empty cluster which affects the numerical stability since there are data constellations where $|\underline{Ck}| = 0$ in RKM algorithm. For verification, our proposed algorithm was tested using 8 UCI datasets and validated using the David Bouldin Index. The experimental results showed that the proposed algorithm of the discernibility initial seed computation of RKM was appropriate to avoid the empty cluster and capable of improving the performance of RKM.

Keywords: Discernibility, Initial Seed Computation, Rough K-Means.

1 Introduction

Clustering is a process of classifying objects into classes based on similarities among data. The process of assigning an object to its cluster is fully based on the data similarity; therefore the characteristics of data may influence the clustering result. The performance of K-Means, as the most widely used clustering algorithm, depends on two key points, namely the initial clustering and the instance order [1]; in which initial clustering itself fully depends on the data distribution. Since the characteristic of the data influences the performance of K-means, many improvements of K-means are being developed. Rough K-means clustering (RKM) [2] is one of the well known extended K-means algorithm.

RKM is the clustering algorithm which addresses the problem of vague data. Its capability to cluster vague data comes from the integration of Rough Set Theory in the process of clustering. While in the original K-Means the cluster is viewed as a crisp cluster only, in RKM the cluster is deployed as an interval clustering. Here, the object is divided in the lower approximation where the object is certainly a member of the cluster, and the boundary area where the object is a member of more than one cluster [2]. Looking at its characteristics, RKM can be considered as a powerful algorithm for clustering vague data. Vague data can be clustered in a boundary area which is useful for further processing.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 161–168, 2010.

Despite its advantages, RKM has a drawback especially on the numerical stability problem [3][4][5]. The problem arises because RKM equation requires that each cluster must have at least a member. This situation is also found in the original K-means and is solved by an initial seed computation [1][6][7][8][9][10]. Unlike in the original K-means, the empty cluster in the RKM will generate the numerical stability problem since there are data constellations where $|\underline{Ck}| = 0$, which refers to the computation of cluster centroid [2][11]. Therefore, several researchers have made improvements on the numerical stability problem [3][4][5].

According to the numerical stability problem, Peters [3] refined RKM by forcing at least one of the objects should be a member of the cluster. Hence one of the objects which is the closest to the centroid of the lower approximation will be assigned to the closest cluster. Miao [4] avoided the empty cluster by using the non-object outlier to the proper cluster and proposed the use of angle measurement to decide the member of clusters. Obviously, all of the previous work on RKM refinement, including that of Zhao [5] and of Lingras [11], focused on the membership function refinement. Although previous researchers had improved the RKM, they ignored the other source of the numerical stability problem i.e. the initial seed, since K-means clustering certainly relies on the chosen initial centroid [1][6][7]. Moreover, when the algorithm is applied, the boundary area should be restricted to avoid a numerical stability problem [3][4][5]. Therefore, to fill the gap of the previous work that heavily focused on refining the membership function to avoid numerical stability this work highlights the initial seed computation to avoid a numerical stability problem.

Many ISCs have been developed since the process of the K-means clustering is deterministic mapping from initial solution to local minima of final result [1][11]. The previous research showed that the use of the ISC did not only improve the performance of K-means but also was able to avoid the empty cluster problem that plagues K-Means. Hence we propose the use of ISC to avoid a numerical stability problem in RKM as an extension of K-means.

In this paper, we review the required characteristics of the previous ISC works, from which we further develop the algorithm based on the discernibility approach of Rough Set Theory which is suitable for the purpose of RKM i.e. processing the vague data. To verify the proposed algorithm, we use David Bouldin (DB) Index, which is a well-known validity measurement in clustering analysis [12].

2 Initial Seed Computation (ISC) on K-Means Clustering

Determining the initial seed points is very important in K-means since the initial centroid will determine the final centroid [1]. The main issue of the initial seed is that the initial centroid should be chosen properly. Currently, there are many studies focusing on the ISC for improving K-means algorithm in order to improve the result of clustering [1][6][7][8][9][10], which is also applicable in RKM to solve its numerical stability problem. Furthermore, the previous characteristics will be discussed below.

There are three properties commonly used as the guidance for initial seed determination, namely, centrality, sparseness and isotropy. According to the characteristics, Arthur [6] suggested a method to choose the area of initial seed, that is, after the first initial seed is selected the next initial seed should be chosen in the uncovered space of its cluster. Kang [7] improved the performance of clustering by following the properties that the initial centorid should be distant enough to each other but close to the final centroid. Redmond et al. used Density Generated Kd-tree [9], a top-down hierarchical scheme, to isolate data. Suppose we have a set of n points, $(x_1 \ldots x_n)$, it will be divided roughly by splitting the data along the median value of the co-ordinates in that dimension, median $(x_{1mmax} ; x_{2mmax} ; \ldots ; x_{nmmax})$. Then the properties of each partition will be the basis ISC. Although they use different approaches to satisfy the properties initial seed, their results are able to improve the performance of K-means significantly. The results from the previous research do not only improve the performance of K-means clustering but also can avoid an empty cluster. However, the previous methods have some limitations, such as the high processing time since they contain complex computation with *O(N2).* Referring to the characteristics of initial seed, we propose the use of the discernibility concept of Rough Set for ISC. Discernibility is one of the important characteristics of Rough Set Theory [13], and therefore it is suitable for the main purpose of RKM.

3 Discernibility of Rough Set Theory (RST)

Discernibility, an important property in RST, is the relation of two objects which can be defined as $\text{discern}_A(B) = \{(x,y) \in U2 : a(x) \neq a(y), \forall\ a \in B\}$. Given an IS A = (U, A), and a subset of attributes $B \subseteq A$, the discernibility matrix of A is M_B, where each entry $m_B(i,j)$ consists of the attribute set that discerns between objects x_i and x_j. The discernibility function of A over attributes $B \subseteq A$ can be defined as $f[B] = V \Lambda\ m_{[B]} (E_i ,E_j)$ where $i, j \in \{1,...,n\}$ and n is the number of classes in the IS. Suppose, given a subset of attributes $A \subseteq A_t$ and a pair of objects $(x_i , x_j) \in U$ x U, i,j $\in\{1, 2, ..., |U|\}$, the quantitative discernibility relation $dis(A)(x_i , x_j)$ is defined as the complement of a quantitative indiscernibility.

$$dis(A)(x_i, x_j) \qquad = 1 - ind(A)(x_i, x_j). \tag{1}$$

satisfies to the properties $dis(A)(x_i , x_i) = 0$ and $dis(A)(x_i, x_j) = dis(A)(x_j , x_i)$ where the quantitative discernibility relation is reflexive and symmetric. The discernibility level, which is possible to represent the granularity of objects, can be used to measure the discernibility among objects. The higher level of discernibility implies that the objects are likely to be treated as discernible. This discernibility will be the basis for our ISC.

4 Proposed Discernibility Based Initial Seed Computation

The objective of ISC is to place the initial centroid close to the intrinsic centroids. Using its objective, the clustering algorithm should rapidly converge to the global optimal structure and the problem of the empty cluster can be avoided. On the other hand, the good cluster should ensure that each centroid should be in the middle of the cluster (minimizing intra-cluster variance), and the centroids should be distant enough to each other (maximizing inter-cluster variance). The proposed algorithm tries to satisfy this objective. We first discretize the entire objects in data sets. This process will be followed by the calculation of the discernibility level which is considered as the degree of a distant object. Then, we select the best combination of initial seed space. The best combination is measured by the highest discernibility level which indicates the highest degree of object distance.

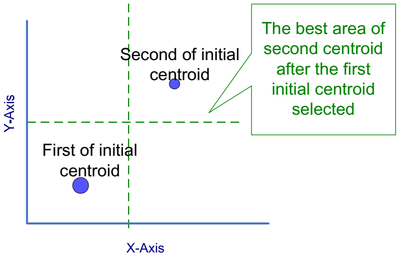

Second of initial centroid
The best area of second centroid after the first initial centroid selected
Y-Axis
First of initial centroid
X-Axis

Fig. 1. The example of best search space combinations of initial centroid point for data set of two attributes and two clusters

The degree of distance is very relative and difficult to determine precisely. Therefore, we adopt the discernibility of RST to introduce the discernibility concept for initialization of seed points. We use the discernibility level terminology to measure the appropriate degree of the distance of initial centroid of RKM. Since the discernibility approach for data separation has the computation load problem, the proposed technique is focused on the discernibility relation of a binary table which is generated from any appropriate discretization method. This approach aims to partition objects roughly in the data set. Referring to the properties of initial seed [6][7], the best chosen initial seed can be achieved when a discernibility degree (α) is maximum as follows $\alpha_A(x_1,\ldots, x_n) = \text{MAX}[\alpha_A(x_i, x_j)]$. Fig. 1 illustrates an example of how a discernibility concept is used. Suppose the data set has two attributes and is identified to have two clusters, we can divide the spaces of initial seed into four spaces using the binary classification. Each attribute will have two spaces as illustrated in Fig. 1.

Using one of the properties of the initial seed which is proved [6][7], the best combination space of initial seed area is achieved when the space area is the highest possible distance. This condition can be achieved when the discernibility degree is maximum. As illustrated in Fig. 1, assuming the first of initial centroids space is in quadrant one then, the best second initial centroids space is in quadrant four. Further the discernibility degree α between two objects can be calculated as in Eq.(2).

$$\alpha_A(x_i, x_j) = \frac{|\{a \in A \mid I_a(x_i) \neq I_a(x_i)\}|}{|A|} \tag{2}$$

Where |A| denotes the cardinality of a set and two objects x_i and x_j, for $\alpha_A(x_i, x_j)$ for $x_i \neq x_j$. Therefore, the initial seed points for the cluster are the points x_i, x_j that give the highest degree α for n object. It can be defined as the maximum degree of any degree α two objects as formulated in Eq(3) where $i \neq j$, $i,j = 1,2, ..., n$.

$$\alpha_A(x_1,..., x_n) = \mathrm{MAX}[\alpha_A(x_i, x_j)] \tag{3}$$

For example, for more than two objects, the discernibility degree α can be computed as $\alpha_A(x_1, x_2,...x_n) = \mathrm{MAX}\{\alpha(x_1,x_2), \alpha(x_1, x_3),..., \alpha(x_1, x_n)\}$. In order to satisfy that all objects have the highest discernibility, the discernibility degree is obtained as the maximum of $\alpha_A(x_1, x_2,...x_n)$. By using this approach, the complexity is related to the binary searching of a space area; therefore, the complexity of discernibility ISC is only about O($n$log$n$). Another advantage of using ISC for RKM is that the maximum discernibility will lead the process in order to keep the consistency and its convergence. Thus, this computation can control the influence of the threshold value. The proposed method is outlined in the following steps.

Step 1. Initial seed computation

1. Convert the information system into the binary classification.
2. Calculate the discernibility degree α for every pair of data points.
3. Find the maximum α among the objects x_i, x_j
4. Select x_i, x_j as the initial seed for RKM algorithm.

Step 2. RKM Clustering

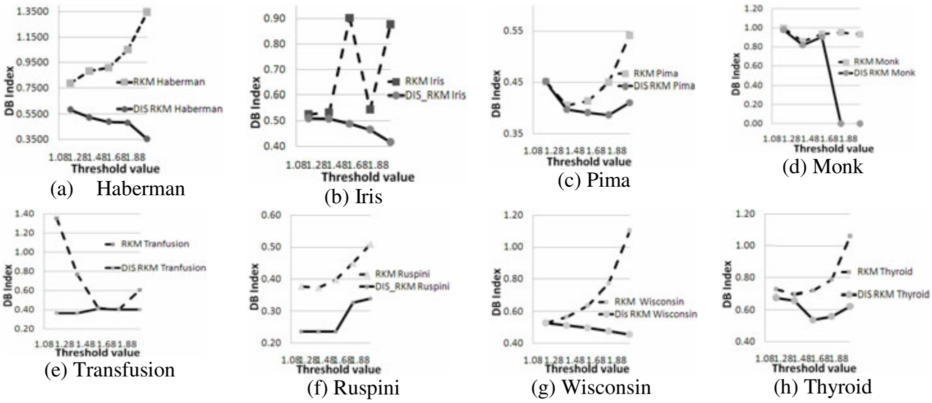
1. Initiate the centroid using the selected object from the first step
2. Calculate the new means of RKM as in [11]
3. Calculate membership of each object of RKM as in [11]
4. Repeat from step 2.2. until convergence.

5 Experiment and Discussion

In this study we tested our proposed method called DIS_RKM upon eight UCI datasets. The datasets are Iris, Monk, Pima, Wisconsin, Ruspini, Haberman, Transfusion, and Thyroid. Two measures were used in the experiment: i) the percentage of data points in the Lower Approximation (%Lower) and ii) the DB_Index. The DB Index was calculated using the formulation in Eq(4).

$$DB = \frac{1}{c}\sum_{i=1}^{c} \max_{j \neq i}\left\{\frac{S(U_i) + S(U_j)}{d(U_i, U_j)}\right\} \tag{4}$$

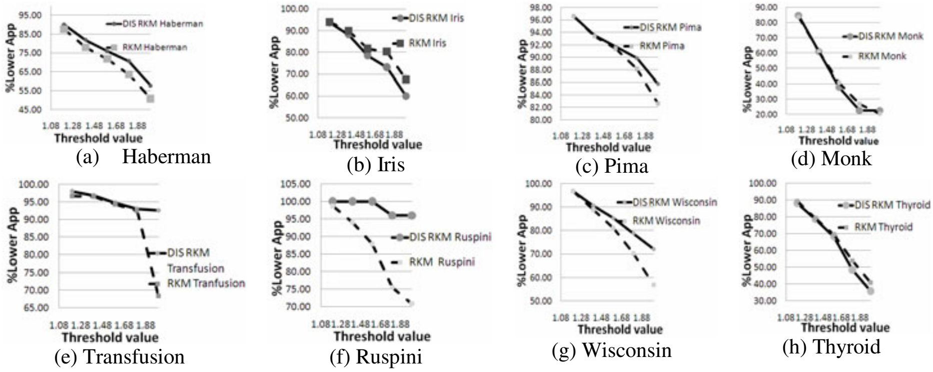
The good RKM is that the values of %Lower and DB Index decrease as the threshold values increase until the centroid move. The proposed algorithm was implemented using Java Netbean 6.8, Processor Intel T9660 Ram 3 GByte and Windows operating system. The experiment used the threshold value to simulate the consistency and stability of the method.



(a) Haberman (b) Iris (c) Pima (d) Monk

(e) Transfusion (f) Ruspini (g) Wisconsin (h) Thyroid

Fig. 2. Change in DB Index with threshold in eight UCI data sets



(a) Haberman (b) Iris (c) Pima (d) Monk

(e) Transfusion (f) Ruspini (g) Wisconsin (h) Thyroid

Fig. 3. Change in %Lower App. with threshold in eight UCI data sets

Fig. 2 and Fig. 3 showed the experimental results obtained using our proposed discernibility ISC for RKM called DIS_RKM and the random initial seed setting for RKM via the original RKM. The DB Index was obtained by using different threshold values. The results showed that the DIS_RKM yielded a lower DB Index than RKM in all of data sets and all of threshold values (Fig. 2), except on Pima data sets where in threshold value 1.2. Since the lower DB Index value represented a better cluster, we concluded that DIS_RKM was able to improve the performance of RKM.

Referring to the membership function in RKM, the threshold values increased, the %Lower decreased and the size of the boundary area increase is means that more vague values are moved to the boundary area. Based on DB Index simulation, the compactness of the cluster improved since the intra-cluster distance was reduced and inter-cluster distance was retained when threshold value increased. However, we can see that the original RKM was not consistent with the changes of thresholds. The DB Index values decreased to a certain level but increased when it reached the threshold 1.6. The situation also proved the robustness of DIS_RKM. By simulating the threshold we can see that DIS_RKM was consistent since it led to the similar centroid where the DB Index values decreased as the threshold values increased. Larger threshold values indicated the larger size of a boundary area. In our proposed approach, the distance between centroid was retained while the threshold changed, but the size and compactness of the lower approximation and boundary area changed. The decreasing values of DB Index were achieved by reducing the average distance among the objects within clusters while the distance among the clusters was retained when the threshold value increased. This condition can be achieved if the centroid did not move when the threshold increased.

The threshold determined directly the membership of the object which influenced the boundary area. Therefore, changing of thresholds will change the boundary area. Theoretically, it is sufficient to control the vague objects of the cluster while maintaining the DB Index. Based on Fig. 3 the %Lower measure showed that DIS_RKM was comparable with the RKM where the increasing values of threshold were followed by the decreasing %Lower approximation. Referring to the turning point of the threshold change, the point where the direction of DB Index changed from a lower to a higher point, the range of DIS_RKM was longer than that of the original RKM. It indicated that the proposed algorithm was more robust. Moreover discernibility based ISC was able to improve the random seed setting by the original RKM. In this experiment we also showed that the higher the threshold, the lower the DB_Index and the %Lower approximation where the size of the boundary area was controlled properly.

According to the issue of the numerical stability DIS_RKM outperforms in four data sets (Haberman, Tranfusion, Wisconsin, Ruspini,) and comparable in data sets (Pima, Monk) due to a slower decline of %Lower DIS_RKM. This result indicated that the ability of DIS_RKM to avoid an empty cluster was better than the original RKM. Furthermore, the final centroid of DIS_RKM converged at a certain point.

In our experiment, the DIS_RKM did not change the rule but improved it by making the RKM simpler since the lower approximation component was decreased gradually. Therefore, we were able to control the boundary area as needed, except on the Ruspini data set where the separation of the cluster was deterministic (not vague). In Ruspini dataset, we were able to show that the ISC improved the performance of RKM in a natural way, not through forcing the data of the cluster. The 100% value of %Lower indicated that no data were forced to the boundary even though the threshold values increased and the data clearly belonged to the lower approximation space.

6 Conclusion

We have introduced the discernibility ISC to improve RKM. This approach is based on the properties of good clusters that the centroids should be distant enough to each other. To implement the ISC we proposed the use of binary discernibility in order to reduce the high computation problem. We observed that the proposed ISC was able to improve the robustness, to enhance the performance and to avoid the empty cluster problem of RKM, particularly when the threshold increased. The experimental results showed improved and consistent clusters as compared to the initial random cluster centers.

Acknowledgments. I would like to thank Atma Jaya University Yogyakarta, Indonesia for the financial support for my research project, UKM for conference funding. I would also like to express my gratitude to Prof. P. Lingras for providing me with the original code of RKM and his RKM papers.

References

1. Peña, J.M., Lozano, J.A., Larrañaga, P.: An empirical comparison of four initialization methods for the K-Means algorithm. Pattern Recognition Letters archive 20(10), 1027–1040 (1999)
2. Lingras, P., West, C.: Interval Set Clustering of Web Users with Rough K-means. Journal of Intelligent Information System 23(1), 5–16 (2004)
3. Peters, G.: Some Refinement of K-means Clustering. Pattern Recognition 39, 1481–1491 (2006)
4. Miao, D.Q., Chen, M., Wei, Z.H., Duan, Q.G.: A Reasonable Rough Approximation of Clustering Web Users. In: Zhong, N., Liu, J., Yao, Y., Wu, J., Lu, S., Li, K. (eds.) Web Intelligence Meets Brain Informatics. LNCS (LNAI), vol. 4845, pp. 428–442. Springer, Heidelberg (2007)
5. Zhou, T., Zhang, Y.N., Lu, H.L.: Rough k-means Cluster with Adaptive Parameters. In: Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, August 19-22,, pp. 3063–3068 (2007)
6. Arthur, D., Vassilvitskii, S.: k-means++: The advantages of careful seeding. In: Proc. ACM-SIAM Symp. Discrete Algorithms (2007)
7. Khang, P., Cho, S.: *K*-means clustering seeds initialization based on centrality, sparsity, and isotropy. In: Corchado, E., Yin, H. (eds.) IDEAL 2009. LNCS, vol. 5788, pp. 109–117. Springer, Heidelberg (2009)
8. Khan, S.S., Ahmad, A.: Cluster Center initialization algorithm for K-means clustering. Pattern Recognition Letter 25(11), 1293–1302 (2004)
9. Redmond, S.J., Heneghan, C.: A method for initializing the k-means clustering algorithm using kd-trees. Pattern Recognition Letters 28(8), 965–973 (2007)
10. He, J., Lan, M., Tan, C.-L., Sung, S., Low, H.-B.: Initialization of cluster refinement algorithms: A review and comparative study. In: Proc. IEEE Int. Joint Conf. Neural Networks, pp. 297–302 (2004)
11. Lingras, P., Chen, M., Miao, D.: Rough Cluster Quality Index Based on Decision Theory. IEEE Transactions On Knowledge And Data Engineering 21 (2009)
12. Halkidi, M., Batistakis, Y., Vazirgianni, M.: On Clustering Validation Techniques. Journal of Intelligent Information Systems 17(2/3), 107–145 (2001)
13. Pawlak, Z.: Rough Set: Theoretical Aspect of Reasoning about Data. Kluwer Publications, Dordrecht (1991)

Fixing the Threshold for Effective Detection of Near Duplicate Web Documents in Web Crawling

V.A. Narayana[1], P. Premchand[2], and A. Govardhan[3]

[1] Department of Computer Science & Engineering,
CMR College of Engineering & Technology,
Hyderabad, India
narayanaphd@gmail.com
[2] Department of Computer Science & Engineering
University College of Engineering, Osmania University
Hyderabad, AP, India
p.premchand@uceou.edu
[3] JNTUH College of Engineering
Nachupally (Kondagattu), Karimnagar Dist, AP, India
govardhan_cse@yahoo.co.in

Abstract. The drastic development of the WWW in recent times has made the concept of Web Crawling receive remarkable significance. The voluminous amounts of web documents swarming the web have posed huge challenges to web search engines making their results less relevant to the users. The presence of duplicate and near duplicate web documents in abundance has created additional overheads for the search engines critically affecting their performance and quality which have to be removed to provide users with the relevant results for their queries. In this paper, we have presented a novel and efficient approach for the detection of near duplicate web pages in web crawling where the keywords are extracted from the crawled pages and the similarity score between two pages is calculated. The documents having similarity score greater than a threshold value are considered as near duplicates. In this paper we have fixed the threshold value.

Keywords: Fingerprint, Similarity score, Near-duplicate, Web crawling and Threshold.

1 Introduction

The employment of automated tools to locate the information resources of interest, and for tracking and analyzing the same, has become inevitable these days owing to the drastic development in the information accessible on the World Wide Web. This has made the development of server side and client side intelligent systems mandatory for efficient knowledge mining [1]. A branch of data mining that deals with the analysis of World Wide Web is known as Web Mining. Web Mining owes its origin to concepts from diverse areas such as Data Mining, Internet technology and World Wide Web, and lately, Semantic Web. Web mining includes the sub areas: web

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 169–180, 2010.

content mining, web structure mining, and web usage mining and can be defined as the procedure of determining hidden yet potentially beneficial knowledge from the data accessible in the web. The process of mining knowledge from the web pages besides other web objects is known as Web content mining. Web structure mining is the process of mining knowledge about the link structure linking web pages and some other web objects. The mining of usage patterns created by the users accessing the web pages is called Web usage mining. The World Wide Web owes its development to the Search engine technology. The chief gateways for access of information in the web are Search engines. Businesses have turned beneficial and productive with the ability to locate contents of particular interest amidst a huge heap [17]. Web crawling, a process that populates an indexed repository of web pages is utilized by the search engines in order to respond to the queries [10]. The programs that navigate the web graph and retrieve pages to construct a confined repository of the segment of the web that they visit. Earlier, these programs were known by diverse names such as wanderers, robots, spiders, fish, and worms, words in accordance with the web imagery.

Generic and Focused crawling are the two main types of crawling. Generic crawlers [3] differ from focused crawlers in a way that the former crawl documents and links of diverse topics whereas the latter limits the number of pages with the aid of some prior obtained specialized knowledge. Repositories of web pages are built by the web crawlers so as to present input for systems that index, mine, and otherwise analyze pages (for instance, the search engines) [2]. The subsistence of near duplicate data is an issue that accompanies the drastic development of the Internet and the growing need to incorporate heterogeneous data [11]. Even though the near duplicate data are not bit wise identical they bear a striking similarity [11]. Web search engines face huge problems due to the duplicate and near duplicate web pages. These pages either increase the index storage space or slow down or increase the serving costs thereby irritating the users. Thus the algorithms for detecting such pages are inevitable [12]. Web crawling issues such as freshness and efficient resource usage have been addressed previously [4], [5], [6]. Lately, the elimination of duplicate and near duplicate web documents has become a vital issue and has attracted significant research [8].

Identification of the near duplicates can be advantageous to many applications. Focused crawling, enhanced quality and diversity of the query results and identification on spams can be facilitated by determining the near duplicate web pages [9, 12, 15]. Numerous web mining applications depend on the accurate and proficient identification of near duplicates. Reduction in storage costs and enhancement in quality of search indexes besides considerable bandwidth conservation can be achieved by eliminating the near duplicate pages [3]. Check summing techniques can determine the documents that are precise duplicates (because of mirroring or plagiarism) of each other [7]. The recognition of near duplicates is a tedious problem.

Research on duplicate detection was initially done on databases, digital libraries, and electronic publishing. Lately duplicate detection has been extensively studied for the sake of numerous web search tasks such as web crawling, document ranking, and document archiving. A huge number of duplicate detection techniques ranging from manually coded rules to cutting edge machine learning techniques have been put forth [11, 12, 13, 20 - 23]. Recently few authors have projected near duplicate detection techniques [14, 24, 25, 26]. A variety of issues such as from providing high detection rates to minimizing the computational and storage resources have been addressed by them. These techniques vary in their accuracy as well. Some of these techniques are computationally pricey to be implemented completely on huge collections. Even though some of these algorithms prove to be efficient they are fragile and so are susceptible to minute changes of the text.

The primary intent of our research is to develop a novel and efficient approach for detection of near duplicates in web documents. Initially the crawled web pages are preprocessed using document parsing which removes the HTML tags and java scripts present in the web documents. This is followed by the removal of common words or stop words from the crawled pages. Then the stemming algorithm is applied to filter the affixes (prefixes and the suffixes) of the crawled documents in order to get the keywords. Finally, the similarity score between two documents is calculated on basis of the extracted keywords. The documents with similarity scores greater than a predefined threshold value are considered as near duplicates.

The rest of the paper is organized as follows. Section 2 presents a brief review of some approaches available in the literature for duplicates and near duplicates detection. In Section 3, the novel approach for the detection of near duplicate documents is presented. In Section 4, the experimental results are presented. The conclusions are summed up in Section 4.

2 Related Work

Our work has been inspired by a number of previous works on duplicate and near duplicate document and web page detection. A system was proposed for registering documents and then detecting copies, either complete copies or partial copies [20]. Algorithms were described for detection, and metrics required for evaluating detection mechanisms covering accuracy, efficiency and security. An efficient way to determine the syntactic similarity of files was developed and was applied it to every document on the World Wide Web [21]. The extent and the types of duplication existing in large textual collections were determined [22]. Mechanism for measuring the intermediate kinds of similarity was explored, focusing on the task of identifying where a particular piece of information originated [23]. The use of simple text clustering and retrieval algorithms for identifying near-duplicate public comments was explored [24]. It was focused on automating the process of near-duplicate detection, especially form letter detection. A clear near-duplicate definition was given and explored simple and efficient methods of using feature-based document retrieval and similarity-based clustering to discover near-duplicates.

The comparison of the two algorithms namely shingling algorithm [21] and random projection based approach [7] on a very large scale set of 1.6B distinct web pages was done [12]. The results showed that neither of the algorithms works well for finding near-duplicate pairs on the same site, while both achieve high precision for near-duplicate pairs on different sites. A combined algorithm was presented which achieves precision 0.79 with 79% of the recall of the other algorithms. DURIAN (DUplicate Removal In lArge collectioN), a refinement of a prior near-duplicate detection algorithm [25]. In the course of developing a near-duplicate detection system for a multi-billion page repository, two research contributions were made [14]. First, it was demonstrated that fingerprinting technique [7] is appropriate for this goal. Second, an algorithmic technique was presented for identifying existing f-bit fingerprints that differ from a given fingerprint in at most k bit-positions, for small k. This technique is useful for both online queries (single fingerprints) and batch queries (multiple fingerprints). Exact similarity join algorithms with application to near duplicate detection and a positional filtering principle was proposed, which exploits the ordering of tokens in a record and leads to upper bound estimates of similarity scores [11]. They demonstrated the superior performance of their algorithms to the existing prefix filtering-based algorithms on several real datasets under a wide range of parameter settings.

3 Novel Approach For Near Duplicate Webpage Detection

A novel approach for the detection of near duplicate web pages is presented in this section. In web crawling, the crawled web pages are stored in a repository for further process such as search engine formation, page validation, structural analysis and visualization, update notification, mirroring and personal web assistants or agents and more. Duplicate and near duplicate web page detection is an important step in web crawling. In order to facilitate search engines to provide search results free of redundancy to users and to provide distinct and useful results on the first page, duplicate and near duplicate detection is essential. Numerous challenges are encountered by the systems that aid in the detection of near duplicate pages. First is the concern of scale since the search engines index hundreds of millions of web-pages thereby amounting to a multi-terabyte database. Next is the issue of making the crawl engine crawl billions of web pages everyday. Thus marking a page as a near duplicate should be done at a quicker pace. Furthermore, the system should utilize minimal number of machines [14].

The near duplicate detection is performed on the keywords extracted from the web documents. First, the crawled web documents are parsed to extract the distinct keywords. Parsing includes removal of HTML tags, java scripts, stop words/common words and stemming of remaining words. The extracted keywords and their counts are stored in a table to ease the process of near duplicates detection. The keywords are stored in the table in a way that the search space is reduced for the detection. The similarity score of the current web document against a document in the repository is calculated from the keywords of the pages. The documents with similarity score greater than a predefined threshold are considered as near duplicates.

3.1 Near Duplicate Web Documents

Even though the near duplicate documents are not bitwise identical they bear striking similarities. The near duplicates are not considered as “exact duplicates” but are files with minute differences. Typographical errors, versioned, mirrored, or plagiarized documents, multiple representations of the same physical object, spam emails generated from the same template, and many such phenomenon’s may result in near duplicate data. A considerable percentage of web pages have been identified as be near-duplicates according to various studies [12 and 15]. These studies propose that near duplicates constitute almost 1.7% to 7% of the web pages traversed by crawlers. The steps involved in our approach are presented in the following subsections.

3.2 Web Crawling

The analysis of the structure and informatics of the web is facilitated by a data collection technique known as Web Crawling. The collection of as many beneficiary web pages as possible along their interconnection links in a speedy yet proficient manner is the prime intent of crawling. Automatic traversal of web sites, downloading documents and tracing links to other pages are some of the features of a web crawler program. Numerous search engines utilize web crawlers for gathering web pages of interest besides indexing them. Seed URLs are a set of URLs that a crawler begins working with. These URLs are queued. A URL is obtained in some order from the queue by a crawler. Then the crawler downloads the page. This is followed by the extracting the URLs from the downloaded page and enqueuing them. The process continues unless the crawler settles to stop [16]. A crawling loop consists of obtaining a URL from the queue, downloading the corresponding file with the aid of HTTP, traversing the page for new URLs and including the unvisited URLs to the queue.

3.3 Web Document Parsing

Information extracted from the crawled documents aid in determining the future path of a crawler. Parsing may either be as simple as hyperlink/URL extraction or complex ones such as analysis of HTML tags by cleaning the HTML content. It is inevitable for a parser that has been designed to traverse the entire web to encounter numerous errors. The parser tends to obtain information from a web page by not considering a few common words like a, an, the, and more, HTML tags, Java Scripting and a range of other bad characters.

3.4 Stop Words Removal

It is necessary and beneficial to remove the commonly utilized stop words such as "it", "can" ,"an", "and", "by", "for", "from", "of", "the", "to", "with" and more either while parsing a document to obtain information about the content or while scoring fresh URLs that the page recommends. This procedure is termed as stop listing.

3.5 Stemming Algorithm

Variant word forms in Information Retrieval are restricted to a common root by Stemming. The postulation lying behind is that, two words posses the same root represent identical concepts. Thus terms possessing to identical meaning yet appear morphologically dissimilar are identified in an IR system by matching query and document terms with the aid of Stemming [19]. Stemming facilitates the reduction of all words possessing an identical root to a single one. This is achieved by removing each word of its derivational and inflectional suffixes [18]. For instance, "connect," "connected" and "connection" are all condensed to "connect".

3.6 Keywords Representation

We possess the distinct keywords and their counts in each of the each crawled web page as a result of stemming. These keywords are then represented in a form to ease the process of near duplicates detection. This representation will reduce the search space for the near duplicate detection. Initially the keywords and their number of occurrences in a web page have been sorted in descending order based on their counts. The similarity score between two documents can be calculated if and only the prime keywords of the two documents are similar. Thus the search space is reduced for near duplicates detection.

3.7 Similarity Score Calculation

If the prime keywords of the new web page do not match with the prime keywords of the pages in the table, then the new web page is added in to the repository. If all the keywords of both pages are same then the new page is considered as duplicate and thus is not included in the repository. If the prime keywords of new page are same with a page in the repository, then the similarity score between the two documents is calculated. The similarity score of two web documents is calculated as follows:

Let T1 and T2 be the tables containing the extracted keywords and their corresponding counts.

T_1	K_1	K_2	K_4	K_5	.....	K_n
	C_1	C_2	C_4	C_5	.....	C_n
T_2	K_1	K_3	K_2	K_4	.....	K_n
	C_1	C_3	C_2	C_4	.....	C_n

The keywords in the tables are considered individually for the similarity score calculation. If a keyword is present in both the tables, the formula used to calculate the similarity score of the keyword is as follows.

$$a = Index\ [K_i]_{T_1}\ ,\ \ b = Index\ [K_i]_{T_2} \qquad (1)$$

$$S_{D_C} = \log(count(a)/count(b)) * Abs(1+(a-b)) \qquad (2)$$

Here ‘a’ and ‘b’ represent the index of a keyword in the two tables respectively. If the keywords of T1 / T2 ≠ φ, we use the formula to calculate the similarity score.

$$S_{D_{T_1}} = \log(\,count\ (a)) * \left(1 + |T_2|\right) \qquad (3)$$

If the keywords of T2 / T1 ≠ φ, we use the below mentioned formula to calculate the similarity score.

$$S_{D_{T_2}} = \log(\,count\ (b)) * \left(1 + |T_1|\right) \qquad (4)$$

The similarity score (SSM) of a document against another document is calculated by using the following equation.

$$SS_M = \frac{\sum_{i=1}^{|N_C|} S_{D_C} + \sum_{i=1}^{|N_{T_1}|} S_{D_{T_1}} + \sum_{i=1}^{|N_{T_2}|} S_{D_{T_2}}}{N} \qquad (5)$$

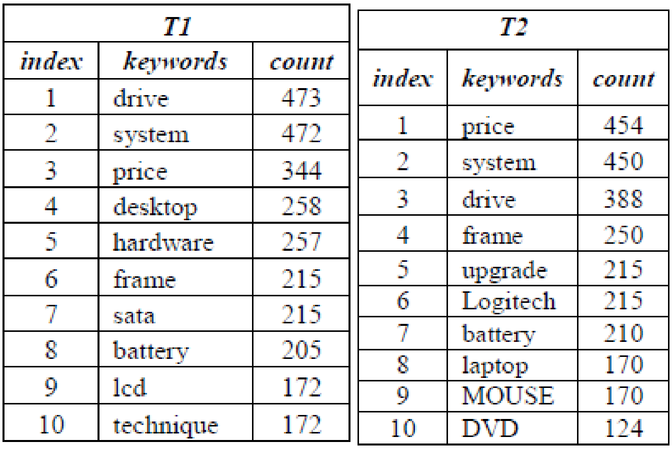
Where the amount of the keywords present in T1 but not in T2 is taken as N_{T_1} and the keywords present in T2 but not in T1 is taken as N_{T_2}

$n = |T_1 \cup T_2|$ and $N = (|T_1| + |T_2|)/2$. The web documents with similarity score greater than a predefined threshold are near duplicates and these are the documents already present in repository. These near duplicates are not added in to the repository for further process such as search engine indexing.

4 Experimental Results

The results of our experiments are presented in this section. The proposed approach is implemented in Java with MS Access as backend. The keywords extracted from the web documents are stored in MS Access. In our experiments, we have analyzed the proposed approach using many documents. The near duplicates have been detected efficiently by the proposed approach. This has been achieved with the aid of the similarity scores. The similarity score would find all pairs of web pages whose duplications are identified by a given predefined threshold. Upon considering the similarity scores, a lower distance measure indicates that the documents are more similar and hence are regarded as near duplicates. If the extracted keywords from the two web documents are almost same, then it is considered as near duplicate and their distance is of minimum value. One such example for near duplicate web documents, the extracted keywords and their calculated similarity scores are as follows:

Document 1 a document
Document 2 another document which is not near duplicate
The keywords from Document 1 and Document 2 are given in Fig. 1.

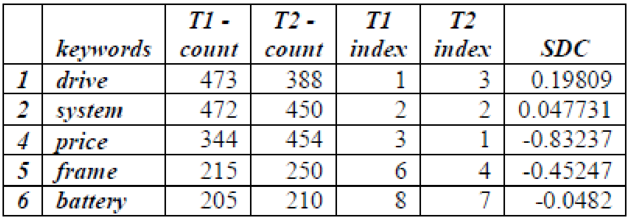


T1		
index	*keywords*	*count*
1	drive	473
2	system	472
3	price	344
4	desktop	258
5	hardware	257
6	frame	215
7	sata	215
8	battery	205
9	lcd	172
10	technique	172

T2		
index	*keywords*	*count*
1	price	454
2	system	450
3	drive	388
4	frame	250
5	upgrade	215
6	Logitech	215
7	battery	210
8	laptop	170
9	MOUSE	170
10	DVD	124

Fig. 1. T1--Keywords from Doc 1, T2--Keywords from Doc 2

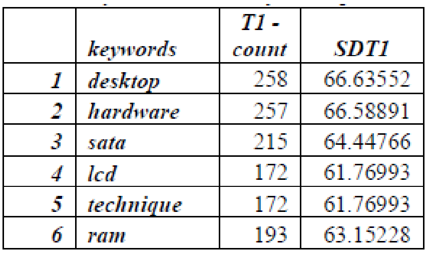
The similarity scores calculated for the keywords present in both the tables are as follows:



	keywords	*T1 - count*	*T2 - count*	*T1 index*	*T2 index*	*SDC*
1	*drive*	473	388	1	3	0.19809
2	*system*	472	450	2	2	0.047731
4	*price*	344	454	3	1	-0.83237
5	*frame*	215	250	6	4	-0.45247
6	*battery*	205	210	8	7	-0.0482

Fig. 2. Similarity scores of the keywords present in T1 and T2

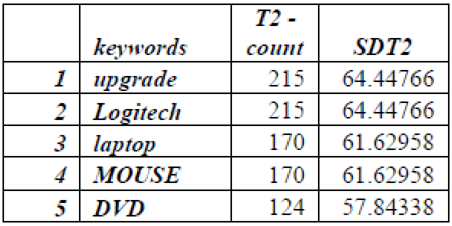
The similarity scores calculated for the keywords present in T1 but not present in T2 are as follows:



	keywords	*T1 - count*	*SDT1*
1	*desktop*	258	66.63552
2	*hardware*	257	66.58891
3	*sata*	215	64.44766
4	*lcd*	172	61.76993
5	*technique*	172	61.76993
6	*ram*	193	63.15228

Fig. 3. Similarity scores of the keywords present in T1 but not in T2

The similarity scores calculated for the keywords present in T2 but not present in T1 are as follows:



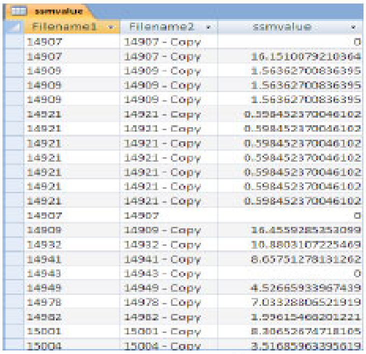
	keywords	*T2 - count*	*SDT2*
1	*upgrade*	215	64.44766
2	*Logitech*	215	64.44766
3	*laptop*	170	61.62958
4	*MOUSE*	170	61.62958
5	*DVD*	124	57.84338

Fig. 4. Similarity scores of the keywords present in T2 but not in T1

Finally, we have calculated the similarity score SSM between document 1 and document 2 with the aid of the similarity scores in Fig. 2-4 and the SSM value is 60.2848. Consequently we have compared this SSM value with the predefined threshold value and this value is greater than the predefined threshold. Based on this, we have concluded that the document 1 and document 2 are not near duplicates.

sum=(SDC+SDT1+SDT2)	693.275
N = (T1 + T2)/2=(11+12)/2=	11.5
SSM = sum/N	60.2848

In our approach of fixing the threshold, we have considered two documents which are duplicates. Then we have added some content in the form of advertisements or counter or timestamp to the duplicate document so that now it becomes a near duplicate document. We have then calculated the similarity scores of both the documents, and according to above formulae we have calculated SSMs for 7,00,000 documents and formulated the SSM values in a table as below and the SSM values are also plotted as shown in Fig. 5.


ssmvalue

Filename1	Filename2	ssmvalue
14907	14907 - Copy	0
14907	14907 - Copy	16.1510079210364
14909	14909 - Copy	1.56362700836395
14909	14909 - Copy	1.56362700836395
14909	14909 - Copy	1.56362700836395
14921	14921 - Copy	0.598452370046102
14921	14921 - Copy	0.598452370046102
14921	14921 - Copy	0.598452370046102
14921	14921 - Copy	0.598452370046102
14921	14921 - Copy	0.598452370046102
14921	14921 - Copy	0.598452370046102
14921	14921 - Copy	0.598452370046102
14907	14907	0
14909	14909 - Copy	16.4559285253099
14932	14932 - Copy	10.8803107225469
14941	14941 - Copy	8.65751278131262
14943	14943 - Copy	0
14949	14949 - Copy	4.52665933967439
14978	14978 - Copy	7.03328806521919
14982	14982 - Copy	1.99615468201221
15001	15001 - Copy	8.30652674718105
15004	15004 - Copy	3.51685963395619

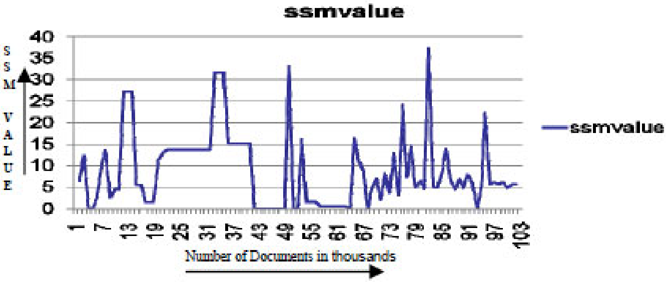


Fig. 5. The SSM value is found to be around 20

From the above graph we find that, generally the value of SSM is less than 20 for most of the documents and then from the above results we calculated the average of all the above SSM values and finalized the approximate threshold value as 19.5043 While experimenting we found some interesting patterns. Some of them are

a. The number of characters present in the advertisements are also affecting the value of SSM. With this information we calculated some results and formulated in a table and constructed graph as follows. As the number of characters increase the SSM value also increases.

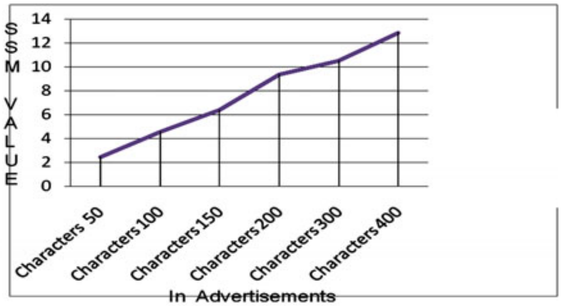


Fig. 6. The SSM value increases as the number of characters in advertisements in document increases

b. Even though timestamps varies, the SSM value is 0. The reason is that the count of keyword is 1.When we are calculating SSM value we are using log i.e, log(1)=0. So there will be no effect on SSM value.
c. When we are calculating SSM there may be a chance of getting negative value, which has occurred when log(count(a)/count(b)) lies between 0 and 1. Hence we may change the formula as mod(log(count(a)/count(b))).

5 Conclusions

Though the web is a huge information store, various features such as the presence of huge volume of unstructured or semi-structured data; their dynamic nature; existence of duplicate and near duplicate documents poses serious difficulties for effective information retrieval. The voluminous amounts of web documents swarming the web have posed a huge challenge to the web search engines making them render results of less relevance to the users. The detection of duplicate and near duplicate web documents has gained more attention in recent years amidst the web mining researchers. In this paper, we have presented a novel and efficient approach for detection of near duplicate web documents in web crawling. The proposed approach has detected the duplicate and near duplicate web pages efficiently based on the keywords extracted from the web pages. Further more, reduced memory spaces for web repositories and improved search engine quality have been accomplished through the proposed duplicates detection approach.

References

1. Narayana, V.A., Premchand, P., Govardhan, A.: A Novel and Efficient Approach For Near Duplicate Page Detection in Web crawling. In: IEEE International Advance Computing Conference, Patiala, pp. 1492–1496 (2009)
2. Pant, G., Srinivasan, P., Menczer, F.: Crawling the Web. In: Web Dynamics: Adapting to Change in Content, Size, Topology and Use. Springer, Heidelberg (2004)

3. Balamurugan, S., Rajkumar, N.: Design and Implementation of a New Model Web Crawler with Enhanced Reliability. Proceedings of World Academy of Science, Engineering and Technology 32 (2008) ISSN 2070-3740
4. Menczer, F., Pant, G., Srinivasan, P., Ruiz, M.E.: Evaluating topic-driven web crawlers. In: Proc. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 241–249. ACM, New Orleans (2001)
5. Broder, A.Z., Najork, M., Wiener, J.L.: Efficient URL caching for World Wide Web crawling. In: International Conference on World Wide Web, pp. 679–689. ACM, Budapest (2003)
6. Chakrabarti, S.: Mining the Web: Discovering Knowledge from Hypertext Data. Morgan Kaufmann, San Francisco (2002)
7. Cho, J., Garcia-Molina, H., Page, L.: Efficient crawling through URL ordering. Computer Networks and ISDN Systems 30(1-7), 161–172 (1998)
8. Charikar, M.: Similarity estimation techniques from rounding algorithms. In: Proc. 34th Annual Symposium on Theory of Computing (STOC 2002), pp. 380–388. ACM, Montreal (2002)
9. Cho, J., Shivakumar, N., Garcia-Molina, H.: Finding replicated web collections. ACM SIGMOD Record 29(2), 355–366 (2000)
10. Conrad, J.G., Guo, X.S., Schriber, C.P.: Online duplicate document detection: signature reliability in a dynamic retrieval environment. In: CIKM, pp. 443–452. ACM, New Orleans (2003)
11. Pandey, S., Olston, C.: User-centric Web crawling. In: Proceedings of the 14th International Conference on World Wide Web, pp. 401–411. ACM, Chiba (2005)
12. Xiao, C., Wang, W., Lin, X.M., Xu Yu, J.: Efficient Similarity Joins for Near Duplicate Detection. In: Proceeding of the 17th International Conference on World Wide Web, pp. 131–140. ACM, Beijing (2008)
13. Henzinger, M.: Finding near-duplicate web pages: a large-scale evaluation of algorithms. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 284–291. ACM, Seattle (2006)
14. Castillo, C.: Effective web crawling. ACM SIGIR Forum 39(1), 55–56 (2005)
15. Manku, G.S., Jain, A., Sarma, A.D.: Detecting near-duplicates for web crawling. In: Proceedings of the 16th International Conference on World Wide Web, pp. 141–150. ACM, Banff (2007)
16. Gibson, D., Kumar, R., Tomkins, A.: Discovering large dense subgraphs in massive graphs. In: VLDB, pp. 721–732. ACM, Trondheim (2005)
17. Spertus, E., Sahami, M., Buyukkokten, O.: Evaluating similarity measures: a large-scale study in the orkut social network. In: KDD, pp. 678–684. ACM, Chicago (2005)
18. Singh, A., Srivatsa, M., Liu, L., Miller, T.: Apoidea: A Decentralized Peer-to-Peer Architecture for Crawling the World Wide Web. In: Proceedings of the SIGIR 2003 Workshop on Distributed Information Retrieval. LNCS, pp. 126–130. ACM, Toronto (2003)
19. Lovins, J.B.: Development of a stemming algorithm. Mechanical Translation and Computational Linguistics 11, 22–31 (1968)
20. Bacchin, M., Ferro, N., Melucci, M.: Experiments to evaluate a statistical stemming algorithm. In: Peters, C., Braschler, M., Gonzalo, J. (eds.) CLEF 2002. LNCS, vol. 2785, pp. 161–168. Springer, Heidelberg (2003)
21. Brin, S., Davis, J., Garcia-Molina, H.: Copy detection mechanisms for digital documents. ACM SIGMOD Record 24(2), 398–409 (1995)
22. Broder, A.Z., Glassman, S.C., Manasse, M.S., Zweig, G.: Syntactic clustering of the web. In: Proceedings of WWW6 1997, pp. 391–404. Elsevier Science, Santa Clara (1997)

23. Conrad, J., Schriber, C.P.: Online duplicate document detection: signature reliability in a dynamic retrieval environment. In: Proceedings of the Twelfth International Conference on Information and Knowledge Management, pp. 443–452. ACM, New Orleans (2003)
24. Metzler, D., Bernstein, Y., Bruce Croft, W.: Similarity Measures for Tracking Information Flow. In: Proceedings of the Fourteenth International Conference on Information and Knowledge Management, CIKM 2005, pp. 517–524. ACM, Bremen (2005)
25. Yang, H., Callan, J.: Near-duplicate detection for eRulemaking. In: Proceedings of the 2005 National conference on Digital Government Research, pp. 78–86. Digital Government Society of North America, Atlanta (2005)
26. Yang, H., Callan, J., Shulman, S.: Next steps in near-duplicate detection for eRulemaking. In: Proceedings of the 2006 International Conference on Digital Government Research, pp. 239–248. ACM, San Diego (2006)

Topic-Constrained Hierarchical Clustering for Document Datasets

Ying Zhao

Department of Computer Science and Technology
Tsinghua University
Beijing, China 100084
yingz@tsinghua.edu.cn

Abstract. In this paper, we propose the *topic-constrained hierarchical clustering*, which organizes document datasets into hierarchical trees consistant with a given set of topics. The proposed algorithm is based on a constrained agglomerative clustering framework and a semi-supervised criterion function that emphasizes the relationship between documents and topics and the relationship among documents themselves simultaneously. The experimental evaluation show that our algorithm outperformed the traditional agglomerative algorithm by 7.8% to 11.4%.

Keywords: Constrained hierarchical clustering, Semi-supervised learning, Criterion functions.

1 Introduction

Incorporating prior knowledge into the clustering process has drawn people's attention recently. Such knowledge could be a given set of labeled data or pair-wise object relatoinships (in case of *semi-supervised clustering* [1,2,3,4]), or an entire or a part of the desired clustering structure (in case of *evolutionary clustering* [5] and clustering under *hierarchical constrtaints* [6]), or a given set of topics or concepts that the datasets contains (in case of *topic-driven clustering* [7]). Regardless of the format of the prior knowledge, the aim of all those clustering problems is to benefit from the prior knowledge to produce more desired clustering solutions.

In [7], we have defined the problem of *topic-driven clustering*, in which case we want to organize a document collection according to a given set of topics. For example, the knowledge management of a law firm would like to organize their legal documents according to the major topics provided by their law librarians based on their knowledge on the practice areas, related law categories, and custom base of the law firm. Another example is to organize the documents from a web bulletin board according to hot topics and current news identified from other sourses to help browsing through the bulletin board. We have discussed the *topic-driven clustering* for flat clustering structure in [7]. In this paper, we focus on the problem in hierarchical clustering setting, because hierarchies are ideal for

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 181–192, 2010.

document interactive visualization and exploration as they provide data-views that are consistent, predictable, and at different levels of granularity.

More formally, we propose the **topic-constraint hierarchical clustering** problem in this paper, in which case a hierarchical tree should be built based on a given set of topics. In other words, a good clustering solution must have two properties: the documents that are more similar to each other should be merged first; documents from a same topic should be merged before documents from different topics. Towards this end, we propose a topic-constraint hierarchical clustering algorithm based on a constrained agglomerative clustering framework and a semi-supervised criterion function to produce cluster constraints that emphasizes the relationship between documents and topics and the relationship among documents themselves simultaneously.

The topic-constraint hierarchical clustering differs from other constrained hierarchical clustering in both the way how the prior knowledage is presented and the way how the prior knowledage is used. In addition, we present a comprehensive experimental evaluation using various datasets and our experimental results show that the proposed topic-constraint hierarchical clustering algorithms is effective with topic prototypes of different levels of specificity.

The rest of this paper is organized as follows. Section 2 dicusses some of the related work. Section 3 describes the criterion functions that are used in various steps of the proposed algorithm. Section 4 describes the algorithms that optimizes the various criterion functions and the topic-constraint hierarchical clustering algorithm. Section 5 provides the detailed experimental evaluation of the various proposed algorithms. Finally, Section 6 provides some concluding remarks.

2 Related Work

The work in constrained hierarchical clustering can be grouped along two dimensions: the way how the prior knowledage is presented and the way how the prior knowledage is used. *Semi-supervised clustering* [1,2,3,4] assumes the prior knowledge (background knowledge) is given by a limited set of labeled data, from which the knowledge of two objects should belong to the same cluster (must-link) or should not belong to the same cluster (cannot-link) can be derived. *Evolutionary clustering* [5] uses a known historical hierarchy to direct forming the new hierarchy. Recently, clustering under *hierarchical constrtaints* [6] assumes a known partial hierarchy to constrain the clustering process.

In terms of how the piror knowledge being used, previous semi-supervised approaches fall into three categories: instance-based, metric-based and the combined approaches. Instance-based approaches explicitly modify the objective function or make certain constraints using must- and cannot-links during the clustering process[1,4,6]. Whereas, metric-based approaches parametrize distance metric and learn the metric parameters in a manner, so that the distance between objects connected by must-links is smaller and the distance between objects connected by cannot-links is larger in general [2,3,6]. Finally the combined approaches integrate both of these techniques in the clustering process [8]. The topic-constraint hierarchical clustering differs from the constrained hierarchical clustering discussed above in both the way how the prior knowledage is presented and the way how the prior knowledage is used. Specifically, in the case of topic-constraint hierarchical clustering, the prior knowledage is neither labeled data, nor some existent hierarchy, but the descriptions of possible topics.

3 Criterion Functions

The essence of clustering consists of two components in most cases: a criterion function that describes the perfect clustering result would be; and an optimization process to achieve a clustering result guided by the criterion function. The optimization process can be partitional, agglomerative, or a mixture of both. In this section, we discuss the criterion functions used in various steps of the proposed topic-constrained agglomerative schemes.

3.1 Document Representation

We use the symbols n, m, and k to denote the number of documents, the number of terms, and the number of clusters, respectively. We use the symbol S to denote the set of N documents that we want to cluster, $S_1, S_2, \ldots, S_k$ to denote each one of the k clusters, $n_1, n_2, \ldots, n_k$ to denote the sizes of the corresponding clusters, and $T_1, T_2, \ldots, T_k$ to denote the topic prototype vectors given as prior knowledge.

The various criterion functions use the vector-space model [9] to represent each document. In this model, each document d is considered to be a vector in the term-space. In particular, we employed the $tf - idf$ term weighting model, in which each document can be represented as

$$(tf_1 \log(n/df_1), tf_2 \log(n/df_2), \ldots, tf_m \log(n/df_m)).$$

where tf_i is the frequency of the ith term in the document and df_i is the number of documents that contain the ith term. To account for documents of different lengths, the length of each document vector is normalized so that it is of unit length ($\|d_{tfidf}\| = 1$), that is each document is a vector on the unit hypersphere.

Given a set A of documents and their corresponding vector representations, we define the ***composite*** vector D_A to be $D_A = \sum_{d \in A} d$, and the ***centroid*** vector C_A to be $C_A = D_A/|A|$.

In the vector-space model, the cosine similarity is the most commonly used method to compute the similarity between two documents d_i and d_j, which is defined to be $\cos(d_i, d_j) = {d_i}^t d_j/(\|d_i\|\|d_j\|)$. The cosine formula can be simplified to $\cos(d_i, d_j) = {d_i}^t d_j$, when the document vectors are of unit length. This measure becomes one if the documents are identical, and zero if there is nothing in common between them (*i.e.*, the vectors are orthogonal to each other).

3.2 Criterion Function for Agglomeration

The UPGMA criterion [10] (also known as group average) is the most popular criterion when the clustering process is done agglomeratively. UPGMA measures the similarity of two clusters as the average of the pairwise similarity of the documents from each cluster. That is,

$$\mathrm{sim}_{\mathrm{UPGMA}}(S_r, S_t) = \frac{1}{n_i n_j} \sum_{d_i \in S_r,\ d_j \in S_t} \cos(d_i, d_j) = \frac{{D_i}^t D_j}{n_i n_j}. \tag{1}$$

3.3 Unsupervised Criterion Function

The internal unsupervised criterion function, denoted by $\mathcal{U}_{\mathcal{I}}$ ([2]) is used by the popular vector-space variant of the K-means algorithm [11,12,13,14]. In this algorithm each cluster is represented by its centroid vector and the goal is to find the solution that maximizes the similarity between each document and the centroid of the cluster that is assigned to.

$$\mathcal{U}_{\mathcal{I}} = \sum_{r=1}^{k} \sum_{d_i \in S_r} \cos(d_i, C_r) = \sum_{r=1}^{k} \|D_r\|. \tag{2}$$

3.4 Supervised Criterion Function

We assume that the description of each topic is available as prior knowledge and can be represented as a vector. Given these topic prototype vectors, the similarity between each document and its topic can be defined as the cosine similarity between the vector of the document d and the prototype vector of the topic T_r. The internal supervised criterion function, denoted by $\mathcal{S}_{\mathcal{I}}$, tries to maximize the similarity between the documents in a cluster to the topic associated with the cluster. The formal definition can be written as

$$\mathcal{S}_{\mathcal{I}} = \sum_{r=1}^{k} \sum_{d_i \in S_r} \cos(d_i, T_r) = \sum_{r=1}^{k} {D_r}^t T_r. \tag{3}$$

3.5 Semi-supervised Criterion Function

In our previous study [7], we have studied various ways of combining unsupervised and supervised criterion functions as a mulit-objective optimization problem. In this paper, we focus on the *weighted* scheme, which allows fine-tuning of the tradeoffs among the objectives with relatively less complexity. Given two criterion functions X and Y, the weighted scheme can be written as

$$M(X, Y) = \alpha X + (1 - \alpha) Y, \tag{4}$$

where α is the preference factor.

The weighted scheme allows a fine-tuned control of the tradeoffs among the objectives by varying the preference factor α. In this paper, we use the weighted scheme to combine $\mathcal{U}_{\mathcal{I}}$ and $\mathcal{S}_{\mathcal{I}}$ together to form $\mathcal{M}_{\mathcal{I}}$, which tries to maximize the similarity between the documents in a cluster to the cluster centriod and the topic associated with the cluster simultaneously.

$\mathcal{M}_{\mathcal{I}} = M(\mathcal{U}_{\mathcal{I}}, \mathcal{S}_{\mathcal{I}}) = \alpha \sum_{r=1}^{k} \|D_r\| + (1-\alpha) \sum_{r=1}^{k} {D_r}^t T_r$

4 Topic-Constrained Hierarchical Clustering

In this section, we will discuss the proposed topic-constrained hierarchical clustering algorithm that produces hierarchical clustering results consistent with given topics. Our topic-constrained hierarchical clustering algorithm is based on a constrained agglomerative clustering framework.

4.1 Constrained Agglomerative Clustering

Traditional agglomerative algorithms build the solution by initially assigning each document to its own cluster and then repeatedly selecting and merging pairs of clusters, to obtain a hierarchical clustering tree from bottom (*i.e.*, its leaves) toward the top (*i.e.*, root).

Traditional agglomerative algorithms may suffer from early bad merging decisions when the documents are not part of particularly cohesive groups, and these errors may be multiplied as the aggloemration progesses. In [15], we proposed a framework called *constrained agglomerative clustering* to help eliminating this type of errors.

In this approach, the clustering process consists of three steps. First, a partitional clustering algorithm is used to compute a k-way clustering solution. Then, each of these clusters, referred as *constraint clusters*, is treated as a separate collection and an agglomerative algorithm is used to build a tree for each one of them. Finally, the k different trees are combined into a single tree by merging them using an agglomerative algorithm that treats the documents of each subtree as a cluster that has already been formed during agglomeration. The choice of the partitional algorithm and agglomerative algorithm in each step would be the one that suits the dataset best, and could be different. In this paper, we use UPGMA as the criterion for agglomeration.

4.2 Topic-Constrained Agglomerative Clustering

Given the *constrained agglomerative clustering* framework, a straight-forward way of incorporating given topics is to make *constraint clusters* represent these topics. Suppose the partitional clustering process of generating constraint clusters tries to optimize a criterion function $\mathcal{CF}$ using an optimization procedure $\mathcal{P}$, we refer the constrained agglomerative clustering algorithm as $\mathcal{CF}$-constaints agglomerative.

$\mathcal{CF}$ can be any criterion functions that we discussed in Section 3. When $\mathcal{CF}$ is $\mathcal{U}_{\mathcal{I}}$, the constraint clusters are generated in a unsupervised fashion. Whereas using $\mathcal{S}_{\mathcal{I}}$, the process of generating constraint clusters is supervised. Our goal is to have constaint clusters of good quality so that agglomeration based on them indeed improves clustering quality, and consistent with the given topics so that the final hierarchical tree is consistent with the given topics as well. Towards this goal, we propose to use the semi-supervised criterion function $\mathcal{M}_{\mathcal{I}}$ as $\mathcal{CF}$, and we call the algorithm *topic-constrained agglomerative clustering.*

The optimization procedure $\mathcal{P}$ is similar to that used in [14], in which a selection of k seeds for the k clusters is followed by a *incremental* refinement. The refinement strategy consists of a number of iterations. During each iteration, the documents are visited in a random order. For each document, d_i, we compute the change in the value of the criterion function obtained by moving d_i to the other cluster. If there exist some moves that lead to an improvement in the overall value of the criterion function, then d_i is moved to the cluster that leads to the highest improvement. If no such move exists, d_i remains in the cluster that it already belongs to. The refinement phase ends as soon as we perform an iteration in which no documents are moved between clusters. When optimizing $\mathcal{S}_{\mathcal{I}}$ and $\mathcal{M}_{\mathcal{I}}$, the initial seeds are the topic vectors, whereas for $\mathcal{U}_{\mathcal{I}}$, the initial seeds are selected randomly from the entire dataset.

5 Experiments

5.1 Datasets and Metrics

In our experiments, we used a total of three different datasets as in [7], whose general characteristics are summarized in Table 1. The datasets *trec6*, *trec7* and *trec8* were derived from the Financial Times Limited (FT) and the Los Angeles Times (LATimes) articles that are distributed as part of the TREC collection [16]. We used the queries of the ad hoc test from TREC-6 [16], TREC-7 [16] and TREC-8 [16] as the topic prototypes, and derived the datasets by including all the relevant document in FT and LATimes to particular queries. The queries that have fewer than 10 relevant documents were eliminated from the datasets. For all datasets, we used a stop-list to remove common words, and the words were stemmed using Porter's suffix-stripping algorithm [17]. Moreover, any term that occurs in fewer than two documents was eliminated.

Each TREC query contains a title, a description and a narrative. The title usually contains 2-3 words as the key words. The description describes what are the contents of the relevant document briefly, and the narrative provides more detailed descriptions. Thus, we could use the titles, descriptions and narratives to form the topic prototypes of different levels of specificity. In particularly, we used the titles to form short topics, titles and descriptions to form medium topics, and all three parts to form long topics.

The quality of a clustering solution was determined by analyzing how the documents of the different classes are distributed in the nodes of the hierarchical trees produced by the various algorithms and was measured using the *FScore measure* [12,15,6] , which identifies for each class of documents the node in the hierarchical tree that best represents it and then measures the overall quality of

Table 1. Summary of datasets used to evaluate the various clustering criterion functions

Dataset	Topic #	Source	# of Docs	# of terms	# of classes
trec6	301-350	FT & LATimes	2619	32790	38
trec7	351-400	FT & LATimes	2838	33963	45
trec8	401-450	FT & LATimes	2804	36347	43

the tree by evaluating this subset of clusters. In determining how well a cluster represents a particular class, the FScore measure treats each cluster as if it was the result of a query for which all the documents of the class were the desired set of relevant documents. Given such a view, then the suitability of the cluster to the class is measured using the F value that combines the standard precision and recall functions used in information retrieval. Specifically, given a particular class L_r of size n_r and a particular cluster S_i of size n_i, suppose n_r^i documents in the cluster S_i belong to L_r, then the F value of this class and cluster is defined to be

$$F(L_r, S_i) = \frac{2 * R(L_r, S_i) * P(L_r, S_i)}{R(L_r, S_i) + P(L_r, S_i)},$$

where $R(L_r, S_i) = n_r^i / n_r$ is the recall value and $P(L_r, S_i) = n_r^i / n_i$ is the precision value defined for the class L_r and the cluster S_i. The FScore of class L_r is the maximum F value attained at any node in the hierarchical tree T. That is,

$$\mathit{FScore}(L_r) = \max_{S_i \in T} F(L_r, S_i).$$

The FScore of the entire hierarchical tree is defined to be the sum of the individual class specific FScores weighted according to the class size. That is,

$$\mathit{FScore} = \sum_{r=1}^{c} \frac{n_r}{n} \mathit{FScore}(L_r),$$

where c is the total number of classes. A perfect clustering solution will be the one in which every class has a corresponding cluster containing the same set of documents in the resulting hierarchical tree, in which case the FScore will be one. In general, the higher the FScore values, the better the clustering solution is.

5.2 Comparison of Various Hierarchical Clustering Algorithms

We evaluated our topic-constrained agglomerative algorithms by comparing the FScore values of the clustering results obtained by the following four shemes. The traditional agglomerative algorithm with UPGMA served as our baseline. Then, three constrained agglomerative algorithms were evaluated against the baseline, in which case the constraints were obtained by optimizing unsupervised critirion function $\mathcal{U}_\mathcal{I}$, supervised criterion function $\mathcal{S}_\mathcal{I}$, and combined criterion function $\mathcal{M}_\mathcal{I}$, respectively. Observed from previous study [7], we know that the parameter α used for combining supervised and unsupervised term in $\mathcal{M}_\mathcal{I}$ can be set with

Table 2. FScores of the clustering solutions obtained by the various clustering methods

Datasets	Agglomerative	Constrained Agglomerative		
		$\mathcal{U}_\mathcal{I}$	$\mathcal{S}_\mathcal{I}$	$\mathcal{M}_\mathcal{I}$
trec6	0.775	0.775	0.738	**0.851**
trec7	0.709	0.708	0.704	**0.790**
trec8	0.768	0.812	0.748	**0.828**

an value in [0.1..0.4] to obtain better partitional clusters. In our comparison, we set $\alpha = 0.2$. The topics used in this set of experiments are long topics.

The results in FScore are shown in Table 2. The entries that are bold-faced correspond to the methods that perform the best for a particular dataset.

From Table 2, we can see that the topic-constrained agglomerative algorithm ($i.e$, optimizing $\mathcal{M}_\mathcal{I}$ to obtain constraints) outperformed the traditional agglomerative algorithm by 7.8% to 11.4%. The constrained agglomerative algorithm with $\mathcal{U}_\mathcal{I}$-constraints performed similarly as the traditional agglomerative algorithm for trec6 and trec7, and achieved better results for trec8. Enforcing constraints obtaining from $\mathcal{S}_\mathcal{I}$ alone did not yeild good clustering results. However, when $\mathcal{S}_\mathcal{I}$ was combined with $\mathcal{U}_\mathcal{I}$ to producing constraints, it outperformed both the constrained algorithm with $\mathcal{U}_\mathcal{I}$-constraints and with $\mathcal{S}_\mathcal{I}$-constraints, and achieved the best performance.

As mentioned above, the parameter α term in $\mathcal{M}_\mathcal{I}$ can be set with an value in [0.1..0.4] to achieve better performance. We did a parameter study on α to see whether the conclution also holds for producting better hierarchical trees. In particularly, we tested the $\mathcal{M}_\mathcal{I}$-constraints agglomerative algorithm with $\alpha = 0.1$ to 0.9 with an increment of 0.1 on the three datasets. The combined scheme becomes the supervised scheme and the unsupervised scheme with $\alpha = 0$ and 1, respectively.

The results for the three datasets are similar and we only show the results for trec6. In Figure 1, we plot the FScore values obtained by the $\mathcal{M}_\mathcal{I}$-constraints scheme against the α values for trec6. We can see that at $\alpha = 0.1..0.4$, the FScore values of the $\mathcal{M}_\mathcal{I}$-constraints scheme are the best (or close to the best), which

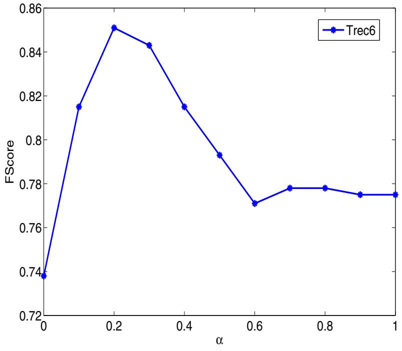


Fig. 1. Relative performance of the selected α for $\mathcal{M}_\mathcal{I}$

suggests that the $\mathcal{M}_\mathcal{I}$-constraints scheme can achieve relative good performance with a broader choice of α values.

5.3 Topic Prototypes of Different Levels of Specificity

The second set of experiments was focused on how the various topic-constrained schemes perform with the topic prototypes of different levels of specificity. For each TREC dataset, we performed the same set of experiments as in Section 5.2 with long, medium, and short topics. The results are similar for all three datasets, and we only show the results of trec6 in in Table 3, in which all the entries are the FScore values of the clustering solutions obtained by the various schemes. Again, the FScore results for the $\mathcal{M}_\mathcal{I}$-constraints scheme were obtained with a fixed α value of 0.2. The entries that are bold-faced correspond to the methods that perform the best for a particular dataset.

A number of observations can be made by analyzing the results in Table 3. First, the $\mathcal{S}_\mathcal{I}$-constraints scheme performs better as the topics become more specific. Second, for all the cases the topic-constrained scheme perform the best. Finally, despite the fact that the short and medium topics alone (used in the $\mathcal{S}_\mathcal{I}$-constraints scheme) perform much worse than the long topics, the proposed topic-constrained scheme is effective with the topic prototypes of different levels of specificity.

Table 3. FScores of the clustering solutions obtained by the various clustering methods with long, medium, and short topics for trec6

Topic Type	Agglomerative	Constrained Agglomerative		
		$\mathcal{U}_\mathcal{I}$	$\mathcal{S}_\mathcal{I}$	$\mathcal{M}_\mathcal{I}$
long	0.775	0.775	0.738	**0.851**
medium	0.775	0.775	0.723	**0.821**
short	0.775	0.775	0.704	**0.799**

5.4 Quality of Constrained and Unconstrained Neighborhoods

This set of experiments were designed to compare the quality of constrained and unconstrained neighborhoods for each document as in [15]. The motivation is that the quality of the most similar documents directly affects the overall performance of agglomerative methods. Unconstrained agglomerative methods consider the most similar documents form the entire collection. On the other hand, constrained agglomerative methods build hierarchical trees within each constraint cluster, thus only the most similar documents of each document from the same constraint cluster are considered. Hence, the relative quality of the neighborhood of each document may indicate how enforcing constains can improve the quality of the resulting hierarchical trees.

Specifically, for each document d we analyzed the class distribution of the t most similar documents, when these documents were identified from the entire document collection and when they were restricted to be part of the same constraint cluster as d. We refer to the first set of documents as the *unconstrained t nearest neighbors* (unconstrained t-nn), denoted as $A_t(d_i)$ for document d_i, and

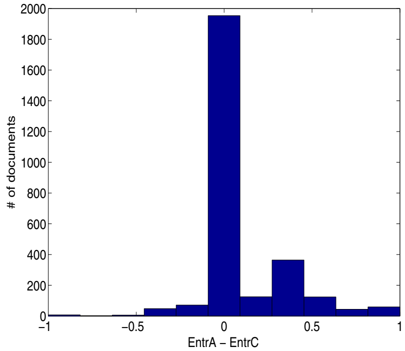


Fig. 2. The histogram of the 5-nn entropy differences of each document without any constraint (EntrA) and with unsupervised cluster constraints obtained by $\mathcal{U}_\mathcal{I}$ (EntrC) for Trec8, with an average entropy difference of 0.0904.

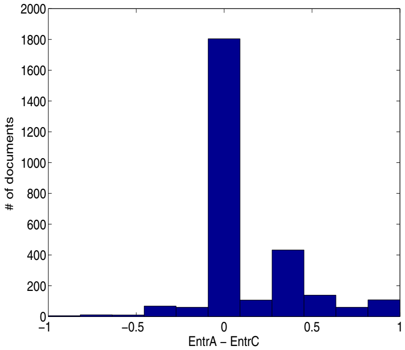


Fig. 3. The histogram of the 5-nn entropy differences of each document without any constraint (EntrA) and with topic-constraints obtained by $\mathcal{M}_\mathcal{I}$ (EntrC) for Trec8, with an average entropy difference of 0.1156.

the second as the *constrained t nearest neighbors* (constrained t-nn), denoted as $C_t(d_i)$.

We evaluated the quality of the nearest neighbors using the entropy measure. Given a particular cluster S_r of size n_r, the entropy of this cluster is defined to be

$$E(S_r) = -\frac{1}{\log q}\sum_{i=1}^{q}\frac{n_r^i}{n_r}\log\frac{n_r^i}{n_r},$$

where q is the number of classes in the dataset and n_r^i is the number of documents of the ith class that were assigned to the rth cluster.

Now, for each document d we computed the entropy value of the unconstrained t-nn (EntrA) and constrained t-nn (EntrC) as follows,

$$\text{EntrA} = E(A_t(d)) \quad \text{EntrC} = E(C_t(d)).$$

Given the definition above, we compare the quality of the constrained and unconstrained 5-nn of each document for trec8. For each document d we computed the difference between the entropy of unconstrained 5-nn and constrained 5-nn (EntrA - EntrC), and we did this for both the $\mathcal{U}_{\mathcal{I}}$-constraints scheme and the $\mathcal{M}_{\mathcal{I}}$-constraints scheme. Figure 2 and Figure 3 show how these differences (EntrA - EntrC) are distributed for the $\mathcal{U}_{\mathcal{I}}$-constraints, and the $\mathcal{M}_{\mathcal{I}}$-constraints scheme, respectively. The X-axis in Figure 2 and Figure 3 represents the differences of the 5-nn entropy values (EntrA - EntrC), whereas the Y-axis represents the number of the documents. Note that since lower entropy values are better, differences that are positive correspond to the instances in which the constrained 5-nn neighborhoods are purer.

From these charts we can see that most of the bars are at or on the right of the origin, which means that the constraints indeed improve the quality of each document's neighborhood. The average entropy difference of the $\mathcal{U}_{\mathcal{I}}$-constraints and $\mathcal{M}_{\mathcal{I}}$-constraints scheme is 0.0904 and 0.1156, respectively, which is consistant with their performance. That is, the $\mathcal{M}_{\mathcal{I}}$-constraints scheme improved the neighborhoods better than the $\mathcal{U}_{\mathcal{I}}$-constraints scheme. Accordingly, the $\mathcal{M}_{\mathcal{I}}$-constraints scheme improved the quality of the resulting hierarchical trees better than the $\mathcal{U}_{\mathcal{I}}$-constraints scheme as well.

6 Conclusion

In this paper, we proposed the *topic-constrained hierarchical clustering*, which organizes document datasets to hierarchical trees consistant a given set of topics, such that the documents from a single cluster in the tree are similar to each other and are about the same topic. The proposed algorithm is based on a constrained agglomerative clustering framework and a semi-supervised criterion function. The experimental evaluation show that our algorithm improved the traditional agglomerative algorithm by 7.8% to 11.4%, and outperformed other constrained agglomerative algorithms with constraints produced by supervised and unsupervised criterion functions. We also showed that the topic-constrained hierarchical clustering performs well with topic prototypes of different levels of specificity, and indeed improves the quality of document neighborhoods.

Acknowledgment

This work was funded by the National Science Foundation China (Project No. 60703058).

References

1. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: 18th International Conference on Machine Learning (ICML 2001), pp. 577–584 (2001)

2. Basu, S., Bilenko, M., Monney, R.: A probabilistic framework for semi-supervised clustering. In: 10th International Conference on Knowledge Discovery and Data Mining (2004)
3. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning with application to clustering with side-information. In: Advances in Neural Information Processing Systerms, vol. 15, pp. 505–512 (2003)
4. Davidson, I., Ravi, S.S.: Agglomerative Hierarchical Clustering with Constraints: Theoretical and Empirical Results. In: 9th European Conference on Principles and Practice of Knowledge Discovery in Databeses, pp. 59–70 (2005)
5. Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary clustering. In: 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 554–560 (2006)
6. Bade, K., Nurnberger, A.: Creating a cluster hierarchy under constraints of a partially known hierarchy. In: 2008 SIAM International Conference on Data Mining (SDM 2008) (2008)
7. Zhao, Y., Karypis, G.: Topic-driven Clustering for Document Datasets. In: 2005 SIAM International Conference on Data Mining (SDM 2005), pp. 358–369 (2005)
8. Bilenko, M., Basu, S., Monney, R.: Integrating constraints and metric learning in semi-supervised clustering. In: 21th International Conference on Machine Learning, ICML 2004 (2004)
9. Salton, G.: Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, Reading (1989)
10. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
11. Cutting, D.R., Pedersen, J.O., Karger, D.R., Tukey, J.W.: Scatter/gather: A cluster-based approach to browsing large document collections. In: 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 318–329 (1992)
12. Larsen, B., Aone, C.: Fast and effective text mining using linear-time document clustering. In: 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 16–22 (1999)
13. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Machine Learning 42(1/2), 143–175 (2001)
14. Zhao, Y., Karypis, G.: Empirical and theoretical comparisons of selected criterion functions for document clustering. Machine Learning 55(3), 311–331 (2004)
15. Zhao, Y., Karypis, G.: Hierarchical clustering algorithms for document datasets. Data Mining and Knowledge Discovery 10(2), 144–168 (2005)
16. TREC. Text REtrieval conference, `http://trec.nist.gov`
17. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)

Discretization of Time Series Dataset Using Relative Frequency and K-Nearest Neighbor Approach

Azuraliza Abu Bakar, Almahdi Mohammed Ahmed,
and Abdul Razak Hamdan

Center for Artificial Intelligence Technology,
Faculty of Information Science and Technology,
University Kebangsaan Malaysia 43600 Bangi,
Selangor Darul Ehsan, Malaysia
{aab,sherif,arh}@ftsm.ukm.my

Abstract. In this work, we propose an improved approach of time series data discretization using the Relative Frequency and K- nearest Neighbor functions called the RFknn method. The main idea of the method is to improve the process of determining the sufficient number of intervals for discretization of time series data. The proposed approach improved the time series data representation by integrating it with the Piecewise Aggregate Approximation (PAA) and the Symbolic Aggregate Approximation (SAX) representation. The intervals are represented as a symbol and can ensure efficient mining process where better knowledge model can be obtained without major loss of knowledge. The basic idea is not to minimize or maximize the number of intervals of the temporal patterns over their class labels. The performance of RFknn is evaluated using 22 temporal datasets and compared to the original time series discretization SAX method with similar representation. We show that RFknn can improve representation preciseness without losing symbolic nature of the original SAX representation. The experimental results showed that RFknn gives better term of representation with lower and comparable error rates.

Keywords: Data mining, discretization, reduction, pre-processing and time series representation, dynamic intervals.

1 Introduction

At the present, the time series data are being generated with extraordinary speed from almost every application domain. Among these data are related to the daily fluctuations of the stock market data, traces of dynamic processes and scientific experiments, medical field and daily rainfall occurrence. As a consequence, in the last decade there has been an increasing number of interest in querying and mining such data which, in turn had resulted active research to introduce the new methodology for indexing, classification, clustering and approximation of time series data [1], [2], [3], and [4]. However, these researches are oriented to the data compression, rather than to the information maximization. Specifically, these representations transform times

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 193–201, 2010.

series data of length N, into a set of n coefficients, where n < N. These data compression processes are only intended to reduce dimensionality on data [5]. However, they do not consider whether the new representation preserves the relevant information in order for the observations to continue to belong to the associated class labels.

A discretized interval should not hide the patterns. It must be chosen carefully or this may lead to the loss of important information. For instance, if the length of the intervals is very large, we may lose some of the details that described the data and it will not be meaningful to discover patterns from the database. However, if the length of each interval is too small then it will not have enough data to produce patterns. Several discretization techniques for time series are discussed in [6]. In most time series data discretization algorithms require the user to specify a set of parameters to perform the transformation. For example, the number of segments (word size) dividing the times series length, and the number of intervals (alphabet size) required to compress the time series value. Without a previous analysis of the temporal data, it is very difficult to know the proper parameter values to obtain a good discrete representation of data. However, in practice it is assumed that the parameters are known [2] [4].

In this work, we propose a new method to improve SAX, which is based on the relative frequency and K-Nearest Neighbor (*RFknn*) method. The main process of the method is to determine the sufficient number of intervals that alphabets size (a) can ensure the efficient mining process and a good knowledge model is obtained without major loss of knowledge. This paper is to five sections. In section 2 the related work is discussed. Section 3 describes the proposed method and main concepts involved in its definition the method RFknn is explained in detail. Section 4 describes the experiments carried out and the data used to evaluate the performance of the method and shows the results obtained over the experiments. Section 5 presents a conclusions and future work are presented.

2 Background of the Work

The majority of the methods in data mining in time series assume that the time series are discrete. Nevertheless, most applications generate and use floating-point data type. Therefore there are numerous approaches to the discretization of time series with floating-point values. They propose different approaches and different measures of function, this is the case proposed in [7]: the purpose of the method is to detect the persistent states of the time series, the method does not require either parameter specification. The applicability of this method is restricted to the existence of persistent states in the time series that is not common in most of the world applications. The method processes a single time series at a time, so that the discretization criterion is not generalized to the complete dataset. Brian, et. al. (2002) represents time series values as a multi connected graph [8]. Under this representation, similar time series are grouped into a graphical model. However, its limitation is that it only works with one time series at a time. Eamonn, et.al. (2003) proposed a new symbolic representation of time series called SAX [9]. Their representation is unique in that it allows dimensionality reduction, and it also allows distance measures to be defined on the symbolic approach where the lower bound corresponding distance measures is defined on the original series. This method is based on piecewise Aggregate Approximation representation PAA [10]. Many work have been proposed to improve SAX, those studies showed that SAX method still an open research area to bring and develop new ideas to improve such representation [11], [12], and 13].

Daniel and Lopez [13] proposed a new algorithm for discretization time series using an approach that applied the Genetic Algorithm called GENEBLA (Genetic Entropy Based Linear Approximation). Genetic algorithm has proven to be an efficient searching algorithm in state space or approximate solutions optimization. It also performs efficiently when the user does not have precise domain expertise, because genetic search possesses the ability to explore and learn from their domain. Several others GA based algorithm used for discretization are discussed [1],[11]. EBLA2 algorithm is proposed for automatic generation of alphabet size and word size to maximize accuracy in classification. Similar to greedy search, heuristic is used to lead to a specific solution obtained from the best result of the iteration. This approach requires no parameters to the search, because the outcome is deterministic. An enhanced version of the algorithm EBLA2 is proposed in [12] called EBLA3. It performs a broader search than EBLA2 using the simulated annealing approach as the discretization schemes of temporal datasets. This allows the result is not entirely deterministic, possible to find better discretization schemes. However, the results obtained by this approach could be improved by enabling populations to generate discretization schemes, which could quickly reach a good solution.

3 Proposed Method

Many approaches and techniques that concentrate on the time series data representation problems have been proposed in the past decade. Most commonly used Piecewise Aggregate Approximation (PAA) [14],[15]. Recently, one promising representation method was proposed, called Symbolic Aggregate Approximation (SAX) [9]. The basic idea of our method proposed in the paper, is based on the last two approaches among these representation techniques. These two methods are the PAA and the SAX representations, which are briefly described below in this section. In this work, we employ the Piecewise Aggregate Approximation (PAA) and Symbolic Aggregate Approximation (SAX) as data representations.

3.1 Piecewise Aggregate Approximation

The basic idea of Piecewise Aggregate Approximation (PAA) is that it represents the time series as a sequence of rectangular basis functions and converts the time series into discrete symbolic sequences. It is a dimensionality-reduction representation method as shown in Eq. (1) where n is the length of sequence; N (Ws) is the number of PAA segments, w is the segment size, j is index value of segment size and Pi is the average value of the segment.

$$\bar{P}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} P_j \ . \qquad (1)$$

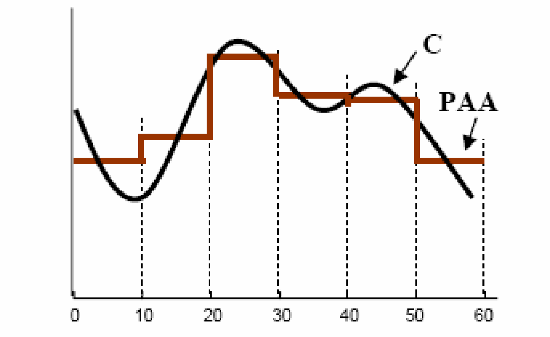


Fig .1. The time series *C* is represented by PAA. In the example above the dimensionality is reduced from *N*=60 to *Ws*= 6.

The advantages of this transformation (PAA) are that 1) it is very fast and easy to implement, and 2) the index can be built in linear time. As shown in Fig.1, in order to reduce the time series from *n* dimensions to *k* dimensions, the data is divided into *k* equal sized segments. The mean value of the data falling within a segment is calculated and a vector of these values becomes the data-reduced representation.

3.2 Symbolic Aggregate Approximation (SAX)

The transformed PAA time series data are then referred to the SAX algorithm to obtain a discrete symbolic representation. Since normalized time series have a Gaussian distribution, we can determine the "breakpoints" that will produce *c* equal-sized areas under the Gaussian distribution curve. The time series data are then dispersed into the discrete symbolic series based on these breakpoints [1]. All PAA coefficients that are below the smallest breakpoint are transformed to the symbol "*a*", and all coefficients greater than or equal to the smallest breakpoint but less than the second smallest breakpoint are transformed to the symbol "*b*".

While there are many different symbolic approximations of time series data in the study, SAX is the first symbolic representation that allows both dimensionality reduction and lower bounding the Euclidean distance. But, as noted before, it has also some disadvantages such as the dimensionality reduction nature that has the possibility of missing important patterns in some datasets. Therefore, we propose a method *RFknn* to automatically generate alphabets size (intervals) for each word size based on the dataset's information itself. This step comes after implementation of PAA, after which SAX representation is applied. One advantage of RFknn is that it generates (intervals) alphabets size based on the data information which would insure that each sequence (word size) will get just the right represented number of intervals because of using the relative frequency function. More details of the proposed method are discussed in the next section.

3.3 The RFknn Method

We proposed the principle of relative frequency and K-Nearest Neighbor (RFknn) Algorithm [16] which is to determine the sufficient number of intervals that can ensure a good knowledge model which will be obtained without major loss of knowledge. The proposed algorithm contains two main functions that are the computation of the relative frequency function and the interval threshold values and the merging of confidence interval values and less confidence interval values function where the less confidence interval values are assigned to the nearest confidence interval values. There are six important steps involved in both phases. The relative frequency *s* is the summation of the absolute frequency $a{=}_{a1},\ldots,a_n$ as shown where *n* is the number of observation, *i* is the time of the intervals occurrence . The relative frequency *rf* and the means of relative frequency P_{rf} are computed as in Eq.(2) and Eq. (3). The observation is set to the confidence interval value (*c*) if *rf* is larger or equal to P_{rf} and less confidence interval value ($\overleftarrow{c}$) if *rf* is less than P_{rf} as in Eq. 4. Then K-NN is used to calculate the distance between the observations contain in confidence interval value (c) and less confidence interval value ($\overleftarrow{c}$) to determine the appropriate interval for the observations values.

The sample *relative frequency* (*rf*) is the most commonly used and is represented by the Eq.(2):

$$rf_i = \sum_{j=1}^{n} \frac{a_j}{S} \;, \tag{2}$$

Where *rf* is the sample relative frequency, Eq. (2) is the sum of all frequency divided by total number of items occurrence *i* denotes number of intervals represented in a series), and *n* is the number of intervals. The percentage of *rf* ($\mathrm{P_{rf}}$) is the most commonly used and is represented by the Eq.(3).

$$P_{rf} = \frac{rf_i}{\sum_{i=1}^{n} rf_i} \;, \tag{3}$$

Where (P_{rf}) is the sample mean, Σrf_i is the sum of all relative frequency ten divided by *n*, where *i=1... n*.

The c and $\overleftarrow{c}$ are the hypothesis to be tested, and c is generally a statement that a population parameter has a confidence value or parameters *rf* over P_{rf}, $\overleftarrow{c}$ is $rf <\; P_{rf}$ more populations are similar as in Eq.(4).

$$x(class) = \begin{cases} c, & rf_i \geq Prf \\ \overleftarrow{c}, & rf_i < Prf \end{cases} . \tag{4}$$

Standard distance error is the estimate of the K-Nearest Neighbor. The estimate of the standard error of the destination values $x_i(c), x_j(\overleftarrow{c})$ is described by the following Eq.(5).

$$d(x_i(c), x_j(\overleftarrow{c})) = \sqrt{\sum_{i=1}^{n} [(x_i(c) - x_j(\overleftarrow{c})]^2} \;, \tag{5}$$

Where d is the sample standard distance and n is the number of samples. x is the represented intervals value, and $\{c,\overleftarrow{c}\}$ are classes which denotes confidence intervals and less confidence intervals. A way to merge less confidence intervals value $x_j(\overleftarrow{c})$ with the confidence intervals value $x_i(c)$ based on the minimum distance between both $x_j(\overleftarrow{c})$ and $x_i(c)$. Therefore the less confidence interval value is merged to nearest confidence interval value, the new value $\hat{x}(\overleftarrow{c})$ of the sample is represented by the Eq. (6).

$$\hat{x}(\overleftarrow{c}) = \begin{cases} x_i(c) & d(x_i(c),x_j(\overleftarrow{c}))=0 \\ x_j(\overleftarrow{c}) & d(x_i(c),x_j(\overleftarrow{c}))>0 \end{cases} \;, \tag{6}$$

Where $\hat{x}$ is the output which computes only the x values that labeled with confidence intervals as class label then assigned less confidence intervals values to x value based on the distance and nearest neighbor. The output of the algorithm is described in this Eq (7).

$$\hat{x}_{new} = \{x_1, x_2, \ldots, x_m\} \,. \tag{7}$$

4 Experiments and Results

The performance of the method was tested using 22 datasets available on the time series classification/clustering web page [17]. The Symbolic Aggregate Approximation (SAX) and PAA [11] are used in the experiment. The *RFknn* is applied to determine the efficient number of intervals of the alphabet size (a). *RFknn* is run after PAA to generate intervals of alphabet size (a) on each sequence. Then, SAX is run with the same generated intervals by *RFknn* of the alphabet size (a) which is generated by *RFknn* and we call this *RFknn*-SAX. The *RFknn* and *RFknn*-SAX are compared with the original Eumonn's 1-NN and 1-NN SAX [9]. The alphabet size produced by *RFknn* denoted as *a-R*, is compared with the fixed alphebte size *a* given by the original SAX a-SAX [9]. The experimental results shows that the proposed *RFknn* performs better in terms of error rates in several dataset compared to 1-NN Eumonn's techniques. The *RFknn-SAX* gives outstanding lower error rates when compared to the original SAX in most datasets with larger number of intervals.

We use 1-NN, to evaluate the *RFknn* method and as well use the Euclidean distance. Classification performance was also evaluated using the raw data (continuous time series). It is important to remark that SAX uses its own similarity measure [9]. Using algorithms and parameters for the classification as described above, a lower error rate indicates better performance in classification. Table1 shows that generally *RFknn* and *RFknn*-SAX reaches similar error rates using the same parameters (alphabet size and word size), with the advantage that *RFknn* does not require these parameters a priori because they are automatically calculated compared to SAX, whereas as SAX reached a little bit high error rates with fixed alphabet size and word size. In some datasets *RFknn*-SAX performance improvement was obtained with *RFknn* generally providing better performance as compared to SAX.

The advantage of the proposed method is that *RFknn* does not require a priori parameters because they are automatically calculated; moreover *RFknn* works very well whenever the time series data length is larger. *RFknn* improves the performance of SAX, especially in datasets where SAX improves the performance of classifications using larger alphabets size. Hence most of the dataset improves the performance of SAX when the word uses the parameters found by *RFknn*. We decided to compare *RFknn* with SAX because SAX is one of the most efficient methods proposed so far. *RFknn* allows greater control over the selection of the chosen number of intervals because it uses relative frequency to compute interval size on the data. In addition, it uses the fundamental of k-nearest neighbor to merge confidence intervals value with less confidence intervals value. Based on those two

Table 1. Error rate obtained for RFknn, RFknn-SAX and SAX using the parameters (alphabets size) suggested by RFknn.

No	Name	1-NN EU Error	1-NN SAX Error	1-NN *RFknn* Error	1-NN *RFknn–* SAX Error	*a-SAX*	*a-R*	Ws
1	Beef	0.467	0.567	0.500	0.400	10	9	128
2	Coffee	0.250	0.464	0.357	0.398	10	5	128
3	OliveOil	0.133	0.833	0.667	0.533	10	27	256
4	Lighting2	0.246	0.213	0.426	0.203	10	40	256
5	Lighting7	0.425	0.397	0.495	0.397	10	10	128
6	ECG200	0.120	0.120	0.110	0.120	10	12	32
7	Facefour	0.216	0.170	0.179	0.085	10	5	128
8	Adiac	0.386	0.890	0.831	0.802	10	15	64
9	Fish	0.217	0.474	0.560	0.598	10	23	128
10	Gun_Point	0.087	0.180	0.093	0.090	10	13	64
11	Trace	0.240	0.460	0.380	0.300	10	13	128
12	OUSleaf	0.483	0.467	0.434	0.399	10	11	128
13	Syntic Control	0.120	0.020	0.011	0.010	10	5	16
14	50Words	0.369	0.341	0.290	0.300	10	9	128
15	CBF	0.148	0.104	0.130	0.102	10	12	32
16	yoga	0.170	0.195	0.177	0.195	10	10	128
17	Face(all)	0.286	0.330	0.305	0.299	10	13	64
18	car	0.267	0.333	0.257	0.394	10	25	256
19	Plane	0.038	0.038	0.043	0.038	10	10	64
20	Two pattern	0.093	0.081	0.061	0.063	10	8	32
21	Swedish leaf	0.211	0.483	0.321	0.291	10	7	32
22	OSU	0.483	0.467	0.389	0.481	10	21	128

functions used by *RFknn* the measure of distance may also contribute to achieve better performance in classification, as SAX uses its own measure of distance to reconstruct part of the original data, while maintaining a relationship between discrete representation and representation of the continuous time series data.

5 Conclusion

In the present work a new improved method for discretization on time series data called RFknn has been proposed. Given a labeled dataset containing temporal data, the RFknn method automatically computes only the alphabet size (a) intervals of each word size. The efficiency of the method was evaluated using twenty two datasets and compared to one of the most efficient representations of data of time series called SAX, *RFknn*–SAX and raw data. Generally, error rates reached using *RFknn* representation were similar to those obtained by *RFknn*–SAX tests using the same parameter (*alphabet size, word size*).

References

1. Waldron, M., Manuel, P.: Genetic Algorithms as a Data Discretization Method. In: Proceeding of Midwest Instruction and Computing Symposium (2005)
2. Jiawei, H., Micheline, K.: Data Mining: Concepts and Techniques. Morgan Kaufmann, CA (2005)
3. Keogh, E.: A decade of progress in indexing and mining large time series databases. In: Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, pp. 1268–1268 (2006)
4. Keogh, E., Kasetty, S.: On the need for time series data mining benchmarks: a survey and empirical demonstration. In: Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, pp. 349–371 (2002)
5. Han, J.: Data mining techniques. In: Proceedings of the 1996 ACM SIGMOD International Conference on Management of data, Montreal, Quebec, Canada, p. 545 (1996)
6. Rakesh, A., Faloutsos, C., Swami, A.: Efficient similarity search in sequence databases. In: Lomet, D.B. (ed.) FODO 1993. LNCS, vol. 730, pp. 69–84. Springer, Heidelberg (1993)
7. Mörchen, F.: Optimizing time series discretization for knowledge discovery. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, Chicago, Illinois, USA, pp. 660–665 (2005)
8. Brian, B., Shivnath, B.: Models and issues in data stream systems. In: Proceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Madison, Wisconsin, pp. 1–16 (2002)
9. Jessica, L., Eamonn, K.: A symbolic representation of time series, with implications for streaming algorithms. In: Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, San Diego, California, pp. 2–11 (2003)
10. John, F.R., Kathleen, H., Myra, S.: An Updated Bibliography of Temporal, Spatial, and Spatio-temporal Data Mining Research. In: Roddick, J., Hornsby, K.S. (eds.) TSDM 2000. LNCS (LNAI), vol. 2007, pp. 147–163. Springer, Heidelberg (2001)

11. Acosta, M., Nicandro, H.G., Daniel-Alejandro, C.R.: Entropy Based Linear Approximation Algorithm for Time Series Discretization. Advances in Artificial Intelligence and Applications 32, 214–224 (2007)
12. Alejandro, G.L.D.: Algoritmo de Discretización de Series de Tiempo Basado en Entropía y su Aplicación en Datos Colposcopicos. Universidad Veracruzana (2007)
13. Acosta-Mesa, D.A.: Discretization of Time Series Dataset with a Genetic Search. In: Aguirre, H., et al. (eds.) MICAI 2009. LNCS, vol. 5845, pp. 201–212. Springer, Heidelberg (2009)
14. Geurts, P.: Pattern extraction for time series classification. In: Siebes, A., De Raedt, L. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 115–127. Springer, Heidelberg (2001)
15. Byoung-Kee, Y., Christos, F.: Fast Time Sequence Indexing for Arbitrary Lp Norms. Book Fast Time Sequence Indexing for Arbitrary Lp Norms, Series Fast Time Sequence Indexing for Arbitrary Lp Norms,Kaufmann. Morgan Kaufmann, San Francisco (2000)
16. Almahdi, M.A., Azuraliza, A.B., Abdul, R.H.: Dynamic data discretization technique based on frequency and K-Nearest Neighbour algorithm. In: Proceedings of the 2nd Conference on Data Mining and Optimization, Malaysia, pp. 27–28 (2009)
17. Keogh, E., Xi, X., Wei, L.: The UCR Time Series Classification Clustering Homepage (2006), http://www.cs.ucr.edu/~eamonn/time_series_data

MSDBSCAN: Multi-density Scale-Independent Clustering Algorithm Based on DBSCAN

Gholamreza Esfandani and Hassan Abolhassani

Computer Engineering Department, Sharif University of Technology, Tehran, Iran
esfandani@ce.sharif.edu, abolhassani@sharif.edu

Abstract. A good approach in data mining is density based clustering in which the clusters are constructed based on the density of shape regions. The prominent algorithm proposed in density based clustering family is DBSCAN [1] that uses two global density parameters, namely minimum number of points for a dense region and epsilon indicating the neighborhood distance. Among others, one of the weaknesses of this algorithm is its un-suitability for multi-density data sets where different regions have various densities so the same epsilon does not work. In this paper, a new density based clustering algorithm, MSDBSCAN, is proposed. MSDBSCAN uses a new definition for core point and dense region. The MSDBSCAN can find clusters in multi-variant density data sets. Also this algorithm benefits scale independency. The results obtained on data sets show that the MSDBSCAN is very effective in multi-variant environment.

Keywords: local core distance, scale independency, multi-density scale-independent clustering, MSDBSCAN.

1 Introduction

Clustering is an unsupervised process to find relations between points of a data set. In clustering we want to partition points into some groups such that the points in each group have high intra-class similarity and low inter-class similarity. Currently, different methods for clustering are proposed. Each of them has its own benefits and disadvantages.

Almost all the algorithms for clustering take a vector of input parameters. Determination of appropriate values for these parameters is challenging. However, quality of clustering has a direct relationship with these parameters' values. Some of the algorithms lack the ability to handle arbitrary shapes and are limited to specific shapes. Some of the algorithms are unable to find correct clusters in multi-variant density environment. In most algorithms, if the data set of algorithm is multiplied by a constant value, the input variables should be set again. Although many algorithms are proposed for solving some of the mentioned problems but some of these problems still remain challenging.

In this paper, we introduce MSDBSCAN in which concepts of *local core distance* (lcd) for each point in a given data set is employed. The lcd represents the distance in

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 202–213, 2010.

which there is at least MinPts objects. To be a core point, the neighbors' lcd of the point should be similar to each other. Also in this paper, we will refine basic definitions of DBSCAN algorithm and adapt them for MSDBSCAN. MSDBSCAN tries to solve the problem of DBSCAN algorithm for handling variant density shapes and finds clusters in multi-variant density environment correctly. Also this algorithm can find and separate overlapping clusters properly.

Another characteristic of MSDBSCAN is its scale independency which means if data set of algorithm is multiplied by a constant value, there is no need to change the input parameters of the algorithm (i.e. it is resolution independent). Current clustering algorithms have not this feature. By scaling the data set, the input parameter should be adapted.

The rest of the paper is organized as follows: Related works on clustering are briefly described in section 2. In section we describe in detail the problems that motivated us to do this work. Also we express some notations and definitions required for the algorithm. The details of algorithm and proof of its scale independency is given in section 4. The discussions on the appropriate tuning of input parameters are the other issue of this section. In section 5 we show the results of experiments that we have done. In addition the time complexity of algorithm is described in this section.

2 Related Work

Clustering algorithms is a complicated task in Data Mining and Knowledge Discovery. Therefore many algorithms are proposed in this field. Existing clustering algorithms can be generally classified into the three following main categories [3]

1-partitioning algorithms: This family of methods tries to find the best partitioning for data points in which intra class similarity is maximum and inter class similarity is minimum. The simplest algorithm in this family is k-means [4] and based on the k-means many other methods are proposed. PAM [5] and k-modes [7] are two of them.

2-Hierarchical clustering algorithms: A hierarchical clustering algorithm works by decomposing data objects into a tree of clusters. This kinds of algorithms can be classified as either agglomerative (top-down) or divisive (bottom-up). BIRCH [6] and CHAMELEON [7] are examples of hierarchical clustering family.

3-Density-based clustering algorithms: This family of algorithms has been developed to discover clusters with arbitrary shapes. This family tries to find clusters based on the dense regions in a shape. If two points are close enough and the region around them is dense, then these two data points join together and contribute in constructing a cluster. DBSCAN [1], OPTICS [2] and LDBSCAN [8] are examples of this approach.

Our algorithm can be regarded as a density based algorithm. Hence in the rest of this section we discuss more on prominent density based algorithms. The most renowned algorithm in this family is DBSCAN which has two input parameters, epsilon and MinPts. According to these parameters the dense regions in data set are defined. A region is dense around a specific point if in its epsilon neighborhood radius there are at least MinPts points. The algorithm calls these as core points. DBSCAN merges two core points if they are enough close to each other. By applying this approach, algorithm is capable of finding arbitrary shapes of clusters.

OPTICS algorithm tries to generalize the DBSCAN. Similar to DBSCAN this algorithm takes two input parameters epsilon and MinPts. Based on these parameters, OPTICS generates an augmented ordering of the data set and makes a visual representation of it. By this representation user can choose a value for epsilon' variable which is less than epsilon to gain the desired results.

LDBSCAN tries to use the concept of *Local Outlier Factor* (LOF) and *Local Reachability Factor* (LRD) [9] in order to detect noise points. It takes four input parameters, namely $MinPts_{LOF}, MinPts_{LDBSCAN}, pct$ and *LOFUB.* The algorithm takes two different MinPts values, one of them for calculating the LOF of each point and the other is used in algorithm as in DBSCAN. In this algorithm a point is core if its LOF is less than input parameter LOFUB. The parameter pct is used to control the fluctuation of local density.

3 Basic Definitions of MSDBSCAN

We define some notations and concepts of the algorithm in this section. Some of these definitions are refined concepts of DBSCAN algorithm while some of them are new.

Definition 1: local core distance of object *o* (lcd(o)): The local core distance of *o* in data set D with respect to MinPts, is defined as a minimum distance which satisfies the following two conditions:

1. For at least MinPts object *o'* in D-{*o*} it holds that $d(o,o^{'}) \le lcd(o)$
2. For at most MinPts-1 object *o'* in D-{*o*} it holds that $d(o,o^{'}) < lcd(o)$

In the above definition *d(o,o')* is the distance between objects *o* and *o'*. Informally, local core distance of *o* is the minimum distance which at least MinPts neighbors exist in it. Let $\varepsilon-$distance-neighborhood of point *o* be denoted by N_{ε}(o) then the $Card(N_{\text{lcd(o)}}(\text{o})) \ge \text{MinPts}$ will be satisfied. For most of points N_{ε}(o) equals to MinPts.

Definition 2: mean of local core distance of object o ($\overline{lcd}(o)$): The mean of local core distance of object *o* in data set D with respect to MinPts equals to the mean of lcd of objects which are in the set $\{o, N_{\text{lcd}}(\text{o})\}$. In other words $\overline{lcd}(o)$ can be obtained from the following formula:

$$\overline{lcd}(o) = \frac{lcd(o) + \sum_{i \in N_{\text{lcd}}(\text{o})} lcd(i)}{Card(N_{\text{lcd}}(\text{o})) + 1} \quad (1)$$

Also the average mean of local core distance of object *o* ($\overline{\overline{lcd}}(o)$) is defined as follow:

$$\overline{\overline{lcd}}(o) = \frac{\overline{lcd}(o) + \sum_{i \in N_{\text{lcd}}(\text{o})} \overline{lcd}(i)}{Card(N_{\text{lcd}}(\text{o})) + 1} \quad (2)$$

Another notation that we use in this paper is variance of the vector $\overline{LCD}$. For each point o in data set, $\overline{LCD}$ is a vector for object o whose members are $\overline{lcd}(p)$ such that $p \in \{o, N_{lcd(o)}(o)\}$. The variance of vector $\overline{LCD}$ with respect to the values of $\overline{lcd}$ situated in $\overline{LCD}$ equals to the following equation:

$$s^2(\overline{LCD}(o)) = \mathrm{var}(\overline{LCD}(o)) = \frac{(\overline{lcd}(o) - \overline{\overline{lcd}}(o))^2 - \sum_{i \in N_{lcd(o)}} (\overline{lcd}(i) - \overline{\overline{lcd}}(o))^2}{Card(N_{lcd(o)})} \qquad (3)$$

In the above equation s is the standard deviation of $\overline{LCD}$.

Definition 3: core point: o will be a core point if the values in its $\overline{LCD}$ vector are almost similar. To formally express this, o will be a core point if the following condition is satisfied:

$$\forall p \in \left\{N_{\mathrm{lcd}}(\mathrm{o})\right\} \rightarrow \overline{lcd}(p) \in \ [\overline{lcd}(o) - \gamma \times s(\overline{LCD}(o)), \overline{lcd}(o) + \gamma \times s(\overline{LCD}(o))] \qquad (4)$$

In the above proposition, s is the standard deviation of $\overline{LCD}$ that was mentioned in Definition 4. The value of γ is the input parameter of the algorithm. Related discussions about γ is given in section 4.1 in details.

This new definition of core point expresses that to be a core point for o depends on the values of its neighbors' $\overline{lcd}$ which should lie in a certain interval. The mentioned interval is determined locally based on the situation of point o. With definition 5, it is possible to have core points with different radiuses in a data set. The reason of choosing $\overline{lcd}$ instead of lcd is to prevent the high impact of outlier points' lcd to the value of lcd.

Definition 4: Scale Independent algorithm: Let D be a data set of points and there is an algorithm A that will be applied to D. With multiplying D into a constant value like ξ, the obtained data set is denoted by ξD. If the result of applying A to D with input parameter vector v, is R1 and the result of applying A to ξD with the same input parameter vector, is R2 then A will be a scale independent algorithm if R1=R2. In other words, the input parameter of algorithm should be independent of the scale of a data set.

4 MSDBSCAN

The general structure of finding clusters in MSDBSCAN is similar to DBSCAN. The algorithm constructs the large clusters by merging the core points. The two core points which are merged should be the neighbors of each others. It is obvious that in merging time the neighbors of these points, are also added to the merged clusters. The core point condition is described in definition 3 of previous section. Below, we present a high-level pseudo code of MSDBSCAN.

```
Begin of MSDBSCAN Algorithm
  // allData is a given DataSet
  //MinPts, γ are the input parameters of the algorithm
  MSDBSCAN(allData, MinPts, γ)

    1.setNeighborsOfEachDataPoint(allData,MinPts);
    2.computeStatisticalItems(allData,gama);
    3.runMSDBSCAN(allData);

End of Algorithm
```

In the above pseudo code, allData is the input dataset and MinPts and γ are the input parameters of the algorithm. *setNeighborsOfEachDataPoint* determines the values of lcd for each points in the data set. Also, this procedure finds the neighbors of each point which their distances are less than or equal to the lcd of that point. To achieve a faster algorithm we materialize the neighbors of each point in this procedure.

In *computeStatisticalItems* procedure the values of $\overline{lcd}, \overline{\overline{lcd}}$ are determined for each point. Additionally this procedure detects the core points of the data set. The main part of the algorithm *is runMSDBSCAN* function. The procedure *runMSDBSCAN* starts from a random point. If the selected point is a core point, the neighbors of this point will join the cluster and this operation will be continued until the cluster reaches to the border points.

The structure of runMSDBSCAN procedure is similar to DBSCAN's structure and the only difference between two algorithms is related to the manner of finding core points. The intuition for this new definition of core point is that if *o* is a core point and is in depth of a cluster, the values of its $\overline{lcd}$ and its neighbors' $\overline{lcd}$ should not vary considerably. Theses variations will be much more, if cluster reaches to its border points. Such definition of core point, releases us from the global parameters. Thus each point in a data set, based on its neighbors' arrangement can be tested for being as a core point. The local view of core point results in the recognition of clusters with variant density. Another advantage of this algorithm is its scale independency property. We prove it below.

Lemma 1: If the data set D is multiplied by a constant factor ξ , the core points of D will remain core points in new data set.

Proof: Suppose x_1, x_2 are two points in n-dimensional space and distance between them denoted by $d(x_1, x_2)$. If each of them is multiplied by a constant value ξ , it is obvious that the distance between them will be $\xi d(x_1, x_2)$. Also it is easy to prove that the values of $lcd, \overline{lcd}, \overline{\overline{lcd}}$ and standard deviation for each point in ξD will be multiplied by ξ .

We use "proof by contradiction" to prove the lemma 1. We assume that the opposite of lemma is correct. With this assumption there is a point *o* which it is core in D and not in ξD . Because *o* is not a core point in ξD , according to definition 3, we have the following proposition for *o*:

$$\exists p \in \{o, N_{\text{lcd}}(\text{o})\} \Rightarrow \ \xi\overline{lcd}(p) \notin [\xi\overline{lcd}(o) - \gamma \times \xi s(\overline{LCD}(o)), \xi\overline{lcd}(o) + \gamma \times \xi s(\overline{LCD}(o))]$$

And because *o* is a core point in D so the following propositions should be true.

$$\forall p \in \{o, N_{\text{lcd}}(\text{o})\} \Rightarrow \overline{lcd}(o) - \gamma \times s(\overline{LCD}(o)) \le \overline{lcd}(p) \ \le \overline{lcd}(o) + \gamma \times s(\overline{LCD}(o)) \Rightarrow$$

$$\xi\overline{lcd}(o) - \gamma \times \xi s(\overline{LCD}(o)) \le \xi\overline{lcd}(p) \le \xi\overline{lcd}(o) + \gamma \times \xi s(\overline{LCD}(o)) \Rightarrow$$

$$\forall p \in \{o, N_{\text{lcd}}(\text{o})\} \rightarrow \xi\overline{lcd}(p) \in [\xi\overline{lcd}(o) - \gamma \times \xi s(\overline{LCD}(o)), \xi\overline{lcd}(o) + \gamma \times \xi s(\overline{LCD}(o))]$$

The last proposition is contradiction with our primary assumption. Therefore all the core points in D are also core points in ξD . Similarly it is easy to prove that the points which are not core in D are not also core points in ξD .

Corollary: MSDBSCAN is scale independent algorithm.

The main part of algorithm is *runMSDBSCAN* procedure. This procedure only dealing with core points and the scale of data points are not important. When the core points of ξD are the same as D, by applying the runMSDBSCAN on ξD the results are the same as applying on D.

4.1 How to Set Parameters?

In section 4, we described how the algorithm works. The only point that remains is how the input parameters should be set. For having an adequate result for clustering it is important to set parameters properly. In MSDBSCAN there are two parameters MinPts and γ .

For a certain MinPts, if the value of γ is so large, the algorithm will return only one cluster and similarly if the value of of γ is small, the result of algorithm will be a large number of small clusters. For tuning the input parameters of algorithm, we must find out the roll of parameter γ in MSDBSCAN.

For this purpose we rewrite the core condition of definition 5 in other way. According to definition 5, *o* will be a core point if it satisfies the following condition:

$$\forall p \in \{N_{\text{lcd}}(\text{o})\} \rightarrow \overline{lcd}(p) \in [\overline{lcd}(o) - \gamma \times s(\overline{LCD}(o)), \overline{lcd}(o) + \gamma \times s(\overline{LCD}(o))]$$

$$\overline{lcd}(o) - \gamma \times s(\overline{LCD}(o)) \le \overline{lcd}(p) \le \overline{lcd}(o) + \gamma \times s(\overline{LCD}(o)) \Rightarrow -\gamma \le \frac{\overline{lcd}(p) - \overline{lcd}(o)}{s(\overline{LCD}(o))} \le \gamma$$

$$\Rightarrow CoreCondition(o) = \max_{p \in \{N_{\text{lcd}}(\text{o})\}}\{\frac{|\overline{lcd}(p) - \overline{lcd}(o)|}{s(\overline{LCD}(o))}\} \le \gamma \qquad (5)$$

The last proposition means that *o* will be a core point if it satisfies the *o*'s CoreCondition inequality. To guess the value of γ for a specific MinPts, we can use cumulative chart for CoreCondition. For each point such as *o* we calculate automatically the maximum value of $\frac{|\overline{lcd}(p) - \overline{lcd}(o)|}{s(\overline{LCD}(o))}$ and set this value as CoreCondition of *o*. Now we draw the CoreCondition cumulative chart. The horizontal coordination of this diagram represents the different values of CoreCondition and the vertical coordination of this diagram represents the percentage of objects which their CoreConditions are less than a certain CoreCondition.

In Fig. 1, we guess about 90% (a percentage in interval [89%-93%]) of points should be core. The CoreCondition cumulative chart for this data set is shown in Fig. 2.

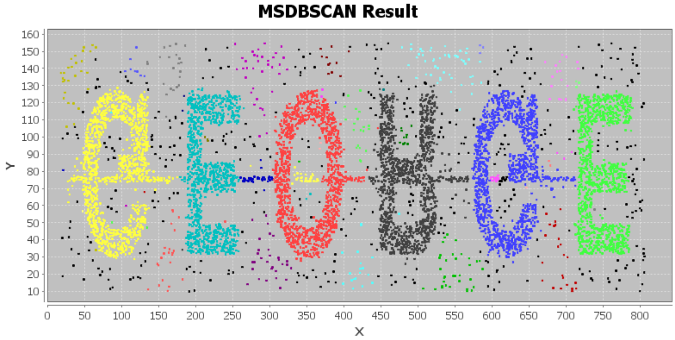


Fig. 1. MSDBSCAN result on t5.8k with MinPts=9, γ =0.37

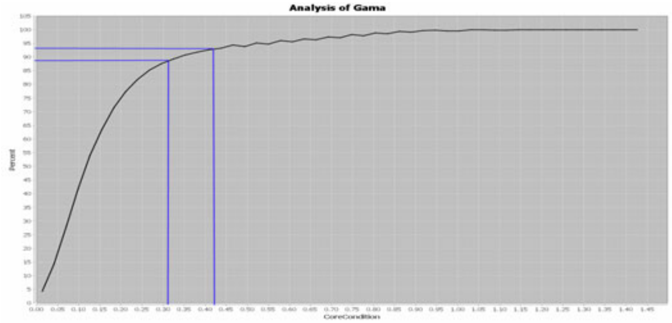


Fig. 2. CoreCondition cumulative chart

5 Experimental Results and Evaluations

In this section, we compare the results of MSDBSCAN to LDBSCAN and OPTICS in practice. We ran algorithm over a series of selected data sets from the chameleon data sets which are available from [10]. To test algorithm in multi-variant density environment, we generated a series of synthetic data sets too. Also we tried to test finding overlapping clusters in synthetic data sets. The results obtained by these tests shown that the MSDBSCAN is better than OPTICS and LDBSCAN in multi-variant density environment. In this section, we will also demonstrate the scale independency of algorithm on real data sets experimentally.

The experiments were performed by using a laptop with 3Gb of RAM, Intel Core 2Duo, CPU 2.1GHz. The OS used for this experiment was Ubuntu 9.10. The clustering algorithm was implemented by using Sun's JDK version 1.6.

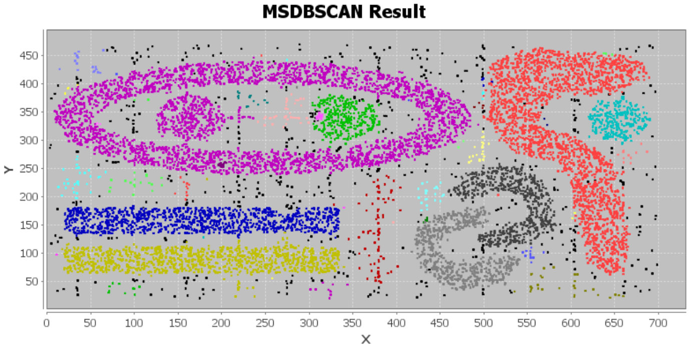


Fig. 3. MSDBSCAN result on t7.10k with MinPts=9, γ =0.25

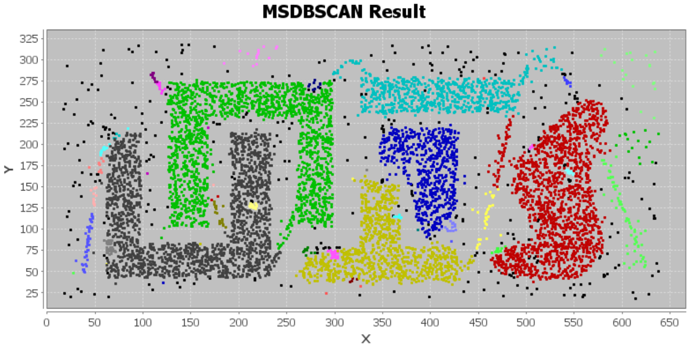


Fig. 4. MSDBSCAN result on t4.8k with MinPts=8, γ =0.26

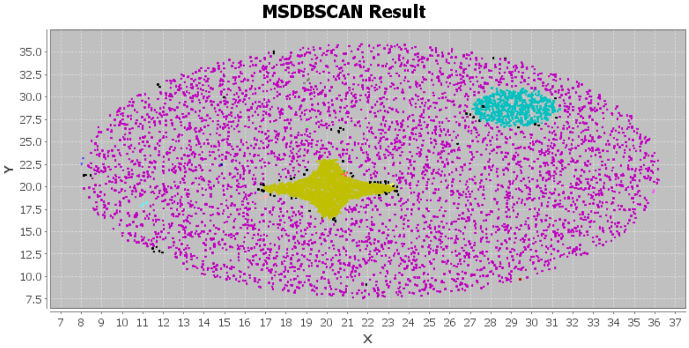


Fig. 5. MSDBSCAN result on synthetic data set with MinPts=7, γ =0.34

The Fig. 3, Fig. 4 are the outputs of MSDBSCAN for chameleon data sets (t7.10k, t4.8k). The output of applying MSDBSCAN on t5.8k was already shown in Fig. 1. The results of MSDBSCAN are so similar to OPTICS algorithm.

Fig. 5 shows the output of MSDBSCAN algorithm on a synthetic data set. The data set contains 9000 points and 3 clusters with different densities. The MSDBSCAN has identified 3 clusters appropriately. The result of OPTICS shows that it can identify only two clusters, yellow and jade. If the epsilon selected for OPTICS, is so small, only yellow cluster can be determined and the other points are identified as outlier points and if the epsilon selected for OPTICS, is too large, all the data points are returned as a single cluster. The average of these two states will return the yellow and jade cluster and other points will be the outliers.

We also tested LDBSCAN algorithm on chameleon data sets and our synthetic data set shown in Fig. 5. The LDBSCAN takes 4 input parameters $MinPts_{LOF}, MinPts_{LDBSCAN}, pct$ and $LOFUB$ The algorithm has not provided any guideline for good determination of these parameters and only gave an interval for some of them. The algorithm has recommended to set $MinPts_{LOF}, pct$ and $LOFUB$ in intervals [10-20],[0.3-0.5] and [1.5-2.5], respectively. Also we considered the interval [6-12] for $MinPts_{LDBSCAN}$. The best result of LDBSCAN through theses values is shown in Fig. 6. The algorithm only found two major clusters. We also applied LDBSCAN on chameleon datasets. The results are so similar to the results of MSDBSCAN except the t7.10k dataset which is shown in Fig. 7.

Our algorithm can also find the arbitrary shapes of clusters in multi-variant density data sets. To test this property, we constructed a data set with different type of shapes. The data set is shown in Fig. 8. This new data set contains about 3000 points. The density of each shape in this data set is different from the density of its neighbors. The algorithm has properly found seven different clusters for this data set.

We applied LDBSCAN on this new synthetic data set and tested all possible values for parameters of LDBSCAN as before. The best clustering for these values is shown in Fig. 9. LDBSCAN only found six clusters among seven existence clusters.

Also, we tested the scale independency of MSDBSCAN in this experiment on t7.10k. The output of algorithm was exactly the same as shown in the Fig. 3.

In the rest of this section we discuss about the time and space complexity of the algorithm. The algorithm has three main procedures which the total time complexity is the maximum time complexity of these procedures. In two first steps of the algorithm, it finds the neighbors of each point and computes a series of operations on them. As mentioned in definition 1 of section 3, in most cases, the average cardinality of lcd-distance-neighborhood of each point almost equals to the MinPt. Therefore, the time complexity of this algorithm depends on the k-nearest-neighbors query and equals to *O(n*time for a k-nn query)*. As the structure of algorithm is similar to DBSCAN, so the runtime of the algorithm can be $O(n \log n)$ if we have spatial index on our data set, otherwise the time complexity of algorithm will be $O(n^2)$. The algorithm also materializes the lcd-distance-neighborhood of each object in the first step. So MSDBSCAN takes almost $O(MinPts \times n)$ memory space.

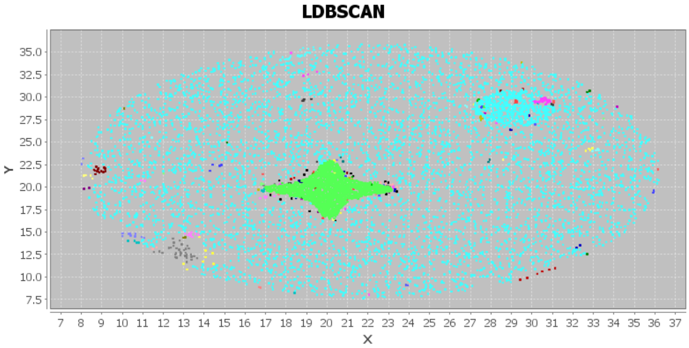


Fig. 6. LDBSCAN result on synthetic data set with $MinPts_{LOF}$=13, $MinPts_{LDBSCAN}$=11, pct=0.3 and LOFUB= 2.5

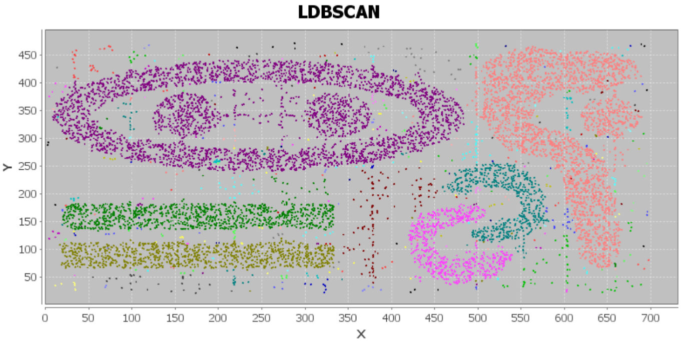


Fig. 7. LDBSCAN result on t7.10k with $MinPts_{LOF}$=12, $MinPts_{LDBSCAN}$=7,pct=0.2 and LOFUB= 2.3

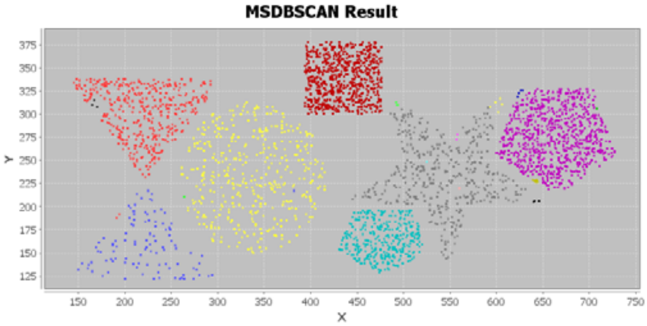


Fig. 8. MSDBSCAN result on synthetic data set with MinPts=5 , γ =0.45

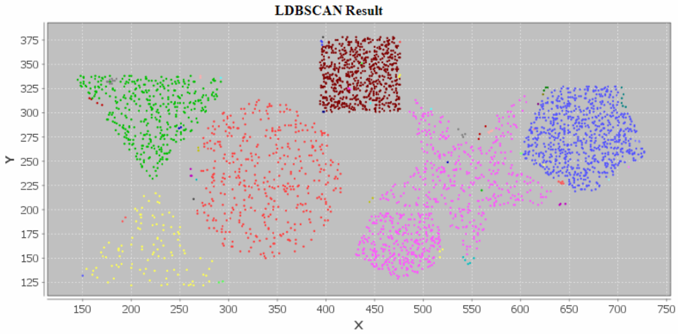


Fig. 9. LDBSCAN result on synthetic data set with $MinPts_{LOF}$=10, $MinPts_{LDBSCAN}$=6, pct=0.2 and LOFUB= 1.9

6 Conclusion and Future Work

In this paper we presented a new density based clustering algorithm MSDBSCAN which relies on localizing concept of core points based on the position of their neighbors. The experiments show that MSDBSCAN is significantly more effective in discovering multi-variant clusters especially overlapping clusters than the well-known algorithms such as OPTICS and LDBSCAN. The scale independency is another characteristic of MSDBSCAN which was proved and experimented in this paper

We are now working to propose a special representation of objects such as mentioned in OPTICS to determine γ. This representation will be complementary to the CoreCondition cumulative chart.

Acknowledgement. The authors would like to acknowledge the financial support of Iran Telecommunication Research Center (ITRC) for this paper.

References

1. Kriegel, H.P., Sander, J., Xu, X., Ester, M.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Menlo Park, CA, pp. 226–231 (1996)
2. Breunig, M.M., Kriegel, H.P., Sander, J., Ankerst, M.: OPTICS: Ordering Points To Identify The Clustering Structure. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, Philadelphia, USA, pp. 49–60 (1999)
3. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 2nd edn., Diane Cerra, USA (2006)
4. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceeding of the Fifth Berekely Symposium on Mathematics, Statistics and Probabilities, vol. 1, pp. 281–297 (1967)
5. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley Interscience, Hoboken (2005)

6. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: an efficient data clustering method for very large databases. SIGMOD Record 25, 103–114 (1996)
7. Karypis, G., Han, E., Kumar, V.: Chameleon: Hierarchical Clustering Using Dynamic Modeling. Computer 32, 68–75 (1999)
8. Duan, L., Xu, L., Guo, F., Lee, J., Yan, B.: A local-density based spatial clustering algorithm with noise. Information Systems 32, 978–986 (2007)
9. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. SIGMOD Record 29, 93–104 (2000)
10. Karypis, G.: Karypis Lab (2010), `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download`

An Efficient Algorithm for Mining Erasable Itemsets

Zhihong Deng[1,2] and Xiaoran Xu[1]

[1] Key Laboratory of Machine Perception (Ministry of Education),
School of Electronics Engineering and Computer Science,
Peking University, Beijing 100871, China
[2] The State Key Lab of Computer Science, Institute of Software,
Chinese Academy of Sciences, Beijing 100190, China
zhdeng@cis.pku.edu.cn, xuxiaoran@pku.edu.cn

Abstract. Mining erasable itemsets first introduced in 2009 is one of new emerging data mining tasks. In this paper, we present a new data representation called PID_list, which keeps track of the id_nums (identification number) of products that include an itemset. Based on PID_list, we propose a new algorithm called VME for mining erasable itemsets efficiently. The main advantage of VME algorithm is that the gain of an itemset can be computed efficiently via union operations on product id_nums. In addition, VME algorithm can also automatically prune irrelevant data. For evaluating VME algorithm, we have conducted experiments on six synthetic product databases. Our performance study shows that the VME algorithm is efficient and is on average over two orders of magnitude faster than the META algorithm, which is the first algorithm for dealing with the problem of erasable itemsets mining.

Keywords: Data mining, Erasable Itemsets, Algorithm, PID_list.

1 Introduction

Since the problem of mining frequent patterns first introduced in [2], it has emerged as a fundamental problem in data mining and plays an essential role in many important data mining tasks such as association rule analysis, cluster analysis, classification, and many other important data mining tasks [2]. Although there are plenty of studies on pattern mining, such as in [3], many new pattern-mining problems have arisen, such as high-utility pattern mining [4], probabilistic frequent itemsets mining [5], erasable itemsets mining [6], and so on, with the extensive application of pattern mining in every walk of life.

The problem of mining erasable itemsets originates from production planning. Consider a manufacturing factory, which produces a large collection of products. Each type of product is made up of a few components (or materials). For manufacturing their products, the factory should spend a large number of money to purchase or store these components. When financial crisis is coming, the factory should carefully plan production because it has not enough money to purchase all needed components as usual. Therefore, a vital question to the managers of the factory is how to plan the

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 214–225, 2010.

manufacture of production due to limited money. They can not purchase all components due to limited money. Obviously, they must stop manufacturing some productions because the corresponding components are unavailable. However, for the sake of commercial interests, the loss of the factory's profit caused by stopping manufacturing some products should be controllable. Hence, the key to the problem is how to efficiently find these components, without which the loss of the profit is no more than the given threshold. These components are also called as erasable itemsets. The paper [6] first introduced the problem of erasable itemsets mining and proposed META algorithm to deal with the problem.

Although META algorithm is capable of finding all erasable itemsets in reasonable time, it has two important weaknesses. The first weakness is that the time efficiency of META algorithm is poor because it scans database repeatedly. The second weakness is that META algorithm can not automatically prune irrelevant data.

In this paper, we present a new algorithm, which can overcome the weaknesses of META, to mine erasable itemsets efficiently. First, we present a new data representation called PID_list, which keeps track of the id_nums of products that include an itemset. Second, we propose a PID_list-based algorithm called VME for mining erasable itemsets efficiently. The main advantage of VME algorithm is that the gain of an itemset can be computed efficiently via union operations on product id_nums. In addition, VME algorithm can also automatically prune irrelevant data. For evaluating VME algorithm, we have conducted experiments on six synthetic product databases. Our performance study shows that the VME algorithm is efficient and is on average over two orders of magnitude faster than the META algorithm, which is the first algorithm for dealing with the problem of erasable itemsets mining.

The organization of the rest of the paper is as follows. Section 2 introduces the formal statement of the problem. Section 3 introduces the PID_list structure and its properties. Section 4 develops a PID_list-based erasable itemsets mining algorithm, VME. Section 5 presents our performance study. In section 6, we conclude with a summary and point out some future research issues.

2 Problem Statement and Preliminaries

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items, which are the abstract representation of components of products, and a product database $DB = \{P_1, P_2, \ldots, P_n\}$, where P_i ($i \in [1\ldots n]$) is a type of product and is presented in the form of < *PID*, *Items*, *Val* >. *PID* is the identifier of P_i. *Items* are all items (or components) that constitute P_i. *Val* is the profit that a manufactory (or factory) reaps (or obtains) by selling all P_i-type products. In this section, related concepts are described and a formal description of erasable itemset is given.

Table 1 shows an example product database. The database has six types of product. $\{i_1, i_2, i_3, i_4, i_5, i_6, i_7\}$ is the set of universal items that comprise the six types of products .Let's take P_3 as an example. We know that we need four different kind of item, which are i_1, i_2, i_3 and i_5, to manufacture P_3-type products. It would profit 50 million dollars by selling all P_3-type products.

Table 1. Example product database

Product	***PID***	***Items***	***Val* (Million $)**
P_1	1	$\{i_2, i_3, i_4, i_6\}$	50
P_2	2	$\{i_2, i_5, i_7\}$	20
P_3	3	$\{i_1, i_2, i_3, i_5\}$	50
P_4	4	$\{i_1, i_2, i_4\}$	800
P_5	5	$\{i_6, i_7\}$	30
P_6	6	$\{i_3, i_4\}$	50

Definition 2.1: Let A ($\subseteq I$) be an itemset (a set of items), the gain of A is defined as:

$$Gain(A) = \sum_{\{P_k \mid A \cap P_k.Items \neq \phi\}} P_k.Val\,. \tag{1}$$

That is, the gain of itemset A is the sum of profits of all products that include at least one item in A as their components.

Let P (=$\{i_6, i_7\}$) be an itemset. From table 1, we know that the products, who's *Items* contain i_6 or i_7, are P_1, P_2 and P_5. Therefore, the gain of P is the sum of $P_1.Val$, $P_2.Val$, and $P_5.Val$. That is, $Gain(P)$ is 100 million dollars.

Definition 2.2: Given a predefined threshold ξ and a product database *DB*, an itemset A is erasable if

$$Gain(A) \leq (\sum_{P_k \in DB} P_k.Val) \times \xi\,. \tag{2}$$

Based on the above definitions, the problem of mining erasable itemsets can be described as follows:

Given a product database, *DB*, and a threshold, ξ, the problem of finding the complete set of erasable itemsets is called the erasable itemsets mining problem.

Erasable itemsets are those itemsets that without these items, the loss of profits does not exceed ξ percents of the original profits. For example, let ξ be 10%. $\{i_6, i_7\}$ is an erasable. If a manufactory does not purchase i_6 and i_7 as raw materials because of economic crisis, the manufactory can not manufacture products that are type of P_1, P_2 or P_5. However, the lost profit is no more than 10% of the original profit. Erasable itemsets is especially useful for manufactures to decide how to purchase raw materials and plan the process of manufacturing products in the case of economic crisis.

Consider the example product database shown in Table 1. Let ξ be 15%. According to Definition 2, the entire erasable itemsets are $\{i_3\}$ (150), $\{i_5\}$ (70), $\{i_6\}$ (80), $\{i_7\}$ (50), $\{i_5, i_6\}$ (150), $\{i_5, i_7\}$ (100), $\{i_6, i_7\}$ (100), and $\{i_5, i_6, i_7\}$ (150). Note that the numbers in parentheses are the gains of corresponding itemsets.

3 PID_list: Definition and Property

Obviously, the key question of erasable itemset mining is how to compute the gain of an itemset. META algorithm employs the method of scanning database to get the gains. However, after some careful examination, we find that there exist more efficient methods if we change the date format of product databases.

To design an efficient data structure for fast mining, let's first examine the database showed by Table 2. By careful observation we find the database in Table 2 is an inversion of the database in Table 1. The data format employed by the database in Table 1 is known as horizontal data format while the data format employed by the database in Table 2 is known as vertical data format.

Table 2. The inversion database

Item	*Inverted List*
i_1	<3, 50>, <4, 800>
i_2	<1, 50>, <2, 20>, <3, 50>, <4, 800>
i_3	<1, 50>, <3, 50>, <6, 50>
i_4	<1, 50>, <4, 800>, <6, 50>
i_5	<2, 20>, <3, 50>
i_6	<1, 50>, <5, 30>
i_7	<2, 20>, <5, 30>

One advantage of vertical data format is that we can very convenient to obtain the gain of itemsets. For example, the gain of $\{i_3\}$ is the sum of the second part of <1, 50>, <3, 50>, and <6, 50>. That is, it is 150 (50+50+50). In fact, not only 1-itemsets but also k-itemset ($k > 1$) can also be easy computed by a similar method, where a k-itemset means an itemset including k items. For example, by combing the *Inverted List* of $\{i_5\}$, {<2, 20>, <3, 50>}, and the *Inverted List* of $\{i_6\}$, {<1, 50>, <5, 30>}, we generate {<1, 50>, <2, 20>, <3, 50>, <5, 30>}. By summing 50, 20, 50, and 30, we get 150, which is the gain of $\{i_5, i_6\}$. As for the rationality, the remainder of this section will be discussed in detail.

The other advantage of vertical data format is that it can automatically prune irrelevant data. Let's see $\{i_3\}$ again. If we want to get the gain of $\{i_3\}$ by database scanning, we must compare it with each product of the database in Table 1 to examine whether it is a component of the product. Obviously, comparing $\{i_3\}$ with P_1, P_2 and P_5 is nonsense because they do not take i_3 as their component at all. If we do this by searching the *Inverted List* of $\{i_5\}$, P_1, P_2 and P_5 are filtered automatically because they are not in the *Inverted List* of $\{i_5\}$.

Before we introduce PID_list, we give two conventions in the paper for the sake of discussion. First, we assume that $I = \{i_1, i_2, \dots, i_m\}$ is the given universal item set and $DB = \{P_1, P_2, \dots, P_n\}$ is the given product database. Second, we denote an itemset $X(\subseteq I)$ by $\{i_{x1}, i_{x2}, \dots, i_{xk}\}$, where $i_{xj} \in I$ for $1 \leq j \leq k$ and $x_s < x_t$ for $1 \leq s < t \leq k$. That is, any itemset is an ordered set and is sorted by the order that items occur in I.

Definition 3.1: For any $y \in I$, the PID_list of 1-itemset $\{y\}$ is $\{<P_{y1}.PID, P_{y1}.Val>, <P_{y2}.PID, P_{y2}.Val>, \dots, <P_{yk}.PID, P_{yk}.Val>\}$, where *y is a component of* P_{yj} for $1 \leq j \leq k$ (that is, $y \in P_{yj}.Items$) and $P_{yv}.PID$ is less than $P_{yu}.PID$ for $v < u$. that is, the elements in the PID_list are sorted by the ascending order of *PID*.

For example, the PID_list of $\{i_2\}$ is <1, 50>, <2, 20>, <3, 50>, <4, 800>.
According to the definition 2.1 and definition 3.1, we have the following Property 3.1.

Property 3.1: Let $\{y\}$ be a 1-itemset and its PID_list is $\{< P_{y1}.PID, P_{y1}.Val >, < P_{y2}.PID, P_{y2}.Val >, \ldots, < P_{yk}.PID, P_{yk}.Val >\}$. The gain of $\{y\}$ can be computed as follows:

$$Gain(\{y\}) = \sum_{j=1}^{k} P_{yj}.Val \ . \tag{3}$$

For example, the gain of $\{i_2\}$ is 920.

Based on Definition 3.1, we define the PID_list of a k-item (k >1) as follows.

Definition 3.2: Let $X = \{i_{x1}, i_{x2}, \ldots, i_{xk}\}$ be a k-itemset. For each i_{xj} $(1 \le j \le k)$, we denote the PID_list of $\{i_{xj}\}$ as PID_list ($\{i_{xj}\}$). PID_list(X), the PID_list of X, is $\{< P_{X1}.PID, P_{X1}.Val >, < P_{X2}.PID, P_{X2}.Val >, \ldots, < P_{Xs}.PID, P_{Xs}.Val >\}$, which meets the following three conditions:

(1) $P_{xv}.PID$ is less than $P_{xu}.PID$ for $1 \le v < u \le s$;
(2) $\forall < P_{Xj}.PID, P_{Xj}.Val > (1 \le j \le s) \in$ PID_list(X) $\Rightarrow \exists\ i_{xv} \in X, < P_{Xj}.PID, P_{Xj}.Val > \in$ PID_list ($\{i_{xv}\}$);
(3) For any $i_{xv} \in X$ $(1 \le v \le k)$, $\forall < P_j.PID, P_j.Val > \in$ PID_list ($\{i_{xv}\}$) $\Rightarrow < P_j.PID, P_j.Val > \in$ PID_list(X).

In fact, PID_list(X) is the union of the PID_lists of all items that belongs to X with the condition that its elements are sorted by the ascending order of *PID*.

Let's see an example. From Table 2, we know the PID_lists of $\{i_3\}$, $\{i_5\}$, $\{i_6\}$ are {<1, 50>, <3, 50>, <6, 50>}, {<2, 20>, <3, 50>}, and {<1, 50>, <5, 30>}. According to Definition 4, the PID_list of $\{i_3, i_5, i_6\}$ is {<1, 50>, <2, 20>, <3, 50>, <5, 30>, <6, 50>}.

If we sum the second part of each element in the PID_list of $\{i_3, i_5, i_6\}$, we would find that it is 200. Surprisingly, the gain of $\{i_3, i_5, i_6\}$ is also 200. This is not an accidental phenomenon. In fact, we have the following Property 3.2.

Property 3.2. Let X be a itemset and its PID_list is $\{< P_{X1}.PID, P_{X1}.Val >, < P_{X2}.PID, P_{X2}.Val >, \ldots, < P_{Xs}.PID, P_{Xs}.Val >\}$. The gain of X can be computed as follows:

$$Gain(X) = \sum_{j=1}^{s} P_{Xj}.Val \ . \tag{4}$$

Proof. If X be 1-itemset, we know equation (4) is right according to Property 3.1.Then, we see the case that X is a k-itemset (k > 1). According to Definition 2.1, the gain of X is the sum of profits of all products that include at least one item in X as their components. The set of these products can be formally represented by $\{P_k \mid X \cap P_k.Items \ne \emptyset\}$. The products that occur in PID_list of X can be formally represented by $\{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$. So, we convert the proof of equation (4) to the proof of $\{P_k \mid X \cap P_k.Items \ne \emptyset\} = \{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$. We split the proof into two steps as follows.

(***a***) $\{P_k \mid X \cap P_k.Items \ne \emptyset\} \subseteq \{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$

For any $P \in \{P_k \mid X \cap P_k.Items \ne \emptyset\}$, we know that there must exist i, $i \in X \cap P_k.Items$. According to the Definition 3.1, <$P.PID$, $P.Val$> must be an element of

PID_list ($\{i\}$) because of $i \in P_k.Items$. According to the condition (3) of Definition 3.2, we have $<P.PID, P.Val> \in$ PID_list(X). That is, $P \in \{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$. So, we have $\{P_k \mid X \cap P_k.Items \neq \varnothing\} \subseteq \{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$.

(***b***) $\{P_{X1}, P_{X2}, \ldots, P_{Xs}\} \subseteq \{P_k \mid X \cap P_k.Items \neq \varnothing\}$

For any $P \in \{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$, we know $<P.PID, P.Val> \in$ PID_list(X) according to the definition of $\{P_{X1}, P_{X2}, \ldots, P_{Xs}\}$. According to the condition (2) of Definition 3.2, we know $\exists\, i \in X$, $<P.PID, P.Val> \in$ PID_list ($\{i\}$). According to Definition 3.1, we have $i \in P.Items$. So, we have $(i \in X) \wedge (i \in P.Items)$. That is, $P \in \{P_k \mid X \cap P_k.Items \neq \varnothing\}$. Therefore, we have $\{P_{X1}, P_{X2}, \ldots, P_{Xs}\} \subseteq \{P_k \mid X \cap P_k.Items \neq \varnothing\}$.

Consolidating (***a***) and (***b***), we prove that equation (4) is right when X is a k-itemset ($k > 1$).

Based on the above analysis, we prove Property 3.2. □

4 Mining Erasable Itemsets Using PID_list

Before presenting our method, let's first explore the following lemmas relevant to erasable itemsets mining.

Lemma 4.1: Let X ($\subseteq I$) and Y ($\subseteq I$) be two itemsets. If Y is a superset of X ($X \subseteq Y$), we have $Gain(X) \leq Gain\ (Y)$.

Proof. Let P_k be any product that satisfies $P_k.Items \cap X \neq \varnothing$. That is, $P_k.Items$ and X share at least one item. Because Y is X's superset, we have $P_k.Items \cap Y \neq \varnothing$. So, we have $\{P_k \mid P_k.Items \cap X \neq \varnothing\} \subseteq \{P_k \mid P_k.Items \cap Y \neq \varnothing\}$. According to Definition 2.1, we know

$$\sum_{\{P_k \mid X \cap P_k.Items \neq \phi\}} P_k.Val \leq \sum_{\{P_k \mid Y \cap P_k.Items \neq \phi\}} P_k.Val\,. \tag{5}$$

This means $Gain(X) \leq Gain(Y)$. □

Let's take Table 1 as an example. The values of itemset $\{i_6\}$, $\{i_6, i_7\}$, $\{i_3, i_6, i_7\}$ are 80, 100, and 200 respectively. It is obvious that $Val(\{i_6\}) \leq Val(\{i_6, i_7\}) \leq Val(\{i_3, i_6, i_7\})$.

Based on Lemma 4.1, we have Lemma 4.2 as follows.

Lemma 4.2 (anti-monotone): if itemset X is unerasable and Y is a superset of X ($X \subseteq Y$), Y must also be unerasable.

Proof. Let Y be a superset of X. We assume Y is erasable. According to Definition 2.2, we have

$$Gain(Y) \leq (\sum_{P_k \in DB} P_k.Val) \times \xi\,\cdot \tag{6}$$

Because X is unerasable, we have

$$Gain(X) > \sum_{P_k \in DB} P_k.Val \times \xi\,\cdot \tag{7}$$

Based on inequation (6) and inequation (7), we know *Gain*(*X*) > *Gain*(*Y*). However, according to Lemma 4.1, we know *Gain*(*X*) ≤ *Gain*(*Y*). These two conclusions conflict each other. Therefore, our assumption is wrong and *Y* must be unerasable. □

Based on Lemma 4.2, we employ an iterative approach known as a *level-wise* search, which is also adopted by Apriori algorithm [3] in frequent patterns mining. The *level-wise*-based iterative approach finds erasable (*k*+1)-itemsets by taking use of erasable *k*-itemsets.

The detailed idea of the *level-wise*-based iterative approach is described as follows. First, the set of erasable 1-itemsets is found. This set is denoted as E_1. E_1 is used to find E_2, which is the set of erasable 2-itemsets, and then E_2 is used to find E_3, and so on, until no more erasable *k*-itemsets can be found. The finding of each E_j requires one scan of the database. To improve the efficiency of the level-wise generation of erasable itemsets, Lemma 4.2 is used to narrow the range of search.

Based on the above discussions, we design an algorithm called VME algorithm. VME is the abbreviation for **V**ertical-format-based algorithm for **M**ining **E**rasable itemsets. The details of the VME algorithm are described as follows.

```
Algorithm: VME
Input: a product database, DB, a itemsets, I, and a
threshold, ξ.
Output: EI, all erasable itemsets in DB.
Method:
Scan DB to get the overall profit, Sum_val;
Scan DB again to find the set of all erasable 1-
itemset, $E_1$, and the PID_list of each erasable 1-
itemset;
For (k = 2; $E_{k-1} \neq \varnothing$; k++) {
  $GC_k$ = Gen_Candidate($E_{k-1}$);
  $E_k = \varnothing$;
  For each k-itemset $P \in GC_k$ {
    Compute P.gain according to Property 3.2;
    If $P.gain \leq \xi \times Sum\_val$ then $E_k = E_k \cup \{P\}$;}}
Return $EI = \cup_k E_k$;
// Generating candidate itemsets and their PID_lists
Procedure Gen_Candidate($E_{k-1}$)
Candidates = $\varnothing$;
For each erasable itemset $A_1 (=\{x_1, x_2, \ldots x_{k-2}, x_{k-1}\}) \in E_{k-1}$ {
  For each erasable itemset $A_2 (=\{y_1, y_2, \ldots y_{k-2}, y_{k-1}\}) \in E_{k-1}$
{
   //Here, $x_{k-1}$ < $y_{k-1}$ means $x_{k-1}$ is ahead of $y_{k-1}$ in $I$ .
    If (($x_1$= $y_1$)∧ ($x_2$= $y_2$)∧…∧ ($x_{k-2}$= $y_{k-2}$) ∧( $x_{k-1}$ < $y_{k-1}$)) then
{
     $X$ = {$x_1$, $x_2$ , …$x_{k-2}$, $x_{k-1}$, $y_{k-1}$};
     If No_Unerasable_Subset($X$, $E_{k-1}$) then {
        $X$. PID_list = $A_1$.PID_list ∪ $A_2$.PID_list;
         Candidates = Candidates ∪ {($X$, $X$. PID_list)}; }
     } } }
Return Candidates;


Procedure **No_Unerasable_Subset**($X$, $E_{k-1}$).
// $X$: a candidate $k$-itemset
//$E_{k-1}$: the set of all erasable ($k$ -1)-itemsets
For each ($k$ -1)-subset $X_s$ of $X$ {
   If $X_s \notin E_{k-1}$ then Return FALSE; }
Return TRUE;
```

5 Experimental Evaluation

In this section, we present a performance comparison of VME with the first erasable itemsets mining algorithms META algorithm. All the experiments were performed on an IBM xSeries 366 Server with 2G Memory. The operating system was Microsoft Windows 2000 Server. All the programs were coded in MS/Visual C++.

For experimental databases, we first generated three databases by IBM generator[1]. These databases are denoted by T15I10D100K, T20I10D100K, and T30I25D100K. Each of the three databases has 100, 000 tuples (or products) and 200 different items. The average production sizes of T15I10D100K, T20I10D100K, and T30I25D100K are 15, 20, and 30 respectively.

However, T15I10D100K, T20I10D100K, and T30I25D100K have no attribute (or column) for representing profit of products. To make these databases more like product databases, we add a new attribute (column or field), which is used to store the profit of a product, for each database. We employ two probability distributions, U(1, 100) and N(50, 25), to generate products' profit. U(1, 100) is an uniform distribution with the range of values of [1, 100]. N(50, 25) is a normal distribution with mean of 50 and variance of 25. Therefore, we have six databases by combining T15I10D100K, T20I10D100K, and T30I25D100K with U(1, 100) and N(50, 25). Table 3 shows the details of these databases.

[1] http://www.almaden. ibm.com/cs/quest/syndata.html

Table 3. The summary of the database

Database	*#Product*	*#Items*	*Probability Distribution of Profits*
T15U1_100	100,000	200	*U*(1, 100)
T15N50_25	100,000	200	*N*(50, 25)
T20U1_100	100,000	200	*U*(1, 100)
T20N50_25	100,000	200	*N*(50, 25)
T30U1_100	100,000	200	*U*(1, 100)
T30N50_25	100,000	200	*N*(50, 25)

The scalability of VME and META on T15U1_100 and T15N50_25 as the threshold increases from 1% to 7% is shown in Figure 1 and Figure 2 respectively. Figure 3 and Figure 4 respectively show the scalability of VME and META on T20U1_100 and T20N50_25 as the threshold decreases from 2% to 10%. Figure 5 and Figure 6 respectively show the scalability of VME and META on T30U1_100 and T30N50_25 as the threshold decreases from 3% to 15%. As shown in Figure 1 – 6, VME scales much better than META. Not matter which database is used, VME algorithm is always over two orders of magnitude faster than META algorithm on average. Especially, the bigger the threshold is, the more distinct the advantage of VME over META is. The efficiency of VME can be explained by two main factors: (1) computing the gain of an itemset via PID_lists is much efficient; (2) PID_lists also provide a natural way to prune irrelevant data automatically.

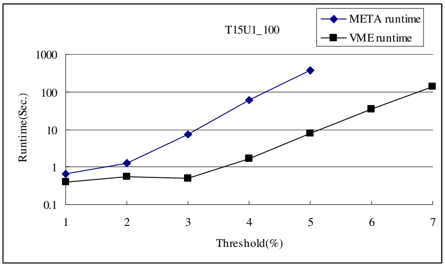


Fig. 1. Comparative performance on T15U1_100

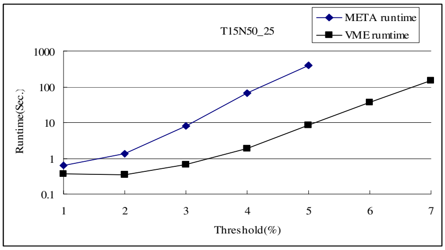


Fig. 2. Comparative performance on T15N50_25

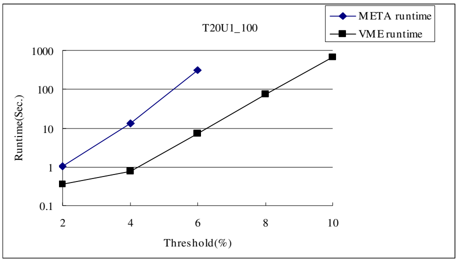


Fig. 3. Comparative performance on T20U1_100

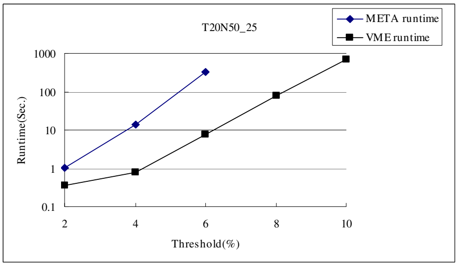


Fig. 4. Comparative performance on T20N50_25

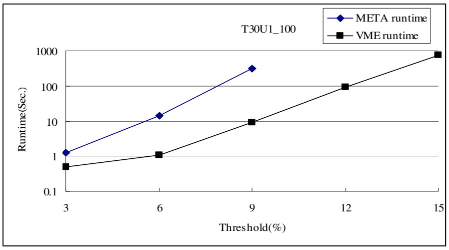


Fig. 5. Comparative performance on T30U1_100

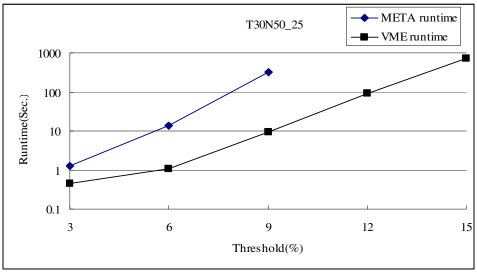


Fig. 6. Comparative performance on T30N50_25

6 Conclusions

In this paper, we present a new data representation called PID_list for storing compressed, crucial information about itemsets of a production database. Based on PID_list, we develop a new algorithm called VME for mining erasable itemsets efficiently. VME algorithm not only makes it easy to compute the gain of an itemset via union operations on product id_nums, but also automatically prune irrelevant data. For evaluating VME algorithm, we have conducted experiments on six synthetic product databases. Our performance study shows that the VME algorithm is efficient and is on average over two orders of magnitude faster than the META algorithm, the first algorithm for mining erasable itemsets.

For the future work, there is a lot of interesting research issues related to erasable itemsets mining. First, we will take efforts towards more efficient algorithms by adopting useful ideas from many proposed algorithms of mining frequent patterns. Second, there have been some interesting studies at mining maximal frequent [7], closed frequent patterns [8] and top-k frequent patterns [9, 10] in recent years. Similar to frequent patterns, the extension of erasable itemsets to these special forms is an interesting topic for future research.

Acknowledgments. This work is partially supported by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2009AA01Z136 and the National Natural Science Foundation of China under Grant No.90812001. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. In: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pp. 207–216. ACM Press, New York (1993)
2. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 1–12. ACM Press, New York (2000)
3. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
4. Hu, J., Mojsilovic, A.: High-utility pattern mining: A method for discovery of High-utility item sets. Pattern Recognition 40, 3317–3324 (2007)
5. Bernecker, T., Kriegel, H., Renz, M., Verhein, F., Zuefle, A.: Probabilistic Frequent Itemset Mining in Uncertain Databases. In: 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 119–127. ACM Press, New York (2009)
6. Deng, Z., Fang, G., Wang, Z., Xu, X.: Mining Erasable Itemsets. In: 8th IEEE International Conference on Machine Learning and Cybernetics, pp. 67–73. IEEE Press, New York (2009)
7. Burdick, D., Calimlim, M.,, Flannick, J., Gehrke, J., Yiu, T.: MAFIA: A Maximal Frequent Itemset Algorithm. IEEE Trans. Knowledge and Data Engineering 17, 1490–1504 (2005)

8. Wang, J., Han, J., Pei, J.: CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Patterns. In: 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 236–245. ACM Press, New York (2003)
9. Han, J., Wang, J., Lu, Y., Tzvetkov, P.: Mining Top-k Frequent Closed Patterns without Minimum Support. In: Second IEEE International Conference on Data Mining, pp. 211–218. IEEE Press, New York (2002)
10. Wang, J., Han, J., Lu, Y., Tzvetkov, P.: TFP: An Efficient Algorithm for Mining Top-k Frequent Closed Patterns. IEEE Trans. Knowledge and Data Engineering 17, 652–664 (2005)

Discord Region Based Analysis to Improve Data Utility of Privately Published Time Series

Shuai Jin, Yubao Liu, and Zhijie Li

Department of Computer Science, Sun Yat-sen University, Guangzhou, 510006, China
jingkb2000@163.com, liuyubao@mail.sysu.edu.cn, gdgzlzj@gmail.com

Abstract. Privacy preserving data publishing is one of the most important issues of privacy preserving data mining, but the problem of privately publishing time series data has not received enough attention. Random perturbation is an efficient method of privately publishing data. Random noise addition introduces uncertainty into published data, increasing the difficult of conjecturing the original values. The existing Gaussian white noise addition distributes the same amount of noise to every single attribute of each series, incurring the great decrease of data utility for classification purpose. Through analyzing the different impact of local regions on overall classification pattern, we formally define the concept of discord region which strongly influences the classification performance. We perturb original series differentially according to their position, whether in a discord region, to improve classification utility of published data. The experimental results on real and synthetic data verify the effectiveness of our proposed methods.

Keywords: privacy preserving publishing, time series, discord region, random perturbation.

1 Introduction

Privacy preserving data publishing is one of the most important issues of privacy preserving data mining, but the problem of privately publishing time series data has not received enough attention. Time series data is widely used in many applications, including some privacy sensitive sections, such as, national defense security, financial transactions and retail record etc. Privacy preserving publishing time series data requires protecting personal or organizational privacy as well as retain most data utility for various data mining purposes, such as series classification, frequent pattern discovery and time series query. In this paper we focus on the data utility for building a classification model, which we call classification utility. The more precise the model, the more data utility the publishing scheme retains.

Traditional privacy preserving publishing methods includes k-anonymity [7], SMC (secure multiparty computation) [16] and data perturbation [10]. SMC could be further classified into horizontal and vertical categories according to different data distribution among multiple sites. We study on the data privacy in the centralized scenario, for example, a hospital entrusts a third party for data mining patients' ECG

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 226–237, 2010.

(electrocardiogram) data. K-anonymity ensures data privacy through generalizing and suppressing which would cause severe information loss for time series data because of its high dimensions.

Different from existing works, we employ a region based analysis to increase classification utility of privately published data. The attribute space formed by all series could be spatially partitioned into regions which have different impact on overall classification performance, for example, all data points in one region might belong to the same class while another region contains multi-class data points. We say these two kinds of regions have different local classification patterns. Data perturbation allocates additive noise to original attributes of data to disguise the original values, however having destroyed local patterns too. Local regions have different resilience to noise, thus we should perturb the regions differentially. For highly resilient regions we introduce more perturbation and less to poorly resilient ones to preserve local patterns. Region based analysis effectively applies to time series data because in the high dimensional space, class distribution on one or few dimensions cannot decide overall data distribution, whereas regions reflect overall class distribution. We have formally defined and discovered one kind of regions which have important influence on classification distribution, which we call discord region. We also propose a region based data perturbation method which preserves patterns underlying in discord regions, thus decreasing the influence brought by noise. The experiments on real and synthetic data show that our method can obviously increase data utility while preserving privacy.

There are some other related works. [1] discusses privacy preserving publishing time series in distributed environment, which succeeded to avoid sites' mutual privacy infringement through only transferring aggregate statistics necessary to posteriori calculation. [3] presents some results about time series protection and re-identification. K-anonymity [7] is extended in [4] to time series data, but for the consideration of data utility, the algorithm finally relaxes the rigorous constraint of k-anonymity. [2] tries to separate original streams from aggregate series using blind source separation technique, then match the data provider through frequency. [4] discusses the evaluation model for time series privacy. The algorithms in [5] distribute noise along the principle component of time series to be published to protect correlation. [6] discusses on protecting the original values of single time series, through adding compressible noise to preserve the trend along time dimension. The detection of discord time series is discussed in [8].

2 Time Series Data Perturbation

We denote a time series T as an ordered set $\{x_1, x_2, \ldots, x_m\}$, x_i is the sequence value at time point t_i, t_{i+1} is the next time point close to t_i, and m is the length of T. A time series database containing n sequences could be viewed as an $n*m$ matrix. The *i-th* row corresponds to time series T_i, the *j-th* column is denoted as $\{T_{ij}\}$ where $1\leq i\leq n$, $1\leq j\leq m$. We extend data perturbation scheme in this paper, which modifies the original matrix and publishes the altered version $TD'_{n\times m}$. As for an original value T_{ij}, we generate a random number p_{ij}, the process could be denoted as $T'_{ij} \leftarrow T_{ij} + p_{ij}$. We perturb data in a column by column manner, and control the degree using the discrepancy (D)

metrics [5]. Discrepancy measures the distance from original values to the published version by calculating Frobenius norm, $D(TD'-TD)=\frac{1}{m}\|TD'-TD\|_F^2 \le \sigma^2$, where σ^2 denotes the quantity of perturbation.

The larger the discrepancy value, the more uncertainty the perturbation introduces, and the higher privacy degree achieved by published data. The existing works [12, 13] show that high perturbation leads to severe information loss, thus we set the higher bound of discrepancy to σ^2. For the *j-th* column $\{T_{ij}\}$, $1\le i \le n$, we generate a perturbation sequence $\{p_{ij}\}$, $1\le i \le n$, where Gaussian random number p_{ij} is of zero mean and standard deviation σ_i. Existing Gaussian white noise addition (*i.i.d perturbation*) distributes the same amount of perturbation to every single attribute and sets $\sigma_1=\sigma_2=\ldots=\sigma_n=\frac{\sigma}{\sqrt{n}}$. Based on existing perturbation methods, we model classifiers (1NN classification model) on 20 UCR time series datasets (http://www.cs.ucr.edu/~eamonn/time_series_data/), and classification error rates are shown in Fig. 1. We observe from Fig.1 that the existing methods seriously deteriorate the classification utility of published data and error rates even increase twice than original data on 9 datasets.

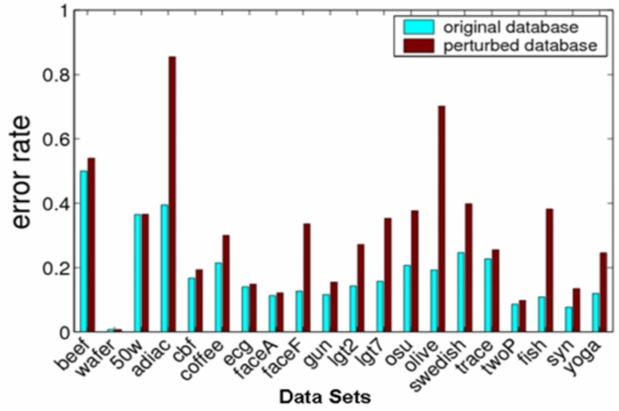


Fig. 1. Deterioration of Publishing Data Utility

We differentially perturb original streams according to the local region where they are, rather than uniformly as *i.i.d perturbation* does, in the purpose of preserving local patterns in regions. We distribute a weight factor w_i for average perturbation degree σ_i ($1\le i\le$n), which is denoted as $\sigma_i \leftarrow w_i * \frac{\sigma}{\sqrt{n}}$ and $\sum_{i=1}^{n} w_i^2 = n$.

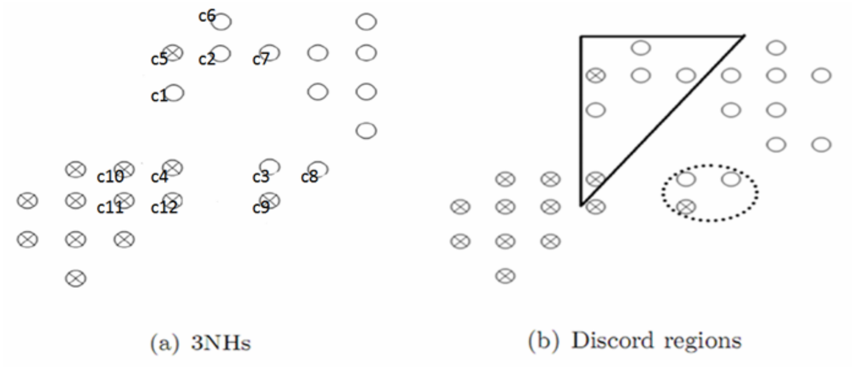
3 Discord Region

Definition 1. Given time series dataset *TD*, $\forall T_i \in TD$, the *k nearest neighborHood* of T_i is defined as KNH(T_i) = { T_i }∪{KNN(T_i)}, where KNN(T_i) means the *k* nearest neighbors of T_i in time series dataset *TD*.

Definition 2. Given a time series T_i in dataset *TD*, if KNH(T_i) contains data points which belong to more than one classes, and we call KNH(T_i) is a discord KNH. Otherwise, we say KNH(T_i) is an accord KNH

Fig. 2 shows the class distribution of a two-dimensional binary class data set, where *k* equals 3 and each point such as c1 and c2 etc means a time series. As for Fig.2 (a), 3NH(c1) = {c1, c2, c4, c5}, 3NH(c2) = {c2, c5, c6, c7}. In particular 3NH(c1) contains data points c1 and c2 that belong to one class, and c4 and c5 are of the other class. 3NH(c4)={c4, c10, c11, c12}, all members belong to the same class. Then 3NH(c1) is a discord KNH and 3NH(c4) is an accord KNH.

Scattered KNHs do not help much in capturing structures and patterns for modeling classification. Because a single accord neighborhood might be embedded into the distribution of the opposite class, which is being viewed as an outlier scenario. On the other hand, continuous accord KNHs or discord KNHs form regions which are of distinct importance for classifying purpose. For example, when an unlabeled data lies within a region formed by accord neighborhoods, we can believe it is of the same class as the neighborhood in high confidence, however, in low confidence to believe in the recognized label when it is in a region formed by discord neighborhoods.



(a) 3NHs

(b) Discord regions

Fig. 2. Formation of Discord Regions

[14] discovered that class overlap region has an important impact on the precision of a classifier. Xi et al. [11] continued to take advantage of the properties of class overlap region and reduce numerosity which results in increased precision. Existing works reveal that class overlap region contains local patterns essential to modeling classification, thus we introduce the concept of discord region to represent areas among classes. For example, in Fig.2(a), both 3NH(c1) and 3NH(c2) are discord, and c2 is one neighbor of c1, we merge all the neighborhoods like these two which result in some regions formed by connected neighborhoods. We call a region like this a *discord region* (DR). Data points not in any discord region form an *accord region*

(AR). In Fig.2 (b) the accord region includes data points except for those in the triangular area and the elliptical area. The 3NH of point c4 is accord, thus cannot be combined with any other discord region. The interesting case happens when considering the neighborhoods of c1 and c3. The reason for not merging those two neighborhoods is because the accord neighborhood of c4 is between them.

4 The Algorithm on Finding Discord Regions

According to the definition of discord region, we design an algorithm to find discord regions in attribute space, which is outlined in Fig.3. The algorithm iteratively proceeds to extend the *k* nearest neighborhood of a data point. If the neighborhood is accord the algorithm continues to process the next series. The value of *k* decides the size of a neighborhood, and influences the formation of final regions. Our experiments show that the size of a discord region grows up with *k*. A proper value of *k* helps find good discord regions, which reflect the real atypical regions among classes.

```
Algorithm 1. FindDiscordRegion

Inputs: original time series data set $TD$ and $k$

Output: all the discord regions

1.  Region = ∅;
2.  Initialize each time series as unprocessed;
3.  For i = 1 to $|TD|$
4.    $R_i = \emptyset$;
5.    Calculate KNH($T_i$) // return the KNH of $T_i$;
6.    If KNH($T_i$) is accord, continue;
7.    For each $p \in$ KNH($T_i$),
8.      If $p$ is not in any existed discord regions, $R_i = R_i \cup \{p\}$;
9.    End For
10.   Set Candidate= KNN ($T_i$); //knn returns the $k$ nearest neighbors of $T_i$
11.   For each $c \in$ Candidate
12.     IF KNH($c$) is accord, continue;
13.     For each $q \in$ KNH($c$)
14.       If $q$ is unprocessed, Candidate= Candidate $\cup \{q\}$, $R_i = R_i \cup \{q\}$;
15.       If $q$ belongs to the accord region, $R_i = R_i \cup \{q\}$;
16.     End For
17.   End For
18.   If $|R_i| >= k$ Region = Region $\cup R_i$;
19. End For
```

Fig. 3. Algorithm Description of Finding Discord Regions

In Fig.3, steps 7 to 9 add the discord neighborhood of a processed data point into the current discord region, step 11 to 17 iteratively check the KNHs of a point's neighbors and expand them if met with the merge condition. We see from step 12 that, if a neighbor's KNH is discord it can be combined with the current region.

Step 8 and step 15 illustrate that a point in the accord region can be combined into a discord region, thus our algorithm favors discord regions, but a discord region is not allowed to overlap with previously formed discord regions according to step 7 and step 14.

A discord region might overlap with another one, for example, in Fig.2, 3NH(c1) overlaps with 3NH(c3) on data point c4. Since we implement the algorithm in a non-overlap manner, if one point T_i belongs to several KNH(T_j) ($1 \le j \le n$), T_i is randomly assigned to a KNH. In Fig.2(b) if c4 is assigned to 3NH(c1), 3NH(c3) is shrunk to the elliptical area. We decide if a shrunk area has less than least "*k*" points it is no more a discord region, and if a shrunk area becomes accord it should be added into the accord region. Then the elliptical area in Fig.2 (b) is still counted as a discord region.

The subprogram KNN at step 10 involves searching whole attribute space, thus producing a run time complexity O(n) where n is the number of data points in the dataset. Spatial index (e.g., R*-tree) can help reduce the searching time complexity to O(log(n)). Since every data point needs only one expansion, the overall time complexity of this algorithm is O(n^2) or O(n*log(n)).

5 Discord Region Based Data Perturbation Algorithms

We design two data perturbation schemes based on discord regions, that is, even discord region based perturbation (even DRP) and recursive DRP. Even DRP distributes total perturbation ^{2} evenly to discord regions rather than the accord region. The description of even DRP is outlined in Fig.4.

```
Algorithm 2: even_DRP
Inputs: time series database TD,  k,  and perturbation (i.e. $\sigma^2$)
Output: published TD'
1. Region = FindDiscordRegion (TD, k); AR=TD-Region;
2. Set the average perturbation $\sigma_i = \sqrt{\frac{perturbation}{|AR|}}$ ;
3. For i=1 to n  // n time series in TD
4.         TD'[i] ← TD[i] + GussianRandom(0, $\sigma_i$);
```

Fig. 4. Algorithm Description of even DRP

Step 1 retrieves all discord regions in dataset *TD*. Step 2 calculates the average perturbation distributed to each series in the accord region. In step 2, |accords| denotes the size (i.e. the number of points) of accord region. Step 4 generates random numbers to be added to original series. Even DRP allocates all the perturbation to the accord region, in the case of a small even negligible accord region, that would cause severe data distortion in the accord region. In purpose of guaranteeing data utility we set the higher bound of individual perturbation to every single series as . We allocate perturbation to the accord region but consider the upper bound of perturbation. If individual perturbation exceeds the bound we distribute perturbation as much as to each series. We have to allocate left perturbation to discord regions, where we decrease the value of *k* and discover tighter discord regions in former discord regions, then allocate left perturbation to the newly discovered accord region. This is a recursive process which will terminate when total perturbation is not more than zero or when *k* decreases to 1. Detailed steps are outlined in Fig.5.

```
Algorithm 3: recursive_DRP
Input: time series database TD, k, perturbation (i.e. $\sigma^2$)
Output: published TD'
1. Region= FindDiscordRegion (TD, k);
2. AR= TD- Region; //the accord region
4. Perturbation_left = perturbation - |AR|*$\beta^2$;
5. IF perturbation_left $\leq 0$
7.     Assign $\frac{pertubation}{\sqrt{|AR|}}$ to each time series in AR and return;
9. Else
10.        Assign $\beta$ to each time series in AR;
11.        IF (k$\leq$1)
12.                Calculate $\sigma_{left} = \frac{pertubation\_left}{\sqrt{|\mathrm{Re}gion|}}$ ;
13.                Assign $\sigma_{left}$ to each sequence in discords and return;
14.    End if
15.    recursive_DRP (Region, k-1, perturbation_ left);
16. End if
```

Fig. 5. Algorithm Description of recursive DRP

Especially, when the condition is true in step 5, recursive DRP only iterates once and generalizes to even DRP, we can see that recursive DRP is the generalized version of even DRP. When the accord region is too large or when the perturbation is too small, the accord region will carry all the perturbation and the algorithm iterates only once.

6 Experimental Results

In this section, we report the experimental results on our algorithms. We firstly compared the effectiveness of proposed perturbation methods in the aspect of improving data utility with the existing Gaussian white noise addition (i.i.d_p) method. In addition, we also report the running time of recursive DRP algorithm.

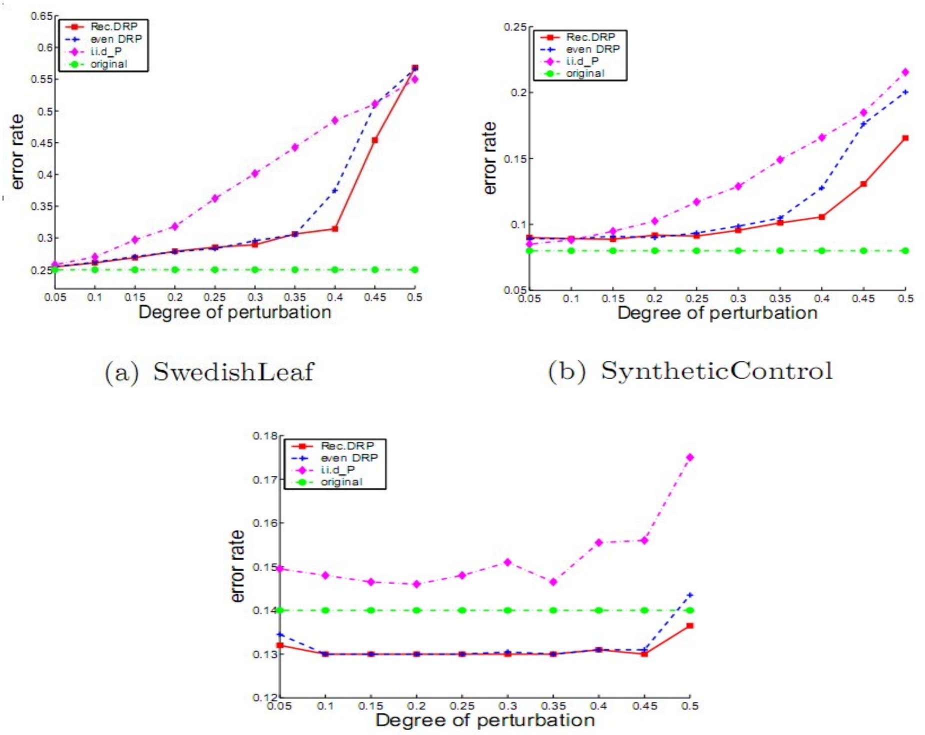
All algorithms are implemented using MATLAB 6.5. The experimental environment is a PC with Pentium Dual-Core 2.6GHzCPU, 2GB memory and Windows OS. The test dataset is the UCR dataset (http://www.cs.ucr.edu/~eamonn/time_series_data/). We have normalized every time series in a data set to zero mean and unit variance. To get convincing results, we vary the discrepancy value from 5% to 40% relative to the original energy of time series. For each experimental run, we test its performance along this range by steps of 5%. We perform each experimental run 20 times, each time with an independent perturbation, and the average values of comparison results are reported.

Table 2. Recursive DRP Publishing Data Quality

Data Set	Size	Length	Recursive DRP vs. i.i.d_p
SyntheticControl	300	60	0.0955, 0.1288
GunPoint	50	150	0.052, 0.164
CBF	30	128	0.1317, 0.2017
FaceAll	560	131	0.1286, 0.1221
OSULeaf	200	427	0.3875, 0.3748
SwedishLeaf	500	128	0.2893, 0.4013
150Words	450	270	0.3738, 0.3683
Trace	100	275	0.1715, 0.2525
TwoPatterns	1000	128	0.1024, 0.0981
wafer	1000	152	0.007, 0067
FaceFour	24	350	0.3333, 0.3438
Lighting2	60	637	0.2825, 0.2742
Lighting7	70	319	0.3464, 0.3621
ECG	100	96	0.13, 0.151
Adiac	390	176	0.4764, 0.864
Yoga	300	426	0.2378, 0.2467
Fish	175	463	0.258, 0.3837
Beef	30	470	0.4667, 0.5444
Coffee	28	286	0.2464, 0.3089
OliveOil	30	570	0.2583, 0.72

6.1 Effectiveness of DRP

There are two parameters k and σ^2 in our algorithms. We test k values from 2 to 6 to find the most proper k with lowest classification error rate on each dataset. Table 2 shows the comparison results between recursive DRP and those of baseline Gaussian white noise addition (i.i.d_p), where σ^2 is set as 30%. In our experiments, the parameter in recursive DRP is set as $2*\frac{\sigma}{\sqrt{n}}$. In the experiments, we find out that discord regions commonly exist in time series datasets, although with different size of discord regions. The only exception is the wafer dataset, which has two classes. Our further investigation reveals that there is a class imbalance as high as 10:1 in wafer. That's the reason why trivial discord region is discovered, thus it is insensitive to discord region based perturbation. For the other left 19 datasets, recursive DRP preserves more data utility than i.i.d perturbation on most datasets and just loses in 3 cases (i.e. OSULeaf, TwoPatterns, Lighting2).



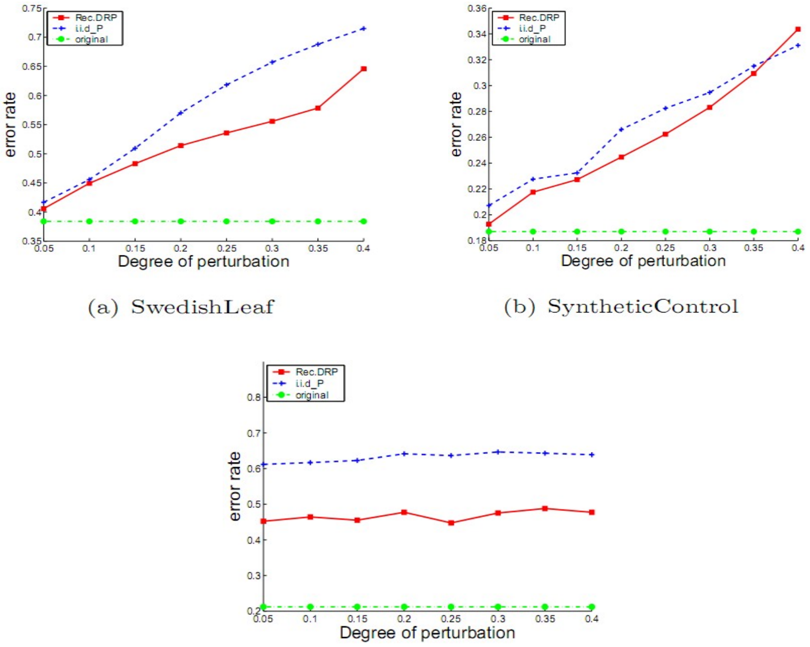
(a) SwedishLeaf

(b) SyntheticControl

(c) ECG

Fig. 6. Classification Error Rates

Fig. 6 shows the publishing error rates of three data perturbation schemes. We test on 1NN classification model. We choose three datasets SwedishLeaf, ECG and SyntheticControl, which are representative in aspect of training set size (i.e. 500, 100, 300 respectively), number of classes (i.e. 15, 2, 6 respectively) and the former two are real datasets while the third one is a synthetic dataset. We observe that the accuracy of classifier decreases with the growth of perturbation. That's because much perturbation destroys structures in data necessary for classification. Both DRP methods outperform i.i.d perturbation. It makes sense to say the discord region analysis helps preserve data utility. In order to compare recursive DRP and even DRP, we extend the perturbation range to 50%. When the amount of perturbation is small, recursive DRP performs the same as even DRP. But recursive DRP can achieve lower error rate than even DRP when perturbation increases.



(a) SwedishLeaf

(b) SyntheticControl

(c) ECG

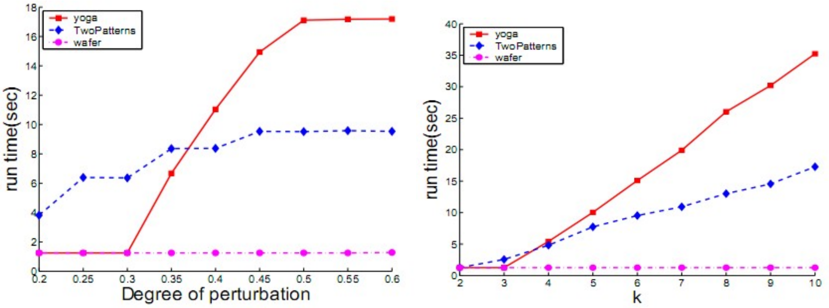
Fig. 7. Classification Error Rates on C4.5

The original line represents the classification error rates of unperturbed data. We notice in Fig.6 that the gap between original data error rate and other publication method error rate increases with the growth of perturbation, but recursive DRP achieves acceptable data quality when the perturbation is not that much(e.g., below 40%). And as we find in Fig.6(c), the recursive DRP published data even outperforms original data. It demonstrates that perturbation not necessarily decreases data utility. In this case perturbed data exhibits new structures which help model classification. This similar phenomenon is also found in [15].

We also test on other classifiers, for example C4.5. The experiment results are shown in Fig.7, where C4.5 is the implementation in WEKA toolkit (http://www.cs.waikato.ac.nz/ml/weka/) without any parameter optimization. From Fig.7 we can also observe the improvement of published data's classification utility with our recursive DRP compared to i.i.d_p .

6.2 Efficiency of Recursive DRP

We also test the efficiency of recursive DRP algorithm. In the experiment, we chose three larger datasets, Wafer, Yoga and TwoPatterns, all containing 1000 time series. We focus on two aspects of recursive DRP algorithm, that is, algorithm efficiency versus the quantity of perturbation and algorithm efficiency versus *k* values.



(a) Runtime vs. perturbation

(b) Runtime vs. k

Fig. 8. Efficiency of Recursive DRP

Fig.8 (a) shows the relation between the former pair. The algorithm does not work for Wafer dataset, for it has little discord region which has been discussed in section 6.1. As for the left two datasets, both of them display a trend of three stages. Stage one where the perturbation is small, the algorithm needs to search for the discord region only once. During stage two, algorithm run time increases with increased perturbation. Stage three, a large amount of perturbation brings the algorithm to reach its full capability (i.e., when *k* is decreased to 1). Fig.8 (b) shows that the algorithm run time increases with *k* linearly. That is because a larger *k* produces a larger discord region where the program searches.

7 Conclusions

Through analyzing different impact of local regions on overall classification performance, we have found that discord region influences published data's classification utility most. Discord regions are poorly resilient to additive noise introduced by perturbation, leading to destruction of local patterns in them. Through perturbing discord regions and accord regions differentially, distributing noise in consistent with class distribution, proposed region based data perturbation can prominently improve data utility. The experimental results on real datasets and synthetic datasets show the effectiveness of our presented methods.

Acknowledgments. This paper is supported by National Natural Science Foundation of China (NSFC) under grant No.60703111, 61070005, 60773198, and 61033010.

References

1. Yabo, X., Ke, W., Ada, W.C.F., Rong, S., Jian, P.: Privacy-Preserving Data Stream Classification. In: Charu, A., Philip, S.Y. (eds.) Privacy-Preserving Data Mining Models and Algorithms, pp. 487–510. Springer, Heidelberg (2008)
2. Ye, Z., Yongjian, F., Huirong, F.: On Privacy in Time Series Data Mining. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 479–493. Springer, Heidelberg (2008)
3. Josenildo, C.S., Matthias, K.: Privacy-preserving discovery of frequent patterns in time series. In: Perner, P. (ed.) ICDM 2007. LNCS (LNAI), vol. 4597, pp. 318–328. Springer, Heidelberg (2007)
4. Nin, J., Torra, V.: Towards the evaluation of time series protection methods. Information Science 179, 1663–1677 (2009)
5. Feifei, L., Sun, J., Papadimitriou, S., Mihaila, G., Stanoi, I.: Hiding in the crowd: Privacy preservation on evolving streams through correlation tracking. In: 23rd International Conference on Data Engineering, pp. 686–695. IEEE, Los Alamitos (2007)
6. Papadimitriou, S., Feifei, L., Kollios, G., Philip, S.Y.: Time series compressibility and privacy. In: 33rd International Conference on Very Large Data Bases, pp. 459–470. ACM, New York (2007)
7. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10, 571–588 (2002)
8. Yubao, L., Xiuwei, C., Fei, W., Jian, Y.: Efficient Detection of Discords for Time Series Stream. In: Li, Q., Feng, L., Pei, J., Wang, S.X., Zhou, X., Zhu, Q.-M. (eds.) APWeb/WAIM 2009. LNCS, vol. 5446, pp. 629–634. Springer, Heidelberg (2009)
9. Lindell, Y., Pinkas, B.: Privacy Preserving Data Mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000)
10. Agrawal, R., Aggarwal, C.C.: Privacy preserving data mining. In: ACM SIGMOD International Conference on Management of Data, pp. 439–450. ACM, New York (2000)
11. Xi, X., Keogh, E., Shelton, C., Wei, L., Ratanamahatana, C.: Fast time series classification using numerosity reduction. In: International Conference on Machine Learning, pp. 1033–1040. ACM, New York (2006)
12. Kifer, D., Gehrke, J.: Injecting utility into anonymized datasets. In: ACM SIGMOD International Conference on Management of Data, pp. 217–228. ACM, New York (2006)
13. Evfimievski, A., Gehrke, J., Srikant, R.: Limiting privacy breaches in privacy preserving data mining. In: ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 211–222. ACM, New York (2003)
14. Abe, S., Lan, M.S.: A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. IEEE Trans. Fuzzy Systems 3, 18–28 (1995)
15. Benjamin, C.M.F., Ke, W., Philip, S.Y.: Top-Down Specialization for Information and Privacy Preservation. In: 21st International Conference on Data Engineering, pp. 205–216. IEEE Computer Society, Los Alamitos (2005)
16. Ivan, D., Yuval, I.: Scalable Secure Multiparty Computation. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 501–520. Springer, Heidelberg (2006)

Deep Web Sources Classifier Based on DSOM-EACO Clustering Model

Yong Feng, Xianyong Chen, and Zhen Chen

College of Computer Science, Chongqing University, Chongqing 400030, China
fengyong@cqu.edu.cn

Abstract. There are many deep web sources providing the services, but we may not be aware of their existence, and not know which sources can satisfy our demands. So that there is a great significant to build a system to integrate the myriad deep web sources in the Internet, and the classification of deep web sources is very important in the integration. In this paper, a clustering model based on dynamic self-organizing maps (DSOM) and enhanced ant colony optimization (EACO) is systematically proposed for deep web sources classification. The basic idea of the model is to produce the cluster by DSOM and EACO. With the classified data instances, the classifier can be established. And then the classifier can be used in real deep web sources classification, and it is observed that the proposed approach gives better performance over some traditional approaches for deep web sources classification problems.

Keywords: deep web sources classification, classifier, dynamic self-organizing maps, ant colony optimization, clustering.

1 Introduction

Information of deep web sources can only be accessed through their query interfaces [1]. A study [2] in April 2004 estimated 450,000 such Web data-sources. With myriad data-sources on the web, the users may be puzzled to find the sources satisfying their information requirements. So that there is a great significant to build a system to integrate the myriad deep web sources in the Internet. Such integration faces great challenges [3]: First, the scale of deep web pages is between 400 to 500 times larger than the static pages, and it is still proliferating rapidly. Second, the query conditions are varying, and there are no rules for building the unified query interfaces. Facing the challenge, a compromising strategy is inevitable: Using domain-based integration [4]. Query interfaces of deep web sources in the same domain can share a lot in common. So the classification of deep web sources is very important in the integration. Furthermore, all deep web sources are autonomous systems. With the birth and death, the number of deep web sources in the integration is changing everyday. Last, the classification also provides a way to organize the myriad deep web sources, just like the category directory in yahoo.

In this paper we proposed a novel algorithm, it firstly use dynamic self-organizing maps (DSOM) and enhanced ant colony optimization (EACO) to produce the clusters,

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 238–245, 2010.

with the classified data instances, the classifier can be established. And then the classifier can be used in real deep web sources classification. This algorithm takes full advantage of the class label information and starting with a small SOM neural network; hence it is likely to be more efficient and is except to generalize well.

The rest of the paper is organized as follows. Section 2 presents our clustering model based on DSOM and EACO. Establishment of the classifier is reported in Section 3. Experiment results are presented in Section 4 and some conclusions are also provided towards the end.

2 Clustering Model Based on DSOM and EACO

Our clustering model based on DSOM and EACO has two main steps:

Step1: DSOM network growth
Step2: EACO clustering
Fig. 1 shows the principle of DSOM and EACO clustering.

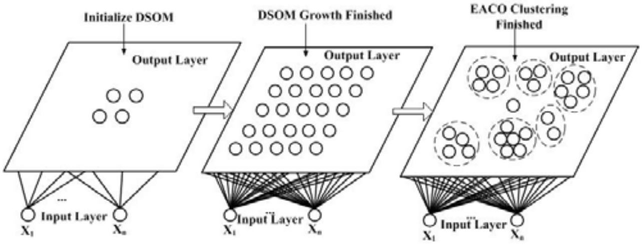


Fig. 1. Principle of DSOM and EACO clustering

2.1 DSOM Network Growth Algorithm

The DSOM determines the shapes as well as the size of network during the training of the network, which is a good solution to problems of the SOM [5].

The DSOM is an unsupervised neural network, which is initialized with four nodes and grows nodes to represent the input data [6]. During the node growth, the weight values of the nodes are self-organized according to a similar method as the SOM.

Definition 1. The winner neural *b* is defined as:

$$\left\| v - w_b \right\| \le \left\| v - w_q \right\|, \forall q \in N \,. \tag{1}$$

Where *v*, *w* are the input and weight vectors, *q* is the position vector for nodes, *N* is the set of natural numbers, and ||·|| is Euclidean distance.

Definition 2. *E* is the error distance between *b* and *v*, *E* is defined as:

$$E = \sum_{j=1}^{d} (v_j - w_{b,j})^2 \cdot \tag{2}$$

Where *d* is dimension of the vector *v*.

Definition 3. *GT* is the growth threshold of DSOM. For the node *i* to grow a new node, it is required that $E \geq GT$. It can be deduced from (2), $0 \leq v_j, w_{b,j} \leq 1$ and $0 \leq E \leq d$ that $0 \leq GT < d$. Therefore, it becomes necessary to identify a different *GT* value for data sets with different dimensionality. This becomes a difficult task. The spread factor *SF* can be used to control and calculate the *GT* for DSOM. The *GT* can be defined as:

$$GT = d \times f(SF)\,. \tag{3}$$

Where $SF \in R, 0 \leq SF \leq 1$, and *f* (*SF*) is a function of *SF*.

Definition 4. *f* (*SF*) is defined as:

$$f(SF) = Sigmoid_{n(t)}(1-SF) = \frac{1-e^{-n(t)*(1-SF)}}{1+e^{-n(t)*(1-SF)}}\,. \tag{4}$$

Where *n*(*t*) is the total number of nodes at t^{th} iteration. *f* (*SF*) gradually saturated with the increase of network training that *GT* is stable, and DSOM algorithm is reaching convergence.

The **DSOM algorithm** is described as follows:

1) Initialization phase.
   a) $V' = \{v'_1, v'_2, \ldots\ldots, v'_n\}$ is the input vector sets, and $v'_i = \{x'_{i1}, x'_{i2}, \ldots\ldots, x'_{id}\}$, where $1 \leq i \leq n$ and *d* is dimension of the vector v'_i. Standardizing *V'* to *V*, if v'_i is a continuous variable, the method can be described as fellows:

$$m_j = \frac{1}{n}\sum_{i=1}^{n} x'_{ij}\,. \tag{5}$$

$$\delta_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x'_{ij} - m_j)}\,. \tag{6}$$

$$x_{ij} = (x'_{ij} - m_j)/\delta_j\,. \tag{7}$$

   Where m_j denotes the average feature instance of v'_i, δ_j denotes the standard deviation feature instance of v'_i, and x_{ij} denotes the feature of the normalized instance. Here, all features are equally weighted in order to enhance the algorithm's generality. If v'_i is a discrete variable, we may code v'_i according to binary code, and then processing like formula (5), (6) and (7).

   b) Initialize the weight vectors of the starting nodes with random numbers between 0 and 1.
   c) Calculate *GT* for the given data set.

2) Growing phase.

   a) Present input to the network.
   b) Find the winner neural *b* such that (1).
   c) Calculate the error distance E_i between *b* and v_i according to (2). If $E_i \geq GT$, then turn d) to grow nodes, or turn e) to adjust the weight value of neighborhood.

d) Grow the new node m, and set $w_m = v_i$.

e) The weight adjustment of the neighborhood can be described by:

$$w_j(k+1)=\begin{cases} w_j(k),\ j\notin N_{k+1} \\ w_j(k)+LR(k)\times(x_k-w_j(k)),\ j\in N_{k+1} \end{cases}. \tag{8}$$

When $k\to\infty$ ($k\in N$), the learning rate $LR(k)\to 0$. $w_j(k)$, $w_j(k+1)$ are the weight vectors of the node j before and after the adjustment, and N_{k+1} is the neighborhood of the wining neuron at $(k+1)^{th}$ iteration.

f) Adjustment the learning rate according to the following formula:

$$LR(t+1) = LR(t)\times\beta . \tag{9}$$

Where β is the learning rate reduction and is implemented as a constant value $0<\beta<1$.

g) Repeat steps b)—f) until all inputs have been presented and node growth is reduced to a minimum level.

2.2 EACO Clustering Algorithm

EACO clustering has two main phases. First, each ant chooses the object at random, and picks up or moves or drops down the object according to picking-up or dropping probability in the output layer of DSOM. Second, clusters are collected from the output layer of DSOM.

Definition 5. Swarm similarity is the integrated similarity of a data object with other data objects within its neighborhood.

A basic formula of measuring the swarm similarity is showed as formula (10).

$$f(o_i)=\sum_{o_j\in Niegh(r)}[1-\frac{d(o_i,o_j)}{\alpha}]. \tag{10}$$

Where $Neigh(r)$ denotes the local region, it is usually a rounded area with a radius r, $d(o_i, o_j)$ denotes the distance of data object o_i with o_j in the space of attributes. The α is defined as swarm similarity coefficient. If α is too large, the algorithm converges quickly, whereas if α is too small, the algorithm converges slowly.

Definition 6. Probability conversion function is a function of $f(o_i)$ that converts the swarm similarity of a data object into picking-up or dropping probability for a ant.

The general idea of configuring probability conversion function is that the more big swarm similarity is, the more small picking-up probability is and the more big dropping probability is, and vice versa. According to the idea, the picking-up probability and the dropping probability are given by:

$$P_p=\frac{1}{2}-\frac{1}{\pi}\arctan\left[\frac{f(o_i)}{k}\right]. \tag{11}$$

$$P_d=\frac{1}{2}+\frac{1}{\pi}\arctan\left[\frac{f(o_i)}{k}\right]. \tag{12}$$

Where k is a plus constant and can speed up the algorithm convergence if it is decreased. Instead of using the linear segmentation function of CSI model [7] and LF model [8], here we propose to use a nonlinear function and can help to solve linearly inseparable problems [9].

The **EACO clustering** algorithm description can be found in our previous work [9].

3 Establishing Deep Web Sources Classifier

In order to obtain precision deep web sources classification model, we need to establish DSOM and EACO-based deep web sources classifier. It can balance the contradiction between the clustering results and the apriori knowledge to some extent, and the accuracy and the scalability of the classifier are also improved.

Accuracy of DSOM and EACO clustering is depend on SF and α to some extent. In order to obtain precision classification, we need to determine stable SF and α. DSOM and EACO-based classifier can help us to balance the contradiction between the clustering results and the apriori knowledge, and stable value of SF and α can also be determined by it.

The DSOM and EACO-based detection classifier is described as fellows:

1) Adjusting *SF* and α, applying DSOM and EACO-based clustering model to training data set, produce clusters $\{C_1, C_2, \ldots\ldots, C_S\}$, where S is the number of the clusters.
2) Computing the centres of $\{C_1, C_2, \ldots\ldots, C_S\}$, get the cluster centres model.
3) Selecting the new training data sets, repeat step 1)-3).
4) Computing the new cluster centres, get average value of each type cluster centre.
5) Updating the cluster centres model using average value of each type cluster centre.
6) Deciding the best varying scale of *SF* and α.
7) Producing the classifier according to the new cluster centres model and the best varying scale of *SF* and α.
8) $\{O_1, O_2, \ldots\ldots, O_S\}$ is the new cluster centres model, x is a data package, computing Euclidean distance $d(O_i, x)$ between x and $\{O_1, O_2, \ldots\ldots, O_S\}$, x belong to the cluster where $d(O_i, x)$ is minimum.

4 Experiment

We conducted experiments to test the performance of our method on datasets (Airfrares.xml, Movies.xml, MusicRecords.xml, Books.xml) [10], which relate to four different domains of deep web sources, they are Airfares, Movies, Musics, and Books, and there are 260 different query interface schemas involved. In this paper, we organized the types of attributes following the reference [11]. We select all the distinct attributes from the training samples. According the deep web model [4], using the probability $P(f)$ <=5%, we filter the concepts as the features.

$$P(f) = N_f / N . \tag{13}$$

where f is feature. N_f is the number of deep web sources that f appeared in the training samples, and N is the total.

In the experiment, we compare our approach with the traditional OLS RBF classifier, SVM (Support Vector Machines) classifier and BP neural network classifier on the same datasets. To evaluate the algorithm we are interested in 3 major indicators of performance: *Precision*, *Recall* and *F*. Supposed that there are *n* deep web sources, $dw_1, dw_2, ..., dw_n$, with domain identified $l_1, l_2, ..., l_m$, let *TP* denote the number of deep web sources that are correctly classified, *FN* denotes number of deep Web sources that are falsely excluded in l_i, FP denotes the number of deep web sources belong to l_j but falsely classified into l_i.

$$Precision = \frac{TP}{TP + FP}, Recall = \frac{TP}{TP + FN}, F = \frac{2Precision * Recall}{Precision + Recall} . \tag{14}$$

The BP algorithm were trained with n_h hidden neurons, n_h was varied from 10 to 50, and the maximum training cycles is 5000. The width parameter of radial function is the most important to the OLS RBF classifier; it varied from 1 and 4 in this paper and the maximum RBFs is 100. In SVM classifier experiment, the kernel function is $K(u,v) = (ugv+1)^d$, where *d* is the order of the kernel function, the value of *d* is bigger and the performance of SVM classifier is higher, so we adjust *d* from 1 to 5. For our DSOM-EACO-based classifier, we adjust *SF* from 0.5 to 0.9 (interval is 0.1) and α from 0.85 to 0.2 (interval is 0.05), *ant-number* = 10, *r* = 12, *k* = 10, *n* = 60×1000.We can decide the best value of *SF* is 0.8 (it could balance between accuracy and efficiency of DSOM and EACO clustering) by the training experiment.

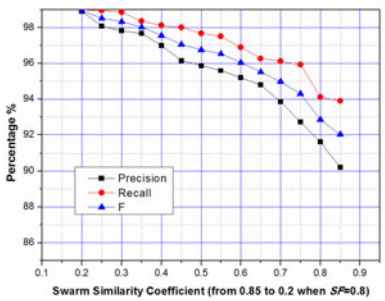


Fig. 2. Experimental results for DSOM-EACO-based classifier

The results from the Fig. 2 shows that, when SF = 0.8, our classifier keeps a good performance that the average *Precision* is higher than 95.21%, the average *Recall* is higher than 96.77% and the average *F* is higher than 96%.

According to testing results (Fig. 2-5), we found that, the performance of DSOM-EACO-based classifier is increased obviously than the traditional RBF network

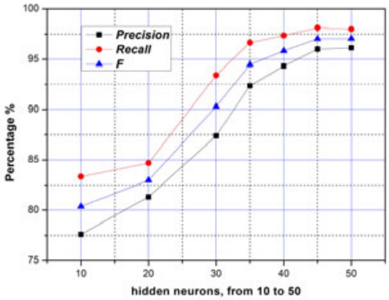


Fig. 3. Experimental results for BP classifier

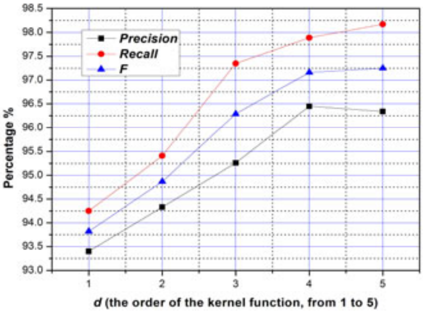


Fig. 4. Experimental results for SVM classifier

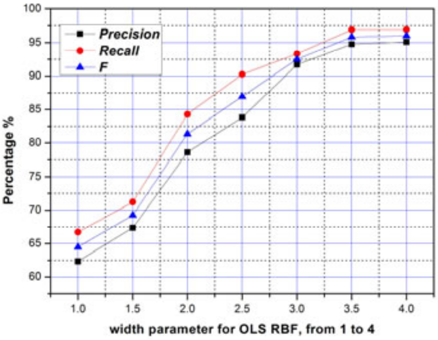


Fig. 5. Experimental results for OLS RBF classifier

classifier and BP. The average *Precision*, *Recall* and *F* are also higher than BP (89.31%, 93.08% and 91.15%), SVM (95.16%, 96.61% and 95.88%), and OLS RBF (81.96%, 85.67% and 83.77%).

5 Conclusions

This paper proposes a deep web sources classifier based on dynamic self-organizing maps (DSOM) and enhanced ant colony optimization (EACO) clustering model. It overcomes the contradiction between the clustering results and the apriori knowledge to some extent, and the accuracy and the efficiency are improved. Experimental results show that the average *Precision*, *Recall* and *F* of our approach maintained a higher performance than BP, SVM and OLS RBF on the same deep web source datasets.

Acknowledgement. This work is supported by the Fundamental Research Funds for the Central Universities (Grant No. CDJZR10180006), the Third Stage Building of “211 Project” (Grant No. S-10218), Natural Science Foundation Project of CQ CSTC (Grant No. 2008BB2183), National Social Sciences Fund Special Projects in Educational Science (Grant No. ACA07004-08) and Key Projects in the National Science & Technology Pillar Program (Grant No. 2008BAH37B04).

References

1. He, B., Patel, Z.M.Z.: Accessing the Deep Web. Communications of the ACM 50, 95–101 (2007)
2. Ghanem, T.M., Aref, W.G.: Databases Deepen the Web. Computer 37, 116–117 (2004)
3. Liu, W., Meng, X.F., Meng, W.Y.: A Survey of Deep Web Data Integration. Chinese Journal of Computers 30, 1475–1489 (2007)
4. Wang, Y., Zuo, W.L., Peng, T., He, F.L.: Domain-Specific Deep Web Sources Discovery. In: Proceeding of Fourth International Conference on Natural Computation, pp. 202–206. IEEE Press, New York (2008)
5. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (1995)
6. Alahakoon, L.D., Halgamuge, S.K., Srinivasan, B.: A Structure Adapting Feature Map for Optimal Cluster Representation. In: Proc. Int. Conf. Neural Information Processing, pp. 809–812. IEEE Press, New York (1998)
7. Wu, B., Shi, Z.Z.: A Clustering Algorithm Based on Swarm Intelligence. In: Proceedings of the 2001 IEEE International Conferences on Info-tech & Info-net, pp. 58–66. IEEE Press, New York (2001)
8. Lumer, E., Faieta, B.: Diversity and Adaptation in Populations of Clustering Ants. In: Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animals to Animats, pp. 499–508. MIT Press, Cambridge (1994)
9. Feng, Y., Wu, Z.F., Zhong, J.: An Enhanced Swarm Intelligence Clustering-based RBF Neural Network Detection Classifier. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 526–533. Springer, Heidelberg (2008)
10. The UIUC web integration repository, `http://metaquerier.cs.uiuc.edu/repository`
11. He, B., Chang, K.C.C.: Automatic Complex Schema Matching Across Web Query Interfaces: A Correlation Mining Approach. ACM Transactions on Database Systems 31, 346–395 (2006)

Kernel Based K-Medoids for Clustering Data with Uncertainty

Baoguo Yang and Yang Zhang*

College of Information Engineering, Northwest A&F University,
Yangling 712100, Shaanxi, China
{ybg0359,zhangyang}@nwsuaf.edu.cn

Abstract. Uncertain data is ubiquitous in real-world applications due to various causes. In recent years, clustering uncertain data has been paid more attention by the research community, and the classical clustering algorithms based on partition, density and hierarchy have been extended to handle the uncertain data. However, these extended algorithms usually work in the input space. In this paper, to well explore the inherent data pattern in the high dimensional feature space, we propose a kernel based K-medoids algorithm for clustering uncertain data. Extensive experiments performed on synthetic and several real datasets demonstrate that our kernel based method has higher clustering accuracy than the state-of-the-art UK-medoids algorithm. Also, it signifies that the uncertain data pattern in the new feature space could be well presented when the kernel function and the K-medoids algorithm are effectively incorporated.

Keywords: Uncertain data, Clustering, Kernel trick, K-medoids.

1 Introduction

The data values are usually assumed as exact or precise in traditional machine leaning algorithms. However, in many real-world applications, data contains inherent uncertainty with different degree when they are generated or collected. These uncertain data are usually represented in terms of an exact value with uncertain region or margins of error over which a probability density function (pdf) is defined.

It is very important to develop effective data mining algorithms to manage uncertain data since data uncertainty is ubiquitous. In this paper, we study the problem of clustering data with multi-dimensional uncertainty. Our main goal is to group a collection of uncertain objects into clusters.

The partition based K-means and K-medoids algorithms are two of the most popular clustering approaches. To deal with uncertain data, both of the two algorithms have been extended to UK-means [1] and UK-medoids [2], respectively. UK-medoids algorithm outperforms significantly than UK-means [2]. However, the existing algorithms are usually work in the input space, for some non-linear separable uncertain data, the data pattern might not be well explored. Motivated by this, we propose to use kernel based method to break through this limitation. Kernel function can be considered as

* Corresponding author.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 246–253, 2010.

a non-linear transformation that map the raw data to a high dimensional feature space where the data are expected to be more separable.

In this paper, we propose a kernel based K-medoids method to cluster uncertain data. We express the distance between uncertain objects in the form of kernels. To our knowledge, this is the first work to use kernel function for the uncertain distance calculation. Experimental results demonstrated that our approach outperforms the existing UK-medoids algorithm in terms of clustering accuracy.

This paper is organized as follows. Section 2 reviews the related work. Section 3 presents the framework for clustering uncertain data in the paper. Section 4 gives the experimental results. In section 5, we conclude this paper and give our future work.

2 Related Work

Clustering Uncertain Data. Recently, the classical K-means algorithm has been extended to cluster uncertain data[1]. The distance between uncertain objects are computed as the ED (expected distance) based on a pdf. It is empirically shown that clustering results are improved if data uncertainty is taken into account during the clustering process [3]. In [3], in order to improve the efficiency of UK-means, the authors proposed some pruning techniques to avoid the redundant EDs computation. However, it depends on specific dataset which cannot achieve high pruning. In [4], Yun and Yang introduce a novel approach, CK-means, to compute the EDs efficiently, which works only for a specific form of distance. And in [2], Gullo *et al.* proposed a K-medoids based clustering algorithm, UK-medoids, to overcome the above weakness, which employs the properly defined distance functions for uncertain objects, and achieves high accuracy.

Besides the partition based uncertain data clustering algorithm, the density based clustering algorithm for uncertain data clustering are also proposed afterwards. Kriegel *et al.* proposed a fuzzy algorithm, *F*DBSCAN [5], which uses fuzzy distance functions to compute core object and reachable probabilities, and the *F*OPTICS [6] algorithm based on fuzzy distance functions applied to manage uncertain data. These algorithms identify regions with high expected density based on the pdfs of the objects. However, a formal and general fuzzy distance function definition has not been provided.

Uncertain Distance Representation. The probabilistic distance functions are used to measure the similarity between uncertain objects, which takes the pdfs defined over the data intervals into consideration. In [1], the expected distance is used by calculating the average of all the possible distances weighted by their probabilities. However, it is very expensive to calculate expected distance due to the pair-wise distance calculation between all pairs of possible locations or samples. To overcome this bottleneck, Ngai *et al.* proposed some pruning techniques to reduce the redundant distance computations [3]. In [7], Xiao and Huang proposed an efficient approach, Approximation by Single Gaussian (ASG), to calculate the expected distance by a function of the mean and variances of samples from uncertain objects. However the above methods for the expected distance calculation between two uncertain objects only worked in their original input space. When the uncertain objects are not linearly separable in the input space, the real distance between uncertain objects cannot be well represented. In this paper, we will express the expected distance in the form of kernel function.

3 Clustering Data with Uncertainty

3.1 Uncertain Data Prototype

Generally, an attribute can be viewed as a dimension, and all attributes of an object will be combined to generate a multidimensional space. Uncertain object can be represented as a set of points while the certain object can be viewed as a single point. Also, an uncertain object can be represented as a region, which can be finite or infinite. Here, we call this region *uncertain region* of an object o_i, denoted as $UR(o_i)$. A pdf (probability density function),f_i, which denotes the probability density of each possible value within $UR(o_i)$, and we can have $\int_{\mathbf{x} \in UR(o_i)} f_i(\mathbf{x}) d\mathbf{x} = 1$ and $\int_{\mathbf{x} \notin UR(o_i)} f_i(\mathbf{x}) d\mathbf{x} = 0$.

3.2 Kernel Function

For a set of samples $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$, where $\mathbf{x}_i \in R^m$, and $\mathbf{x}_i$ is mapped from the input space R^m to a new feature space R^s by a mapping function ϕ. The kernel function is defined as the dot product [8] in the feature space R^s: $H(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$.

The major advantage for kernel function is that it is constructed without knowing the concrete form of the mapping function ϕ.

3.3 Uncertain Distance Calculation by Kernel Trick

Given a set of uncertain objects $O = \{o_1, ..., o_n\}$, considering two uncertain objects o_i, o_j, $\forall o_i, o_j \in O$, whose pdfs are $f_i(\mathbf{x}_i)$, $f_j(\mathbf{x}_j)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are locations of o_i and o_j, respectively. The uncertain regions of o_i and o_j are defined as $UR(o_i)$ and $UR(o_j)$.

We can define the uncertain distance function over O as $\Delta : \ O \times O \times \Re \rightarrow \Re_0^+$, which returns the probability of a distance value and holds the following condition:

$$\int_0^\infty \Delta(o_i, o_j, \tau) d\tau = 1, \forall o_i, o_j \in O \ . \tag{1}$$

According to the pdfs related to the uncertain objects, Δ can be written as:

$$\Delta(o_i, o_j, \tau) = \textstyle\int_{UR(o_i)} \int_{UR(o_j)} Q(D(\mathbf{x}_i, \mathbf{x}_j), \tau) f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \ . \tag{2}$$

Where the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ is denoted as $D(\mathbf{x}_i, \mathbf{x}_j)$, τ is a non-negative real value, and $Q(\cdot)$ is a verdict function, $Q(x, y) = 1$, if $x = y$; $Q(x, y) = 0$ otherwise. In fact, $\Delta(o_i, o_j, \tau)$ returns the probability that the distance between o_i and o_j is equal to τ. Then the expected distance between o_i and o_j in terms of $\Delta(o_i, o_j, \tau)$ is expressed as follows,

$$ED(o_i, o_j) = \int_0^\infty \Delta(o_i, o_j, \tau) \tau d\tau \ . \tag{3}$$

Combining (2) with (3), the expect distance could be recomputed as:

$$ED(o_i, o_j) = \int_{UR(o_i)} \int_{UR(o_j)} D(\mathbf{x}_i, \mathbf{x}_j) f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \ . \tag{4}$$

In analogy with [7], the squared Euclidean distance is also used as the distance function $D(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2$ in this paper owing to its easier integration compared with other distance metrics.

The key issue of extending UK-medoids to kernel based UK-medoids for uncertain data is the expected distance computation in the new space. Let $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ denote the image of $\mathbf{x}_i$ and $\mathbf{x}_j$ by an implicit mapping function, respectively. The expected distance between o_i and o_j in the new space can be rewritten in the form of kernel function $H(\cdot)$ as:

$$\begin{aligned}
&ED(o_i, o_j) = \int_{UR(o_i)} \int_{UR(o_j)} D(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \\
&= \int_{UR(o_i)} \int_{UR(o_j)} ||\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)||^2 f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \\
&= \int_{UR(o_i)} \int_{UR(o_j)} (\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i) - 2\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) + \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_j)) f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \\
&= \int_{UR(o_i)} \int_{UR(o_j)} (H(\mathbf{x}_i, \mathbf{x}_i) - 2H(\mathbf{x}_i, \mathbf{x}_j) + H(\mathbf{x}_j, \mathbf{x}_j)) f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \\
&= \int_{UR(o_i)} H(\mathbf{x}_i, \mathbf{x}_i) f_i(\mathbf{x}_i) d\mathbf{x}_i - 2 \int_{UR(o_i)} \int_{UR(o_j)} H(\mathbf{x}_i, \mathbf{x}_j) f_i(\mathbf{x}_i) f_j(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \\
&\qquad + \int_{UR(o_j)} H(\mathbf{x}_j, \mathbf{x}_j) f_j(\mathbf{x}_j) d\mathbf{x}_j \\
&= E(H(\mathbf{x}_i, \mathbf{x}_i)) - 2E(H(\mathbf{x}_i, \mathbf{x}_j)) + E(H(\mathbf{x}_j, \mathbf{x}_j)) \; .
\end{aligned} \tag{5}$$

It is obvious from (5) that the expected distance between two uncertain objects can be denoted using the expected kernel $E(H(\cdot))$. If we consider each uncertain object o as a tuple $(A^{(1)}, ..., A^{(m)})$, and each attribute $A^{(w)}$ is denoted by a pair $(I^{(w)}, f^{(w)})$, $w \in [1, .., m]$. In this paper, we define the interval of $A^{(w)}$ as $I^{(w)} = [l^{(w)}, r^{(w)}]$, and $f^{(w)}$ is the probability density function defined over $I^{(w)}$. According to the above denotations, the uncertain objects o_i and o_j can be denoted as: $o_i = (I_i, f_i)$, $o_j = (I_j, f_j)$, where $I_i = (I_i^{(1)}, ..., I_i^{(m)})$, $f_i = (f_i^{(1)}, ..., f_i^{(m)})$, $I_j = (I_j^{(1)}, ..., I_j^{(m)})$ and $f_j = (f_j^{(1)}, ..., f_j^{(m)})$.

For $\forall \mathbf{x}_i \in I_i$, $\forall \mathbf{x}_j \in I_j$, we can have:$\mathbf{x}_i = (x_i^{(1)} \in I_i^{(1)}, ..., x_i^{(m)} \in I_i^{(m)})$, $\mathbf{x}_j = (x_j^{(1)} \in I_j^{(1)}, ..., x_j^{(m)} \in I_j^{(m)})$. In this case, as to (5), we can use the uncertain interval I_i and I_j to replace the uncertain region $UR(o_i)$ and $UR(o_j)$, respectively. The pdf $f_i(\mathbf{x}_i)$ and $f_j(\mathbf{x}_j)$ can be rewritten as:$f_i(\mathbf{x}_i) = \prod_{w=1}^{m} f_i^{(w)}$, $f_j(\mathbf{x}_j) = \prod_{w=1}^{m} f_j^{(w)}$.

In this paper, we focus on the case that each attribute of the uncertain objects is an interval, and the associated pdf is generated using a certain number of sample points in the interval. We compute the integrals in the distance calculation by means of samples based on corresponding pdfs. The expected distance can be calculated using (6):

$$\begin{aligned}
&ED(o_i, o_j) = \tfrac{1}{n_i} \tfrac{1}{n_j} \sum_{u=1}^{n_i} \sum_{v=1}^{n_j} ||\phi(\mathbf{x}_{i,u}) - \phi(\mathbf{x}_{j,v})||^2 \\
&= \tfrac{1}{n_i} \tfrac{1}{n_j} \sum_{u=1}^{n_i} \sum_{v=1}^{n_j} (\phi(\mathbf{x}_{i,u}) \cdot \phi(\mathbf{x}_{i,u}) - 2\phi(\mathbf{x}_{i,u}) \cdot \phi(\mathbf{x}_{j,v}) + \phi(\mathbf{x}_{j,v}) \cdot \phi(\mathbf{x}_{j,v})) \\
&= \tfrac{1}{n_i} \sum_{u=1}^{n_i} H(\mathbf{x}_{i,u}, \mathbf{x}_{i,u}) - 2 \tfrac{1}{n_i} \tfrac{1}{n_j} \sum_{u=1}^{n_i} \sum_{v=1}^{n_j} H(\mathbf{x}_{i,u}, \mathbf{x}_{j,v}) + \tfrac{1}{n_j} \sum_{v=1}^{n_j} H(\mathbf{x}_{j,v}, \mathbf{x}_{j,v}) \\
&= \overline{H}(\mathbf{x}_i, \mathbf{x}_i) - 2\overline{H}(\mathbf{x}_i, \mathbf{x}_j) + \overline{H}(\mathbf{x}_j, \mathbf{x}_j) \; .
\end{aligned} \tag{6}$$

Here $\mathbf{x}_{i,u}$ and $\mathbf{x}_{j,v}$ represent the u-th sample of object o_i and the v-th sample of object o_j, respectively; $\mathbf{x}_{i,u} = (x_{i,u}^{(1)} \in I_i^{(1)}, ..., x_{i,u}^{(m)} \in I_i^{(m)})$, $\mathbf{x}_{j,v} = (x_{j,v}^{(1)} \in I_j^{(1)}, ..., x_{j,v}^{(m)} \in I_j^{(m)})$; n_i and n_j denote the numbers of samples for objects o_i and o_j; $H(\cdot)$ is the kernel function and $\overline{H}(\cdot)$ denotes the average for kernel values.

3.4 The Kernel Based UK-Medoids Algorithm

In this subsection, we give our kernel based K-medoids algorithm for clustering uncertain data. The framework of our method is introduced in Algorithm 1.

Algorithm 1. Kernel based UK-medoids

```
Input:
    Uncertain objects set, $O = \{o_1, ..., o_n\}$;
    Total number of clusters, $k$;
Output:
    The final partitioned clusters, $\varsigma = \{C_1, ..., C_k\}$;
 1: // The preparation phase:
 2: Compute the expected kernels to construct the overall expected kernel matrix $EH$;
 3: Compute the expected distances in the new space $ED(o_i, o_j), \forall o_i, o_j \in O$ by means of the
    expected kernels in the matrix $EH$;
 4: // The clustering phase:
 5: Select the initial medoids set $M = \{m_1, ..., m_k\}$;
 6: $M' \leftarrow M$;
 7: $M \leftarrow \emptyset$;
 8: $\varsigma = \{C_1, ..., C_k\} \leftarrow \{\emptyset, ..., \emptyset\}$;
 9: for all $o \in O$ do
10:     $m_t \leftarrow \arg\min_{o' \in M'} ED(o, o')$;
11:     $C_t = C_t \cup \{o\}$;
12: end for;
13: for all $C \in \varsigma$ do
14:     $m \leftarrow \arg\min_{o \in C} \sum_{o' \in C} ED(o, o')$;
15:     $M \leftarrow M \cup \{m\}$;
16: end for;
17: If $M \neq M'$, go to line 6 otherwise stop, and return the final clusters $\varsigma = \{C_1, ..., C_k\}$;
```

In the preparation phase (lines 2-3) of Algorithm 1, the expected kernel matrix and all the expected distances between any pair of objects are calculated only once and are used at each iteration of clustering. The clustering phase is given in lines 5-17, which is similar to the work in [2], the k initial medoids are selected by random chance in this paper. The main loop is conducted in the succedent lines. In lines 9-12, all the objects are assigned to their closest clusters in terms of the expected distance between objects and the medoids in the new space. And in lines 13-16, the medoid in each cluster are recomputed. The goal is to find a new medoid of each cluster, which is the object with minimal total distance to other objects in its cluster. The loop will stop if there is no change with respect to the previous iteration, otherwise go to line 6 to repeat.

Note that, the computation of both expected kernels in matrix EH and the expected distances in matrix ED are all separated from the clustering process. The structures for both EH and ED are given in (7).

$$EH = \begin{bmatrix} eh_{1,1} & eh_{1,2} & \dots & eh_{1,n} \\ eh_{2,1} & eh_{2,2} & \dots & eh_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ eh_{n,1} & eh_{n,2} & \dots & eh_{n,n} \end{bmatrix}, ED = \begin{bmatrix} ed_{1,1} & ed_{1,2} & \dots & ed_{1,n} \\ ed_{2,1} & ed_{2,2} & \dots & ed_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ ed_{n,1} & ed_{n,2} & \dots & ed_{n,n} \end{bmatrix}. \tag{7}$$

where $eh_{i,j}$ denotes $E(H(\mathbf{x}_i, \mathbf{x}_j))$, and $ed_{i,j}$ represents $ED(o_i, o_j)$. In fact, both EH and ED are symmetry matrixes. Hence, in order to save memory space, we only need to store the up-triangle part in memory.

4 Experiment Evaluation

4.1 Data Sets

Synthetic Dataset. Delta Set[1] is formed by 424 points of two classes with each class 212 points, which is a nonlinearly separated bidimensional set [9], namely, each data point has two attributes.

Real Datasets. We choose three datasets with numerical value attributes, namely, *iris*, *wine*, and *sonar* from the popular UCI Machine Learning Repository[2].

The UCI data sets used in our experiments are shown in Table 1. Column 1 gives the name of each dataset, column 2 lists the total counts of objects for the corresponding dataset, and the last two columns give the corresponding number of attributes and classes, respectively.

4.2 Experimental Settings

Originally, the datasets we selected are certain with deterministic values. To make the attributes uncertain, we convert each numerical value into an uncertain interval. For attribute A, we scan all its attribute values and get its maximum value $A.max$ and minimum value $A.min$. We can have $length(A) = A.max - A.min$. For the interval $I = [l, r]$ within $[A.min,\ A.max]$, let $\lambda = lenth(I)/length(A)$, we call the dataset with λ extent uncertainty. For each tuple $\mathbf{x}$, the value $x^{(w)}$ on the w-th attribute $A^{(w)}$ can be converted into a λ extent interval with $l^{(w)} = x^{(w)} - \lambda\xi * lengh(A^{(w)})$ and $r^{(w)} = x^{(w)} + \lambda(1-\xi) * length(A^{(w)})$, where $\xi \in [0, 1]$ is a controlling parameter.

In this paper, we consider two density functions, i.e., uniform and normal (Gaussian) pdfs. For uniform distribution, the pdf defined over $I^{(w)} = [l^{(w)}, r^{(w)}]$ can be denoted as $f^{(w)} = \frac{1}{r^{(w)} - l^{(w)}}$. As for normal distribution, we use the original real value $x^{(w)}$ as the mean, and $\frac{1}{4}(r^{(w)} - l^{(w)})$ as the standard deviation to represent the pdf. Hence, the pdfs can be generated by S sample points from the interval according to the corresponding distribution (i.e., normal distribution or uniform distribution).

[1] Delta dataset can be downloaded from: ftp://ftp.disi.unige.it/person/CamastraF/delta.dat

[2] http://archive.ics.uci.edu/ml/

To conduct a fair comparison between Uk-medoids and our kernel based method, the set of initial medoids is also randomly selected, for both methods, the experiment results are averaged over 50 different runs to avoid the bias introduced by random initialization.

The algorithms mentioned above are implemented in MATLAB, our experiments are executed on a PC with Pentium IV 3.2 GHz CPU and 2.0 GB main memory. We use Gaussian kernel as our kernel function, for Delta Set, the kernel width $\sigma = 0.032$, and for the three UCI datasets, namely, *iris*, *wine*, *sonar*, the parameter $\sigma = 0.3, 0.5, 0.48$, respectively. For all datasets, we set the number of sample points $S = 20$, and the controlling parameter $\xi = 0.5$.

4.3 Evaluation Criterion

In this paper, we report our experiment results mainly resorting to the clustering accuracy, which is widely used for measuring the performance of clustering in the research community[10], [11]. Suppose there is a collection of uncertain objects $O = \{o_1, ..., o_n\}$, let $\varsigma = \{C_1, ...,C_k\}$ be the output partition resulted by a clustering algorithm, and $\mathrm{Z} = \{\mathrm{Z}_1, ..., \mathrm{Z}_k\}$ be the predefined (real) partition of the objects in O.The overall *Accuracy* of clustering can be defined as follows:

$$Accuracy = \frac{1}{n}(\sum_{i=1}^{k} \max_{j \in [1..k]} |C_i \cap \mathrm{Z}_j|) \ . \tag{8}$$

Where $|C_i \cap \mathrm{Z}_j|$ denotes the number of common members between every two clusters (i.e., the yielded cluster C_i vs. the referenced cluster Z_j).

4.4 Experimental Results and Analysis

In this subsection, we report the clustering performance obtained by our kernel based method and the UK-medoids algorithm and make a comparison.

Table 1. The UCI data sets used in our experiments

dataset	# objects	# attributes	# classes
iris	150	4	3
wine	178	13	3
sonar	208	60	2

Table 2. Quality results (average clustering accuracy), $\lambda = 0.4$

dataset	**pdf**	**UK-medoids**	**Our method**
Delta	Uniform	0.5024	**0.5479**
	Normal	0.5021	**0.6418**
iris	Uniform	0.8373	**0.8764**
	Normal	0.8393	**0.8625**
wine	Uniform	0.9162	**0.9325**
	Normal	0.8630	**0.8740**
sonar	Uniform	0.5466	**0.5893**
	Normal	0.5465	**0.5507**

The detailed quality results with two different pdfs (i.e., uniform and normal) for clustering on all our selected datasets with $\lambda = 0.4$ are given in Table 2. Experimental results yielded by the two clustering methods are listed in column 3 and 4 over 50 runs on each data set. It is obvious that our kernel based method performs better than UK-medoids algorithm in terms of average accuracy. We can also find that there is no significant change between the two different pdfs on these datasets.

5 Conclusion and Future Work

In this paper, we proposed a novel kernel based K-medoids algorithm for clustering uncertain objects. We compute the distance between any pair of uncertain objects in a feature space using the expected kernels. Experimental results conducted on synthetic and real-life datasets demonstrate that our kernel based method outperforms the UK-medoids algorithm in terms of accuracy.

In our future work, for large scale uncertain datasets, we plan to devise an incremental approach for our kernel based UK-medoids to address the bottleneck of the disk-space and I/O requirements.

Acknowledgements. This research is supported by Chinese Universities Scientific Fund (QN2009092). And the authors also gratefully acknowledge the support of K. C. Wong Education Foundation, Hongkong.

References

1. Chau, M., Cheng, R., Kao, B., Ng, J.: Uncertain data mining: An example in clustering location data. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS (LNAI), vol. 3918, pp. 199–204. Springer, Heidelberg (2006)
2. Gullo, F., Ponti, G., Tagarelli, A.: Clustering Uncertain Data Via K-Medoids. In: Greco, S., Lukasiewicz, T. (eds.) SUM 2008. LNCS (LNAI), vol. 5291, pp. 229–242. Springer, Heidelberg (2008)
3. Ngai, W., Kao, B., Chui, C., Cheng, R., Chau, M., Yip, K.: Efficient clustering of uncertain data. In: Perner, P. (ed.) ICDM 2006. LNCS (LNAI), vol. 4065, pp. 436–445. Springer, Heidelberg (2006)
4. Yun, C., Yang, J.: Reducing UK-Means to K-Means. In: Proceedings of the 7th IEEE International Conference on Data Mining Workshops, ICDMW 2007, pp. 483–488 (2007)
5. Kriegel, H., Pfeifle, M.: Density-based clustering of uncertain data. In: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD 2005, pp. 672–677 (2005)
6. Kriegel, H., Pfeifle, M.: Hierarchical density-based clustering of uncertain data. In: Proceedings of the 5th IEEE International Conference on Data Mining, ICDM 2005, pp. 689–692 (2005)
7. Xiao, L., Hung, E.: An efficient distance calculation method for uncertain objects. In: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2007, pp. 10–17 (2007)
8. Shawe-Taylor, J., Cristianini, N.: Kernel methods for pattern analysis (2004)
9. Camastra, F., Verri, A.: A Novel Kernel Method for Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(5), 801–805 (2005)
10. Caruana, R., Elhawary, M., Nguyen, N., Smith, C.: Meta Clustering. In: Perner, P. (ed.) ICDM 2006. LNCS (LNAI), vol. 4065, pp. 107–118. Springer, Heidelberg (2006)
11. Mccallum, A., Nigam, K., Ungar, L.: Efficient clustering of high-dimensional data sets with application to reference matching. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2000, pp. 169–178 (2000)

Frequent Pattern Mining Using Modified CP-Tree for Knowledge Discovery

R .Vishnu Priya[1], A.Vadivel[1], and R.S. Thakur[2]

[1] Department of Computer Applications National Institute of Technology, Tamilnadu, India
[2] Maulana Azad National Institute of Technology, Madyapradesh, India
vissaru@yahoo.co.in, vadi@nitt.edu, ramthakur2000@yahoo.com

Abstract. Mining frequent pattern from databases is useful for knowledge discovery. In this paper, we propose modified CP-Tree, which scans entire transactions only once and constructs the tree by inserting the transactions one by one. The constructed tree consists of an item list along with its occurrence. In addition, a sorted order of items with its frequency of occurrence is maintained and based on the sorted value, the tree is dynamically rearranged. In rearranging phase, the nodes are rearranged in each branch based on sorted order of items. Each path of the branch is removed from the tree, sorted based on sorted order of items and inserted back as a branch into the tree. We have evaluated the performance of the proposed modified tree on benchmark databases such as CHESS, MUSHROOM and T10I4D100K. It is observed that the time taken for extracting frequent item from the tree is encouraging compared to conventional CP-Tree.

Keywords: Frequent Pattern Mining, Modified CP-Tree, Knowledge Discovery.

1 Introduction

A fundamental component in data mining tasks is finding frequent patterns in a given dataset. Apriori is the first technique to find frequent patterns and it uses a "bottom up" approach. In order to find frequent patterns, Apriori requires multiple database scans and it requires a large amount of memory to handle the candidate patterns. To overcome from this drawback, there is an algorithm called frequent-pattern tree. The FP-tree construction takes exactly two scans: The first scan collects the set of frequent items and the second scan constructs the FP-tree. Even though FP-tree can mine frequent items by scanning the database twice there is another algorithm called CP-tree, which mine the frequent patterns with one scan. The main idea in CP-tree is to periodically reorganize the prefix-tree in frequent-dependent item order after inserting some transactions into the prefix-tree using previous item order. Through repeated reorganization, CP-tree provides better mining performances as a result and it also supports both incremental and interactive mining. Although the above mentioned algorithms can mine the frequent patterns efficiently, the prefix tree construction time for mining frequent patterns is found to be quite higher. In our, approach, we address this drawback by constructing the prefix tree structure in very short time compared to

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 254–261, 2010.

the prefix tree. The proposed tree structure is constructed by rearranging the nodes in each branch from already constructed tree based on item sorted list (items are arranged in descending order based on it count). By this process, the proposed tree takes less time for frequent pattern mining compared to CP-tree algorithm and it also support both incremental and interactive mining support.

We organize the rest of the paper as follows. We discuss the related work in the next section. In section 3, we discuss the procedure for tree construction and rearrangement. The functionality and frequent patterns mining of the modified tree is explained in section 4. In section 5, we present the experimental result and conclude the paper in the last section of the paper.

2 Related Work

In 1993, the first known frequent pattern mining algorithm is Apriori [1], which was proposed by Agrawal et al. Since then various algorithms have been proposed for improving the performance of Apriori-based algorithm. This kind of approaches requires multi database scan and a huge number of candidate sets are generated to find frequent patterns. In Apriori, performance issues such as memory space and execution time is found to be high. This restricts use of these methods for incremental, interactive and data stream mining. This problem can be handled by using frequent pattern tree algorithm proposed by Han et al. [5], which mine frequent patterns with two scans and eliminate the generation of candidate sets. Among tree based approaches, the highly compact prefix tree structure called FP- tree is a new approach for mining frequent patterns. Various studies have been carried out on FP-tree by focusing either on its performance improvement or proposing new application domain. However, it has been noticed that though FP-tree mine frequent patterns with two database scan, it may not be suitable for incremental, interactive and data stream mining. Various algorithms have been proposed for discovering frequent patterns from incremental databases such as FUP2 [3] and UWEP [2]. However, these approaches have to generate large number of candidates and repeatedly scan the database. Similarly, the tree-based approaches such as Adjusting FP-tree for Incremental Mining [7], Extending FP-Tree for Incremental Mining [9] and Fast Updated Frequent Pattern Trees [6] have been proposed to perform incremental mining by adjusting the FP-tree structure. Koh and Shieh proposed the AFPIM algorithm and the prefix tree constructed only for frequent items with two database scan. While new transactions are updated, the old tree is not being used and instead, it rescans again the entire updated database to build a new FP-tree and thus tree construction time is more. Xin Li has proposed the EFPIM algorithm for constructing tree with two database scans by using extra variable called pre minsupport (value less than minsupport). EFP tree constructed for an updated database is found to be an extension of the old one, which means all existing nodes and their arrangement is not changed. FUFP has been proposed by Tzung-Pei Hong, where for new transactions, while originally large item becomes small, items are directly removed from the FUFP- tree. Thus, when an originally small item is becoming large, it is added to the end of the header table and then inserted into the leaf nodes of the FUFP tree. While doing so, the frequent descending order is lost and the computational time is more. Compressed and Arranged Transaction Sequences Tree

(CATS-tree), Canonical-Order tree (CAN-tree) has been proposed for overcoming the drawbacks of AFPIM and EFPIM. Cheung and Zaiane [4] have designed the CATS tree and it is constructed by one scan. However, the tree construction is found to be expensive in terms of time. This is due to the fact that this method searches for common items and tries to merge the new transactions into an existing tree path. This leads to the requirement of swap/merging the nodes. While mining frequent items in CATS-tree, one need to travel both upward and downward direction. Kai-Sang Leung [8] proposed the CAN-tree algorithm, it construct prefix tree in canonical order with one database scan. Since, it is arranged in canonical order, frequent descending order of items is not maintained and this result in very high mining time compared to FP-tree. To resolve the drawback of CAN-tree, the Compact Pattern (CP-tree) [10] was proposed by Syed Tanbeer, which mines the frequent patterns with one scan. In this approach, the tree is constructed for some transactions based on predefined item order given by the user. Since, the frequency of occurrence for items are found from the constructed tree and based on the items sorted order, the dynamically rearranged tree is created. This process is periodically continued till end of the transactions in the dataset. However, it incurs very high computation time due to, the repeated reorganization of the old branch based on new Isort (items are arranged in descending order based on its frequency of occurrences) in the rearranging phase while the new transactions are inserted in construction phase. In contrary, if no change has been made in new Isort, at least we have to check whether old branch is in the same order of new Isort in rearranging phase. In contrast to all the above method discussed, our proposed modified tree scan the database once and the time for frequent pattern mining is found to be low compared to the original CP-Tree.

3 Construction and Rearrange Phase of the Proposed Tree

3.1 Constructing Phase

In our approach, the transactions are inserted into the tree one by one. While inserting an item, the order of the item in the transaction is maintained. This is due to the fact that our aim is to mine frequent patterns by performing only one scan of the Database. In Table 1, we have presented the sample transactions considered for mining the frequent pattern. The TID is the Transaction ID and Pi is the product purchased during a particular transaction.

Table 1. Sample transactions

TID	Transactions
1	P_2 P_1 P_6 P_7 P_8
2	P_4 P_5 P_1
3	P_2 P_1 P_6 P_7 P_4
4	P_1 P_2 P_3
5	P_1 P_4 P_7
6	P_2 P_4 P_7 P_3 P_9

The constructed tree will have the itemlist along with its occurrence. In addition, a sorted order of items with its frequency of occurrence is maintained and based on the sorted value, we dynamically rearrange the tree. Below, we illustrate the process of tree construction with a suitable example. Initially, an empty node is created with null as the label. The first transaction in Table 1, which is <P2 P1 P6 P7 P8>, scanned once and each item in this transaction is inserted as a branch in the tree along with its frequency of occurrence. Let the inserted branch in the tree is <P2:1 P1:1 P6:1 P7:1 P8:1>. After inserting the first transaction, the next transaction < P4 P5 P1> is scanned once. The first item of the transaction which is P4, is checked with first item of the inserted branch, which is P2 in our case. If items are found to be equal then the count of the item in that branch is incremented by one and the current node is not inserted and the process continues to check the next item. Otherwise the node is inserted as a new branch or path with its count. However, in this case, which is P4= P2, and both are not equal and thus, it is inserted as a new branch. Similarly, let the second inserted branch in the tree is <P4:1 P5:1 P1:1>. In the same way, all of the remaining transactions are inserted into the tree and its Isort is generated. In Figure 1, we show the constructed tree and the corresponding Isort.

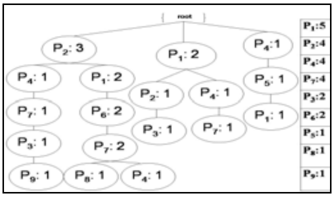


Fig. 1. The constructed tree with itemset

3.1.1 Algorithm for Tree Construction

```
Input: Dataset DB, Total No of transactions N.
Output: Tree, Item Sorted List I_sort.
Method:
Read DB, N and set j=1.
Create the root of a tree T and label it   as “null”.
while (j<=N){
Scan the transaction tj once.
Let the transaction be [p/P], where p is the first element and P is the remaining element in the
transaction, then call construct_tree ([p/P], T).
 Fun construct_tree([p/P],T){
   If T as a child C such that C.item_name=p. item_name then increment C’s count by 1.
   else
    Create a new_node C;
     C’s count=1;C’s parent_link be linked to T;
     C’s node_link be linked to rest of the items in the transaction.}}
Increment j variable  }
Find occurrences of each item in the tree and arranged each item based on occurrences. The
arranged items are stored in I_sort.
```

Once, the tree is constructed by using all the transactions, it is necessary to rearrange the tree in such a way that the nodes with frequency of occurrence will be on the top and the one with less frequency of occurrence will be in the bottom of the tree. This is being carried out for effective frequent pattern mining. The rearranging phase of the proposed tree is explained below.

3.2 Rearranging Phase

In rearranging phase, the nodes are rearranged in each branch based on Isort. The rearrangement using Isort can be performed by various sorting techniques and in our approach; we have used array based technique. Each path of the branch is removed from the tree, sorted based on Isort and inserted back as a branch into the tree. This rearrangement process is continues till the entire branches in the tree are arranged based on Isort. Now we illustrate the rearranging phase with a suitable example. The rearranging phase starts by removing the second path of the branch P2 <P2:1 P1:1 P6:1 P7:1 P8:1> from the constructed tree (Fig. 1) and the removed branch is stored in temp array. The items in temp array are sorted based on item sorted list. Let the sorted path < P1:1 P2:1 P7:1 P6:1 P8:1> is inserted as a branch in the tree, which is shown in Fig. 2.

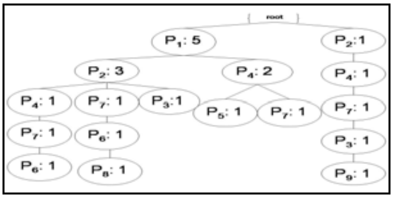


Fig. 2. The rearranged tree

After insertion, the third branch, which is < P4:1 P5:1 P1:1> is removed from the tree (Fig. 1), then stored in temp array. The items in temp array are sorted based on Isort then the first item <P1:1> in Isort is checked with first item of the branch <P1:1>, in this case both are equal. Thus, the count of the item in the branch is incremented by count of the item is in temp array. After checking the first item, the second item <P4:1> is checked with second node of the branch <P2:1>. In this case, both are not equal then <P4:1> is linked as a child of <P1:5> and rest of the items in temp array is linked as the child of <P4:1> as shown in Fig. 2. This process is continued till all the branches in constructed tree are inserted into rearranged tree.

3.2.1 Algorithm for Tree Rearrangement

```
Input: $I_{sort}$ and Tree T.
Output: $T_{sort}$.
Method:
Read $I_{sort}$ and Tree T
for each branch $B_i$ in T
for each unprocessed path $P_j$ in $B_i$
Remove the unprocessed path $P_j$ from T
If each node in $P_j$ is sorted based on $I_{sort}$
  {for each node $n_k$ in $P_j$ from the $leaf_p$ node
   for each sub_path S from $n_k$ to $leaf_c$ with $leaf_c$ not equal to $leaf_p$
   If frequency of each node in S from $n_k$ to $leaf_c$ are less than frequency of node $n_k$
      P= path from the root to $leaf_c$
10.        Execute the step from 16 - 19
11.      else
12.        P= S from $n_k$ to $leaf_c$
13.          If P is sorted path
14.            Execute the step from 6 - 14 }
15.  else
16.      { Reduce the frequency of each node in $P_j$ to the frequency of leaf node
17.        Stored each node in P into the temp array
18.        Insert sorted nodes as a branch into T at the location from where it was taken }
19.  Terminate while all the path in T are found to be sorted based on $I_{sort}$.
```

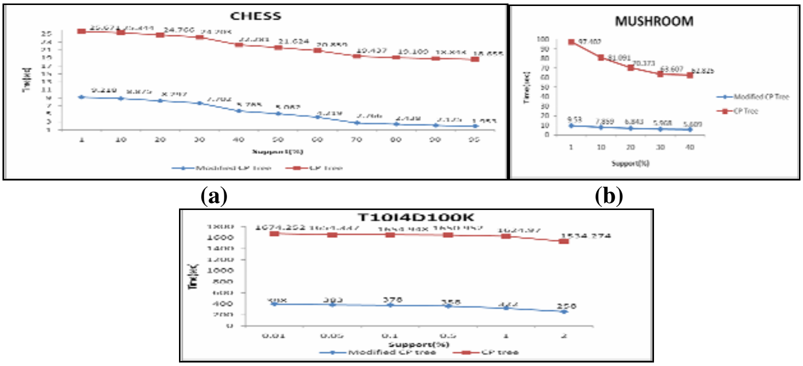
4 Frequent Patterns Mining and Functionalities of the Proposed Tree

Mining frequent pattern is one of the important tasks in Data mining. In this section, we illustrate mining frequent patterns from the proposed tree using FPtree mining technique [5]. For mining, the conditional pattern base is found for all the items from bottom to top in the Isort. Each and every item, from the bottom, in Isort is considered for finding the conditional pattern base. In conditional pattern base for the current item, we store the set of prefix paths from the root of the rearranged tree till the current item and each items present in the prefix path carries the occurrence of the corresponding current item in that path. Now, for the current item, a small Ilist is generated for the conditional pattern base of the current item. After Ilist is created, a new conditional tree is constructed for the conditional pattern base of the current item by eliminating all infrequent items having a count less the minimum support value from the Ilist. By recursively passing through this tree, we get frequent patterns of the current item and similarly, the frequent pattern is found for all the items in the Isort.

While new transactions are updated in the dataset, each updated transaction is inserted as a branch of the rearranged tree and now it acts as a constructed tree based on Isort and the new Isort for that tree is created. Now, the same procedure presented in Section 3 is followed for rearranging the tree based on the new Isort. Using rearranged tree, we find frequent patterns and thus while there is a new transactions in the database, it is not required to scan the entire transactions and begin the mining process from the scratch. Similarly, while minimum support given by the user is too low or too high, there is a possibility that some of the important patterns of interest may be missed. Interactive mining provide the user an opportunity to interactively alter the minimum support. In Interactive mining, the database usually remains unchanged and only minimum support value is changed. Thus, the tree is constructed only once for a given database and it is used in mining with various minimum support values. By doing so, the tree construction cost can be reduced to some extent over several runs of the mining process. Similar to other interactive mining algorithms, mining time using the proposed tree reduced considerably by caching the frequent patterns mined from previous round and reusing them for the current round with different minimum support value.

5 Experimental Results

In this section, we present the performance of the proposed tree on frequent patterns mining from the datasets. All the experiments have been performed in computer with Intel(R) Core(TM) 2DUO 3GHz CPU and 1.96 GB RAM. We implement our proposed algorithm and the CP-tree algorithm in java. For evaluation, we have used two dense benchmark datasets such as CHESS [11], MUSHROOM [11] and one sparse dataset such as T10I4D100K[11]. In CP-tree the user supplies a value for "n", which represents the number of transactions after restructuring phase to be done. In our work we choose the values of n=500 for Chess, n=1000 for Mushroom and n=25,000 for T10I4D100K are chosen for the execution purpose. In each graph x-axis shows vary in minimum support value (%) in ascending order and y-axis shows the execution time represent in seconds. In Fig.3, we can see that the execution time of our proposed tree is low comparing to the execution time of the CP-tree for mining frequent patterns. While the minimum support values increases gradually the execution time for mining patterns is decreases. This is due to the fact that, for lower support the frequent patterns are more, hence it requires high mining times that increase the rate of change in overall runtime. In contrary, for high minimum support the frequent patterns to mine are less, hence it requires low mining time.



(a) (b)

(c)

Fig. 3. Time for extracting patters from datases.Fig.3(a).Time for extracting frequent pattertns from CHESS dataset. Fig.3(b).Time for extracting frequent patterns from MUSHROOM dataset.Fig.3(c).Time for extracting frequent from T10I4D100K datasets.

6 Conclusion

Mining frequent patterns from databases is useful for knowledge discovery. Tree based approaches has been proposed by the researchers and the FP-tree, CP-tree etc are considered as suitable structure for frequent pattern mining. However, these approaches take large computational time. In this paper, we have considered this issue and proposed a modified tree of CP tree for frequent pattern mining. The proposed method scans the transactional database only once and builds tree in lesser time. The nodes in each branch from already constructed tree are dynamically rearranged based on Isort. The tree is constructed for all the items in the transactional database, even though, items are not frequent. While the new transactions updated, the already constructed tree is rearranged for updated transactions. Thus, the proposed tree also supports both interactive and incremental mining. We have evaluated the performance of the proposed modified tree on benchmark databases such as CHESS, MUSHROOM and T10I4D100K. It is noticed that the time taken by the proposed tree for extracting frequent item is encouraging compared to the other relevantly proposed tree.

Acknowledgments. The work done by Dr. A.Vadivel is supported by research grant from the Department of Science and Technology, India, under Grant SR/FTP/ETA-46/07 dated 25th October, 2007.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings. ACM-SIGMOD International Conference on Management of Data (SIGMOD), Washington, DC, pp. 207–216 (1993)
2. Ayan, N.F., Tansel, A.U., Akrun, E.: An efficient algorithm to update large itemsets with early pruning. In: Proceedings of the Fifty ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 287–291 (1999)
3. Cheung, D.W., Lee, S.D., Kao, B.: A general incremental technique for maintaining discovered association rules. In: Proceedings of the Fifth International Conference on Database Systems for Advanced Applications, pp. 185–194 (1997)
4. Cheung, W., Za, O.R.: Incremental mining of frequent patterns without candidate generation or support constraint. In: Proceedings of the Seventh International Database Engineering and Applications Symposium, IDEAS 2003 (2003)
5. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 1–12 (2000)
6. Hong, T.-P., Lin, C.-W., Wu, Y.-L.: Incrementally fast updated frequent pattern trees. Expert Systems with Applications 34(4), 2424–2435 (2008)
7. Koh, J.-L., Shieh, S.-F.: An efficient approach for maintaining association rules based on adjusting FP-tree structures. In: Lee, Y., Li, J., Whang, K.-Y., Lee, D. (eds.) DASFAA 2004. LNCS, vol. 2973, pp. 417–424. Springer, Heidelberg (2004)
8. Leung, C.K., Khan, Q.I., Li, Z., Hoque, T.: CanTree: a canonical-order tree for incremental frequent-pattern mining. Knowledge and Information Systems 11(3), 287–311 (2007)
9. Li, X., Deng, X., Tang, S.: A fast algorithm for maintenance of association rules in incremental databases. In: Li, X., Zaïane, O.R., Li, Z.-h. (eds.) ADMA 2006. LNCS (LNAI), vol. 4093, pp. 56–63. Springer, Heidelberg (2006)
10. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: Efficient single-pass frequent pattern mining using a prefix-tree. Information Sciences 179, 559–583 (2008)
11. http://fimi.cs.helsinki.fi/data

Spatial Neighborhood Clustering Based on Data Field

Meng Fang[1], Shuliang Wang[1], and Hong Jin[2]

[1] International School of Software,Wuhan University, Wuhan 430079, China
whumoe@163.com, slwang2005@whu.edu.cn
[2] State Key Laboratory of Software Engineering, Wuhan University, China
anya_1024@163.com

Abstract. Based on the theory of data field, each sample point in the spatial database radiates its data energy from the sample space to the mother space. This paper studies the use of the data field as a basis for clustering. We put forward a novel method for clustering, which is a kind of natural clustering method called spatial neighborhood clustering. In the data field, the potential center is identical to the cluster center. The key step of the cluster algorithm is to find the potential centers in the grid units of data field. The spatial neighborhood cluster method makes use of the distribution property of the potential value point as the potential center in the data field to discriminate the maximum potential value in a certain neighborhood window. Then the cluster centers can be acquired corresponding to the maximum potential values and the number of cluster centers is automatically amount to that of potential centers.

Keywords: clustering; spatial data mining; data field; spatial neighborhood discriminating.

1 Introduction

In general, clustering can be considered as the process of organizing objects into groups whose members are similar in some way [6,11]. In spatial data mining, clustering algorithm should be more efficient when dealing with large spatial database and can handle arbitrary shape of cluster [2,9]. Generally, the cluster algorithm can be divided into two categories such as partition algorithms and hierarchy algorithms [10,11]. Partitional algorithms obtain a partition of the dataset into classes such that the instances in a class are more similar to each other than to instances in different classes, for example, popular k-means and k-medoid method. Hierarchical algorithms divide the data set into subsets as tree structure until each subset contains only one object which can be constructed by split or merge. It needs no argument but the stop condition should be defined. For instance, BIRCH [3], CURE [4] and CLIQUE [5] are representative of this type of algorithm.

Data field theory is more suitable for the tasks of spatial data mining [1]. In particular, when dealing with attribute space or feature space and mining natural clustering knowledge in different level, it shows certain applicability in this area. One of our contributions is that the proposed method based on data field theory doesn't need to determine the number of clusters; it clusters data set naturally according the distribution of

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 262–269, 2010.

data set itself. Additionally, our approach is more efficient than the previous algorithm—cutting potential centers algorithm [1]. This paper is organized as follows. The related work and the previous clustering algorithm based on data field theory are discussed in Section 2. In Section 3 spatial neighborhood clustering method is proposed and an experiment is given in Section 4. We conclude with a summary in Section 5.

2 Related Work

In the study of clustering algorithm based on data field, there is a clustering method called "cutting potential centers algorithm". Those most relevant to our work are discussed briefly in this section and we emphasize the limitations which are addressed by our approach. Firstly, we describe fundamental assumption, data field.

In the data field, each data point radiates data energy to the whole data field in order to demonstrate its existence and action in the tasks of spatial data mining. Data field is a mathematical abstraction and hypothesis of data radiation process. Through data radiation, it formulates a data field which amounts to a virtual space field describing the influences that each data point has to the whole data field. In data field theory field-strength function is used to express the law of spatial radiation. Field-strength function describes the distribution of data energy and measures field strength in the data field of different locations.

The cutting potential centers method eliminates the influences of the already found potential centers when finding the next potential center. In the data field, potential center is always surrounded by points which also have large value of field strength. So when finding the next potential center, it is reasonable to eliminate the influences of the potential centers already found before. It repeats the process of cutting the potential center to find the next cluster center until a sufficient number of cluster centers are acquired. Firstly, it doesn't provide a formally description of how to determine the number of cluster centers and should be determined by subjective choices or a certain stop condition. To select the number of categories naturally based on the actual distribution of the spatial data set can be improved. Secondly, when the dataset become very large, the time of repeating of the elimination will be unavailable.

3 Approach

3.1 Grid Units of Data Field

To begin with, our approach selects a field-strength function to construct the data field. Field-strength function is used to express the law of spatial radiation in data field theory. The selection of field-strength function should fully consider the following conditions such as the characteristic of the data itself, the feature of spatial data radiation, the standard of the field-strength function's determination and the application domain of the spatial data field and so on [1]. This paper introduces the nuclear radiation derivative field-strength function as equation 1 shows:

$$s = e^{-\frac{r^2}{2k^2 C_T(x)}}, \tag{1}$$

where s represents the data field strength and r represents the radiation radius, with $C_T(x)$ determines the energy strength of the data radiation. On the condition that the value of $C_T(x)$ is uncertain, it can be set to a uniform value 1. Data radiation factor k is relevant to the data quantity and distribution which can regulate the field-strength function of the data field. The decay speed of nuclear radiation field is the fastest, which is more suitable than other deviate field-strength functions for data radiation.

Then the data field is constructed and divided into rectangle units or cube units or multi-dimension stereoscopes according to the dimension of the data. In more general terms, if the observed data has n attributes or dimensions, the corresponding grid units can be constructed with n-dimension and each dimension indicates different attributes of the data.

For each unit, we compute the sum of data field potential, which is depended on the field-strength function and the distance between the grid unit and other data points. The data field potential can be given by

$$p = -\int_c s \cdot dl, \tag{2}$$

where c is an arbitrary path connecting the point with zero potential to the data point. There are a lot of data points in the space and each data point radiates data energy to the whole grid units. The data field potential for each unit is computed by

$$p_{unit} = \sum_{d=i}^{n} p_i, \tag{3}$$

where n is the number of the data point. The computational complexity is $O(K)$, where K is the number of grid units. Empirically, we set K according to the dataset, $K << n$. (Some strategies can be applied when constructing the grid units to ensure $K <= n$, which are beyond the scope of this paper.)

3.2 Spatial Neighborhood Clustering Based on Data Field

Our approach provides a neighborhood window and searches the potential centers as cluster centers by computing the point which has the maximum potential value in the neighborhood window. Usually, more than one potential center can be found, depending on the natural characteristic of the dataset itself.

Given a data set, the process of this clustering method is as follows:

Input: data set
Output: the clustering result
Process:

(1)The data field is generated according to the data set, then the whole data field is divided into grid units(see section 2.1). The grid units can be viewed as the candidate set for cluster centers.

(2)According to the nuclear radiation field-strength function, the potential value for every grid unit can be computed.

(3)Traverse each grid unit, all of the cluster centers can be found by using spatial neighborhood discriminate algorithm.

(4)In accordance with cluster centers already recorded, the cluster results can be acquired.

More details about spatial neighborhood discriminate algorithm will be given. The spatial neighborhood discriminate algorithm is the most important part in the spatial neighborhood clustering method.

After the data field divided into grid units, these can be viewed as the candidate set for cluster centers. In the data field, potential center is a special point which has the maximum potential value in multi-dimensional space. The potential value of the points in the domain of neighborhood window of the maximum point is less than the maximum point's. According to this property, the problem for finding the potential centers is transformed into finding the local maximum value of the grid units. It proposes the spatial neighborhood discriminate algorithm as follows:

Input: data set
Output: the cluster centers
Process:

(1) Establish the data field and divide it into grid units, then the grid units is the candidate set of the cluster centers.

(2) Select any grid unit as the inspection point.

(3) Specify the inspection point's neighborhood window.

(4) Compute the potential value of the points in this neighborhood window:

> If the potential value of this point is larger than the value of any other units in the domain of its neighborhood window, then the inspection point is the cluster center and records this value and its position. Otherwise, the inspection point is not the cluster center.

(5) Select any other gird unit in the multi-dimensional grid units as the inspection point, and goto step 3 until all the grid units have been already detected.

In the multi-dimensional grid units, arbitrarily selecting a unit as the inspection unit, the neighborhood window of this unit will be analyzed. Firstly we specify the neighborhood window. One way to construct the window is to find the adjacent units of the selected unit. If its potential value is larger than any other units in the domain of its neighborhood window, then the inspection unit is the maximum point in accordance with the maximum principle. On the contrary, if the potential value of the inspection point is less than the potential value of the point in the domain of neighborhood window, then the inspection point is not the maximum point in this domain. Making use of this computing process to detect every point in the multi-dimensional space in turn, all the maximum units can be recorded. Followed by descending order in accordance with the potential value, the maximum sequence in the multi-dimensional grid space can be acquired corresponding to the potential centers of the data field.

4 Experiment

According to the spatial neighborhood clustering method, a case study is carried out. And we compare our approach with the other relevant methods, such as cutting potential centers algorithm and k-means clustering.

4.1 Methodology

Here is the pseudo-code of the spatial neighborhood discriminate algorithm implemented by MATLAB:

```
for each grid unit $m(i_1,i_2,\dots,i_n) \in M$
    $m(i_1,i_2,\dots,i_n) \leftarrow p(i_1,i_2,\dots,i_n)$, p is the potential function
candidate set of cluster centers $A \leftarrow \emptyset$
for each grid unit $m(i_1,i_2,\dots,i_n) \in M$
    boundary management  $m(i_1,i_2,\dots,i_n)$
    generate the neighborhood window
        neighborhood window$\leftarrow m(i_1+j_1,i_2+j_2,\dots,i_n+j_n)$, $j \in \{-1,0,1\}$
    do if $m(i_1,i_2,\dots,i_n)$ is true by using the spatial neighborhood
            discriminate algorithm with neighbors of $m(i_1,i_2,\dots,i_n)$
        then $m(i_1,i_2,\dots,i_n)$ is cluster center and $A \leftarrow A$□$\{m(i_1,i_2,\dots,i_n)\}$
return A
```

The main part of this algorithm is divided into two parts. In the first part the data field is divided into units. In the second part the spatial neighborhood discriminate is implemented. The time complexity of the spatial neighborhood discriminate algorithm is *O(n)*.Comparing to the time complexity of the cutting power center algorithm *O(k*n)*, where *k* is the number of iteration, obviously the former is better.

After the cluster centers have been found, a point over a certain range around one cluster center can be labeled a class. Or computing the distances between the point and each cluster centers, it chooses the nearest cluster center as class.

4.2 Experiment Process

Taking two-dimensional data space as an example and a data set is selected as figure 1 shows. In the experiment a data set of *23100* points were used and run on a

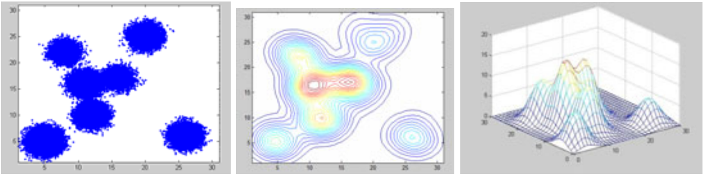
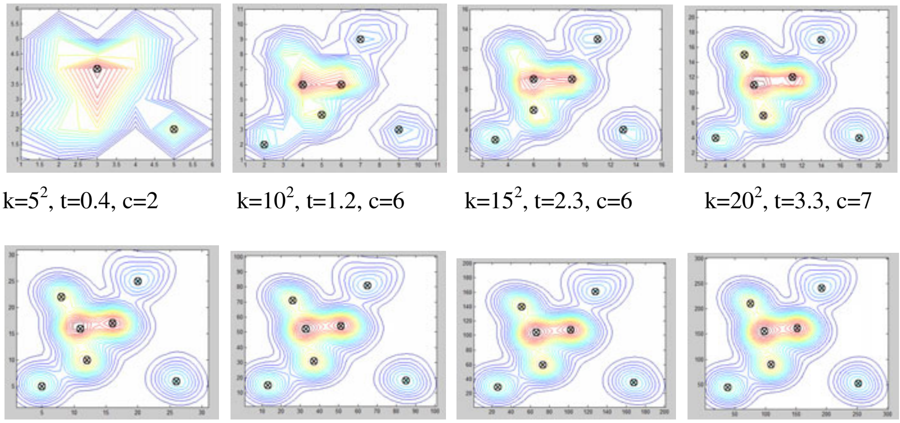


Fig. 1. The data set of 23100 points, and its equipotential lines and potential diagram

separate PC (*2.0* GHz Intel Pentium Dual Core, *2* G RAM). Then the data field was divided into grid units and every grid unit's neighborhood window contains *3*3* units. Figure 1 also shows the equipotential lines and potential diagram through our approach.

In fact, the data set of the experiment has a tendency that it can be divided into *7* parts. In the experiment, different *k* is provided, which is the number of grid units. Figure 2 shows the different result.



k=5^2, t=0.4, c=2 k=10^2, t=1.2, c=6 k=15^2, t=2.3, c=6 k=20^2, t=3.3, c=7

k=30^2, t=7.6, c=7 k=100^2, t=51.1, c=7 k=200^2, t=218.8, c=7 k=300^2, t=476.2, c=7

Fig. 2. Cluster centers of eight groups by different value of *k*, which is the number of grid units and its average running time t(s) and the number of cluster centers c

The number of the candidate cluster centers will decrease when the size of grid units cut down and the average of running time will do so too. The distribution of potential will also be smoother and the variation of potential became less. And when *k* is increase to a threshold, the cluster centers become stable. In the experiment, the threshold of *k* is *20*20* and its average running time is *3.3*s. It spends most of the time on the process of constructing grid units. However, the running time of the spatial neighborhood discriminate algorithm is little. For example, the time of the spatial neighborhood discriminate algorithm is less than *0.1*s when the *k=300*300*. Here, we choose the result of *30*30* grid units which is one of them and figure 3 shows the result of the clustering.

By using the spatial neighborhood discriminate algorithm, the seven cluster centers are found. The average running time is *7.6* seconds. And time consumed in the spatial neighborhood discriminate is *0.011* seconds. Most of the time served constructing the gird units. Considering the practical data set, the calculating of the cluster center is accurate. In this dataset *7* potential centers are found. It indicates that the dataset has *7* cluster centers according to its nature distribution, in other words, data set is divided into seven classes automatically.

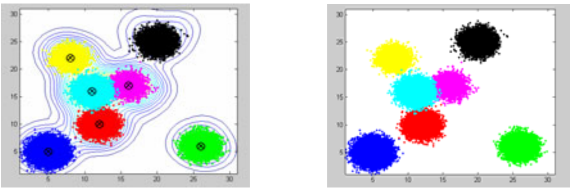


Fig. 3. The result of the clustering by our approach

4.3 Comparison Experiment

In order to compare the efficiency, we also implement the cutting potential center algorithm, which will modify the field strength function and recalculate the potential value in the data field, and k-means clustering under the same conditions. We should point out there will be *7* classes rather than naturally selecting number of class when using the later clustering methods. The results are presented in table 1.

Table 1. The cluster centers of data set by different algorithms and its average running time

Spatial neighborhood clustering	**K-means clustering(c=7)**	**Cutting potential center(c=7)**
coordinate(x,y)	coordinate(x,y)	coordinate(x,y)
(9.9, 15.6)	(9.5, 15.3)	(9.9, 15.6)
(15.3, 16.2)	(15.7, 15.9)	(15.3, 16.2)
(11.1, 9.3)	(10.8, 8.8)	(10.5, 15.6)
(7.8, 21.3)	(7.2, 21.3)	(10.8, 15.6)
(3.9, 4.5)	(3.7, 4.5)	(11.7, 15.6)
(25.5, 5.4)	(25.1, 5.3)	(15.3, 15.6)
(19.5, 24.3)	(19.2, 24.9)	(15.6, 15.6)
t=7.6s	t=5.8s	t=8.2s

The result between spatial neighborhood clustering and k-means clustering(c=*7*) is almost the same. However, the cutting potential center(c=*7*) is not accurate after the second iteration. When checking the distribution of potential of this data set, we find that some points' potential is negative, and that violate the property of data field theory. The reason is that when continuously cutting the potential center, it cannot assure the point whose potential value is small or approach zero will not be negative when continue to cut the potential center.

With respect to the running time, the k-means clustering has lest cost, and our approach has less cost and it is close to the time of k-means clustering. In fact, if *k=20*20* in the experiment, the average running time is less than the k-means's and its value is *3.3*s. Additionally, the spatial neighborhood clustering method is more efficient than the cutting potential centers method. It consumed 7.6 seconds, comparing 8.2 seconds by cutting potential centers method. In fact it only needs to traverse all the grid units one time, however, cutting potential centers method should traverse times.

5 Conclusion

In this paper, our clustering method is a kind of natural clustering method based on data field. It uses the spatial neighborhood discriminate algorithm to find the cluster centers. Through judging the grid unit in the domain of spatial neighborhood window if that is the maximum point, then the potential center will be acquired. This spatial neighborhood discriminate algorithm only needs to handle the data set one time. So it consumes less time than the cutting potential center method especially for the condition that the number of cluster centers and the amount of data are large. Another advantage is that the computing process of the spatial neighborhood discriminate algorithm will not modify the data field itself. Our experimental results validate that our approach can find all the cluster centers automatically and its cost-time is available.

Acknowledgments. This paper is supported by National 973 (2006CB701305, 2007CB310804), National Natural Science Fund of China (60743001), Best National Thesis Fund (2005047), and Natural Science Fund of Hubei Province (CDB132).

References

1. Deren, L., Shuliang, W., Deyi, L.: Spatial Data Mining Theories and Applications. Science Press, China (2006)
2. Ester, M., et al.: A density—based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, USA (1996)
3. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: an efficient data clustering method for very large databases. In: Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data 1996, pp. 103–114 (1996)
4. Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. In: Proceedings of ACM SIGMOD International Conference on Management of Data, New York, NY, pp. 73–84 (1998)
5. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of International Conference on Very Large Databases, Santiago, Chile, pp. 487–499 (1994)
6. Fotheringham, A., Peter, R.: Spatial Analysis and GIS. Taylor and Francies, Abington (1994)
7. Tung, H., Jean, H., Jiawei, H.: Spatial clustering in the presence of obstacles. In: IEEE Transactions on Knowledge and Data Engineering, pp. 359–369 (2001)
8. Murray, T., Shyy, K.: Integrating attribute and space characteristics in choropleth display and spatial data mining. International Journal of Geographical Information Science 14(7), 649–667 (2000)
9. Usama, F., Gregory, P., Padhraic, S., Ramasamy, U.: Advances in knowledge Discovery and Data Mining. AAA/MIT Press, Menlo Park (1996)
10. Ming-Syan, C., Jiawei, H., Philip, Y.: Data mining: an overview from database perspective. IEEE Transactions on Knowledge and Data Engineering (1997)
11. Leonard, K., Peter, R.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons, Chichester (1990)

Surrounding Influenced K-Nearest Neighbors: A New Distance Based Classifier

I. Mendialdua, B. Sierra, E. Lazkano, I. Irigoien, and E. Jauregi

Department of Computer Science and Artificial Intelligence
University of the Basque Country
P. Manuel Lardizabal 1, 20018 Donostia-San Sebastian
Basque Country, Spain
{inigo.mendialdua,b.sierra,e.lazkano,itziar.irigoien,
ekaitz.jauregi}@ehu.es

Abstract. The nearest neighbor classification method assigns to an unclassified point the class of the nearest of a set of previously classified points. An extension to this approach is the K-NN method, in which the classification is made taking into account the K nearest points and classifying the unclassified point by a voting criteria from this k points. We present a new method that extends the K-NN limits, taking into account, for each neighbor, its I nearest neighbors. Experimental results are promising, obtaining better results for two class problems than the original K-NN.

Keywords: Nearest Neighbor, Supervised Classification.

1 Introduction

In supervised classification problems [1] there are two extremes of knowledge which the modeler may consider. Either (s)he may have complete statistical knowledge of the underlying joint distribution of the observation x and the category θ, or (s)he may have no knowledge of the underlying distribution except that which can be inferred from samples. In the first extreme, a standard Bayes analysis will yield an optimal decision procedure and the corresponding minimum (Bayes) probability of error classification R^*. In the other extreme, a decision to classify x into the category θ is allowed to depend only on a collection of n correct samples $(x_1, \theta_1), (x_2, \theta_2), ..., (x_n, \theta_n)$, and the decision procedure is by no means clear. This problem is in the domain of supervised classification, and no optimal classification procedure exists with respect to all underlying statistics.

If it is assumed that the classified samples (x_i, θ_i) are independently identically distributed according to the distribution of (x, θ), certain heuristic arguments may be made about good decision procedures. For example, it is reasonable to assume that observations which are close together (in some appropriate distance metric) will have almost the same posterior probability distributions on their respective classifications.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 270–277, 2010.

Thus to classify the unknown sample x we may wish to weight the evidence of the nearby x_i's most heavily. Perhaps the simplest non-parametric decision procedure of this form is the nearest neighbor (NN) classification method, which assigns to x the category of its nearest neighbor.

The first formulation of a rule of the NN type and primary previous contribution to the analysis of its properties it is presumed to have been made by Fix and Hodges [2]. They investigated a method that is known as K Nearest Neighbors (K-NN), which assigns to an unclassified point the class most heavily represented among its k nearest neighbors.

In this paper we present a modification to the K-NN method. Besides the K nearest neighbors, our method also considers their I surrounding neighbors classes. The objetive is to minimize the influence of the outliers in the final decission.

This paper is organized as follows. In section 2 we review the K-NN classification method while section 3 is devoted to Related Works in distance based classifiers; the new proposed method is introduced in section 4, in section 5 we show the experimental results obtained and in the final section 6 concluding remarks are presented..

2 The *K*-NN Classification Method

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be a correctly classified sample in classes $\theta_1, \ldots, \theta_M$, where $\mathbf{x}_i$ takes values in a metric space upon which a distance function d is defined. We will consider the pairs $(\mathbf{x}_i, \theta^i)$ where $\mathbf{x}_i$ is the p-variate observation upon the ith individual and θ^i is the class or category which that individual belongs to. We usually say that "$\mathbf{x}_i$ belongs to θ^i" when we mean precisely that the ith individual, upon which measurements $\mathbf{x}_i$ have been observed, belongs to category $\theta^i \in \{\theta_1, \ldots, \theta_M\}$.

Consider a new pair $(\mathbf{x}, \theta)$, where only the measurement $\mathbf{x}$ is observable, and where we estimate θ by using the information contained in the set of correctly classified points. We shall call

$$\mathbf{x}' \in \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \ , \tag{1}$$

the *nearest neighbor* (NN) of $\mathbf{x}$ if

$$\min_{i=1,\ldots,n} d(\mathbf{x}_i, \mathbf{x}) = d(\mathbf{x}', \mathbf{x}) \ . \tag{2}$$

The NN classification decision method gives to $\mathbf{x}$ the category θ^i, precisely the category of its nearest neighbor $\mathbf{x}_i$. In case of tie between several neighbors, it has to be broken by modifying the decision rule.

An immediate extension to this decision rule is the so called K-NN approach [3], which assigns the candidate $\mathbf{x}$ the class which is most frequently represented in the k nearest neighbors to $\mathbf{x}$.

3 Related Work

Much research has been devoted to the K-NN rule [4]. One of the most important results is that K-NN has very good asymptotic performance. Broadly speaking, for a very large design set, the expected probability of incorrect classifications (error) R achievable with K-NN is bounded as follows:

$$R^* < R < 2R^* \ , \tag{3}$$

where R^* is the optimal (minimal) error rate for the underlying distributions. This performance, however, is demonstrated for the training set size tending to infinity, and thus, it is not really applicable to real world problems in which we usually have a training set of about hundreds or thousands cases, too few for the number of probability estimations to be performed.

Some distance based approaches, such that of Weinberger et al. [5] try to increase the obtained accuracy in distance based classification by looking for a specific distance, in an automatic way, for each classification problem. The proposed approach could be used to deal with unbalanced or biased databases; a similar idea can be found in other distance based methods [6]. By the other side, *PEBLS* instance based inducer (Cost and Salzberg [7]) incorporates MVDM distance metric to deal with symbolic features, a modification of Stanfill and Waltz's VDM metric [8].

Improvement in classification can also be obtained by selecting and/or weighting features (see [9] for an example). Probabilistic voting approaches have also been used ([10], [11]); the main idea here is that each case among the K nearest ones make a weighted vote in favour of the class it belongs to, being the weight the probability each case has to belong to its own class. A very well kown approach is the so called Instance Based Learning (IBL), based on the work of Aha [12] and Wettschereck [13]; there are several versions of the algorithm [14].

By the other side, there are distance based classifiers which aim to deal with the so called multi labeling problem [15], in which, given a new case to be classified, a different number of categories could be given to it. For instance, and taking as example the document categorization area, a newspaper article relating the wedding of some country president would obtain Politics and Society as category labels, being both adequate for the document.

4 Surrounding Influenced-K-Nearest Neighbors

When the modeler has to approach the K-NN classification problem, it depends in its K nearest neighbors and it is possible that some of this may be outliers, so in this case the assigned category is not suitable. In view of this problem we think in a new solution to solve this. This solution extends the limits to further data.

Our method is similar of K-NN method, Figure 1 shows that similarity. We get the K Nearest Neighbors of the data that we want to classify, but instead of select its category we look to the I nearest neighbors of each of the K points and

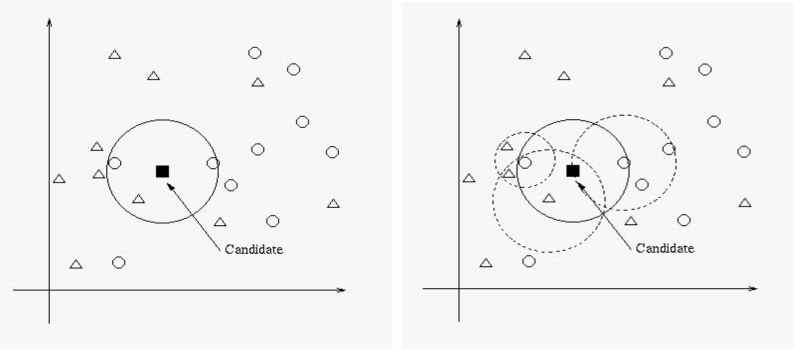


Fig. 1. Example of *SI-K*-NN algorithm where $K = 3$ and $I = 2$. First get K Nearest Neighbors and then for each K points get I Nearest Neighbors.

we take into account their categories. So we have two groups; the K-NN group, where there are the K nearest neighbors of the data that we are clasificating, and the I-NN group, where there are the I nearest neighbors of each K points. For all IxK neighbors we count the categories and the category which has the largest sum is assigned to the new data.

5 Experimental Results

5.1 Datasets

Twenty databases are used to test our hypothesis. All of them are obtained from the *UCI Machine Learning Repository* [16]. The characteristics of the databases are given in Table 1.

Table 1. Datasets used. Different characteristics of the twenty databases used in the experimental setup.

Domain	Num. of Instances	Num. of Attributes	Num. of Classes
Australian Credit	690	14	2
Balance	625	4	3
Blood	748	5	2
Breast Cancer	569	32	2
Car	1728	6	4
Glass	210	9	7
Haberman	306	3	2
Ionosphere	351	34	2
Iris	150	4	3
Letters	20000	16	26
Magic Telescope	19020	11	2
Optdigit	5620	64	10
Pendigit	10992	16	10
Pima	768	7	2
Spambase	4601	57	2
Statlog (Heart)	270	13	2
Statlog (Img Seg)	2310	19	7
Statlog (Landsat Sat)	6435	36	7
Statlog (Shuttle)	58000	9	7
Wine	178	13	3

5.2 Experiment-Setup

We have applied 5x2 fold cross-validation to each database [17]. In our experiments we have run the *SI-K*-NN with different *K* and *I* values, and we compare obtained results with the results that we obtained with *K*-NN method. In Table 2 we can see the results obtained with the *K*-NN method while Table 3 and Table 4 show the results obtained by the new proposed *SI-K*-NN method.

Table 2. Accuracy level percentage of the K-NN method for the databases using different K numbers

Datu-Basea	K = 1	K = 3	K = 5	K = 7	K = 9	Average
Australian Credit	80,927	84,231	86,028	85,681	86,202	84,614
Balance	77,820	81,474	83,974	87,179	87,756	83,641
Blood	69,679	74,010	74,171	75,454	75,294	73,721
Breast Cancer	95,633	96,760	96,901	96,830	96,901	96,605
Car	85,138	90,532	91,458	90,925	90,416	89,694
Glass	64,672	67,476	64,672	62,803	63,738	64,672
Haberman	64,705	66,405	68,366	70,457	70,849	68,156
Ionosphere	85,828	84,914	84,685	84,228	83,428	84,617
Iris	92,533	93,866	94,666	94,933	94,933	94,1864
Letters	94,338	94,276	93,966	93,598	93,126	93,8608
Magic Telescope	83,730	83,158	82,860	82,637	82,374	82,952
Opt Dig	98,398	98,476	98,370	98,163	98,085	98,298
Pen Dig	99,232	99,144	98,941	98,791	98,635	98,949
Pima	68,947	72,781	74,135	74,511	74,360	72,947
Spambase	88,852	89,060	89,139	88,939	88,426	88,883
Statlog (Heart)	76	79,851	81,333	81,481	81,481	80,029
Statlog (Img Seg)	95,844	95,168	94,233	94,129	93,991	94,673
Statlog (Landsat Sat)	89,574	90,071	89,686	89,344	89,070	89,549
Statlog (Shuttle)	99,930	99,866	99,822	99,782	99,739	99,828
Wine	94,382	95,730	96,404	96,179	96,629	95,865

In Table 5 we compared the results obtained with *K*-NN and *SI-K*-NN method. To make this comparison we have selected the average of *K*-NN and the best average between the *I* of *SI-K*-NN. As it can be seen, instead of sorting alphabetically, we have sort out the table depending on the number of classes that each database has. At first it may seem that the results are worse, but when it is sorted depending on the number of classes, it shows more interesting results. Viewed this way in multi-class problems, our method doesn't improve in any database. But in case that the class number is lower the results are better. When the number of classes is two the number of improved results is a bit more than the opposite, five better and four worse. It's the same when the number of classes is three, two better and one worse. The reason of this improvement is that our new method expands to a distant values, so the more classes there are, the more easier is to appear different classes and therefore the main class loses strength. Instead, the less classes there are, the less probability that the wrong class gains importance.

Table 3. Accuracy level percentage of the *SI*-*K*-NN method for the databases using different *K* and *I* numbers

		K = 1	K = 3	K = 5	K = 7	K = 9	Average
Australian Credit	I = 1	81,565	82,841	84,812	85,333	85,971	84,104
	I = 2	81,565	84,754	85,101	85,449	85,623	84,499
	I = 3	84,522	85,739	85,855	85,913	85,565	85,519
	I = 4	84,290	85,913	85,507	85,101	85,391	85,241
	I = 5	85,623	86,029	85,391	85,275	85,333	**85,530**
Balance	I = 1	75,962	80,769	82,949	83,910	84,551	81,628
	I = 2	75,962	83,077	84,551	85,128	85,577	82,859
	I = 3	76,218	85,000	85,705	85,705	85,897	**83,705**
	I = 4	75,833	84,872	85,385	85,385	86,090	83,513
	I = 5	77,051	84,103	85,000	85,000	85,385	83,308
Blood	I = 1	70,214	70,588	71,230	70,749	71,872	70,930
	I = 2	73,155	72,834	73,583	74,118	74,866	73,711
	I = 3	72,727	73,636	74,332	73,529	74,278	73,701
	I = 4	73,904	74,866	74,759	74,385	74,759	**74,535**
	I = 5	73,743	74,171	74,492	73,850	74,385	74,128
Breast	I = 1	93,803	95,563	95,704	95,704	96,127	95,380
	I = 2	93,803	95,634	96,056	95,845	96,268	95,521
	I = 3	93,873	95,845	96,127	95,915	96,197	95,592
	I = 4	94,577	95,845	95,915	95,915	95,845	95,620
	I = 5	95,775	95,986	96,127	95,704	95,986	**95,915**
Car	I = 1	76,204	81,505	82,778	83,241	84,144	81,574
	I = 2	82,755	83,958	84,699	85,486	85,810	**84,542**
	I = 3	79,097	83,657	84,838	85,139	85,926	83,731
	I = 4	81,713	83,912	84,884	85,255	85,486	84,250
	I = 5	80,347	83,796	84,699	85,116	85,394	83,870
Glass	I = 1	59,626	61,121	60,374	60,000	60,000	60,224
	I = 2	59,626	62,243	61,121	60,187	60,561	**60,748**
	I = 3	58,505	61,869	60,187	60,561	61,495	60,523
	I = 4	59,065	61,308	59,626	60,187	61,121	60,262
	I = 5	59,252	59,813	59,813	58,879	59,626	59,477
Haberman	I = 1	64,837	67,974	69,150	69,542	69,412	68,183
	I = 2	64,837	69,150	70,719	70,719	70,196	69,124
	I = 3	67,974	70,719	70,588	70,719	70,327	70,065
	I = 4	67,190	69,804	70,196	70,588	70,850	69,725
	I = 5	69,935	71,242	70,588	70,719	70,458	**70,588**
Ionosphere	I = 1	83,543	84,000	83,771	83,657	82,629	83,520
	I = 2	83,543	84,000	83,771	83,771	82,743	**83,566**
	I = 3	82,400	83,086	82,971	83,200	82,400	82,811
	I = 4	82,514	83,314	82,629	82,171	81,600	82,446
	I = 5	82,286	82,286	81,486	80,686	80,000	81,349
Iris	I = 1	92,533	94,400	94,933	96,000	95,733	**94,720**
	I = 2	92,533	93,867	94,933	95,200	95,467	94,400
	I = 3	92,267	94,667	95,200	95,733	95,467	94,667
	I = 4	92,533	94,400	94,933	95,733	95,200	94,560
	I = 5	92,267	94,667	94,933	95,733	95,467	94,613
Letters	I = 1	90,622	91,572	91,750	91,506	91,156	**91,321**
	I = 2	90,622	91,654	91,408	91,066	90,536	91,057
	I = 3	90,562	91,482	91,074	90,672	90,130	90,784
	I = 4	90,594	91,250	90,904	90,270	89,760	90,556
	I = 5	90,360	91,010	90,554	90,008	89,474	90,281
Magic Gamma Telescope	I = 1	81,708	82,008	81,855	81,735	81,567	**81,775**
	I = 2	81,708	81,924	81,720	81,577	81,468	81,679
	I = 3	81,708	81,886	81,685	81,499	81,367	81,629
	I = 4	81,966	81,836	81,655	81,457	81,340	81,651
	I = 5	81,794	81,718	81,493	81,335	81,184	81,505
Optical Digit	I = 1	97,900	98,221	98,100	97,922	97,915	**98,011**
	I = 2	97,900	98,157	98,078	97,865	97,829	97,966
	I = 3	97,765	98,121	98,007	97,843	97,815	97,910
	I = 4	97,786	98,121	98,000	97,786	97,737	97,886
	I = 5	97,715	98,093	97,957	97,786	97,722	97,855
Pen Dig	I = 1	98,897	98,967	98,759	98,661	98,537	**98,764**
	I = 2	98,897	98,952	98,766	98,624	98,472	98,742
	I = 3	98,810	98,876	98,675	98,574	98,428	98,672
	I = 4	98,857	98,846	98,643	98,490	98,395	98,646
	I = 5	98,712	98,715	98,530	98,428	98,341	98,545
Pima	I = 1	72,105	72,256	71,729	73,083	73,158	72,466
	I = 2	72,105	74,511	74,586	74,286	73,835	**73,865**
	I = 3	72,406	73,233	73,684	74,361	73,835	73,504
	I = 4	72,932	73,459	73,759	73,835	73,759	73,549
	I = 5	73,609	73,008	73,985	74,211	74,211	73,805

Table 4. Accuracy level percentage of the *SI-K*-NN method for the databases using different *K* and *I* numbers

		K = 1	K = 3	K = 5	K = 7	K = 9	Average
Sat Img	I = 1	86,602	87,641	87,721	87,653	87,578	87,439
	I = 2	86,602	88,225	88,032	87,914	87,684	87,692
	I = 3	87,087	88,119	87,846	87,833	87,504	87,678
	I = 4	87,293	88,287	87,858	87,709	87,529	**87,735**
	I = 5	87,386	87,883	87,684	87,566	87,324	87,569
Shuttle	I = 1	99,869	99,857	99,815	99,782	99,734	**99,811**
	I = 2	99,876	99,827	99,819	99,779	99,732	99,806
	I = 3	99,826	99,818	99,788	99,768	99,721	99,784
	I = 4	99,828	99,819	99,781	99,741	99,717	99,777
	I = 5	99,778	99,774	99,751	99,732	99,708	99,749
Spambase	I = 1	84,383	86,809	87,217	86,930	86,748	86,417
	I = 2	84,383	87,061	87,461	87,417	87,174	86,699
	I = 3	85,400	87,217	87,000	86,696	86,678	86,598
	I = 4	86,426	87,409	87,165	86,913	86,652	**86,913**
	I = 5	85,783	87,096	86,887	86,617	86,417	86,560
Statlog Heart	I = 1	74,074	74,815	77,481	78,222	80,741	77,067
	I = 2	74,074	78,370	80,148	79,407	81,333	78,667
	I = 3	78,074	80,741	80,741	81,333	82,222	80,622
	I = 4	79,556	80,889	80,741	81,778	82,815	81,156
	I = 5	81,481	80,889	81,185	82,074	82,370	**81,600**
Statlog Image Segmentation	I = 1	93,108	93,870	93,576	93,455	93,299	**93,461**
	I = 2	93,108	93,680	93,662	93,558	93,091	93,420
	I = 3	92,537	93,385	93,541	93,177	92,675	93,063
	I = 4	92,641	93,420	93,385	92,918	92,589	92,990
	I = 5	92,242	93,299	93,195	92,623	92,346	92,741
Wine	I = 1	91,011	93,933	94,607	95,730	95,730	94,202
	I = 2	91,011	93,708	94,157	95,056	95,056	93,798
	I = 3	91,685	93,933	95,056	95,730	95,281	94,337
	I = 4	92,360	95,506	95,506	95,730	95,955	**95,011**
	I = 5	91,910	95,056	95,281	95,730	96,404	94,876

Table 5. Best accuracy level percentage of the *K*-NN and *SI-K*-NN method for the databases sorting by the number of classe

Num. of Classes	Domain	K-NN	SI-K-NN
2 Classes	Australian Credit	84,614	↑ 85,530
	Blood	73,721	↑ 74,535
	Breast Cancer	96,605	↓ 95,915
	Haberman	68,156	↑ 70,588
	Ionosphere	84,617	↓ 83,566
	Magic Telescope	82,952	↓ 81,775
	Pima	72,94	↑ 73,865
	Spambase	88,883	↓ 86,913
	Statlog (Heart)	80,0296	↑ 81,600
3 Classes	Balance	83,641	↑ 83,705
	Iris	94,186	↑ 94,720
	Wine	95,865	↓ 95,011
>3 Classes	Car	89,694	↓ 84,542
	Glass	64,672	↓ 60,748
	Letters	93,860	↓ 91,321
	Opt Dig	98,298	↓ 98,011
	Pen Dig	98,949	↓ 98,764
	Statlog (Landsat Sat)	89,549	↓ 87,735
	Statlog (Shuttle)	99,828	↓ 99,811
	Statlog (Img Seg)	94,673	↓ 93,461

6 Conclusion and Further Results

A new method extending the *K*-NN idea is presented in this work. The new method, called *SI-K*-NN is created with the idea of reducing the influence of outliers in the final decision.

This new method is used in different problems, and its final results are compared with those obtained by using the K-NN method. We do not expect our new method to be better than the K-NN one in all the classification problems, but it works similar and in some cases improve the results.

As further work we are going to do new experiments in which we will consider the I and K nearest neighbors classes giving them different weights.

References

1. Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)
2. Fix, E., Hodges Jr, J.L.: Discriminatory analysis, nonparametric discrimination. Technical Report Project 21-49-004, USAF school of Aviation Medicine, Randolf field (1951)
3. Cover, T.M., Hart, P.E.: Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory 13, 21–27 (1967)
4. Dasarathy, B.V.: Nearest Neighbor (NN) Norms: NN Pattern Recognition Classification Techniques. IEEE Computer Society Press, Los Alamitos (1991)
5. Weinberger, K.Q., Saul, L.K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification. The Journal of Machine Learning Research 10, 207–244 (2009)
6. Tan, S.: Neighbor-Weighted K-Nearest Neighbor for Unbalanced Text Corpus. Expert Systems with Applications 28, 667–671 (2005)
7. Cost, S., Salzberg, S.: A weighted nearest neighbor algorithm for learning with symbolic features. Machine Learning 10, 57–78 (1993)
8. Stanfill, C., Waltz, D.: Toward Memory-Based Reasoning. Communications of the ACM 29(12), 1213–1228 (1986)
9. Tahir, M.A., Bouridane, A., Kurugollu, F.: Simultaneous Feature Selection and Feature Weighting Using Hybrid Tabu Search/K-Nearest Neighbor Classifier. Pattern Recognition Letters Archive 28, 438–446 (2007)
10. Martínez-Otzeta, J.M., Sierra, B.: Analysis of the Iterated Probabilistic Weighted k-Nearest Neighbor Method, a New Distance-Based Algorithm. 6th International Conference on Enterprise Information Systems (ICEIS), vol. 2, pp. 233–240. Porto (2004)
11. Sierra, B., Lazkano, E.: Probabilistic-Weighted k Nearest Neighbor Algorithm: a New Approach for Gene Expression Based Classification. In: Sierra, B., Lazkano, E. (eds.) Proceedings of KES 2002, pp. 932–939. IOS Press, Amsterdam (2002)
12. Aha, D.: Tolerating, Irrelevant and Novel Attributes in Instance-Based Learning Algorithms. International Journal of Man-Machine Studies 36, 267–287 (1992)
13. Wettschereck, D.: A Study of Distance-Based Machine Learning Algorithms. Ph.D. Thesis, Oregon State University (1994)
14. Aha, D., Kibler, D., Albert, M.K.: Instance-Based learning algorithms. Machine Learning 6, 37–66 (1991)
15. Younes, Z., Abdallah, F., Denux, T.: Multi-Label Classification Algorithm Derived from K-Nearest Neighbor Rule with Label Dependencies. In: Younes, Z., Abdallah, F. (eds.) 16th European Signal Processing Conference, Lausanne (2008) **Lausanne**
16. Blake, B.L., Merz, C.J.: UCI Repository of Machine Learning databases. Department of Information and Computer Science, University of California, Irvine (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
17. Demsar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. Journal of Machine Learning Research 7, 1–30 (2006)

A Centroid k-Nearest Neighbor Method

Qingjiu Zhang and Shiliang Sun

Department of Computer Science and Technology, East China Normal University,
500 Dongchuan Road, Shanghai 200241, P.R. China
qjzh08@gmail.com, slsun@cs.ecnu.edu.cn

Abstract. k-nearest neighbor method (kNN) is a very useful and easy-implementing method for real applications. The query point is estimated by its k nearest neighbors. However, this kind of prediction simply uses the label information of its neighbors without considering their space distributions. This paper proposes a novel kNN method in which the centroids instead of the neighbors themselves are employed. The centroids can reflect not only the label information but also the distribution information of its neighbors. In order to evaluate the proposed method, Euclidean distance and Mahalanobis distance is used in our experiments. Moreover, traditional kNN is also implemented to provide a comparison with the proposed method. The empirical results suggest that the propose method is more robust and effective.

Keywords: Distance metric learning, k-nearest neighbor, Euclidean distance, Mahalanobis distance.

1 Introduction

Distance metric learning is to learn a distance metric from given examples. In recent years, distance metric learning is theoretically and empirically proved to be able to significantly improve the learning performance. Considerable research has been conducted on it. From the view of availability of the training outputs, it can be divided into two categories: unsupervised and supervised distance metric learning [1]. The supervised distance metric learning can be further divided into two categories, global and local metric learning. k nearest neighbor method (kNN) is one of the famous local metric learning approaches.

Though kNN classifies the query point according the k nearest neighbor examples, it also has its assumptions. Suppose $\{x_1, x_2, ..., x_k\}$ are the k nearest neighbors of the query point x. The label of x will be assigned to the most frequent class among those neighbors. This method is theoretically and empirically supported by early researchers' work [2], [3]. Lots of new methods focused on kNN have been proposed in recent years [4], [5], [6], [7]. There are also some other algorithms that employ the labels of k nearest neighbors to weight classifiers participating in the classification [8]. However, kNN has the assumption that the class conditional probabilities of the local nearest neighbors is a constant [9], [10]. This assumption can be regarded as that the class conditional

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 278–285, 2010.

probabilities are distributed smoothly among neighbors. Moreover, different setting of the parameter k may lead to quite different results, which makes the kNN quite unstable for k.

It is clear that all the information used to identify the query point x just comes from the k nearest neighbors. We can assume that around x there exists a region in which only k neighbors are included. However, there are a variety of distributions of those k neighbors. They may be around the query point, or at a side of the query point. The traditional kNN does not reflect this information at this point. Moreover, the hypothesis is often expected by a majority voting measurement which is not reliable due to few neighbors. Consequently, the performance will be improved if the above problems are solved.

In this paper, a novel kNN method based on centroids is proposed. The distribution of k nearest neighbors is properly taken into consideration. Real training examples are not directly used to identify the query points. Instead, centroids generated by real examples with probabilistic labels are used to predict. Thus, we further suggest our new method: centroid k-nearest neighbor method. In order to evaluate the proposed method, traditional kNN is also implemented together with the proposed method on 12 classification problems.

The rest of this paper is organized as follows. In Section 2, how to create the centroids is stated in detail. The novel method and its algorithm are presented in Section 3. In Section 4, experimental results involving 12 real-world problems are reported. At last, conclusions are stated in Section 5.

2 Centroids for *k*NN

Centroids can reflect the distribution and label information of the neighbors, which can be used to further improve the performance of kNN.

2.1 *C*-Means Clustering

In classification problems, lots of methods devote themselves to enlarging the distance between classes while narrowing the distance within classes. There are lost of this kinds of algorithms used in both supervised and unsupervised learning. For instance, the Fisher linear discriminant analysis method [11] project the original space into a new space in which distance between classes is enlarged while distance within class is narrowed. Some clustering methods also cluster unlabeled examples according to the distances between class and within class [12], [13].

The c-means clustering is an unsupervised method which assigns each point to the cluster whose center (also called centroid) is the nearest [14]. The center is the average of all the points in the cluster. That is, its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster. The center is called centroid or center example in this paper. Obviously, centroids play an important role in the c-means clustering. And it can briefly reflect the local distribution information. It will help if this kind of information is taken into consideration to learn supervised problems.

2.2 Classification with Centroids

Centroids are combined with kNN to solve classification problems in this paper. If the centroid of a query point is closer to that query point, it will be more creditable to predict the query point. Therefore, in the proposed method the query example is predicted from the centroids of its neighbors instead of directly from its neighbors. The centroid is constructed by the nearest neighbors according to

$$x'_k = \frac{1}{k}\sum_{j=1}^{k} x_j, \tag{1}$$

where x'_k denotes the centroid of the k nearest neighbors of the query point x. Thus, j nearest neighbors will construct a j-th centroid. Consequently, k centroids will be constructed. Fig. 1 shows an instance of different centroids constructed by different numbers of neighbors. The figure illustrates that a smaller number of nearest neighbors may not play as good as a larger number of nearest neighbors.

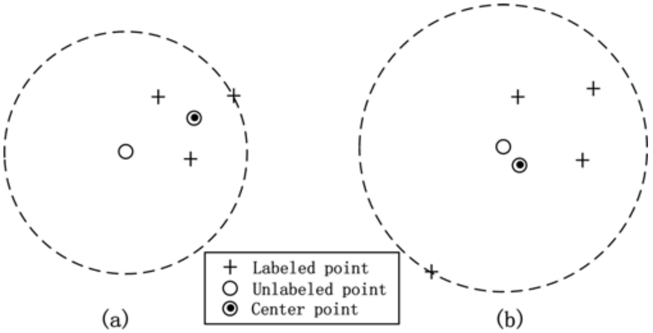
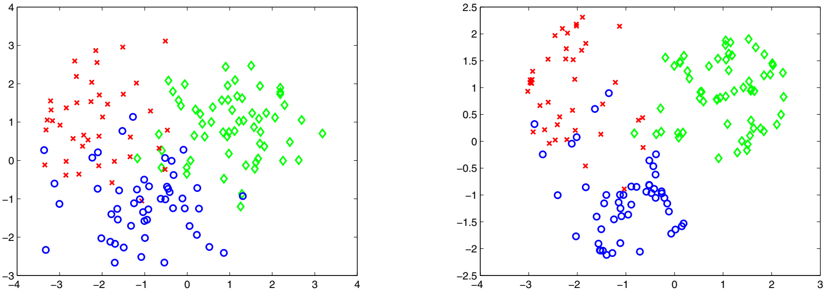


Fig. 1. (a) and (b) show the centroids of the three nearest neighbors and four nearest neighbors

Fig. 2 shows the distributions of a two dimensional toy data and its representation of the centroids. The original dataset has three classes which are drawn from a Gaussion distribution respectively. If each point is substitute by the center of nine nearest neighbor, the data distribution will appear quite differently. In the figure we can find that examples belonging to the same class are clustered together. In other words, distances between classes are enlarged while distances within class are narrowed.

If the centroids are used, two problems may arise.

- The neighbors corresponding to the nearest centroid may not belong to the same class.
- The centroids may become closer to the query point as the parameter k becoming very large.



(a) (b)

Fig. 2. (a) and (b) respectively show the distribution of the original data and the distribution of the centroids constructed by the nine nearest neighbors

To overcome this two problems, we defined a new distance d_c in which not only the distance information but also the label information are involved. The form of d_c is

$$d_c = c_1 + \alpha c_2, \tag{2}$$

where c_1 and c_2 are the distance and labeled information respectively. α is a balance parameter. c_1 and c_2 are calculated according to

$$c_1 = e^{-\frac{d^2}{\sigma^2}}, \quad \sigma = ||(d'_1, ..., d'_k)^T||_2 \tag{3}$$

and

$$c_2 = \sum_{i=1}^{t} p_i \log_t(p_i), \quad p_i = \frac{w_i + \frac{1}{kt}}{\sum\limits_{j=1}^{t} w_j + \frac{1}{k}} = \frac{ktw_i + 1}{t(k^2+1)}, \tag{4}$$

where d'_k means the real distance between the query point and the center of its k nearest neighbors. And this kind of distance measurement can be Euclidean distance and Mahalanobis distance, etc. t and w_i denote the number of classes and the number of neighbors belonging to class j respectively. c_1 is a kind of Gaussian function to calculate the confidence of the nearest neighbors, which ensures the closer neighbor has a larger value. It assumes the query point is the center of the around neighbors. Therefore, the parameter σ should be the norm of the distances between the neighbors and the query point. c_2 utilizes entropy to ensure that the centroid with more confident label have a larger value and a priority to be selected. Thus, the centroid with the largest d_c will be the most creditable one.

3 Centroid *k*NN

Based on the strategy of centroids, a centroid k-nearest neighbors method (CkNN) can be deduced. When centroids are used in kNN, k centroids will be constructed according the strategy of the above section. The label of the query point will be assigned to the label of its nearest centroid. However, its label can also be decided by the k nearest centroids by a majority voting method. That means those constructed points are implemented under a kNN again to predict the final hypotheses. Noting that the k of the second kNN is no larger than the value of k of the first kNN. Now the algorithm of the CkNN can be described in pseudocode in Table 1.

Table 1. The algorithm of CkNN

```
Given:
    Labeled training set $L$ and unlabeled test set $U$,
    The parameters $k$ and $k'$ of the first and second $k$NN,
For each example $x$ in $U$:
    Calculate the Euclidean distance from each training example to $x$,
    Find out the $k$ nearest neighbors $X=\{x_1, x_2, ...x_k\}$,
    Calculate the $k$ centroids corresponding to each group nearest neighbors,
    Calculate the components $c_1$ and $c_2$ for each centroid,
    Calculate the distance $d_c$ from $x$ to each centroid,
    Identify the $x$ using $k'$ nearest neighbors method.
```

4 Experiments

Datasets used in the experiments are from UCI [1] which is public and commonly used as benchmarks by scientists in the field of machine learning. Twelve datases involving different domains are used in this paper. The parameter k for the kNN ranges from one to nine.

In order to evaluate the proposed method, two kinds of distance measurements are launched, and they are Euclidean distance and Mahalanobis distance. For a n–dimensional problem, the Euclidean distance between two examples x and m_i is calculated according to

$$d(x, m_i) = ||x - m_i|| = \sqrt{\sum_{j=1}^{n} (x(j) - m_i(j))^2}, \tag{5}$$

where $x(j)$ denotes the j-th feature value. However, the form of Mahalanobis distance is

$$d(x, m_i) = \sqrt{(x - m_i)^T \sum\nolimits^{-1} (x - m_i)}, \tag{6}$$

where $\sum$ denotes the covariance matrix of the training set.

[1] http://archive.ics.uci.edu/ml/

The set of experiments is carefully designed. The comparison between CkNN and traditional kNN is implemented on all used datasets. Moreover, in order to clearly reflect the effectiveness of the proposed method, all the experiments are implemented under a ten fold cross-validation (CV) method. In the proposed method, there is a parameter α. Its value is also calculated under a CV method on the training set.

4.1 Euclidean Distance

Table 2 shows the classification results (in accuracy) of the experiments when Euclidean distance is calculated. It is clear that the proposed method has a smaller variance on the k results, which reflects the proposed method is more stable and robust. The CkNN also has a higher accuracy on the mean value. Moreover, when the value of k is set to one (Nearest neighbor method), we can find that the proposed method is better than the traditional method almost on all the datasets.

Table 2. The empirical results when Euclidean distance is used

D	Method	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	Mean	Var	Max
1	kNN	73.6	65.2	68.8	67.6	66.4	68.8	70.4	70.0	68.4	68.8	0.024	73.6
1	CkNN	73.6	67.2	68.4	68.0	70.0	68.0	68.4	67.6	69.6	68.9	0.019	73.6
2	kNN	71.2	67.2	67.2	66.4	65.2	64.8	61.6	62.4	60.0	65.1	0.034	71.2
2	CkNN	72.8	70.0	69.2	68.8	67.6	66.4	66.8	65.6	65.6	68.0	0.023	72.8
3	kNN	87.5	81.4	85.6	82.2	84.4	81.7	83.3	82.2	83.3	83.5	0.020	87.5
3	CkNN	88.1	82.5	84.2	83.3	84.4	82.8	83.6	82.8	83.1	83.9	0.017	88.1
4	kNN	96.0	94.7	96.0	96.0	95.3	94.7	95.3	96.0	96.7	95.6	0.007	96.7
4	CkNN	95.3	95.3	96.7	96.0	95.3	95.3	95.3	94.7	94.7	95.4	0.006	96.7
5	kNN	83.6	84.6	84.2	84.2	84.2	84.2	82.1	81.7	80.8	83.3	0.014	84.6
5	CkNN	87.9	85.4	85.0	84.2	84.6	83.8	84.6	83.3	82.5	84.6	0.015	87.9
6	kNN	60.0	50.0	68.3	68.3	61.7	60.0	58.3	51.7	46.7	58.3	0.076	68.3
6	CkNN	63.3	53.3	70.0	63.3	63.3	66.7	63.3	61.7	60.0	62.8	0.046	70.0
7	kNN	80.8	81.8	84.4	84.9	85.6	86.4	86.9	85.9	86.4	84.8	0.022	86.9
7	CkNN	82.3	82.8	85.4	83.9	85.4	84.9	87.2	86.9	86.7	85.0	0.018	87.2
8	kNN	60.0	63.3	66.7	67.4	67.4	66.7	67.4	66.3	66.7	65.8	0.025	67.4
8	CkNN	65.2	67.0	66.7	68.5	69.3	69.3	68.2	67.0	66.7	67.5	0.014	69.3
9	kNN	55.0	44.4	41.3	40.6	39.4	42.5	38.1	36.3	33.8	41.3	0.061	55.0
9	CkNN	56.9	43.1	50.0	47.5	43.8	42.5	42.5	41.3	41.9	45.5	0.051	56.9
10	kNN	90.3	92.0	91.5	91.9	93.1	92.6	92.9	92.5	93.4	92.2	0.009	93.4
10	CkNN	92.5	91.7	92.3	92.2	92.5	92.3	92.6	92.5	92.8	92.4	0.003	92.8
11	kNN	98.2	95.5	94.6	89.1	86.4	85.5	82.7	80.9	80.0	88.1	0.067	98.2
11	CkNN	98.2	95.5	97.3	96.4	93.6	89.1	89.1	89.1	86.4	92.7	0.044	98.2
12	kNN	82.7	81.2	84.0	84.0	85.2	86.7	88.5	87.8	88.8	85.4	0.027	88.8
12	CkNN	86.7	85.8	87.8	88.1	87.9	87.9	87.8	87.6	87.6	87.5	0.007	88.1

4.2 Mahalanobis Distance

Table 3 presents the results of the proposed under the Mahalanobis distance. In the 12 classification problems, we can find that CkNN has a smaller variance value over the ks, which means that CkNN palys more stably than the traditional kNN. As to the mean value, CkNN outperforms kNN on almost all the datasets. Moreover, when k=1 (Nearest neighbor method) CkNN still runs better than kNN. Although CkNN is not as good as kNN on few datasets, but it outperform kNN in the majority of all the datasets. The above results are sufficient to show that the proposed method is better than kNN.

Table 3. The empirical results when Mahalanobis distance is used

D	Method	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	Mean	Var	Max
1	kNN	93.6	86.0	93.6	92.8	94.0	91.6	93.2	92.8	93.6	92.4	0.025	94.0
1	CkNN	93.2	88.0	95.6	95.6	96.4	94.4	94.4	94.0	94.4	94.0	0.025	96.4
2	kNN	67.2	64.4	69.2	67.2	64.8	64.4	62.4	62.8	62.4	65.0	0.024	69.2
2	CkNN	70.8	66.4	70.0	70.4	70.0	67.2	66.4	66.8	66.0	68.2	0.020	70.8
3	kNN	63.9	63.9	63.9	63.9	63.9	63.9	63.9	63.9	63.9	63.9	0.000	63.9
3	CkNN	63.9	63.9	63.9	63.9	639	63.9	63.9	63.9	63.9	63.9	0.000	63.9
4	kNN	92.0	90.7	91.3	90.0	90.7	88.7	88.0	86.7	87.3	89.5	0.019	92.0
4	CkNN	92.0	92.0	93.3	92.0	91.3	90.0	89.3	88.7	88.0	90.7	0.018	93.3
5	kNN	90.4	92.1	89.6	86.7	87.1	88.3	87.9	84.2	85.4	88.0	0.025	92.1
5	CkNN	93.3	92.1	91.3	90.8	90.0	89.2	88.8	88.3	88.3	90.2	0.018	93.3
6	kNN	83.3	60.0	68.3	66.7	63.3	63.3	61.7	60.0	53.3	64.4	0.083	83.3
6	CkNN	80.0	60.0	75.0	73.3	70.0	70.0	70.0	70.0	66.7	70.6	0.055	80.0
7	kNN	75.1	76.2	79.5	80.0	82.1	79.5	80.8	80.0	81.8	79.4	0.024	82.1
7	CkNN	78.7	77.7	79.2	79.2	80.8	81.5	81.8	81.5	82.1	80.3	0.016	82.1
8	kNN	74.1	72.6	79.6	78.5	80.7	79.6	82.2	78.9	79.6	78.4	0.031	82.2
8	CkNN	75.2	74.1	77.4	75.9	80.0	80.0	79.6	80.7	80.4	78.2	0.025	80.7
9	kNN	61.3	52.5	45.6	50.0	51.3	50.0	51.9	52.5	53.1	52.0	0.041	61.3
9	CkNN	63.8	53.1	51.9	52.5	51.9	53.8	53.1	51.3	51.9	53.7	0.039	63.8
10	kNN	83.4	78.0	82.0	78.6	81.5	76.9	80.6	76.2	79.1	79.6	0.024	83.4
10	CkNN	83.7	78.0	79.4	78.8	80.5	79.5	80.2	79.4	79.5	79.9	0.016	83.7
11	kNN	94.6	90.0	91.8	87.3	84.6	80.0	76.4	75.5	75.5	83.9	0.074	94.6
11	CkNN	94.6	88.2	90.9	90.0	90.0	86.4	87.3	87.3	87.3	89.1	0.026	94.6
12	kNN	78.2	77.8	83.3	82.7	86.4	87.0	88.2	89.1	89.4	84.7	0.045	89.4
12	CkNN	85.1	82.4	86.9	86.0	87.6	87.3	88.2	88.2	88.1	86.6	0.019	88.2

5 Conclusions

This paper proposed a new kNN. The real data are not directly used to calculate the distance. Instead, they are substituted by the centroids of the nearest neighbors. The empirical results illustrates that the proposed method can significantly improve the effectiveness of kNN.

Further investigations on centroids are possible. In this paper, centroids are used to replace real neighbors to solve the classification problems. From the results we can conclude that centroids could afford as sufficient information as the real points for the problems. Consequently, all the examples (including training dataset and test dataset) replaced by their centroids may be regarded as an additional view of the original dataset. If the additional view could provide complementary information with the original view, centroids will become a view-creating method for the multiple-view learning problems.

Acknowledgements. This work is supported by the National Natural Science Foundation of China under Projects 60703005 and 61075005.

References

1. Yang, L.: Distance Metric Learning: A Comprehensive Survey. Department of Computer Science and Engineering. Michigan State University (2006)
2. Cover, T., Hart, P.: Nearest Neighbor Pattern Classification. IEEE Transaction on Information Theory, 21–27 (1967)
3. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. Wiley, New York (1973)
4. Achtert, E., Kriegel, H.P., Kröger, P., Renz, M., Züfle, A.: Reverse k-Nearest Neighbor Search in Dynamic and Deneral Metric Databases. In: Proceedings of the 12th International Conference on Extending Database Technology, pp. 886–897 (2009)
5. Sun, S., Huang, R.: An Adaptive k-nearest Neighbor Algorithm. In: Proceedings of the 7th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 91–94 (2010)
6. Weinberger, K.Q., Saul, L.K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification. Journal of Machine Learning Research 10, 207–244 (2009)
7. Jin, C., Guo, W.: Efficiently Monitoring Nearest Neighbors to a Moving Object. In: Proceedings of International Conference on Advanced Data Mining and Applications, pp. 239–251 (2007)
8. Sun, S.: Local Within-class Accuracies for Weighting Individual Outputs in Multiple Classifier Systems. Pattern Recognition Letters 31(2), 119–124 (2010)
9. Hastie, T., Tibshirani, R.: Discriminant Adaptive Nearest Neighbor Classification. IEEE Pattern Analysis and Machine Intelligence 18(6) (1996)
10. Friedman, J.: Flexible Metric Nearest Neighbor Classification. Stanford University Statistics Department, Tech. Rep. (1994)
11. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley and Sons, New York (2001)
12. Ng, A.Y., Jordan, M.I., Weiss, Y.: On Spectral Clustering: Analysis and An Algorithm. In: Advances in Neural Information Processing Systems (2001)
13. MacKay, D.J.C.: Information Theory, Inference and Learning Algorithms. Cambridge University Press, Cambridge (2003)
14. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An Efficient K-means Clustering Algorithm: Analysis and implementation. IEEE Transaction on Pattern Analysis and Machine Intelligence 24, 881–892 (2002)

Mining Spatial Association Rules with Multi-relational Approach

Min Qian[1,2], Li-Jie Pu[1,*], Rong Fu[1], and Ming Zhu[1]

[1] School of Geographic and Oceangrahphic Sciences, Nanjing University,
210093, Nanjing, China
ljpu@nju.edu.cn
[2] School of Resource and Environment, Guizhou University,
550003, Guiyang, China
moneyminn@gmail.com

Abstract. Working on spatial data models for geographic phenomena have always been viewed from a spatial context and emphasizing spatial change. But the problem is that spatial relationships are embedded in space, unknown a priori. To achieve such issue, spatial association rules mining techniques are needed. In this paper, we propose a multi-relational mining method to deal with it. We use a non-parametric way by using Vironoi-diagram based neighborhood, classification method is implemented to pre-process the rules condition, association rules are pre-defined, and a close Apriori-base algorithm is proposed to cope with it. Then the framework is evaluated by the real-world dataset, and some thoughtful association rules are given.

Keywords: spatial association rules mining, multi-relational approach, Vironoi-diagram, neighborhood, Apriori-base algorithm.

1 Introduction

Land use and land cover change is hot issue in the world, researchers wonder which is the main factor makes the land changed and which type the land will be changed to? Lots of the work have been reported and explain it in large scale [1,2], results give some useful interpretations about the land use change and its driving forces in the rapidly developing regions of China. But reasoning from the famous Tobler's First Law of Geography [3], which states the relationship between the result and the dependency is that the larger distance among the entities, the more negligible their effects on each other. So if we will test the same question the small scale, for example, we are interested in a county why a piece of cultivate land will be changed into construction land, the result might be very different from what we have known. Koperski and Han [4] introduce spatial association rules to describe associations between objects based on spatial neighborhood relations can solve this problem. In term of the spatial characteristics, such as location, the nature of neighboring land use and the proximity of other spatial entities (roads, airports, etc.).

[*] Corresponding author.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 286–293, 2010.

Although spatial relationships are the main characteristics to be considered in spatial data mining, but not necessarily all spatial relationships hold interesting information. In this paper, we focus on Data Mining in the spatial data which we are interested in. That is, association rules involving spatial relations among spatial objects and building on a multi-relational (MR) data mining approach. For the different definition between the MR database and spatial data, MR techniques can only give the spatial location of the entities themselves, however, they do not involve information regarding neighboring entities, and these relationships need to be made more explicit. So we propose a pre-processing technique which uses Voronoi-diagram to setup neighborhood and to label each entity in the zone, and then based on Apriori algorithm, we propose a Apriori-like algorithm named Label_Apriori_gen to reduce the number of candidate itemsets by pruning the non-class relevant itemsets as early as possible, thus significantly reduce the computational complexity of frequent itemsets generation.

1.1 Related Work

In the literature several approaches to discovering spatial association rules, since Keperski and Han proposed a top-down method [4], the multi-level technique has been widely used in spatial association rule mining. This method implements progressive deepening search technique, from the high concept levels for large patterns to the lower ones until there are no large patterns. Approximate spatial computation algorithms, such as R-tree or plane-sweep techniques are used to operating on minimum bounding rectangles. A successful application has been reported is applied in exploring the spatial and the temporal relationships to urban growth in Denver in USA which improves efficiency [5]. A MR data mining approach is proposed to solve the limitations of the restrictive data representation formalism known as a single table of a relational database, moreover, it develops in the field of Inductive Logic Programming (ILP) and has proven effective with high accuracy in multi-relational spatial association rule mining [6]. Another important method is mining collocation pattern which is introduced by Huang and Shekhar in [7], their co-location technique allows for multiple definitions of neighborhood relations, the neighborhood of the relation may be defined using topological relationships (e.g. connected, adjacent), metric relationships (e.g. Euclidean distance) or a combination. Such definition of neighborhood incurs substantial computational cost when one wants to enumerate all neighborhoods.

1.2 Contributions

The work listed above is little different from ours. In our work, we explore a MR approach to mining spatial association rules. It defines a neighborhood relationship between entities via non-parametric Voronoi-diagram approach which filters out irrelevant relationships. This method extends the traditional theory of spatial association rule mining with implementation MR technique. Then we experiment it on real-world data.

The remainder of this paper is organized as follows: Section 2 defines the neighborhood, rules, and algorithm pertaining to finding spatial association rules. Experiments and results on this method are described in Section 3. And we conclude in Section 4 with a summary and direction for the future work.

2 Definitions

2.1 Neighborhood Definition

Since spatial data only indirectly implies relationship via the spatial location of the entities, some works have to be done explicitly determine whether entities are related, definition of neighborhood, which is always the basis for spatial calculations seem to be important. There are two types of the state-of-the-art, namely, topological relationships [8,9] and buffer zones [10]. Topological relationships are two planar regions without holes consisted of eight predicates to describe the intersections of their boundary, interior and exterior, such as disjoint, meet, overlap, etc. They are familiar to people and easy to understand. But in the real-world, it would be no use while people define themselves as living in the neighborhood of factory instead of adjacent to it even if they lived in the close vicinity. Buffer zones are useful method for finding the neighborhood of an entity. They have been widely used in many events such as flood control region estimation, wild animal protection area planning, etc. Generally the range of the buffer zones is a circle based on the specified centre and radius. Unfortunately, buffer zones do have major drawbacks. The entity types in the range of the zone can not be selected, that is to say maybe too large or irrelevant entities are involved in and change the resulting rules. So a neighborhood definition which can take into account the number of entities and their distribution without parameter is needed, and here Voronoi-diagram is adopt. Voronoi-diagrams partition a plane into regions, called Voronoi cells, which contain the area that is closest to the entity contained in the Voronoi cell, naturally representing relationships between entities [11]. In order to create different characters and meaningful neighborhood relationships between multiple types of entities in spatial data, we construct a different neighborhood structure using Voronoi-diagrams and defined as follows:

Definition 1. A Voronoi neighborhood If two entities $m^{t,q}$ and $n^{r,s}$ are neighborhood, they will:

- Different entity type: $t \cap r \neq \phi$ i.e.: $t ? r$;
- Same entity type: $t \cap r \neq 1$ i.e.: $t = r$.

This definition eliminates irrelevant entities and proposes the proximity for each entity type is derived from the distribution and number of entities of that entity type.

2.2 Rules Definition

Let D be the spatial dataset, and $S = \{s_1, s_2, ..., s_l\}$ be the set of spatial attributes, $A = \{a_1, a_2, ..., a_m\}$ be the set of non-spatial attributes, and $CL = \{cl_1, cl_2, ...cl_n\}$ be the set of class labels.

Let $I = S \cup A \cup CL = \{s_1, s_2, ...s_l, a_1, a_2, ...a_m, cl_1, cl_2, ...cl_n\}$ be the set of all items in D. Continuous attributes are transformed into categorical attributes, then all the categorical attributes are transformed into binary attributes. Let $T = \{t_1, t_2, ...t_n\}$ be the set of the relational table, which contains N tuples following the schema I (I contains l+m+n number of items). Thus an items $i \in I$ is a binary variable whose value is 1 if the item is present in t_i (i=1,…, N) and 0 for otherwise.

Definition 2. An association rule r is of the form $X \rightarrow Y$, where $X \subseteq I, Y \subseteq I$, and $(X \cup Y) \cap CL \neq \phi$.

– The rule r holds in the D with support(sup) and confidence(con) where

$$\sup(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

$$con(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

– The support count

$$\sigma(\alpha) = |t_i| \alpha \subseteq t_i, t_i \in T \mid$$

An association rule is strong if it satisfies user-specified minimum support (min_support) and minimum confidence (min_confidence) thresholds.

Definition 3. A classification rule is of the form $L_0 ? L_1 \wedge L_2 \wedge ... \wedge L_n$. Where each L_i is a literal, and L_0 specifies a class-lable. The literals are implicitly existentially quantified. As an example:

Rule 1: alterable (C,'yes')? cultivate (C), neighbour (C, F), factory (F), value (F, V), V>7,000,000

States that "a cultivate land is alterable if there exist a neighboring factory with output value greater than 7,000,000 yuan/year". Here F for factory is implicitly existentially quantified. Cultivate land (C) is entity table land use, neighbor denotes a relationship between cultivate land and factory, and output value is a feature of factory. But it is likely more than one factory with the output value greater than 7,000,000, so we need to explicit the problem, and set up multi-relationship. The method of Chelghoum et.al [12] is implemented to pre-process the MR table in summarizing the relationships between entity t_i and t_j. And the example of the table is list below. Table 1 shows the object which we are interested in their relationships. Table 2 shows the spatial distance between the objects and the targets. Table 3 shows the target objects. And the Table 4 is the complete relation table which implements aggregation method to evaluate the maximum distance between objects and targets.

Table 1. Table of Object O

ObjectID	Shape	Type
001	Polygon	cultivate
002	Polygon	grass

Table 2. Table of Association A

ObjectID	Distance	ID1
001	20,000	1
001	10,000	2
001	40,000	3
002	1000	3
002	800	2

Table 3. Table of Target T

ID1	Type
1	Factory
2	House
3	Facory

Table 4. Table of Completion C (O, T, A, {max})

ObjectID	Shape	Type	Distance_Fac	Distance_Hou
001	Polygon	Cultivate	40,000	10,000
002	Polygon	Grass	1000	800

2.3 Algorithm

After the specific relationships materialized, we extend Apriori algorithm called Label_Apriori_Gen on transactions by utilizing a give class structure. This structure is used to get the class-label of training entities, then use those rules to predict the label for new testing entities, after all entities are labeled, the same algorithm can be used to generate association rules. There are pseudocode which illustrate the algorithm in C. Label_Apriori_Gen deals with candidate 1- and 2-itemsets, which is based on the K-itemsets generation (k>2). First, the algorithm constructs candidate 1-itemsets from frequent 1-dataset (step 4-7). Second, to generate candidate 2-itemsets that focus on class labels, the algorithm use subset function to divide the class-label items from other items (step 8). Third, the algorithm enumerates class-label items within the the class-label items and other items (step 9-13). Then pruning the non-class-label items and generate candidate 2-itemsets formed between class labels (step 14-19). Last, step 20 generates the frequent itemset.

Algorithm 1. Candidate Generation and Pruning: Label_Apriori_Gen

```
1. input: transaction dataset D, class-label dataset CL
2. output: frequent itemset $C_k$
3. procedure
4. $L_1$=find_frequent_1_itemset(D);
5. for (k=2; $L_{k-1} \neq \phi$ ; k++) {
```

```
for frequent 1-itemset $f \in L_1$;
C$_1$←f;
(C$_{1\_label}$, C$_{1\_other}$)= subset(C$_1$,CL);
for candidate itemset $c_1 \in C_{1\_label}$ {
c$_1$.count++;
for candidate itemset $c_2 \in C_{1\_other}$
$c = c_1 \cup c_2$;
C$_2$← c; }
for(i = 0, j = length(L$_1$); i < count; i++) {
a=*C$_2$;
b=*L$_1$;
if (a[i]=b[i]∩a[j]≠b[j])
$c = a[0] \cup b[j]$;
C$_2$← c; }
return C$_k$; }
```

3 Experiments

We applied the MR technique to the real-world database, Yixing city, the prefecture-level city of Wuxi, China, which economy is the one of the most increasing rapidly city. The purpose of the experiments is to find out whether the lost of prime farmland is related to the surroundings especially the factory, main road, airport, or the population center. In our experiments, the current land use map from 2000-2005 was available with the location of the main roads and some factories in Yixing city, also available was the same period GDP distribution in the city. We implemented GIS to pre-process the relationship as a transaction in our experiment. Raw attributes were transformed to create binary attributes including spatial attributes, non-spatial attributes, and class labels. We chose 0.1 for min_support and 0.7 for min_confidence for global rule mining, found that low support (24.6%) for global land use change when there was factories output value over 2,000,000 yuan/year.

Rule 2: alterable (C,'yes')? cultivate (C), neighbor (C, F), factory(F), value (F, V), V>3,000,000 (24%, 70%)

Then we focus on obviously land use change regions, and found some interesting rule in these regions (Figure 1). The rule generated from region 1 is:

Rule 3: alterable (C,'yes')? cultivate (C), neighbor (C, F), factory(F), value (F, V), V>2,000,000 (53%, 86%)

And a rule extracted from region 2 is:

Rule 4: alterable (C,'yes')? cultivate (C), neighbor (C, F), factory(F), value (F, V), V>3,000,000 (70%, 87%)

Our experiments show that maybe fixed assets are not the major factors which induce the cultivate land use change in Yixing city, there are some other factors can not be discovered yet.

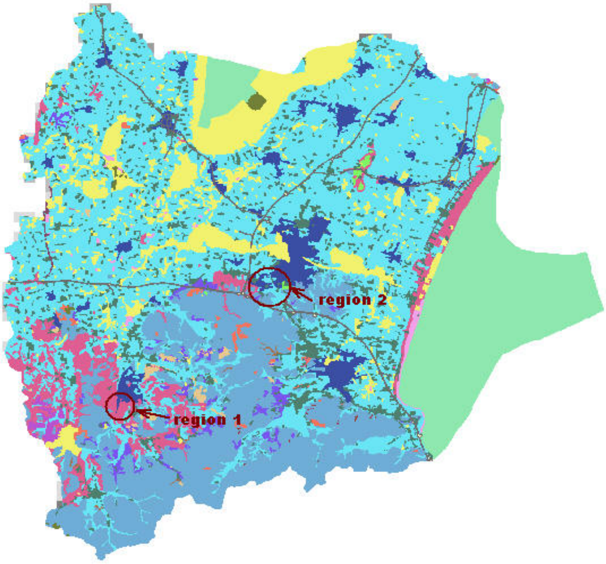


Fig. 1. Yixing land use change result and interesting regions

4 Conclusion and Ongoing Work

This paper presents a multi-relational method in mining spatial association rules which could transact the dataset in relational way instead of multi-level technique. And we introduce a pre-processing method to filter the transaction data using Voronoi-diagram based neighborhood definition. To generate association rule in spatial dataset, we propose a Label-Aprioi-gen method and illustrate algorithm. Then the frame work is test by real-world dataset to find the factors which change the cultivate land to another use.

Our future work will carry on this technique, optimizing its algorithm, maybe in parallelization way. Secondly, more spatial entities will be included in the next experiment to find out their affection on land use change. Last, our approach needs to be evaluated on larger spatio-temporal dataset, that is to say, we may not content with what the relationship between the neighborhoods but also when the land changed from cultivate to another.

Acknowledgements. This study was supported by the Natural Science Foundation of China (Grant no.40871255), the Projects in the National Science & Technology Pillar Program during the Eleventh five-year (Grant no.2006BAJ05A02 and no.2006BAJ05A08).

References

1. Zhang, J., Chen, F., Pu, L.J., Han, S.C., Ye, H., Peng, B.Z.: Analyzing Land Use Change and Its Driving Forces in SXC Region over the Past 20 Years. R. Sc. 29, 61–69 (2007)
2. Lu, R.C., Huang, X.J., Zuo, T.H., Xiao, S.S., Zhang, X.Y., Zhao, X.F.: Construction of Land Use Information Tupu in the Rapidly Developing Regions: A Case Study of the Surrounding Regions of Taihu Lake in Jiangsu Province. R. Sc. 31, 1133–1141 (2009)
3. Tobler, W.R.: A Computer Movie Simulating Urban Growth in the Detroit Region. E. Ge. 46, 234–240 (1970)
4. Koperski, K., Han, J.W.: Discovery of Spatial Association Rules in Geographic Information Databases. In: Egenhofer, M.J., Herring, J.R. (eds.) SSD 1995. LNCS, vol. 951, pp. 47–66. Springer, Heidelberg (1995)
5. Mennis, J., Liu, J.W.: Mining Association Rules in Spatio-Temporal Data: An Analysis of Urban Socioeconomic and Land Cover Change. T. GIS. 9, 5–17 (2005)
6. Agrawal, R., Imielinsk, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: ACM SIGMOD International Conference on Management of Data, pp. 207–216. ACM Press, Washington (1993)
7. Huang, Y., Shekhar, S., Xiong, H.: Discovering Colocation Patterns from Spatial Data Sets: A general approach. I. Tr. Know. Data. Enginee. 16, 1472–1485 (2004)
8. Egenhofer, M.J.: Deriving the Composition of Binary Topological Relations. J. Vis. Lang. Comput. 5, 133–149 (1994)
9. Santos, M.Y., Amaral, L.A.: Geo-spatial Data Mining in the Analysis of a Demographic Database. S. Co. 9, 374–384 (2005)
10. Appice, A., Ceci, M., Lanza, A., Lisi, F., Malerba, D.: Discovery of Spatial Association Rules in Geo-referenced Census Data: A Relational Mining Approach. I. Da. Ana. 7, 541–566 (2003)
11. Benbenik, R., Rybiński, H.: Mining Spatial Association Rules with No Distance Parameter. A. Sof. Comp. 5, 499–508 (2006)
12. Chelghoum, N., Zeitouni, K.: Spatial Data Mining Implementation—Alternatives and Performances. In: GeoInfo 2004, Brazilian Imopsium on GeoInformatics, Brazil, pp. 127–153 (2004)

An Unsupervised Classification Method of Remote Sensing Images Based on Ant Colony Optimization Algorithm

Duo Wang[1,2] and Bo Cheng[1]

[1] Center for Earth Observation and Digital Earth Chinese Academy of Sciences, China
[2] Graduate University of Chinese Academy of Sciences (GUCAS), China, 100086
somegreenhand@163.com, bocheng@ceode.ac.cn

Abstract. Remote sensing images classification method can be divided into supervised classification and unsupervised classification according to whether there is prior knowledge. Supervised classification is a machine learning procedure for deducing a function from training data; unsupervised classification is a kind of classification which no training sample is available and subdivision of the feature space is achieved by identifying natural groupings present in the images values. As a branch of swarm intelligence, ant colony optimization algorithm has self-organization, adaptation, positive feedback and other intelligent advantages. In this paper, ant colony optimization algorithm is tentatively introduced into unsupervised classification of remote sensing images. A series of experiments are performed with remote sensing data. Comparing with the K-mean and the ISODATA clustering algorithm, the experiment result proves that artificial ant colony optimization algorithm provides a more effective approach to remote sensing images classification.

Keywords: unsupervised classification, pheromone, data discretization, ant colony optimization algorithm.

1 Introduction

Remote sensing images classification has always been a significant content of remote sensing research area [1]. It can be divided into supervised classification and unsupervised classification. But most of the time priori knowledge is difficult to obtain due to the influence of complex background, various character, and images noise. Hence, unsupervised classification is of great value in remote sensing images processing. Cluster analysis is the primary method of unsupervised classification [2].

The first Ant colony optimization (ACO) algorithm, proposed by Italian scholar M.Dorigo, has been applied successfully to solve different optimization problems, such as the traveling salesman problem(TSP)[3] and quadratic assignment problem (QAP)[4][5]. It is based on the cooperative behavior of real ant colonies, which are able to find the shortest path from a food source to their nest, shown in Fig.1. In essence, ACO is an evolutionary approach where several generations of artificial ants search for good solutions. Every ant of a generation builds up a solution step by step

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 294–301, 2010.

going through several probabilistic decisions until a solution is found [6]. Ants found a good solution mark their paths through the decision space by putting some amount of pheromone on the edges of the path. The following ants of the next generation are attracted by the pheromone [7]. After a transitory period, the shorter paths will be more frequently visited and pheromone will accumulate faster on them. This in turn causes more ants to select these paths. On the contrary, pheromone continuously evaporates as time, so pheromone on the long paths will become less and less. This positive feedback effect eventually causes that all the ants will potentially use the shortest path.

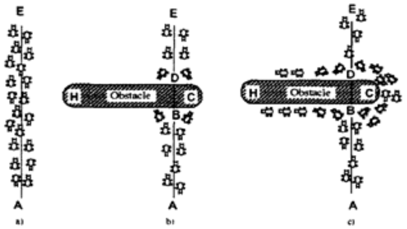


Fig. 1. Characteristic of ant finding food

Wu Zhonglong [8] has discussed the mining classification rule based on ant colony algorithm. Wang Shugen[9] studied automatic classification of remote sensing images based on artificial ant colony algorithm. Han yanfang[10] discussed images segmentation based on improved ant colony algorithm.

2 Method

Based on the characteristics of the remote sensing images, the ACO classification method can be mainly divided into two stages. First of all, a discrete process is needed for each band of remote sensing images. Secondly, spread ants all over the images, and construct rules to let the ants search a whole path automatically.

2.1 Data Discretization

Many classifiers can only handle discrete data, such as decision trees, rough sets and so on. For remote sensing data mining, discretization of continuous data is a necessary step of classification rules extraction.

Although each band value of remote sensing data is discrete value from 0 to 255, too many regions in each band will affect the classification's efficiency and quality. Therefore, using discrete technology to partition the data into limited regions is needed. Whether the discretization is proper involves the classification's quality. In this paper, we introduce the concept of entropy for the discretization of remote sensing data.

For ease of expression, we define the decision table as follows: a decision table is composed by a four- tuple (U, R, V, f). In the four-tuple, U is a collection of objects, which is the field we discuss. $R=C\cup D$. C and D represent condition attribute set and decision attribute set respectively. V is the collection constructed by the range set. F is the information function.

We assume X as the training set with |X| entities. $X \subseteq U$. The number of entities whose decision attribute equals j is k_j. In that way, entropy of the training set can be calculated as follows:

$$H(X) = -\sum_{j=1}^{r} p_j \log_2 P_j, \quad P_j = \frac{k_j}{|X|} \tag{1}$$

Less the entropy is, more dominant one or two decision attribute is, and less chaos the set X is. If we insert a point c_i^a into set X, for the entities whose decision attribute equals j, we define the number of entities, who belong to X and whose attribute value is less than c_i^a, as $l_j^X(c_i^a)$, and the number of entities whose attribute value is more than c_i^a as $r_j^X(c_i^a)$. So c_i^a can divide set X into two subsets, X_l and X_r, whose entropy are as follows:

$$H(X_l) = -\sum_{j=1}^{r} p_j \log_2 P_j \quad P_j = \frac{l_j^X(c_i^a)}{l^X(c_i^a)}, \quad l^X(c_i^a) = \sum_j^r l_j^X(c_i^a) \tag{2}$$

$$H(X_r) = -\sum_{j=1}^{r} q_j \log_2 q_j \quad P_j = \frac{r_j^X(c_i^a)}{r^X(c_i^a)}, \quad r^X(c_i^a) = \sum_j^r r_j^X(c_i^a) \tag{3}$$

We redefine the entropy after we introduce the point c_i^a :

$$H^X(c_i^a) = \frac{|X_l|}{|U|} H(X_l) + \frac{|X_r|}{|U|} H(X_r) \tag{4}$$

We assume that $L = \{Y_1, Y_2, ..., Y_m\}$ are the equivalent classes separated by set P which is composed by the chosen points. Then if we insert a new point c, the entropy will be changed as:

$$H(c,L) = H^{Y_1}(c) + H^{Y2}(c) + ... + H^{Ym}(c) \tag{5}$$

Less $H(c,L)$ is, more significant the point c is. Because point c makes the new classes' decision attribute tend to be unitary. We assume that P is the set composed by the points which have been chosen. L is the equivalent class set separated by P. B is backup point set. So the entropy discretization algorithm can be described as follows:

(1) Calculate $H(c,L)$ for each point c that belongs to set B.

(2) If $H \le \min H(c,L)$, end the process.

(3) Add point $c_{\min}$ which makes $H(c,L)$ least to set P.

(4) For all the X belongs to set L, if point $c_{\min}$ divides X into X_1 and X_2, remove X from set L and add X_1 and X_2 into L.

(5) If entities in the equivalent classes of L have the same decision, end the circle, otherwise go back to step (1).

2.2 Rule Construction

Rule construction imitates the behavior of ants to find food. Ants repeatedly choose nodes until they construct a complete path. If the nodes are completely random, it will cost a long time. In order to shorten constringency time, we introduce a rule for ants to search path.

The detailed steps are as follows:

(1) Initialization: random scatter n classes, m ants over the input images. Each class of ants only searches their own pixels. N is the number of equivalent classes we have sorted out before.

(2) Laying pheromone on the path: the ant located at cell k, first explores cell k obeying the rules of this kind of equivalent classes, then searches the adjacent 8 cells for pheromone following ant searching rules. When pheromone is found, the ant reaches the pixel by this path, explores the pixel, and lays pheromone on this path.

(3) Ant searching rules: one of the coefficients associated with ant is $W(\sigma)$. It is related to pheromone concentration σ and is expressed as:

$$W(\sigma) = (1 + \frac{\sigma}{1+\sigma\delta})^{\beta} \tag{6}$$

Here $1/\sigma$ is pheromone sensor capacity of ant; β is coefficient describing how strong ant is attracted by pheromone.

The other coefficient is the ant movement direction coefficient $w(\Delta_\theta)$. There is an angle of ant's deflection from its earlier direction of travel (45° multiple). The value of $w(\Delta_\theta)$ can be checked out in Fig. 2. In the figure it is assumed that earlier direction of ant travel was north.

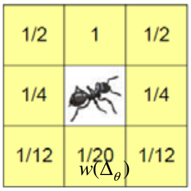


Fig. 2. Ant movement direction coefficient

So the probability for an ant to move to surrounding cell i from cell k is:

$$P_{ik} = \frac{W(\sigma_i)w(\Delta_\theta)}{\sum_{j=1}^{k} W(\sigma_i)w(\Delta_\theta)} \tag{7}$$

(4) Pheromone updating: at the end of each iteration pheromone amount left by all ants is evaporated. The pheromone updating formula is as follows:

$$\tau_{ij}(t+1) = (1-\rho)\bullet\tau_{ij}(t) + (\frac{Q}{1+Q})\bullet\tau_{ij}(t) \tag{8}$$

ρ is the evaporate coefficient of pheromone, Q is the quality coefficient of classification rules.

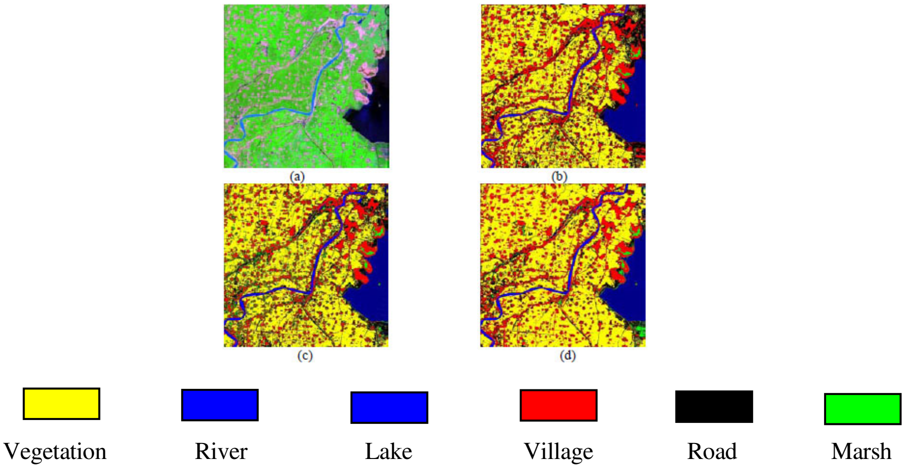
(5) Pheromone accumulation: ants always follow the path with more pheromone. The follow behavior of a great deal ants can make pheromone accumulate with time and at last distinguish their own class from other classes.

3 Experiments and Results

Experiments have been carried on by taking TM images as examples. The range we choose from the TM images contains 1024 x 1025 cells. The fake color images composed by band 5, 4, 3 of TM images is shown in Fig.3-a.

After the data discretization process, the images can be approximately divided into 6 classes, village, lake, river, road, vegetation and marsh. Each class has their own construction rules. So we spread 6 colonies of ants over the images for further classification.

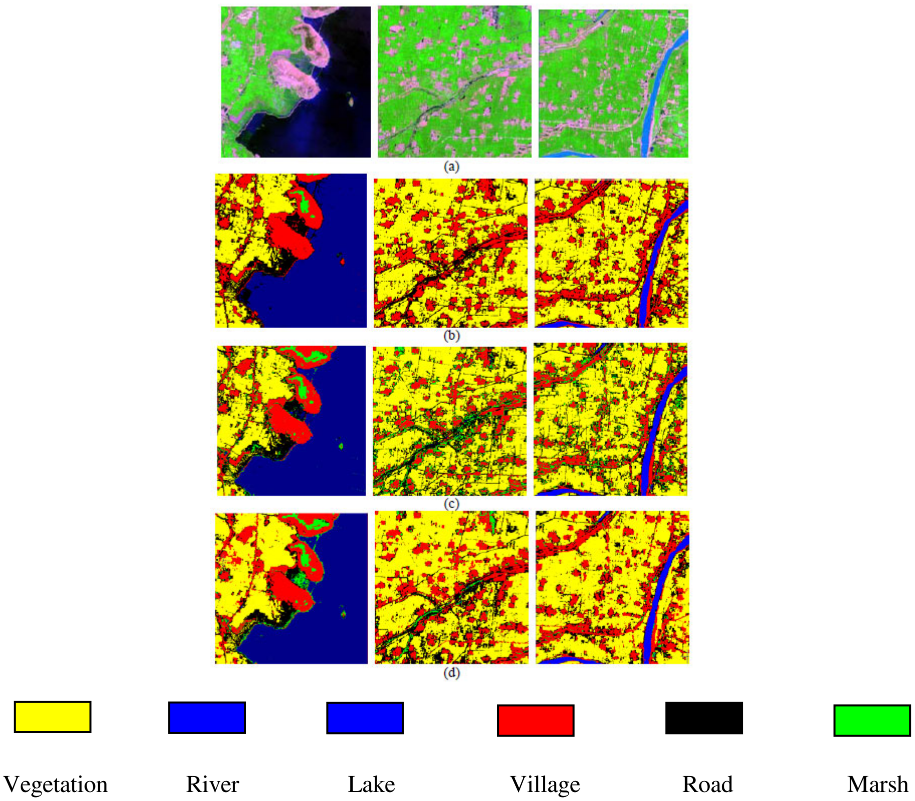
The two most frequently used algorithms in unsupervised classification are the K-mean and the ISODATA clustering algorithm. Fig.3-b, c, d respectively shows the classification results of K-mean, ISODATA, and ACO clustering algorithm.



(a) Original TM I mages(5, 4, 3) (b) ACO algorithm
(c) ISODATA algorithm (d) K-mean algorithm

Fig. 3. Comparison of three unsupervised classification methods

In order to compare results of the three methods clearly, we partially zoom in the images, shown in Fig.4. Through comparing the same place between different methods, we can find that: The effect of ACO algorithm is better than the other two with less mistaken classification.



(a) Original TM images(5, 4, 3) (b) ACO algorithm
(c) ISODATA algorithm (d) K-mean algorithm

Fig. 4. Comparison of partially magnified results

In order to compare the precision of the 3 methods, we use the data set which has been validated before to test precision. The precision evaluated results are show in Table 2, 3, 4, in the way of confusion matrix.

Table 1. Confusion matrix of ACO algorithm

Class	village	vegetation	lake	river	road	marsh	total	UseAcc
village	4668	25	1	180	104	217	5195	89.86
lake	118	1595	0	13	186	0	1912	83.42
river	102	0	5025	0	0	0	5127	98.01
road	0	0	0	2802	0	0	2802	100
vegetation	521	152	0	64	866	32	1635	52.97
marsh	4	0	0	0	8	500	512	97.66
total	5413	1772	5026	3059	1164	749	17183	
ProAcc	86.24	90.01	99.98	91.60	74.40	66.76		
	Overall Accuracy = 89.9494%			Kappa Coefficient = 0.8693				

Table 2. Confusion matrix of ISODATA algorithm

Class	village	vegetation	lake	river	road	marsh	total	UseAcc
village	4497	18	0	90	39	191	4835	93.01
lake	116	1531	0	14	156	0	1817	84.26
river	152	0	5025	0	82	0	5259	95.55
road	13	0	0	2867	0	0	2880	99.55
vegetation	406	188	0	88	793	14	1489	53.26
marsh	229	35	1	0	94	544	903	60.24
total	5413	1772	5026	3059	1164	749	17183	
ProAcc	83.08	86.40	99.98	93.72	68.13	72.63		
	Overall Accuracy = 88.7912%			Kappa Coefficient = 0.8550				

Table 3. Confusion matrix of ISODATA algorithm

Class	village	vegetation	lake	river	road	marsh	total	UseAcc
village	4499	32	0	92	57	201	4881	92.17
lake	127	1711	0	18	195	0	2051	83.42
river	120	0	5026	0	81	0	5227	96.15
road	1	0	0	2863	0	0	2864	99.97
vegetation	430	29	0	83	608	29	1179	51.57
marsh	236	0	0	3	223	519	981	52.91
total	5413	1772	5026	3059	1164	749	17183	
ProAcc	83.11	96.56	100	93.59	52.23	69.29		
	Overall Accuracy = 88.6108%			Kappa Coefficient = 0.8526				

From the table we can see that the overall accuracy of ACO algorithm is 89.9%, and kappa coefficient is 0.869. The overall accuracy of ISODATA algorithm is 88.8%, and the kappa coefficient is 0.855. The overall accuracy of K-mean algorithm is 88.6%, and kappa coefficient is 0.853. The result indicates that the accuracy of ACO algorithm is higher than the other two in remote sensing images classification

4 Conclusion

Swarm intelligence remote sensing classification is one of the hotspots. More complicated the local ground situation is, more difficult the current remote sensing classification is to obtain higher classification accuracy. This paper raised an unsupervised method of ant optimization algorithm for remote sensing classification. Ant colony algorithm has the following characteristics: (1) Parallel calculation: ant search process is independent. They communicate with each other by pheromone only. (2) Positive feedback: positive feedback accelerates the search for optimal solutions. (3) Robust: ACO depends little on the initial conditions compared with other algorithms. (4) Autonomous: little manual intervention is engaged in the solution search process. The experiment shows that ant optimization algorithm is more effective in remote sensing images processing.

As the swarm intelligence research is still at the exploratory stage. And the real ant colony behavior is not yet clear. A complete theoretical system has not been formed. Therefore it's just a start to solve such a complicated remote sensing images classification problem with ant colony algorithm. Further improvement of this method is needed in the future.

References

1. NishiiR, S.: Supervised Images Classification by Contextual Boost Based on Posteriors in Neighborhoods. J. IEEE Transaction on Geoscience and Remote Sensing 43, 2547–2554 (2005)
2. Shuang, L., Shengyan, D., Shuming, X.: Comparion and research on remote sensing classificision methods. J. Henan University Trans. 32, 70–73 (2002)
3. Dorigio, M., Colorni, A., Maniezzo, V.: The ant system: optimization by a colony of cooperating agents. J. IEEE Trans. Syst. Man Cybern. B. 26, 29–41 (1996)
4. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. J. IEEE Trans. On Evolutionary Computation 1, 53–66 (1997)
5. Gambardella, L.M., Taillard, E.D., Dorigo, M.: Ant colonies for the quadratic assignment problem. J. Journal of the Operational Research Society 50, 167–176 (1999)
6. Maniezzo, V., Colorni, A.: The ant system applied to the quadratic assignment problem. J.IEEE Trans. Knowledge and Data Engineering 11, 769–778 (1999)
7. Dorigo, M., Dicaro, G.: Ant colony Optimization: A New Meta-heuristic. In: Proc. of 1999 IEEE Congress on Evolutionary Computation Proceedings (CEC 1999), pp. 1470–1477. IEEE Press, Washington (2001)
8. Zhenglong, W., Rujing, W., Minggui, T., Meisheng, X.: Mining Classification Rule Based on Colony Algorithm. J. Computer Engineering and Application 20, 30–33 (2004)
9. Shugen, W., Yun, Y., Ying, L., Chonghua, C.: Automatic Classification of Remotely Sensed Images Based on Artificial Ant Colony Algorithm. J. Computer Engineering and Application 29, 77–80 (2005)
10. Yanfang, H., Pengfei, S.: An improved ant colony algorithm for fuzzy clustering in images segmentation. J. Neurocomputing 70, 665–671 (2007)

A Novel Clustering Algorithm Based on Gravity and Cluster Merging

Jiang Zhong[1], Longhai Liu[1], and Zhiguo Li[2]

[1] College of Computer Science, Chongqing University
Chongqing 400044, China
[2] Shanghai Baosight Software Corporation,
Chongqing, 400039, China
{Zhongjiang,liulonghai,lizhiguo}@cqu.edu.cn

Abstract. Fuzzy C-means (FCM) clustering algorithm is commonly used in data mining tasks. It has the advantage of producing good modeling results in many cases. However, it is sensitive to outliers and the initial cluster centers. In addition, it could not get the accurate cluster number during the algorithm. To overcome the above problems, a novel FCM algorithm based on gravity and cluster merging was presented in this paper. By using gravity in this algorithm, the influence of outliers was minimized and the initial cluster centers were selected. And by using cluster merging, an appropriate number of clustering could be specified. The experimental evaluation shows that the modified method can effectively improve the clustering performance.

Keywords: Fuzzy C-means Algorithm, Gravity, Cluster Merge.

1 Introduction

In many data mining applications, the partitioning of a data set can be considered to be the result of clustering procedures. Clustering is the process of grouping a data set into clusters so that objects in the same clusters are as similar as possible and objects in different clusters are as dissimilar as possible. Dissimilarities are usually assessed based on the attribute values describing the objects. Often, distance measures are used. As an instance of unsupervised learning, clustering is increasingly employed in many fields, including machine learning, data mining, pattern recognition, image analysis and bioinformatics.

Fuzzy c-means (FCM) algorithm is a method of clustering which allows one piece of data to belong to two or more clusters. This method was developed by Dunn [1] in 1973 and improved by Bezdek [2] in 1981 and is frequently used in data mining. Since the traditional FCM algorithm does not employ the spatial correlation information, it would be sensitive to noise and outliers. Also, it must be given the predefined number of clusters. To overcome these drawbacks of FCM algorithm, Davé [3] proposed the robust version of fuzzy C-means (robust-FCM). In robust-FCM, noise is described as a further cluster, which is denoted as a noise cluster. This method can

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 302–309, 2010.

effectively relieve the impact of noise. Frigui and Krishnapuram [4] added a regularization term to the cost function of FCM, in this method, clusters compete for membership of data items and the number of clusters is progressively reduced until an optimum is reached. Sun et al. [5] proposed to select a smaller C value, and then some of the unreasonable large clusters were split into several small clusters and eventually get an optimum number of clusters. However, these FCM methods are computationally expensive, and the speed and stability of the clustering are greatly affected by the initial cluster centers.

In this paper, a novel FCM algorithm based on gravity and cluster merging was proposed. Firstly, it uses the gravity computing process to find and remove outliers in the datasets. Secondly, the gravity-based initial cluster centers selection method is used to solve the problem of sensitive to the initial cluster centers. At last, the algorithm gets the clustering results through cluster merging process. Some experiments results with real dataset show the feasibility and efficiency of the novel method. After a brief review of the FCM in Section 2, a novel FCM algorithm based on gravity and cluster merging is introduced in Section 3, the empirical evaluations of the proposed approach are presented in Section 4, the conclusion and review the approach are presented in section 5.

2 Overview of the Fuzzy C-Means Algorithm

Fuzzy C-Means (FCM) is a well-known unsupervised clustering algorithm, and it is an iterative method that produces optimal C partitions for the data set by minimizing the objective function J_m, which is the weighted sum of squared errors within groups and is defined as follows:

$$J_m(U,V)=\sum_{k=1}^{C}\sum_{i=1}^{n}u_{ik}^{m}\,\|\,x_i-\mu_k\,\|^2, 1\le m\le\infty \ . \tag{1}$$

In Eq. (1), $U=[u_{ik}]$ represents a fuzzy c-partition of the data set, u_{ik} is the degree of membership of x_i to the kth cluster and it satisfies the condition (2). $V=\{\mu_1,\mu_2,...,\mu_C\}$ is a vector of unknown cluster prototype and $\mu_i\in R^p$, m is any real number greater than 1. $\|\,x_i-\mu_k\,\|$ represents the distance between vector x_i and cluster prototype μ_k .

$$\sum_{k=1}^{C}u_{ik}=1, i\in\{1,2,...,N\} \ . \tag{2}$$

Fuzzy partition is carried out through an iterative optimization of (1) with update of membership u_{ij} and the cluster centers μ_i by (3) and (4):

$$u_{ik}=1\Big/\sum_{j=1}^{C}\left(\frac{\|\,x_k-\mu_i\,\|^2}{\|\,x_k-\mu_j\,\|^2}\right)^{\frac{1}{m-1}}, 1\le i\le c, 1\le k\le n \ . \tag{3}$$

$$\mu_i = \sum_{k=1}^{n} (u_{ik})^m x_k \Big/ \sum_{k=1}^{n} (u_{ik})^m ,1 \le i \le C \ . \tag{4}$$

The criteria in this iteration will stop when $\max_{ij}(| u_{ij} - u_{ij} |) < \varepsilon$, where ε is a small constant value bigger than zero.

3 Dynamic Clustering Based on Gravity and Cluster Merging

Although the traditional FCM algorithm could get well results in many cases, it is sensitive to outliers and the initial cluster centers. In addition, it cannot get the cluster number by itself. We proposed a variation FCM algorithm based on gravity and cluster merging to overcome the above problems. Firstly, gravity-based mechanism was used to detect outliers and select the candidate initial cluster centers; secondly, an optimum number of clusters were obtained through merging clusters.

3.1 Gravity-Based Outlier Detection

3.1.1 Definitions

The traditional FCM algorithm does not employ spatial correlation information; therefore it can not accurately reflect the spontaneous category relationship between the data. The gravity mechanism [6] was introduced to FCM in this paper, because the gravity is not only relevant to distance, but also relevant to the mass.

Newton's law of universal gravity is one of the most useful laws in all the scientific subjects and it is described as follows:

There is universal gravity between every pair of objects, and the gravity is directly proportional to the product of the two objects' masses and inversely proportional to the square of the distance between the two objects.

$$F(m_1, m_2) = G\frac{m_1 \times m_2}{r^2} \ . \tag{5}$$

Where F is the magnitude of the gravitational force between the two point masses, m1 and m2 are the mass of the two objects, r is the distance between m1 and m2, and G is the gravitational constant.

Regard all the data objects as n points whose masses are well-proportioned, the n points are asymmetrically distributed in data space. In order to calculate the gravity between two objects in the data set, we define the mass of object in following.

Definition 1. The Eps neighborhood of object p, denoted as NEps (p), can be defined by $NEps(p) = \{q \in X, dist(p,q) \le Eps\}$, where $dist(p,q)$ is the distance between object p and object q.

Definition 2. The mass of object p: the mass of object p can be defined by the number of objects which is contained in the Eps neighborhood**,** that is $Mp = | NEps(p) |$. The mass of the object reflects the distribution of objects around it. The greater density around the object, the larger mass the object has.

Definition 3. The outlier: If there is no other objects besides object p in the neighborhood Eps, that is Mp = 1, the object p is called an outlier.

Definition 4. The gravity between two data objects: Taken the objects in the data set as particles with mass in Euclidean space, according to Newton's law of gravity, we can deduce that the gravity between object p and object q is:

$$F(p,q) = G\frac{m_p \times m_q}{r^2} = G\frac{|Neps(p)| \times |Neps(q)|}{dist^2(p,q)} \ . \tag{6}$$

Where G is the gravitational constant, which has no influence in the clustering problems.

3.1.2 Identify Outliers by Gravity

The existing outlier detection methods mainly include distance-based method, density-based method and depth-based method. These methods identify outliers well in some cases, but the defect of these methods is that the identified result is sensitive to the threshold. To solve this problem, a gravity-based outlier detection method is proposed in this paper.

One of the features of outliers is sparseness, and another feature is that the outliers are distant from most of other points. These two features reflected in the gravity formula (6) are that the value of m is small and the value of r is large. According to the gravity formula, after calculating the gravity, we find that the gravity between the outlier and other points is very small, and the feathers of outliers are more obvious and easier to be identified. Set ε as the threshold, when the sum of gravity between point x_i and the other points is smaller than ε , we regard it as an outlier and remove it from the dataset. The value of ε is determined by the experience function.

The advantage of this method is as follows: after the gravity calculation, the feathers of the outliers are more obvious.

3.2 Gravity-Based Method for Selecting Candidate Initial Cluster Centers

As the objective function of the fuzzy clustering is non-convex and the FCM algorithm is based on the hill-climbing technique, once the initial cluster centers are selected inappropriately, the function will converge to a local minimum or fall into a saddle point rather than a global minimum.

To overcome the above problems, we propose the gravity-based method to select the candidate initial cluster centers. Firstly, calculate the distances between all the points, the mass of each point, the gravity from each point to all other points and the sum of gravity. Secondly, select the point with the biggest sum of gravity as the first candidate initial cluster center. And then, for the rest points, repeat select the point with the biggest sum of gravity and the distances to all candidate cluster centers are bigger than the threshold β as the next cluster center, until there is no other points to be selected.

The purpose of setting a threshold β is to avoid the selection duplication and the cluster centers excessively dense. Experimental results show that the initial cluster centers are mostly in the middle of clusters by using this method. Therefore, compared to traditional FCM and max-min distance algorithm, the proposed method can get better results.

3.3 Cluster Merging

Clustering strategies have been developed to improve the clustering performance. The multi-clustering fusion algorithm [7] which was proposed by Frossyniotis .etc uses voting mechanism to merge clusters. In this paper, we proposed a cluster merging algorithm to overcome the defect of FCM that it cannot specify the cluster number by itself. Figure 1 shows the clustering results, C1 and C2 represent the cluster centers of the two clusters.

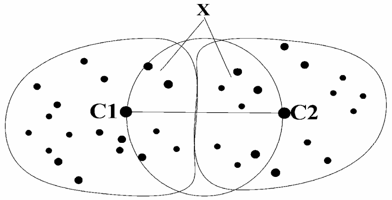


Fig. 1. Merge the adjacent clusters

According to the basic idea of the clustering, similar samples are classified into the same cluster as much as possible. Take the distance between C1 and C2 as the diameter and draw a circle. All points in the sphere formed the point set X. If X obeys uniform continuous distribution, the two clusters could be merged.

We use the evaluation function to determine if the gravity obeys continuous uniform distribution in $[F_{\min}, F_{\max}]$. The idea of evaluation function in this paper is as following: n data objects are evenly divided into m segments in the scope of $[F_{\min}, F_{\max}]$, and there are $\bar{m} = n/c$ data objects in each segment. Actually, the gravity range of the segment k is $[F_{\min} + (k-1)\frac{F_{\max} - F_{\min}}{m}, F_{\min} + k\frac{F_{\max} - F_{\min}}{m}]$, and there are about m_k points in this segment. If they are distributed uniformly, the number of points in each segment should be the expectation $\overline{m}$. The actual number of points in segment k is m_k. In order to measure the deviation degree between the actual value and the expectation (that is Mean), we will calculate the value of variance. The equation of variance function is $D = \sum_{k=1}^{c} (m_k - \bar{m})^2$.

If the points which are in the sphere obey the uniform distribution, then $m_k \approx \bar{m}$, that is $m_k - \bar{m} \approx 0$, and $D \approx 0$. In this paper, if two clusters meet the condition $D \leq (m \times \bar{m}^2)/2$, they will be merged.

The selection of the parameter m is based on the number of n. The larger value n has, the greater value m is. Experiments have proved that the appropriate value of m is ranged in $[\sqrt[4]{n}, \sqrt[2]{n}]$, and n is the points' number in the sphere.

3.4 The Novel FCM Clustering Algorithm

The novel FCM algorithm based on gravitation and cluster merging could be summarized as follows:

Algorithm 1. The Novel FCM Clustering Algorithm

```
Input:
  Data Set S, the initial number of clusters C, threshold $\beta$
Output:
  Membership matrix $\boldsymbol{U}$
  1: Calculate the mass of each point according to the method proposed in Section 3.1;
  2: Calculate the sum of gravity between $x_i$ and all the other points;
  3: Remove the points as outliers whose sums of gravity are less than the threshold $\alpha$ ;
  4: Select the point with the largest sum of gravity as the initial cluster center;
  5: for all $x_i \in S$ , do
       If the point $x_i$ meets the following criteria and $x_i$ is not a candidate cluster center,
       select it as the next candidate cluster center:
         1. The distances between $x_i$ and all cluster centers are greater than $\beta$ ;
         2. The value of gravity of $x_i$ is the largest one;
     End for
  6: Apply the FCM algorithm to group the data objects into clusters;
  7: for every pair of clusters, testify whether they meet the merge criteria. If the pair
       meets the merge criteria, the two clusters are merged into one. Until there are no
       clusters to be merged.
  8: Add each outlier into its nearest cluster and update the membership matrix $\boldsymbol{U}$
  9: Return $\boldsymbol{U}$
```

As the first few steps have already selected the initial cluster centers, the number of clusters and cluster prototypes do not need to artificially set. During the initial cluster centers selection process, there is no uniform standard for the parameter β by now. In practice, we can take a smaller value of β . In this way, we can obtain more cluster centers in step 5, and then achieve the final clustering result through cluster merging.

4 Experimental Evaluation

4.1 Data Set

In order to see the improvement in the results, we compared modified FCM to the traditional FCM. The first comparison is performed on the well-known Iris database. The classes are not spherical shaped and only one class is linearly separated from the others. The second comparison is performed on breast-cancer-Wisconsin database, which contains two classes with 458 instances. Every breast cancer instance is described by nine numerical attributes. There are 16 instances that contain a single missing attribute value, and we used the attribute mean to fill in the missing value in the experiments.

4.2 Experimental Results and Analysis

The performance of algorithm is evaluated by Recall ratio and Precision. Recall rate is the ratio of the correctly grouped instance number to the instance number in the same clusters. Precision is the ratio of the correctly grouped instance number to the instance number grouped into the same cluster.

Table 1. The comparison performed on Iris database

	Recall ratio (%)			Precision (%)		
classes	Traditional FCM	Modified FCM	Improvement	Traditional FCM	Modified FCM	Improvement
Setosa	100.00	100.00	0.00	100.00	100.00	0.00
versicolor	94.00	94.00	0.00	78.30	81.00	2.70
veirginica	74.00	78.00	4.00	92.50	92.90	0.40
average	89.33	90.67	1.34	90.27	91.30	1.03

Table 1 demonstrates the experimental result on Iris database. Compared to the traditional FCM, the modified method improves 1.34% on the average Recall ratio and improves 1.03% on the average Precision.

Table 2. The comparison performed on breast-cancer-Wisconsin

	Recall ratio (%)			Precision (%)		
classes	Traditional FCM	Modified FCM	Improvement	Traditional FCM	Modified FCM	Improvement
Benign tumors	97.60	96.07	-1.53	95.31	98.00	2.69
Malignant tumors	91.29	96.27	4.98	95.24	92.80	-2.44
average	94.45	96.17	1.72	95.28	95.40	0.12

Table 2 demonstrates the experimental result on breast-cancer-Wisconsin. Compared to the traditional FCM, the modified method improves 1.72 % on the average Recall ratio and improves 0.12% on the average Precision. As can be seen, both Recall ratio and Precision have improved to some extent by using the modified method.

5 Conclusions and Future Work

In this paper, we present a novel FCM algorithm based on gravity and cluster merging. Gravity-based method was used to detect the outliers at the first step, then this method was used to select cluster centers; and cluster merging method was used to merge clusters once the clusters meet the merge criteria in the end. The experiments show that the modified method has better performance than the traditional FCM clustering algorithm□ and it improved the clustering accuracy.

Clustering high-dimensional data is a high challenging research field, especially when the data are very sparse and highly skewed. In the future, we will study the applications of the proposed method in high-dimension clustering.

Acknowledgments. This work is supported by the Third Stage Building of "211 Project" (Grant No. S-10218) and Natural Science Foundation Project of CQ(CSTC,2010BB2046).

References

1. Dunn, J.C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics 3, 32–57 (1973)
2. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
3. Davé, R.N.: Characterization and detection of noise in clustering. Pattern Recognition Letters 12, 657–664 (1991)
4. Frigui, H., Krishnapuram, R.: Clustering by competitive agglomeration. Pattern Recognition 30, 1109–1119 (1997)
5. Sun, H.J., Wang, S.R., Jiang, Q.H.: FCM-Based Model Selection Algorithms for Determining the Number of Clusters. Pattern Recognition 37, 2027–2037 (2004)
6. Indulska, M., Orlowska, M.E.: Gravity Based Spatial Clustering. In: Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems, pp. 125–130. Year of Publication, Virginia (2002)
7. Frossyniotis, D., Pertselakis, M., Stafylopatis, A.: A Multi-Clustering Fusion Algorithm. In: Proceedings of the Second Hellenic Conference on Artificial intelligence, pp. 225–236. Year of Publication, Thessaloniki (2002)

Evolution Analysis of a Mobile Social Network

Hao Wang[1, 2] and Alvin Chin[2]

[1] Beijing University of Posts and Telecommunications, Beijing 100876, China
[2] Nokia Research Center, Beijing 100176, China
ext-hao.10.wang@nokia.com, alvin.chin@nokia.com

Abstract. Smart phones and ubiquitous wireless connections are helping people to build and maintain mobile social relationships. We present an in-depth and complete evolution analysis on the user activities and social graph of a mobile social network using data obtained from Nokia Friend View. Our results show that (1) user activities in Friend View are highly correlated and the power law fitted exponents for user activities distribution are slowly becoming larger over time, which appears to be contrary to the famous "rich get richer" assertion in the preferential attachment model because users in Friend View regard the reciprocity as important during the interaction and (2) both undirected friend network and directed comment network in Friend View are small-world and scale-free networks over time with slowly decreasing clustering coefficient. However, compared to online social networks where users have a large number of friends but loose weakly-tied subgroups, users in Friend View tend to have close strongly-tied cohesive subgroups. The results can help us understand users' social activities and interactions over time in mobile social networks.

Keywords: Mobile social network analysis, Small-world evolution.

1 Introduction

Due to the mobility of users and the ubiquity of the mobile phone, social networks are now being created and maintained using the mobile phone, which are called mobile social networks. Mobile social networking applications such as Loopt, Foursquare and Gypsii use location of users in order to provide services such as finding people and places nearby, providing relevant content based on location, and creating specific topic channels from which other people can subscribe to.

Previous work has studied the structure and properties of online social networks (OSN) [1-4] and their evolution [2, 3, 5-7]. However, many researchers use crawled datasets to analyze and model different types of social networks and therefore these results are subject to bias due to the incomplete social information on users about linkage and activity from using only one or two snapshots of networks. Very few works deal with in-depth social network analysis of a mobile social network (except for [8]) and study its evolution from beginning to end. There are still, however, many questions that have not been addressed. How do the activities and interactions of users in mobile social networks change and evolve over time? What is the correlation between users' activities and interactions and how does this correlation evolve over time? Is a mobile social network still a small-world and scale-free network over time

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 310–321, 2010.

and how does the social graph evolve? If we can address these questions, it can help us understand more about users in mobile social networks, thus improving the user experience and mobile social services provided to the user.

In this paper, we perform an in-depth analysis of a mobile social network and study its complete evolution using data obtained from Friend View, which is a service that was created by Nokia Research Center to allow mobile users to find others, update their status and location, add friends, and keep updated with others. We study the number of users' activities such as posting status messages, making comments to others' status, and making friends along with the network properties such as degree, shortest path length, network density and clustering coefficient in the friend network and comment network and show how they evolve. Results illustrate that user activities in Friend View are highly correlated and the power law fitted exponents for user activity distributions are slowly becoming larger over time, meaning fewer people are being connected with those that have high degree of activity (or the most popular users), therefore new Friend View users appear to not necessarily connect to the users with the most activity. This appears to contradict the famous "rich get richer" assertion in the preferential attachment model in which new users are more likely to get connected to those that have a high degree of connection. We believe the reason for this behavior is because users in Friend View regard the reciprocity during interaction as important, and not the amount of activity. Also, Friend View remains a small-world and scale-free network that has decreasing clustering coefficient over time. In addition, the comment network is more closely connected compared to the friend network in Friend View and other OSNs.

Our contributions are the following. First, we study the complete evolution of a mobile social network based on the entire lifetime dataset (which we believe we are the first to do) from beginning to end, while other researchers do complete evolution for only OSNs and evolution analysis using the snapshots of crawled datasets. Second, we discover that the power law fitted exponents for user activity distribution slowly becomes larger over time, which appears to contradict the "rich get richer" assertion in the preferential attachment model mostly associated with current social network models. Third, the friend network and comment network are both small-world and scale-free networks over time with assortative mixing pattern.

The paper is organized as follows. Section 2 describes related work on the study of the properties and evolution of OSNs and mobile social networks. In Section 3, we introduce and describe the overall dataset statistics of Nokia Friend View. We study the evolution of Friend View in Section 4 and quantify the evolution of user activities, the friend network and the comment network and show how the assortative mixing pattern evolves. Finally, we conclude the paper in Section 5.

2 Related Work

There has been much work in studying the properties and structure in OSNs such as Digg [9], Twitter [3, 10], MySpace and YouTube [11], Wealink [2], Yahoo! 360 and Flickr [4], Cyworld [1], and Xiaonei (now RenRen) [12], using social network measures such as degree, network density, clustering coefficient, number of users, number of links, and network diameter. These researchers compare and contrast their results with other OSNs. For example, comparisons have been made between Twitter, WWE blogging network, and Brightkite [8] and have been discovered to be small-world networks. Our work differentiates from them in that we study the evolution of users' social activities and the structural properties in a mobile social network.

Wilson et al [13] studied user interactions in Facebook to answer the question as to whether social links are valid indicators of real user interaction. Cha et al [14] used Flickr user interactions and favorite photo activities to study how far and quickly information was propagated in the Flickr OSN. Our work differs in that we do not focus on exploring users' interaction and social activities in a static network, but rather investigate how users' social activities change and evolve over time.

Recently, there has been increasing work on studying the evolution of OSNs. Many researchers have used crawled datasets from OSNs in order to model the evolution. For example, Kumar et al [4] analyzed the structure of Yahoo! 360 and Flickr and Barabasi et al [6] analyzed scientific collaborations over time, and together using social network analysis, they each created a model of evolution and used simulation to test the model. Garg et al [15] analyzed the social content aggregator FriendFeed and suggested that proximity bias plays an important role in the link formation process. Viswanath et al [16] studied the evolution of activity between users in Facebook to capture how social links can grow stronger or weaker over time and found that although the links of the activity network change rapidly over time, many graph properties of the activity network remain unchanged. Hu et al [2] studied the evolutions of degree, network density, clustering coefficient and modularity, in order to reveal the properties and evolutionary patterns of Wealink. Our work, however, differs in that we do not attempt to create a model of evolution or do simulation, but rather, we examine the structural evolution of both the undirected friend network and the directed comment network respectively in a mobile social service.

With the proliferation of GPS location becoming standard in mobile phones and the convenience of internet access over 2.5G and 3G cellular networks as well as Wi-Fi networks, location-based social networks on mobile phones are now becoming a reality. Social networks are now being created using the mobile phone and use location to tailor specific content (such as Brightkite) and people (such as BuddyCloud and Aka-Aki), and are also used for search (such as Loopt) and communication (such as Gypsii). For example, a mobile micro-blog [17] was created for sharing and querying geo-tagged content through mobile phones. CenceMe [18] used sensor-enabled mobile phones to share information in a mobile social network and used context for detecting and recognizing user's everyday activities. Cluestr [19] is a group and community aware mobile social network that allows users to efficiently select contacts to address them as a group. However, there exists few works into analyzing the user behavior and network properties of mobile social networks because it is difficult to obtain this data. Perhaps one of the first analytical studies into mobile social networks is that of Dodgeball [20], where the author provided an ethnographic study using qualitative analysis (participant observation, user observations and in-depth interviews). Li and Chen [8] have conducted quantitative analysis of a commercial location-based mobile social network (Brightkite). Dong et al. [21] used the mobile phone call logs from users in a city to build a large mobile social network and analyzed the network properties. Our work differs from [8, 21] in that we study the evolution of user's activities and evolution of properties of both the undirected friend network and the directed comment network in a mobile social network.

3 Friend View and Dataset Description

Friend View is a location-enhanced micro-blogging application launched in Nokia Beta Labs in the beginning of November 2008, and was discontinued at the end of September 2009 because it was an experimental project. It allows users to post messages about their status with location about where they are from GPS-enabled Nokia S60 phones and share their activities with close friends. In Friend View, users can keep up with all their friends' locations and recent status messages on a map and make comments to related conversations. Users can add friends adding a new friend who has commented on a status message originated by the user's friend.

We obtain the entire complete dataset from the Friend View project team that consists of the time that all users joined Friend View, status messages posted with or without GPS location, comments users received and made, and their list of friends. The dataset contains full information about users' activities in 11 months For privacy reasons, all the data used in this paper are appropriately anonymized. Since much work has studied the structure of a static snapshot of a dynamic and evolving social network by crawling a subset [1, 8, 16], we believe that in-depth analysis about the entire lifetime of the Friend View dataset provide interesting and reliable results.

During the 11 months of operation, a total of 34980 users are registered to Friend View, with 8443 users having friends, and a total of 20873 friendship links being created. A total of 62736 status messages are posted by 16176 users, providing an average of 3.88 status messages per user. 37599 status messages from 13597 users have GPS location (we see this location information as a kind of checkin that is popular in current location-based social networks such as FourSquare and Gowalla), 9363 status messages that come from 2395 users receive comments, and 22251 comments are posted from 2283 users, providing an average of 3.91 status messages with comments per user and 2.38 comments per status message. An average of 9.75 comments is made per user for the status messages that have comments.

On average, the interactions or activities (status messages and comments) are low compared with other OSNs like Facebook. In our opinion, this reflects the fact that mobile location-based social networks are still not as widely used, due to the privacy concerns of sharing one's location to all. In addition, despite the convenience of using the mobile phone anytime and anywhere for posting status and making comments, it is still easier and faster to write statuses and comments with a full-sized keyboard than with a keypad or small keyboard on the phone.

4 Evolution of Friend View

We analyze the evolution in Friend View and compare it with other OSNs. We extract 11 snapshots of the dataset with an interval of one month. We first present the usage and social activity evolution of Friend View. Then we examine the social network evolution of Friend View by considering the graph's properties from both the user's friend network and comment network and compare them with that of OSNs.

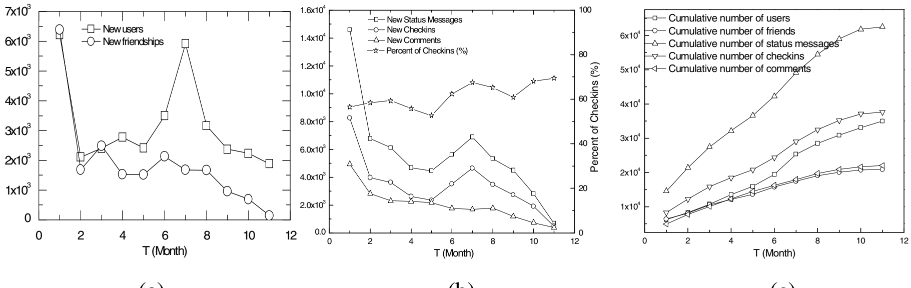
4.1 Usage Evolution

The evolution of new users and friendships made and users' social activities including statuses, checkins and comments are shown in Figure 1 (a) and (b). In Figure 1 (c), the evolution of the cumulative number of users and users' activities are illustrated.

The growth patterns for the number of new users and new friends are similar in shape in Figure 1 (a). The highest number occurs during the beginning when many users join Friend View and are eager to try out the service, and then overall dramatically decreases until the end of the service, with the exception of a few spikes. In the 6th month (April 2009), the number of new users increases and it is interesting to note that in the 7th month (May 2009), the number of new users encounters a huge spike and almost reaches the level from the beginning. We are not sure why this exists, however it may be as a result of a touch version of Friend View that was released.

The cumulative growth patterns for the number of users and friends in Figure 1 (c) appear to follow an S type of shape, but are less steep than those in OSNs such as Wealink [2] and Cyworld [1]. The cumulative trends start small, rapidly accelerate and then stabilize. In the case of the cumulative number of users, it still continues to grow while with the cumulative number of friends, it appears to plateau off during the 10th and 11th months. The cumulative number of status messages, checkins and comments also follow a similar distribution, with the cumulative number of comments closely resembling the cumulative number of friends. This may be due to the fact that Friend View allows users to add commenters as friends.

As shown in Figure 1 (b), the growth patterns for the number of user activities such as posting status messages, uploading GPS locations when checking in at a place and commenting on others' statuses and checkins, share similar trends with Figure 1 (a). At the first release, users were excited with the mobile location-based service and were very active in the networks. With the huge spike of new users joining Friend View in the 7th month (May 2009), the number of status messages and checkins also encounter a relatively huge spike, although it did not reach the level from the first month. However, we see no increase for new comments from May 2009 compared with that from April 2009. From that, the social activities in Friend View reached a



(a) (b) (c)

Fig. 1. Evolution of the number of new users, friendships, activities and the cumulative results

saturation point from which it reduced progressively. It is interesting that the percent of checkins over status messages is stable and rises gradually from 56% in the beginning to about 70% in the end. Therefore, users did not lose their interest in sharing where they were with close friends when they posted their current status.

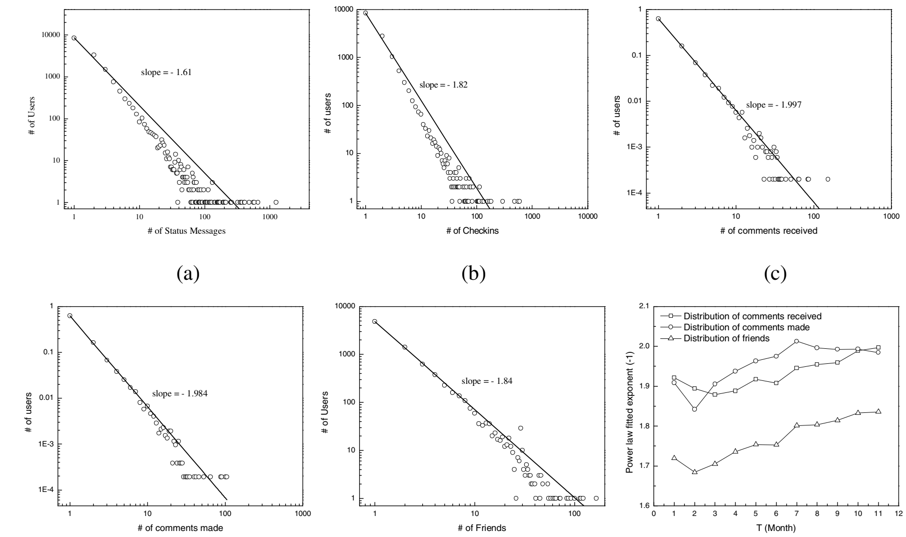
We now examine if there are any interesting user behavioural relationships between the various user activities by performing a correlation analysis. Table 1 gives the Pearson correlation coefficient between different pairs of users' social activities. The Pearson coefficient between a user's number of status messages and number of checkins is 0.87, indicating strongly that a user in Friend View that posted many status messages is also likely to share many locations. This explains why the percent of checkins over status messages remains high when users' activities decline. Users that posted many status messages are also very likely to receive many comments (Comments In) about their status messages/checkins (Pearson coefficient = 0.58 for status messages and 0.55 for checkins) and make many comments to (Comments Out) others' status messages/checkins (Pearson coefficient = 0.5 for status messages and 0.47 for checkins) and associate with many friends (Pearson coefficient = 0.38 for status messages and 0.27 for checkins). Users who post many comments are very likely to receive many comments for their status messages/checkins with Pearson coefficient between comments out and comments in being 0.95. Therefore, the correlation coefficients in Table 1 could explain the similar patterns and trends in Figure 1 for users and friendships and for user's behavior patterns of social activities.

Table 1. Pearson correlation coefficient between user's activities

	Status	Checkins	Comments In	Comments Out	Friends
Status	1				
Checkins	0.87583	1			
Comments In	0.57724	0.5483	1		
Comments Out	0.50489	0.47416	0.95431	1	
Friends	0.37636	0.27109	0.29547	0.27732	1

4.2 Social Activity Evolution

We present the distribution of users' social activities including the number of users' status messages, checkins, comments received, comments made and friends. These distributions are based on the whole dataset and are shown in log-log scale as illustrated in Figures 2 (a)-(e). As expected, the distributions follow a power law with different fitted exponents. The majority of users have a small number of activities in Friend View, for example, many users posted one, two and three status messages /checkins. However, we do encounter few users in the long tail of the distribution that have excessive number of status messages/checkins that amount in the hundreds and even thousands. Although the maximum number of comments a user received and made and the maximum number of friends a user had are not very large (smaller than 200), the distributions are more perfectly fitted by power law.



(a) (b) (c)

(d) (e) (f)

Fig. 2. User activity distribution (based on whole dataset) with power law fitted exponent for (a) Status messages, (b) Checkins, (c) Comments received, (d) Comments made, (e) Friends and (f) Power law fitted exponent evolution for comments received, comments made and friends.

Additionally, we analyze and show in Figure 2 (f) the distributions of the number of comments received and made and the number of users' friends, based on the cumulative dataset up to the month indicated on the x- axis and find how the distributions change and evolve. The comments and friends that users are involved in are more important for the social interactive communities formed in Friend View in that they reflect real user activity rather than social linkage alone [13]. We do not show all the three distributions (number of comments received and made and number of users' friends) at each cumulative month here, but find that they normally can be fitted as a power law distribution. In Figure 2 (f), we plot the evolution of the power law exponent for each distribution. During the evolution of Friend View, generally, the absolute values of power law exponent for the three distributions are slowly becoming larger, although not too much. Large absolute slope of power law fitting line in log-log scale may imply that more users appear in the head of the distribution curve's long tail with relatively more users having small number of comments received and made and small number of friends. It seems to contradict the famous "rich get richer" assertion in the preferential attachment model [22], where new users are more likely to comment and link with those that have high degree. Two possible reasons for this may be: (1) users in Friend View interact with each other through mobile phones and so it is difficult for them to find out the popular users to add as a friend and select interesting activities to join and (2) users in the directed comment network of Friend View regard the reciprocity in comment interaction as important [13]. This explanation can be confirmed in Section 4.4.

4.3 Social Network Evolution

We study the evolution within the friend network and comment network using the cumulative dataset between the beginning of Friend View to the Tth month since its release. The friend network is an undirected graph formed based on a pair of friends, while the comment network is a directed graph based on users making comments to others's statuses. To study how the friend and comment social graphs change and evolve over time, we analyze the shortest path length and network clustering coefficient, as well as the network density, to understand if the friend and comment networks are always small-world and scale-free networks over time.

Figure 3 shows evolution for the average shortest path length for both the friend network and comment network. The friend network has an average shortest path length of around 6 (with a slight decrease from 6.0 to 5.8 during the first three months and then slightly rises from 5.8 to 6.4 over time), indicating each user can reach another user at an average of only 6 hops. This exactly corresponds to six degrees of separation, a general property of social networks [23, 24]. Similarly for OSNs, the average shortest path length for Wealink [2] slowly rises from 5.5 at the beginning to 7.5 near the end (similar to Friend View), and slowly decreases for the Purdue University network in Facebook [25] from 6.6 at the beginning of March 2007 to 4.4 at the beginning of March 2008 (different than Friend View). Average shortest path length of other undirected friend networks in one or cumulative snapshots are reported to be 3.72 for Xiaonei [12] and 6 for MSN Messenger [26].

In Figure 3, the average shortest path length of the comment network is almost the same to that of the friend network for the first five months. Then, it deviates and directly rises to 8.8. This may be explained by Figure 1 which shows that the number of new comments slowed down over time. Similar work in [25] shows that the average shortest path length over time for a conference network in Twitter [25] quickly decreases from 5.5 on 12/07/09 to 4.2 on 12/18/09 (different to Friend View). Average shortest path lengths of other directed networks in one or cumulative snapshots are 6.01 for Flickr [4], 8.26 for Yahoo! 360 [4], and 6.84 for Sina blog [12].

Figure 4 (a) shows the evolution of clustering coefficient and network density for both the friend network and comment network. The clustering coefficient can provide evidence of the existence of subgroups based on the property that social networks have many connected triads exhibiting a high degree of transitivity [27]. It shows that the two properties for the two networks are positively correlated with each other (Pearson coefficient is 0.98457 for comment network and 0.98457 for friend network)

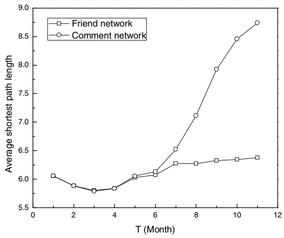
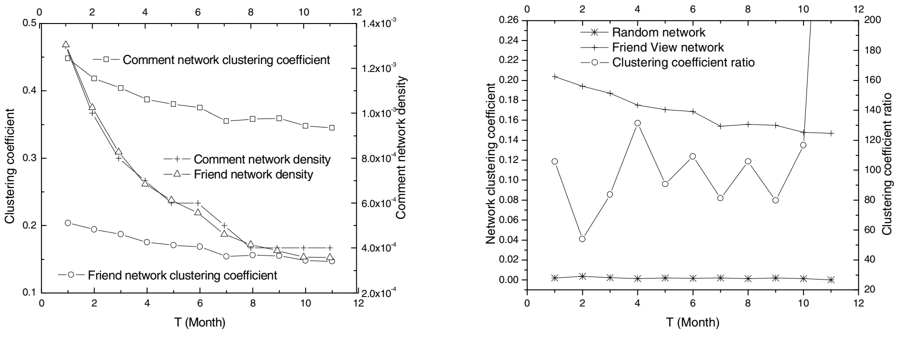


Fig. 3. Evolution of the average shortest path length for friend and comment networks



(a) (b)

Fig. 4. Evolution of (a) clustering coefficient and network density and (b) clustering coefficient of the friend network and its randomized version and coefficient ratio over randomized version

and both follow a similar declining curve, which is similar to the results of Wealink [2] but different to the results of Facebook and Twitter [25]. This means that the Friend View friend network and comment network both become less dense and less clustered (fewer subgroups) over time, as we demonstrated in [28]. The reason is that users form larger subgroups over time as more users join Friend View [28]. However, compared to OSNs where users have a large number of friends but loose weakly-tied connections, users in Friend View tend to have close strongly-tied cohesive subgroups, because the decreasing clustering coefficient for both the friend network and the comment network are still considerably larger than that of OSNs such as Sina Blog and Xiaonei [12], Wealink [2], and Facebook and Twitter [25].

In addition, in Figure 4 (a), network density of the friend and comment networks are almost similar to each other over time. In the first five months, the friend network is slightly denser than the comment network and in the next months, the comment network is slightly denser than the friend network. However, the clustering coefficient of the comment network is apparently larger than that of the friend network, which means that users in the comment network were more cohesively connected than users in the friend network. An explanation is that user A receives a comment from user B and then visits user B's status list where user A finds an interesting comment from user C, so user A visits user C's status and makes a comment.

Figure 4 (b) shows the evolution for the clustering coefficient of the corresponding random network of the friend network, where the order of magnitude of the friend network's clustering coefficient is higher than that of the corresponding random network [29]. We do not show the related curve for the comment network due to the fact that the comment network has considerably larger clustering coefficient than the friend network (0.345 for the comment network compared with 0.147 for the friend network at the end of Friend View's service and similar pattern during each timeslot) and obviously smaller number of users involved than friend network (2283 users in the comment network compared with 8443 users in the friend network at the end and similar pattern during each timeslot). The clustering coefficient of the comment network is an order of magnitude higher than its corresponding random network. From Figure 4 (b), it is clearly shown that the clustering coefficient for the Friend View friend network is significantly higher than for the random network although it decreases over time. At the same time, from Figure 3, since the average shortest path length is small and is around 6, then we can conclude that the Friend View friend network is indeed a small-world network over time [24]. Besides, from Figures 2 (e) and (f), the friend network degree distribution follows a power law with rising slope over time, so the friend network is also a scale-free network over time. Similar results can be also be inferred for the comment network of Friend View.

4.4 Assortativity Evolution

To determine if there exists a correlation between the undirected connection of adjacent users in the friend network and directed connection in the comment network in Friend View, and how the correlation evolves over time, we use assortativity. Assortativity is defined as the Pearson correlation coefficient of the degrees of each side of a link. A positive assortativity indicates that nodes with large degree in the network tend to be connected by other similar nodes with many connections, and vice versa. We plot the assortativity coefficient for the friend and comment networks in Figure 5.

First, all the assortativity coefficients for the friend and the comment network are always positive over time, so the two networks both have an assortative mixing pattern, confirming the conventional wisdom that social networks are assortative mixing [30]. Second, the assortativity undergoes noticeable decline during the first three months and then remains stable and smooth, except that the assortativity coefficients for comment in-in and comment out-out drop sharply to the same level of comment in-out. Therefore, during evolution, both friend and comment networks of Friend View still have an assortative mixing pattern, unlike Wealink which transitions from initial assortativity to subsequent disassortativity [2]. Thirdly, for the friend network, users with many friends tend to make friends with others who also have a lot of friends. In the comment network, obviously, users who post many comments tend to make comments to status/checkin of others who also make many comments out as we see a large assortativity coefficient for comments out-out (similar to that of Sina blog [12]). Users who received many comments for their status/checkin tend to make comments to status/checkin of others who also received many comments as we see a large assortativity coefficient for comments in-in (contrary to that of Sina blog [12]).

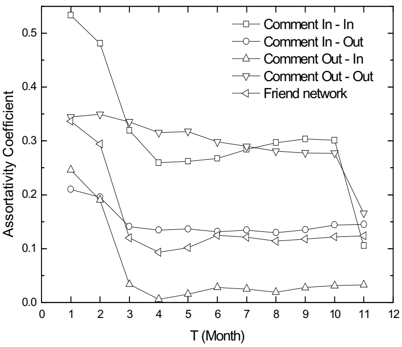


Fig. 5. Evolution of assortativity coefficient for friend network and comment network

That is to say, users in the comment network tend to connect with counterparts through comment activities. The reason is that users in the directed comment network of Friend View regard the reciprocity as important for interaction, which is different to that of the active users in the directed Sina blog which just tend to connect celebrities and do not require the celebrity's response [12].

5 Conclusions

We study the complete evolution of a mobile social network based on the entire lifetime dataset from Friend View which we believe we are the first to do from beginning to end, while others do evolution for only OSNs and evolution analysis using crawled datasets. We discover that Friend View does not follow the "rich get richer" scheme in the preferential attachment model of social networks, but rather power law fitted exponents for user activity distribution are slowly becoming larger over time. We find that the friend network and comment network in Friend View are both small-world and scale-free networks with assortative mixing pattern over time.

By analyzing the evolution of a mobile social network, it helps to understand its operation and user behaviour, thus helping to identify which time periods to focus on for improving its growth. This can help us create many social applications, such as a dynamically social interface that highlights the people in the network that have a certain property like degree and visualize their ties (similar to Vizster [31]), improve friend recommendations by using the social graph and the network evolution for suggesting important people to connect to, and discovering and labeling communities of closely-connected and cohesive people. It is our belief that we can create an analytical framework that can be used to study any mobile social network in comparison with other social networks.

References

1. Hyunwoo, C., Haewoon, K., Young-Ho, E., Yong-Yeol, A., Sue, M., Hawoong, J.: Comparison of Online Social Relations in Volume Vs Interaction: A Case Study of Cyworld. In: The 8th ACM SIGCOMM IMC, pp. 57–70. ACM, New York (2008)
2. Haibo, H., Xiaofan, W.: Evolution of a Large Online Social Network. Physics Letters A 373(12-13), 1105–1110 (2009)
3. Akshay, J., Xiaodan, S., Tim, F., Belle, T.: Why We Twitter: Understanding Microblogging Usage and Communities. In: 1st SNA-KDD, pp. 56–65. ACM, New York (2007)
4. Kumar, R., Novak, J., Tomkins, A.: Structure and Evolution of Online Social Networks. In: The 12th ACM SIGKDD, pp. 611–617. ACM, New York (2006)
5. Lars, B., Dan, H., Jon, K., Xiangyang, L.: Group Formation in Large Social Networks: Membership, Growth, and Evolution. In: The 12th ACM SIGKDD, pp. 44–54. ACM, New York (2006)
6. Barabasi, A.L., Jeong, H., Neda, Z., Ravasz, E., Schubert, A., Vicsek, T.: Evolution of the Social Network of Scientific Collaborations. Physica A 311(3-4), 590–614 (2002)
7. Jure, L., Jon, K., Christos, F.: Graph Evolution: Densification and Shrinking Diameters. ACM Transactions on Knowledge Discovery from Data 1(1), 2 (2007)
8. Nan, L., Guanling, C.: Analysis of a Location-Based Social Network. In: Intern. Confer. on Computational Science and Engineering, pp. 263–270. IEEE, Los Alamitos (2009)

9. Yingwu, Z.: Measurement and Analysis of an Online Content Voting Network: A Case Study of Digg. In: 19th ACM WWW, pp. 1039–1048. ACM, New York (2010)
10. Haewoon, K., Changhyun, L., Hosung, P., Sue, M.: What Is Twitter, a Social Network or a News Media? In: 19th ACM WWW, pp. 591–600. ACM, New York (2010)
11. Vassia, G.: Voters, Myspace, and Youtube. Social Science Computer Review 26(3), 288–300 (2008)
12. Feng, F., Lianghuan, L., Long, W.: Empirical Analysis of Online Social Networks in the Age of Web 2.0. Physica A 387(2-3), 675–684 (2008)
13. Christo, W., Bryce, B., Alessandra, S., Krishna, P.N.P., Ben, Y.Z.: User Interactions in Social Networks and Their Implications. In: 4th EuroSys, pp. 205–218. ACM, New York (2009)
14. Meeyoung, C., Alan, M., Krishna, P.G.: A Measurement-Driven Analysis of Information Propagation in the Flickr Social Network. In: 18th ACM WWW, pp. 721–730. ACM, New York (2009)
15. Sanchit, G., Trinabh, G., Niklas, C., Anirban, M.: Evolution of an Online Social Aggregation Network: An Empirical Study. In: 9th ACM IMC, pp. 315–321. ACM, New York (2009)
16. Bimal, V., Alan, M., Meeyoung, C., Krishna, P.G.: On the Evolution of User Interaction in Facebook. In: 2nd ACM WOSN, pp. 37–42. ACM, New York (2009)
17. Shravan, G., Jack, L., Romit, R.C., Landon, C., Al, S.: Micro-Blog: Sharing and Querying Content through Mobile Phones and Social Participation. In: ACM MobiSys, pp. 174–186. ACM, New York (2008)
18. Emiliano, M., Nicholas, D.L., Kristóf, F., Ronald, P., Hong, L., Mirco, M., Shane, B.E., Xiao, Z.: Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of the Cenceme Application. In: 6th SenSys, pp. 337–350. ACM, New York (2008)
19. Reto, G., Michael, K., Roger, W., Martin, W.: Cluestr: Mobile Social Networking for Enhanced Group Communication. In: ACM GROUP 2009, pp. 81–90. ACM, New York (2009)
20. Humphreys, L.: Mobile Social Networks and Social Practice: A Case Study of Dodgeball. Journal of Computer-Mediated Communication 13(1), 341–360 (2007)
21. ZhengBin, D., GuoJie, S., KunQing, X., JingYao, W.: An Experimental Study of Large-Scale Mobile Social Network. In: 18th ACM WWW, pp. 1175–1176. ACM, New York (2009)
22. Barabasi, A.L., Albert, R.: Emergence of Scaling in Random Networks. Science 286(5439), 509–512 (1999)
23. Milgram, S.: The Small World Problem. Psychology Today 2(1), 60–67 (1967)
24. Watts, D.J., Strogatz, S.H.: Collective Dynamics of 'Small-World'networks. Nature 393(6684), 440–442 (1998)
25. Nesreen, K.A., Fredrick, B., Jennifer, N., Ramana, K.: Time-Based Sampling of Social Network Activity Graphs. In: 8th Workshop on Mining and Learning with Graphs (2010)
26. Jure, L., Eric, H.: Planetary-Scale Views on a Large Instant-Messaging Network. In: 17th ACM WWW, pp. 915–924. ACM, New York (2008)
27. Stanley, W., Katherine, F.: Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge (1994)
28. Alvin, C., Mark, C., Hao, W.: Tracking Cohesive Subgroups over Time in Inferred Social Networks. New Review of Hypermedia and Multimedia 16(1-2), 113–139 (2010)
29. Sergei, M., Kim, S.: Specificity and Stability in Topology of Protein Networks. Science 296(5569), 910–913 (2002)
30. Newman, M.E.J.: Random Graphs with Clustering. Physical Review Letters 103(5) (2009)
31. Jeffrey, H., Danah, B.: Vizster: Visualizing Online Social Networks. In: InfoVis, pp. 32–39. IEEE, Los Alamitos (2005)

Distance Distribution and Average Shortest Path Length Estimation in Real-World Networks

Qi Ye, Bin Wu, and Bai Wang

Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia
Beijing University of Posts and Telecommunications, Beijing, China, 100876
yeqibupt@gmail.com, {wubin,wangbai}@bupt.edu.cn

Abstract. The average shortest path length is one of the most important and frequent-invoked characteristics of real-world complex networks. However, the high time complexity of the algorithms prevents us to apply them to calculate the average shortest path lengths in real-world massive networks. In this paper, we present an empirical study of the vertex-vertex distance distributions in more than 30 artificial and real-world networks. To best of our knowledge, we are the first to find out the vertex-vertex distance distributions in these networks conform well to the normal distributions with different means and variations. We also investigate how to use the sampling algorithms to estimate the average shortest path lengths in these networks. Comparing our average shortest path estimating algorithm with other three different sampling strategies, the results show that we can estimate the average shortest path length quickly by just sampling a small number of vertices in both of real-world and artificial networks.

Keywords: Social Network Analysis, Average Shortest Path length, Vertex-vertex Distance, Sampling.

1 Introduction

Recently, massive network data sets are accumulating at a tremendous pace in various fields. Most of the real-world networks follow one prominent structural phenomena: the small-world phenomenon [1]. By using the measure of average shortest path length measure, small-world networks can be seen as systems that are both globally and locally efficient [2]. So we often use average shortest path length as a measure of network efficiency, and this efficiency measure allows us to give a precise quantitative analysis of the information flow efficiency in the networks. However, very few studies have formally investigated the issue of the characteristics of vertex-vertex distance distributions in massive real-world networks. In this paper, we present an empirical study on the vertex-vertex distance distributions in more than 30 real-world and artificial networks. Our main contributions can be summarized as follows:

- We propose one of the largest detailed studies of vertex-vertex distance distributions in more than thirty real-world networks. To best of our knowledge,

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 322–333, 2010.

we are the first to find the vertex-vertex distance distributions in these networks conform well to the normal distributions with different means and variations.
- We prove that if the vertex-vertex distances in a network follow the same distribution, we can easily estimate its average shortest path distance accurately by sampling a small fraction of vertex pairs.
- To estimate the average shortest path length in massive networks, we compare different sampling strategies. We suggest a simple and fast sampling algorithm to estimate the average shortest path length.

This remainder of the paper is organized as follows. Section 2 surveys the related work. In Section 3, we show the distributions of vertex-vertex distances in various real graphs. Section 4 first evaluates different sampling algorithms in these real-world networks. We also show the performance on the sampling algorithms in 6 pure types of artificial networks. Section 5 concludes this paper.

2 Related Work

The problem of finding shortest paths in networks is one of the most basic and well studied problems in graph theory and computer science. Zwick [3] gives a survey about the exact and approximate distance computing algorithms in networks. The shortest-path problem can be roughly divided into two categories: Single-Source Shortest Paths (SSSP) problem and the All-Pairs Shortest Paths (APSP) problem. To extract shortest-path length of a single source vertex to all other vertices in a unweighted graph with n vertices and m edges, it takes $O(m+n)$ time complexity by using Breadth First Search (BFS) [4]. If the graph is a weighted one, we can use the classical Dijkstra algorithm to handle the SSSP problem. Its time complexity is $O(m \log n)$ if we use a simple heap-based priority queue, and its time complexity is $O(m + n \log n)$ if we use the Fibonacci heap as the priority queue [3]. Floyd-Warshall algorithm [5] uses dynamic programming to solve the all-pairs shortest paths (APSP) problem in time complexity $O(n^3)$, and it costs time $O(mn)$ to get all-pair shortest paths in a sparse unweighted network by using BFS algorithm. So the exact average shortest distance in a sparse unweighted network would take $O(mn)$ time by using the BFS algorithm. Exact computation of single source shortest paths problem from every vertex is infeasible for many massive networks, and there are many approximate algorithms have been proposed. ALT algorithms employ landmarks to prune the search space of the shortest path computation [6]. Potamias et al. [6] show that selecting the optimal set of landmarks is an NP-hard problem. Kleinberg et al. [7] discuss the problem of approximating network distances in real networks via embeddings using a small set of beacons.

Watts and Strogatz [1] find a large number of real-world networks share the small-world phenomenon. Albert et al. [8] first report the World Wide Web displays the small-world property. Newman [4] shows the basic statistics for number of published networks, such as number of vertices and edges, mean degrees, mean vertex-vertex distances, clustering coefficients, degree correlation coefficient, etc. We also note they do not give the average (mean) shortest path length for the large networks containing hundreds of thousands of vertices and edges as the high computation complexity. Faloutsos et al. [9] study the power-laws of Internet topology, and they propose the effective diameter to show how many hops are required to reach a "sufficiently large" part of the network. Based on this idea we can estimate the effective distance between vertices more accurately. Latora and Marchiori [2] use the network efficiency E which is the mean of all the inverse of vertex-vertex shortest path lengths to measure the efficiency of the vertices exchanging information in networks. Lee et al. [10] study the statistical properties of the sampled scale-free networks in various real-world networks by three subgraph sampling algorithms. They exploit the properties of sampled networks, such as degree distribution, betweenness centrality distribution, mean shortest path length, clustering coefficient, etc.

3 Vertex-Vertex Distance Distributions

In this section, we present an empirical study of the vertex-vertex distance distributions in more than 30 real-world and artificial networks.

3.1 Mean Geodesic Distance Functions

There, an undirected graph G with n vertices and m edges is considered, and G is assumed to be unweighted and without self-loops. We can define the mean shortest path length ℓ between any vertex-vertex pairs in the network. There are several definitions of mean shortest path length have been proposed, and Newman gives the following definitions:

$$\ell = \frac{1}{\frac{1}{2}n(n+1)} \sum_{i \geq j} d_{i,j} . \tag{1}$$

However in most real-world networks, there are usually a giant component and many small ones in each network. As there exist vertex-vertex pairs that have no connecting path between them in real-world networks, Newman also defines the mean "harmonic mean" distance, i. e., the reciprocal of the average of reciprocals:

$$\ell^{-l} = \frac{1}{\frac{1}{2}n(n+1)} \sum_{i \geq j} d_{i,j}^{-1} . \tag{2}$$

In Eq. 2, infinite values of $d_{i,j}$ then contributes zeros to the sum. We also note that the definition of "harmonic mean" distance is in fact the reciprocal of the definition of the *efficiency* of G proposed by Latora and Marchiori [2]. However, we notice that as in Eq. 2 and Eq. 1 the value of $d_{i,i}$ which is the distance from vertex i to itself should be zero, and the value of $d_{i,i}^{-1}$ will be infinite. So we do not calculate the distances from each vertex to itself just as the definition of average shortest path defined by Latora and Marchiori [2] and redefine the

Table 1. Data sets of real-world networks

Network	$\|V\|$	$\|E\|$	$\|V_{gc}\|$	$\|E_{gc}\|$	ℓ	ℓ_h	D	σ	R^2	$\ell_{0.01}$
Karate	34	78	34	78	2.41	2.03	5	0.93	0.84	-
Dolphin	62	159	62	159	3.36	2.64	8	1.48	0.89	-
Pol-book	105	441	105	441	3.08	2.52	7	1.42	0.91	3.09
Football	115	613	115	613	2.51	2.22	4	0.73	0.91	2.55
Word	121	425	121	425	2.54	2.26	5	0.77	0.99	2.50
Email	1133	5451	1133	5451	3.61	3.33	8	0.90	1.00	3.64
Jazz	198	2742	198	2742	2.24	1.95	6	0.78	0.97	2.27
Elegans-neu	297	2148	297	2148	2.46	2.25	5	0.67	0.99	2.39
Elegans-meta	453	2025	453	2025	2.66	2.46	7	0.75	0.97	2.66
Pol-blog	1490	16715	1222	16714	2.74	2.51	8	0.77	0.99	2.70
Net-sci	1589	2742	379	914	6.04	4.92	17	2.34	0.98	5.98
Power-grid	4941	6594	4941	6594	18.99	15.90	46	6.51	0.99	18.96
CA-GrQc	5242	14484	4158	13422	6.05	5.58	17	1.57	0.99	5.97
Hep-th	8361	15751	5835	13815	7.03	6.44	19	1.91	0.99	7.03
PGP	10680	24316	10680	24316	7.49	6.76	24	2.27	0.98	7.43
Astro-ph	16706	121251	14845	119652	4.80	4.45	14	1.28	0.98	4.81
Cond-Mat	40421	175693	36458	171736	5.50	5.20	18	1.26	0.99	5.51
As06	22963	48436	22963	48436	3.84	3.62	11	0.90	0.99	3.82
Gnu04	10876	39994	10876	39994	4.64	4.43	10	0.91	1.00	4.62
Gnu05	8846	31839	8843	31837	4.60	4.39	9	0.91	1.00	4.57
Gnu06	8717	31525	8717	31525	4.57	4.37	10	0.90	1.00	4.58
Gnu08	6301	20777	6299	20776	4.64	4.41	9	0.97	1.00	4.62
Gnu09	8114	26013	8104	26008	4.77	4.54	10	0.96	1.00	4.77
Gnu24	26518	65369	26498	65359	5.41	5.23	11	0.94	1.00	5.43
Gnu25	22687	54705	22663	54693	5.54	5.35	11	0.96	1.00	5.56
Gnu30	36682	88328	36646	88303	5.75	5.56	11	0.97	1.00	5.74
Gnu31	62586	147892	62561	147878	5.94	5.76	11	0.96	1.00	5.93
Enron	36692	183831	33696	180811	4.03	3.81	13	0.92	0.98	4.03
EuIns	265214	364481	224832	339925	4.12	4.03	14	0.61	0.97	4.12
Epini	75879	405740	75877	405739	4.31	4.09	15	0.97	0.98	4.33
Slash08	77360	469180	77360	469180	4.02	3.85	12	0.82	0.99	4.04
Slash09	82168	504230	82168	504230	4.07	3.89	13	0.83	0.99	4.06
Amazon0302	262111	899792	262111	899792	8.83	8.41	38	1.89	0.99	8.83
Amazon0312	400727	2349869	400727	2349869	6.45	6.02	20	1.23	0.99	6.45
Amazon0505	410236	2439437	410236	2439437	6.45	6.20	22	1.26	0.99	6.45

the definitions as follows. The all mean shortest path length in network G is redefined as the following:

$$\ell = \frac{1}{\frac{1}{2}n(n-1)} \sum_{i>j} d_{i,j}. \tag{3}$$

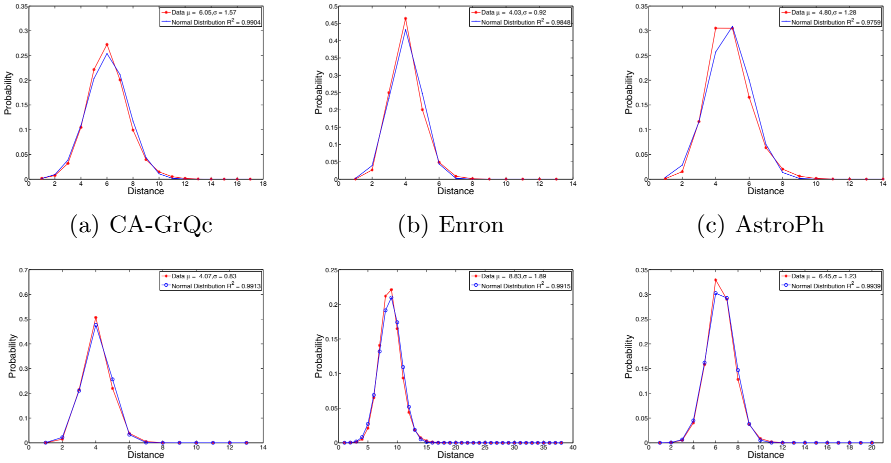
And we also redefine the mean "harmonic mean" distance as:

$$\ell^{-l} = \frac{1}{\frac{1}{2}n(n-1)} \sum_{i>j} d_{i,j}^{-1}. \tag{4}$$

In this paper, we restrict attention to the giant components of these graphs, so we can freely use Eq. 3 and Eq. 4 in these giant components.

3.2 Data Sets

To study the vertex-vertex distance distributions in real-world networks, we run detailed experiments on a lot of public available network data sets. The networks



(a) CA-GrQc (b) Enron (c) AstroPh

(d) slash09 (e) Amazon0302 (f) Amazon0312

Fig. 1. Vertex-vertex distance distributions in giant components in the real-world networks

are provided by Newman[1], Arenas[2] and Leskovec[3]. Please see these web sites for more details. Table 1 shows the basic properties of each network, such as the vertex number $|V|$, the edge number $|E|$, the vertex number in the giant component $|V_{gc}|$ and edge number in the giant component $|E_{gc}|$, the average shortest path length ℓ, the "harmonic mean" shortest path length ℓ_h and the diameter D, etc.

3.3 Vertex-Vertex Distance Distributions

We study the vertex-vertex distance distributions in the giant component $G_{gc} = \{V_{gc}, E_{gc}\}$ of the each network just as the method used by Watts and Strogatz [1]. Table 1 shows the mean shortest path length ℓ and the "harmonic mean" distance ℓ_h of each graph. There still exists significant difference between

the values of ℓ and ℓ_h, so we do not regard the 'harmonic mean" ℓ_h as an accurate value to show the ℓ even if there all the vertices are reachable. Fig. 1 shows the vertex-vertex distance distributions of some real-world networks in Table 1. We also get the mean shortest path ℓ length and the standard deviation σ of the distance distribution for each network. We find that the vertex-vertex distance distributions in the real-world networks are almost invariably symmetrical. By using the same values of ℓ and σ, we fit each vertex-vertex distance distribution with a normal distribution with the same values of mean and deviation. To calculate the difference between the normal distribution and the real-world

[1] http://www-personal.umich.edu/∼mejn/netdata/

[2] http://deim.urv.cat/∼aarenas/data/welcome.htm

[3] http://snap.stanford.edu

distribution, we also get the the coefficient of determination R^2 of these two distributions. To our surprise, we find all the vertex-vertex distance distributions of networks in Table 1 conform well with the fitted normal distributions. The coefficients of determination R^2 in the networks which contains more than 1000 vertices are all above 0.97. As shown in Table 1, most of vertex-vertex distances are narrowly distributed near the mean shortest path length ℓ, and most of the standard deviations σ are below 2 except that of the Power-grid network. We also note that for different snapshots of the same network, the vertex-vertex distributions may vary greatly due to the changes of the network sizes, such as the Amazon0302 and Amazon0312 networks.

4 Sampling Algorithms

4.1 Sampling Algorithms

Given a graph $G = (V, E)$, an induced graph $G[T]$ is a subgraph that consists vertex set T and all the edges whose endpoints are contained in T. There are many ways to create a sampled subgraph G' from the original $G(V, E)$. In the following parts, we will use 4 sampling algorithms called random vertex subgraph sampling, random edge subgraph sampling, snowball subgraph sampling and the random vertex sampling to estimate the mean shortest path length of the original graph. The sampling fraction of the algorithms is defined as the ratio of selected vertices to the total vertex number n in the original graph, and we also just calculate the average vertex-vertex distances in the giant components.

- **Random vertex subgraph sampling (RVS):** Select a vertex set S from the original graph G randomly, and keep the edges among them. The subgraph G' can be formalized as getting the induced subgraph $G' = G[S]$ of the vertex set S [10].
- **Random edge subgraph sampling (RES):** Select edges randomly from the original graph G, and the vertices attached to them are kept. The subgraph G' containing the links is the sampled subgraph [10].
- **Snowball subgraph sampling (Snowball):** Select a single vertex from the graph randomly from the original graph and all its neighbors are picked. Then all the vertices connected to the selected ones are selected. This process is continued until the desired number of vertices are selected. To control the number of vertices in the sampled graph, a certain number of vertices are randomly selected from the last layer [10].
- **Random vertex sampling (RV):** Selecting a vertex set S randomly from the giant component. Perform a BFS computation from each vertex $i \in S$ to all vertices in G, and calculate the mean shortest path length. In a sparse graph which contains n vertices and m edges, it takes $O(m + n)$ time by using the Breadth First Search (BFS) algorithm to get all the shortest path lengths from one source vertex $i \in S$ to all other vertices. The sampling algorithm takes $O((m + n)|S|) = O(n|S|)$ to the average shortest path in sparse networks. We also note that Leskovec et al. [14] use similar method to

get the mean vertex-vertex distance in the MSN network, and Potamias [6] use similar method to calculate the closeness values of vertices. Although these these similar methods have been proposed so far, but none of them has been subjected to strict tests to evaluate their performance.

4.2 Mean and Variance of Random Vertex Sampling

Suppose in a graph G the vertex-vertex distance follows the same distribution with a finite mean ℓ and a finite variance σ^2, instead of considering its probability distribution. Let us select k pairwise vertices from the original graph, where k is very large integer. Let $D_1, D_2, \cdots, D_k$ to be a sequence of independent and identically distributed pairwise shortest path length random variables. It is easy to prove that the mean pairwise vertices shortest path length also follows the normal distribution with the mean shortest path length

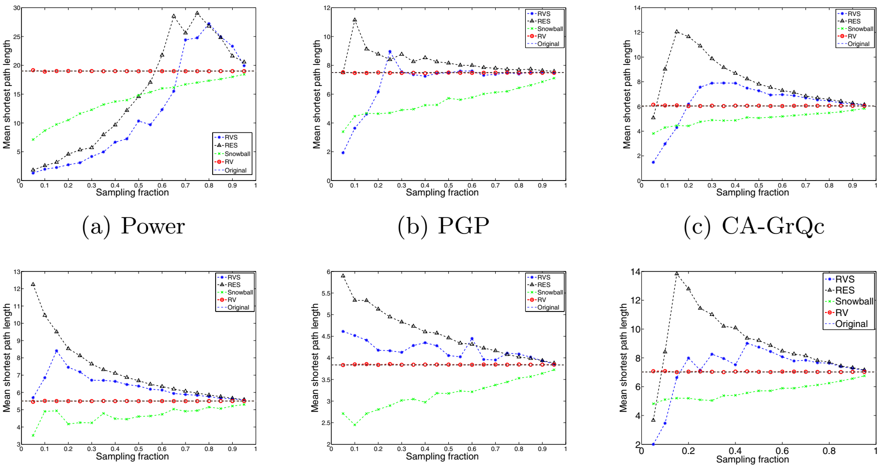
$$\ell_{rand} = \sum_{i=1}^{k} \frac{D_i}{k} \to \ell,$$

and its variance $\sigma^2_{rand} = \sum_{i=1}^{k} \frac{D_i}{k} = \frac{\sigma^2}{k}$ regardless the vertex-vertex distance distributions in the original graphs.

By using the random vertex (RV) sampling algorithm, suppose we select a vertex i from the original graph and calculating the sum of distances $\sum_{j>i} d_{i,j}$ from vertex i to all other vertices. The mean of the shortest path lengths ℓ_i from vertex i to other vertices is $\ell_i = \frac{\sum_{j>i} d_{i,j}}{n-1}$. Let us assume all the distances $d_{i,j}$ follow the same distribution with a finite mean ℓ and a finite variance σ^2. The mean of all vertex-vertex distances from the source vertex i is $\ell_i = l$ and its standard deviation is $\sigma_i = \frac{\sigma}{\sqrt{n-1}}$. So we can get that the mean distance ℓ_{rand} of the vertex set S is $\ell_{rand} = \sum_{i=1}^{|S|} \frac{\ell_i}{|S|} = \ell$ and its standard deviation σ_{rand} is $\frac{\sigma}{\sqrt{|S|(n-1)}}$. We can conclude that the random vertex sampling algorithm is a very efficient way to estimate the average shortest path lengths in massive graphs with just selecting a small number of vertices from the original graphs.

4.3 Empirical Study of Sampled Subgraphs

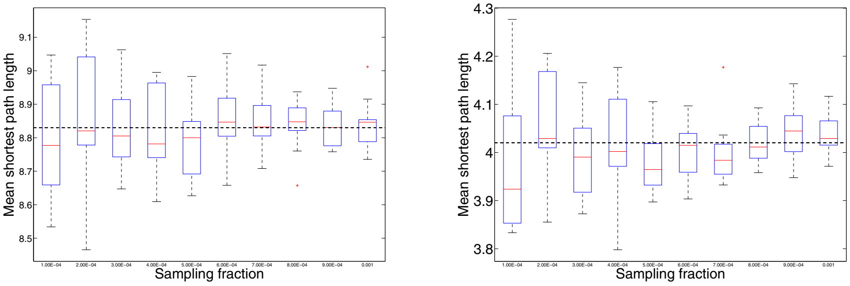
Fig. 2 shows the estimated average shortest path lengths of some networks in Table 1 got by different sampling algorithms, and the horizontal dashed lines are the values for the original mean shortest path length of each network. All the sampling ratios of the four sampling methods are the fraction of the number of chosen vertex number $|S|$ to the total node number $|V|$ in the original graph. Our random vertex (RV) sampling algorithm performs well in all graphs, and the estimated values are almost coincide with the original mean values in all the networks even the sampling ratio is very small. We can also find some interesting characteristics of other 3 sampling algorithms. In all these real-world networks shown in Fig. 2, the average shortest path lengths got by the random edge subgraph (RES) sampling algorithm are usually larger than those got by



(a) Power (b) PGP (c) CA-GrQc

(d) Cond-Mat (e) AS06 (f) HepTh

Fig. 2. Estimated average shortest path lengths according to different sampling ratios of different sampling algorithms. The horizontal dashed lines are the values for the original mean shortest path length of each network.



(a) Amazon0302 (b) slash08

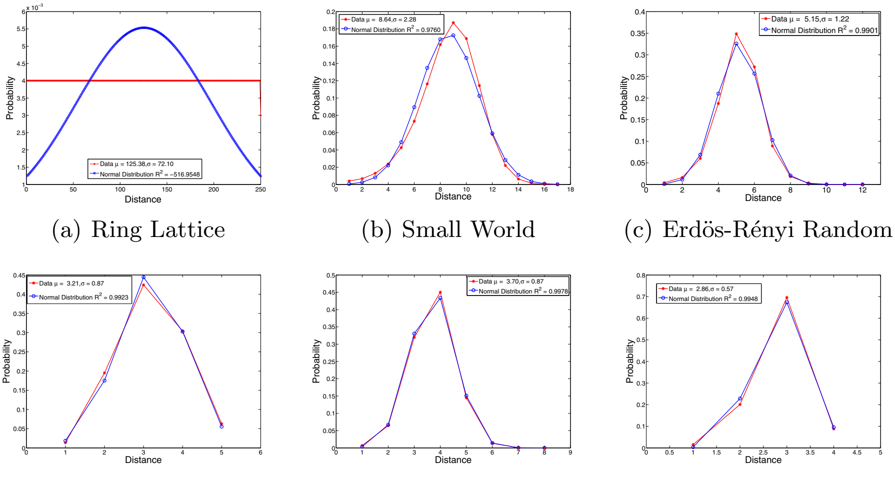
Fig. 3. Box plots of the changes of mean shortest path length for random vertex (RV) sampling algorithm according to very small sampling ratios of the giant components. The horizontal dashed lines are the values for the original mean shortest path length of each network.

the random vertex subgraph sampling algorithm. When we get enough vertices in the subgraphs, the estimated mean shortest path length got by the the Snowball sampling algorithm increases with the growing of the sampling fractions. Obviously, for the Snowball sampling, the diameter of the sampled subgraph is expanded gradually from the source vertex as the increasing of the number of vertices. We can also get that the values of estimated average shortest path length got by the random edge subgraph sampling (RES) are usually larger than those got by the random vertex subgraph (RVS) sampling in most cases.

To show the stability of the random vertex (RV) sampling algorithm in massive networks, we use the random vertex (RV) sampling algorithm by different sampling fractions to estimate the average shortest path length of the Amazon0302 network and slash8 network, and the results are shown in box plots in Fig. 3. To show the distributions of the estimation mean shortest path lengths in these 2 networks, the sampling the fraction is from 0.01% to 0.1%, and each point in this feature is calculated for 20 times. The horizontal dashed lines are also the values for the original mean shortest path length of each network. We can find that the random vertex (RV) sampling algorithm is very stable, and the median values of the boxes are distributed near the original mean shortest path length. To study the accuracy of the random vertex (RV) sampling algorithm in more networks, we sample 1% vertices from the networks and calculate the estimated mean shortest path length $\ell_{0.01}$ for each network in Table 1. In Table 1 the hyphen character indicates the networks are too small to sample 1% vertices in the giant components. As shown in Table 1, one can see that our random vertex (RV) sampling algorithm matches the experiments very well except when the graphs are very small.

4.4 Model Generated Graphs

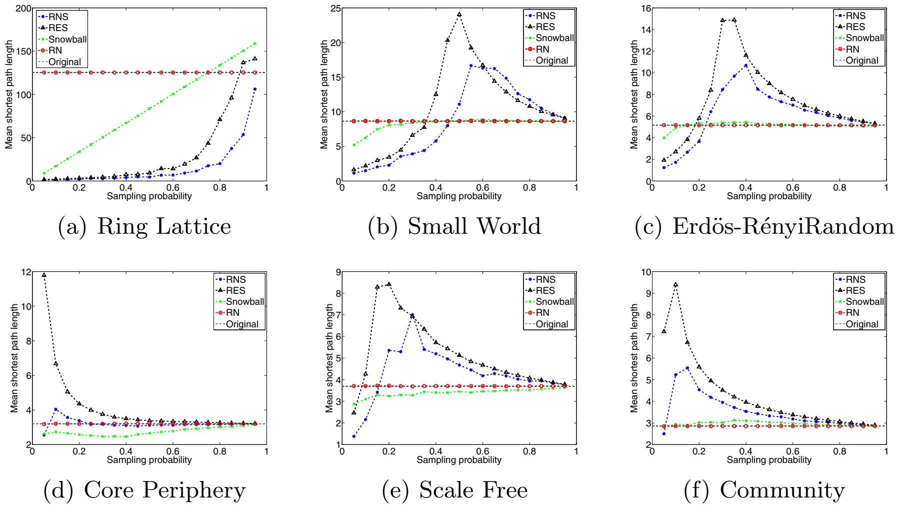
We also study the performances of the sampling algorithms for different artificial networks with different pure topological properties. In order to show the phenomena found in real-world networks, many network topology generators have been proposed [11] [12][13][1] [15]. Airoldi and Carley [11] proposed 6 pure topological types: ring lattice type, small world type, Erdös random type, core-periphery type, scale-free type and cellular type. We note that the cellular type models are



(a) Ring Lattice (b) Small World (c) Erdös-Rényi Random

(d) Core Periphery (e) Scale Free (f) Community

Fig. 4. Mean shortest path length distributions in the of giant components in the sampled networks generated by 6 typical network models



(a) Ring Lattice

(b) Small World

(c) Erdös-RényiRandom

(d) Core Periphery

(e) Scale Free

(f) Community

Fig. 5. Estimated mean shortest path length for each sampling algorithm according to different sampling ratios in the 6 artificial networks. The horizontal dashed lines are the values for the original mean shortest path length of each network.

actually to generate networks with built-in communities, so we call the cellular type model as community type model. To compare the sampling algorithms in these 6 typical topological networks, we choose the ring lattice model [11], Watts & Strogatz small world model [1], Erdös-Rényi random model [11], Albert & Barabasi model [11] [12], core-periphery model [11] and LFR community model [15], respectively. All these algorithms are available in Java in the network analysis framework **JSNVA** [16] as a part of **TeleComVis** [17]. To control for possible sources of variations we are not interested, such as the size of the network and density, each generated network contains 1000 vertices. Fig. 4 shows the vertex-vertex distance distributions in the 6 artificial networks. As the ring lattice network is a quite regular network, as shown in Fig. 4(a), almost all the vertex-vertex distances in the network generated by the ring lattice model have the same distribution probabilities, and the probabilities deviate greatly from the normal distribution. However, we can find that the other random networks generated by the other 5 type models, the vertex-vertex distance distributions conform well to the normal distributions.

Fig. 5 plots the estimated average shortest path lengths according to different sampling ratios in 6 artificial networks. Each value in Fig. 5 is computed by averaging the observations based on 10 samplings. The result shows that our random vertex (RV) sampling algorithm also perform best in these artificial networks. As shown in Fig. 5, we can also get that in all these typical topological networks the average shortest path lengths got by the random edge subgraph (RES) sampling algorithm are usually larger than those got by the random vertex subgraph (RVS) sampling algorithm. In the ring lattice random network, all of the RVS sampling algorithm, RES sampling algorithm and Snowball sampling algorithm perform bad when the sampling fractions are not large enough. As shown in Fig. 5(a), when the sampling fractions are below 0.6, the estimated shortest path lengths got by the RVS and RES sampling algorithms are still quite small. We also notice the Snowball sampling algorithm perform better than the RVS sampling algorithm and RES sampling algorithm in the other 5 pure topology artificial networks. However, there is a very good agreement between our random vertex (RV) sampling algorithm predictions and the real average shortest path length in each artificial networks. Our random vertex (RV) sampling algorithm performances much better than other sampling algorithms in all the cases both of real-world and artificial networks.

5 Conclusion

In this paper, we present an empirical study of the vertex-vertex distance distributions in more than 30 real-world networks. Our most striking finding is that in these networks conform well to the normal distributions with different means and variations. As scalability is a key issue for exploring the characteristic path length in massive network exploration, to estimate the average in massive networks, we prove that if the vertex-vertex distances in a network follow the same distribution, we can easily estimate its average shortest path distance accurately by just sampling a small fraction of vertex pairs. We suggest a simple and intuitive sampling algorithm to get the average shortest path length which is very fast and scales well to both of the massive graphs and the different pure topology artificial networks. The results show that we can estimated the average shortest path lengths quickly by just sampling a small number of vertices in massive networks. In real-applications, we can calculate the efficiency of massive networks quickly, such as estimation the efficiency of massive call graphs in real-time.

Acknowledgments. We thank M. E. J. Newman, Alex Arenas and Jure Leskovec for providing us the network data sets. This work is supported by the National Science Foundation of China (No. 90924029, 60905025, 61074128). It is also supported by the National Hightech R&D Program of China (No.2009AA04Z136) and the National Key Technology R&D Program of China (No.2006BAH03B05).

References

1. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
2. Latora, V., Marchiori, M.: Efficient Behavior of Small-World Networks. Phys. Rev. Lett. 87, 198701 (2001)
3. Zwick, U.: Exact and approximate distances in graphs—a survey. In: Proceeding of the 9th Annual European Symposium on Algorithms, pp. 33–48 (2001)
4. Newman, M.E.J.: The Structure and Function of Complex Networks. SIAM REVIEW 45, 167–256 (2003)

5. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT, Cambridge (2001)
6. Potamias, M., Bonchi, F., Castillo, C., Gionis, A.: Fast Shortest Path Distance Estimation in Large Networks. In: Proceeding of CIKM, pp. 867–876 (2009)
7. Kleinberg, J., Slivkins, A., Wexler, T.: Triangulation and embedding using small sets of beacons. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 444–453. IEEE Computer Society, Washington (2004)
8. Albert, R., Jeong, H., Barabási, A.-L.: Diameter of the world wide web. Nature 401, 130–131 (1999)
9. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the Internet topology. In: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, pp. 251–262. ACM, New York (1999)
10. Lee, S.H., Kim, P.J., Jeong, H.: Statistical properties of sampled networks. Phys. Rev. E. 73, 16102 (2006)
11. Airoldi, E.M., Carley, K.M.: Sampling algorithms for pure network topologies: stability and separability of metric embeddings. SIGKDD Explorations 7, 13–22 (2005)
12. Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. Rev. Mod. Phys. 74, 47–97 (2002)
13. Barabási, A.L., Albert, R.: Emergence of Scaling in Random Networks. Science 286, 509–512 (1999)
14. Leskovec, J., Horvitz, E.: Planetary-scale views on a large instant-messaging network. In: Proceeding of the 17th International Conference on World Wide Web, pp. 915–924 (2008)
15. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E. 78, 46110 (2008)
16. Ye, Q., Zhu, T., Hu, D., et al.: Cell Phone Mini Challenge Award: Exploring Temporal Communication in Mobile Call Graphs. In: 3rd IEEE Symposium on Visual Analytics Science and Technology, pp. 207–208. IEEE Press, Columbus (2008)
17. Ye, Q., Wu, B., et al.: TeleComVis: Exploring Temporal Communities in Telecom Networks. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, pp. 755–758. Springer, Bled Slovenia (2009)

Self-adaptive Change Detection in Streaming Data with Non-stationary Distribution

Xiangliang Zhang[1,⋆] and Wei Wang[2]

[1] Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology, Saudi Arabia
[2] Interdisciplinary Centre for Security, Reliability and Trust (SnT Centre),
University of Luxembourg, Luxembourg

Abstract. Non-stationary distribution, in which the data distribution evolves over time, is a common issue in many application fields, e.g., intrusion detection and grid computing. Detecting the changes in massive streaming data with a non-stationary distribution helps to alarm the anomalies, to clean the noises, and to report the new patterns. In this paper, we employ a novel approach for detecting changes in streaming data with the purpose of improving the quality of modeling the data streams. Through observing the outliers, this approach of change detection uses a weighted standard deviation to monitor the evolution of the distribution of data streams. A cumulative statistical test, Page-Hinkley, is employed to collect the evidence of changes in distribution. The parameter used for reporting the changes is self-adaptively adjusted according to the distribution of data streams, rather than set by a fixed empirical value. The self-adaptability of the novel approach enhances the effectiveness of modeling data streams by timely catching the changes of distributions. We validated the approach on an online clustering framework with a benchmark KDDcup 1999 intrusion detection data set as well as with a real-world grid data set. The validation results demonstrate its better performance on achieving higher accuracy and lower percentage of outliers comparing to the other change detection approaches.

Keywords: Change detection, Data stream, Self-adaptive parameter setting, Non-stationary distribution.

1 Introduction

Mining data streams [1,2,3,4,5,6], one major task of Data Mining, aims at providing a compact description of the data flows generated by e.g., telecommunications, sensor networks, or Internet traffic. Mining data streams works under severe algorithmic constraints due to the size and dynamics of the data flow, since the underlying distribution of the data flow continuously evolves with the

[⋆] The work in algorithm part was done when the first author was a Ph.D. student at INRIA and at Université Paris-Sud 11, France, and the application part for intrusion detection was done when the first author visited SnT, University of Luxembourg.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 334–345, 2010.

users, the usage and the application context, etc. Mining data streams provides solutions for various applications, e.g., autonomic computing [7], anomaly detection [1], or computing approximate statistics [8]. In our previous work, for example, in order to diagnose EGEE grid (Enabling Grid for E-SciencE [1]), we exploited the gLite reports on the lifecycle of the jobs and on the behavior of the middleware components for providing the summarized information of grid running status [9].

One important issue in mining data streams is the inherent property of their **non-stationary distribution**. In network intrusion detection system, for example, in experiments carried out by MIT Lincoln Lab for the 1998 DARPA evaluation [10], network traffic over 7 weeks contains 5 GB of compressed binary tcpdump data which were processed into about five million connection records. The behaviors of each network connection are clearly in non-stationary distribution due to the evolution of the phenomenon, e.g., the traffic, the users and the modes of usage. More generally, the non-stationary data distribution exists in many other application domains, such as sensor network signals, web logs and grids.

One of the major challenges for handling this non-stationary distribution of data streams is to detect the changes in the underlying data distribution. Usually the changes can be categorized into *normal concept drift (emergence of new pattern)*, *abnormal behaviors* and *noise*. In this paper, we refer these changes as *outliers*. According to the different categories of outliers, the detection of changes can help for

– data mining – indicates when to learn a new model in case of *normal concept drift (emergence of new pattern)* identified;
– anomaly detection – triggers alerts/alarms in case of *abnormal behaviors* identified, e.g., intrusions;
– data cleaning – filters the *noise*.

The goal of this paper aims at adaptively detecting changes in streaming data and at identifying the category of the outliers that were discovered by an online updating model. Among several existing online updating models [6,2,3], we used our previously proposed algorithm StrAP [4] as it gives better performance in terms of clustering accuracy [4,7].

In our recent work of [11], we proposed a novel approach of change detection, which helps to discover the unusual running status of grid. Through analyzing the experimental results of real-world EGEE grid jobs, it was shown that the frequency of detected changes is related to the appearance of the unusual grid status, e.g., heavy load, large number of distinct errors and specific device fault. These discovered faults of device are not claimed in the grid logs, but firstly suspected by causing the frequent detection of changes and then confirmed by the StrAP model.

To observe the evolution of distribution, the proposed approach firstly uses a weighted standard deviation for measuring the deviation of outliers from the

[1] http://www.eu-egee.org/, the largest grid infrastructure in the world.

online updating model. Based on a cumulative statistical test, the so-called Page-Hinkley test (PH) [12,13], the evidence of changes in distribution is cumulatively collected. The parameter used for reporting the changes is self-adaptively adjusted according to the distribution of data streams, rather than set by a fixed empirical value. This approach is strictly data distribution-free under the null hypothesis.

This paper extends our work in [11]. We use the self-adaptive approach of change detection for **enhancing the quality of modeling data streams**. The timely reported changes of distribution are expected to improve the accuracy of online clustering model and to reduce the percentage of outliers which are used to update the model for enhancing the efficiency. As an extension of the work in [11], these two advantages comprehensively verify the ability of the proposed approach for effectively detecting changes in data streams with non-stationary distribution. In order to validate the approach, besides using the real-world EGEE grid data in [11], we mainly employ the benchmark KDDcup 1999 intrusion detection data set in our experiments. The validation results show that our approach outperforms the other methods of change detection in the original StrAP algorithm [4,7], in terms of higher accuracy of clustering and lower percentage of outliers.

The remainder of this paper is organized as follows. In Section 2, we introduce and discuss the related work. In Section 3, we present the self-adaptive approach for change detection. Section 4 reports the validation results on EGEE jobs and on KDDcup 1999 data. Conclusion and perspectives follow in Section 5.

2 Related Work of Change Detection

In general, the "changes" of distribution in streaming data can be detected by comparing the current window of streaming data items to a reference data distribution. For example, sketch based techniques can be used for detecting the changes of individual items with a big frequency. Another widely used approach is non-parametric change detection. It has few parameters to set, but must specify when to call a change significant as did in [14,15].

Dasu et al. use relative entropy, also called the Kullback-Leibler distance, to measure the difference between two given distributions [14]. The KL-distance is known to be related to the optimal error in determining whether the two distributions are the same and draws on fundamental results in hypothesis testing. The first step of this approach is to partition the reference data which is a part of the firstly arrived streaming data. Then discrete probability p over the partitioned regions is computed. After that, the same space division over a window of recent streaming items is applied and discrete probability q is computed. Then the KL divergence of recent stream items and the reference data is computed by $D(p||q) = \sum_x p(x) log_2 p(x)/q(x)$. A statistical inference procedure based on the theory of bootstrapping is used to tell the significant of KL divergence.

Song et al. defined a statistical test called the *density test* for detecting changes [15]. The density test identifies whether the newly observed data points S' are sampled from the underlying distribution that produced the baseline data set S.

Aggarwal proposed an approach of change detection by analyzing the trend of data [16]. Data density as well as changes of data density are estimated to discover the evolution of data. The density of the data at a given point is estimated by *Kernel density estimation* (KDE). The rate at which the changes in the data density are occurring is estimated by *Velocity density estimation* (VDE) based on some user-defined temporal window. Based on KDE and VDE, *temporal velocity profiles* and *spatial velocity profiles* are created. These two profiles are then used in order to predict three kinds of data evolution: *dissolution*, *coagulation* and *shift*.

The change detection methods in [14,15,16] are all based on stationary reference data (a.k.a. baseline data). In many application areas, however, the data is typically streaming and not stationary. In this paper, we employ a self-adaptive method of change detection based on the outliers that were discovered by a dynamic online model. It has the autonomic ability of change detection due to the adaption of parameters according to the data distribution and its inheritance of the functionality of streaming StrAP [4].

3 Self-adaptive Change Detection in Non-stationary Data Distribution

Part of the algorithm has been described in [11]. In this Section, for the sake of completeness, we firstly describe the standard Page-Hinkley statistical test [12,13]. In the following Section 3.2, we define a weighted standard deviation of the outliers w.r.t. the detection models in order to measure the trend of occurring outliers in streaming data. Finally in Section 3.3, we discuss a self-adaptive threshold for change detection.

3.1 Page-Hinkley Statistical Test

Formally, let p_t denote the observation of the variable at time t in streaming data with a non-stationary distribution. To detect the changes of the distribution of variable p_t, the PH test first computes the average of $p_1, \ldots, p_t$, denoted by $\bar{p}_t$. It then considers the variable m_t defined as the sum of the difference $p_\ell - \bar{p}_\ell + \delta$ between p_ℓ and $\bar{p}_\ell$ for $\ell = 1...t$, where δ is a tolerance parameter and usually set to a small value, e.g., 10^{-2}. The maximum value M_t of m_l for $\ell = 1...t$ is also computed. The difference between M_t and m_t is monitored as the measurement of changes, denoted by PH_t. The change of distribution is reported when the difference PH_t is larger than a given threshold λ. The variables in PH test are described as follows [12,13]:

$$\begin{aligned} \bar{p}_t &= \tfrac{1}{t}\textstyle\sum_{\ell=1}^{t} p_\ell \\ m_t &= \textstyle\sum_{\ell=1}^{t} (p_\ell - \bar{p}_\ell + \delta) \\ M_t &= max\{m_\ell, \ell = 1...t\} \\ PH_t &= M_t - m_t \\ \texttt{If } PH_t &> \lambda, \quad \text{change is detected} \end{aligned} \tag{1}$$

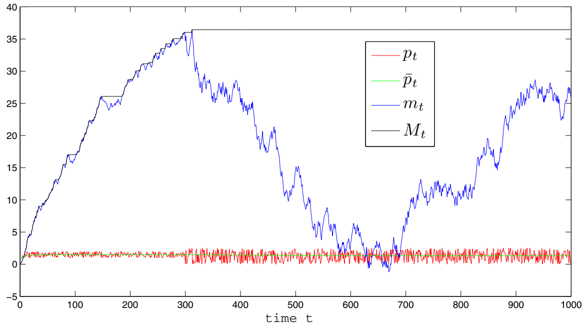


Fig. 1. The demonstration of change detection by PH

Fig. 1 shows an example of a distribution changing along time, as well as the variables in Equation (1) illustrating how PH is used to detect the changes. In this figure, red line p_t is the non-stationary distribution (change happened after 300). $\bar{p}_t$, m_t and M_t are computed from Equation (1), where δ is set to a very small value 10^{-2}. The gap between M_t and m_t, i.e., PH_t, keeps increasing after the change happened at 300. A threshold λ can be set to report the detected change.

Setting parameter λ is an important issue for detecting the changes. Fixing λ by an empirical value hinders reporting changes in a timely fashion. A too large value of λ would delay or miss the detection of changes, while a too small one would falsely alarm the changes frequently. Considering the evolution of distribution in data stream, a self-adapted threshold λ catches better the changes.

In our previous work, the λ adaption problem is considered as a sequential control task that can be handled using an exploration-exploitation algorithm [7]. The setting of λ is adjusted in order to obtain better clustering model. The quality of clustering model is measured by the Bayesian Information Criterion (BIC) [17], which is defined as the *distortion loss of the model* + the *size of model* + the *proportion of outliers*. The proportion of outliers comparing with the clustering model is defined as the observation object p_t in PH test. In order to find out the appropriate setting of λ for minimizing the BIC cost, we used two approaches. One is e-greedy action selection approach for selecting discrete value for λ. The other is Gaussian Process regression approach for generating continuous value for λ. In both of these two approaches, we record the quality measurement of the clustering model and the score of each single value of λ that have been used. The next candidate value of λ is selected or generated w.r.t. the quality of model and the score of used λ values.

The approaches we proposed previously in [7] formalize the adaptation of the threshold λ by an optimization problem. This is one of the solutions for adaptively setting λ. However, there are several difficulties. The first difficulty is that it defines the observation object p_t as the proportion of outliers w.r.t. the clustering model. This definition of p_t ignores the time-dependence of outliers, which is important for categorizing outliers. The densely arriving outliers would be the *normal concept drift* while irregularly arriving outliers can be *noise* or *abnormal* behavior. The second difficulty is that the BIC object function for optimization has to be computed frequently, increasing the computational cost.

In this paper, the proposed approach self-adaptively detects the changes by i) defining the observation object p_t based on the weighted standard deviation of outliers; ii) adapting the threshold λ according to the evolution of distribution.

3.2 Definition of p_t Based on Weighted Standard Deviation

Let u_t denote an outlier, a data item which arrives at time t and deviates from the model of a non-stationary distribution. For $t > 1$, there will be a sequence of outliers $u_i, \ldots, u_t$. It is worth noting that two neighbors u_i and u_j in the outlier sequence might not have the relationship $i + 1 = j$, because the subscript i or j is the time when u_i or u_j arrived. If u_t comes densely with similar values, these outliers could be a new pattern. If u_t irregularly comes with a value in a large range, they can be noise or rare anomalies. Therefore, we define the weighted standard deviation τ_t by simultaneously considering the value of u_t and the time l_t when u_t appears:

$$\tau_{1\to t} = \sqrt{\frac{1}{t}\sum_{i=1}^{t}\omega_i(u_i - \bar{u}_{1\to t})^2} \qquad (2)$$

where $\bar{u}_{1\to t} = \frac{1}{t}\sum_{i=1}^{t} u_i$ is the moving average of u_i, the coefficient $\omega_i = log(l_i - l_{i-1}) + 1$, l_i and l_{i-1} are the time when u_i and u_{i-1} arrive. It is obvious that when $\omega_i \equiv 1$ or $l_i - l_{i-1} = 1$ (u_i comes uniformly), weighted standard deviation (std) is the estimation of squared variance of u_t.

From Equation (2), we know that τ_t will keep decreasing towards a small value (close to 0) when u_t comes densely with similar values. PH test can be used to detect the changing trend of τ_t, by defining p_t as the weighted std τ_t computed in a sliding window in size of d along u_t:

$$p_t = \tau_{(t-d+1)\to t} = \sqrt{\frac{1}{d}\sum_{i=t-d+1}^{t}\omega_i(u_i - \bar{u}_{(t-d+1)\to t})^2} \qquad (3)$$

3.3 Setting of Threshold λ

As discussed in Section 3.1, when PH test is used for detecting the changes of u_t, the threshold λ reporting a change is expected to be self-adapted in the process of handling streaming data. Therefore, in our approach the threshold λ is adjusted in real-time, i.e., λ_t is computed at each time step t as shown in Proposition 1.

Proposition 1. *The threshold for detecting changes by PH test can be computed as*

$$\lambda_t = \begin{cases} 0 & \textit{if } PH_t = 0 \\ f * \bar{p}_t & \textit{otherwise} \end{cases}$$

or

$$\lambda_{t_0} = \begin{cases} 0 & \textit{if } PH_t = 0 \\ f * \bar{p}_{t_0} & \textit{otherwise} \end{cases}$$

where f is a constant called the λ factor, which is the number of required witnesses seeing the changes, e.g., $f = 30$. t_0 is the moment when $PH_{t_0} \neq 0$. $\bar{p}_t$ and $\bar{p}_{t_0}$ are the moving average computed after Equation (1).

Proof. As defined in Equation (1), a change is detected if $PH_t > \lambda$. In order to see how to set λ, firstly we study how the non-negative PH_t increases. As known from Equation (1), if $PH_{t-1} \neq 0, PH_t \neq 0$, the increase from PH_{t-1} up to PH_t is

$$PH_t - PH_{t-1} = (M_t - m_t) - (M_{t-1} - m_{t-1})$$

Since $PH_t > PH_{t-1} > 0$, we have $M_t \equiv M_{t-1}$. Then

$$PH_t - PH_{t-1} = m_{t-1} - m_t = m_{t-1} - (m_{t-1} + p_t - \bar{p}_t + \delta)$$

$$= \bar{p}_t - p_t - \delta$$

Therefore, since $PH_{t_0} > 0$ (i.e., $\bar{p}_{t_0} > p_{t_0} + \delta$), for $t \geq t_0$, we have

$$PH_t = \sum_{i=t_0}^{t} (\bar{p}_i - p_i - \delta) \tag{4}$$

From Equation (4), it is seen that PH_t is the collection of deviation of p_t from $\bar{p}_t$. The scenario of changes happening is the weighted std p_t decreasing towards 0. To be sure of the effective changes, not fake anomalies, evidence should be collected in longer time. We define a λ factor, called f, to be the number of steps when PH_t keeps increasing. As p_t is decreasing towards 0, δ is a very small value (10^{-2}), and $\bar{p}_t$ decreases slowly, we have $PH_t \approx f * \bar{p}_t$ after f steps. Therefore, the first option for setting λ is

$$\lambda_t = f * \bar{p}_t$$

To avoid computing λ_t frequently, it can be set immediately when $PH_t > 0$. Then the second way for setting λ is

$$\lambda_{t_0} = f * \bar{p}_{t_0}$$

where t_0 is the moment when $PH_{t_0} \neq 0$.

In Proposition 1, λ_t is computed according to inner variable $\bar{p}_t$ which reveals the changing trend of p_t. The only shortcoming is the empirically defined constant f. Fortunately, the meaning of f is the number of waiting steps before making the decision. It is independent and has no relationship with any other variables, e.g., p_t, u_t. Therefore, we can set it as a common value, e.g., 30.

4 Experiments and Results

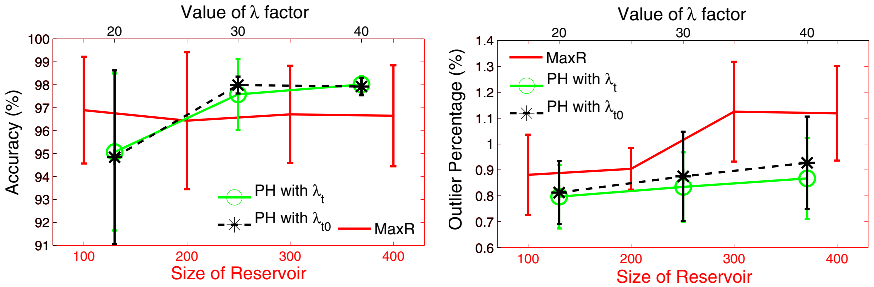
We validate the self-adaptive approach for change detection under the framework of StrAP algorithm proposed in our paper [4]. StrAP combines a clustering method called Affinity Propagation (AP) with the change point detection test. AP is used for summarizing the streaming data while change detection is used for catching the non-stationary distribution in the streaming data. In the original StrAP framework, there are several ways for detecting the changes. One is called “MaxR” which triggers a change through specifying the maximum number of outliers referred to as “size of reservoir”, as an outlier is put into the reservoir as soon as it is discovered. The other ways are based on PH test with threshold λ specified by a fixed-value, or with threshold λ adapted by optimization approaches, e.g., e-greedy selection and Gaussian Process Regression [7].

The goal of the validation is to show that the approach of self-adaptive change detection can achieve better clustering accuracy with less percentage of outliers, comparing to the other ways in original StrAP method. The cluster accuracy measures the model of StrAP in a fashion of supervised learning. Higher accuracy means the better quality of the model in terms of summarizing the patterns of data streams. The lower percentage of outliers indicates the better quality of the model in terms of tracking the evolving distribution of data streams.

The first data set we used for the validation is network connection data used in KDD Cup 1999 Intrusion Detection contest [18,19]. The task is to build a predictive model (i.e., a classifier) capable of distinguishing between “bad” connections, called intrusions or attacks, and “good” normal connections. A connection is a sequence of TCP packets starting and ending at some well defined times, flowing from a source IP address to a target IP address under some well defined protocol. Each connection is labeled among 23 classes, the *normal* class and the specific kinds of attack, such as *buffer_overflow*, *ftp_write*, *guess_passwd*, *neptune*.

The KDDcup 1999 data set we used includes 494,021 network connection records (71MB). Each record is described by 41 attributes among which we only use the 34 numeric ones. We treat the 494,021 connections as a stream (one connection per timestep). It is important to note that the data distribution is not stationary. The later-arriving part of the data includes specific attack types that are not in the beginning part of the data. The connections labeled by one type of attack can have different distribution if their arrival time steps have long intervals. This makes the task more realistic.

In the StrAP model, each cluster includes the connections behaving similarly and belonging to the same type. Each new arriving connection is associated to one cluster, or identified as an outlier. With the help of the class labels in the associated cluster, this online clustering process can classify the arriving connections. The accuracy is defined as the percentage of correctly classified connections. It is worth noting that the clustering model is online updated by detecting the changes to cope with the evolving data distribution.



(a) Clustering accuracy

(b) Percentage of outliers

Fig. 2. Performance of StrAP on KDDcup 1999 dataset: comparing self-adaptive change detection approach and "MaxR" depending on parameter setting

Fig. 2 shows the clustering accuracy and percentage of outliers using StrAP streaming data clustering method. We compare the performance of self-adaptive change detection approach to the baseline "MaxR" method in StrAP [4].

In Fig. 2, the bottom x axis shows the 4 different settings on "size of reservoir". On each of them StrAP was independently run 9 times, and the average with std of performances were computed. Similarly, for the approach of self-adaptive change detection, on each of 3 different settings on λ factor f (top x axis in Fig. 2), the performances of StrAP was averaged on 9 independent running results.

Fig. 2 (a) shows the averaged accuracy and Fig. 2 (b) shows the averaged percentage of outliers. It is seen that "MaxR" has stable accuracy on the "size of reservoir" setting and results in more outliers with the increasing of "size of reservoir". The approach of self-adaptive change detection has higher accuracy when $f \geq 30$ and lower percentage of outliers as expected. The two manners of self-adapting threshold λ_t and λ_{t_0} in Proposition 1 have the similar performance in terms of accuracy and of outlier percentage.

The second data set we used for the validation is the real-world EGEE jobs data. As described in [9], this data set includes 5,268,564 jobs from EGEE grid during 5 months (from 2006-01-01 to 2006-05-31). Each job is labeled by its final state, successfully finished (*good job*) or failed (*bad job*, including about 45 error types). The 5 million jobs include about 20 main error types (more than 1,500 occurrences). Using StrAP algorithm on these streaming jobs produces the visible summarized results to the administrators. The clustering accuracy is evaluated to guarantee that the compact description is correct.

In Fig. 3, we show the clustering accuracy of StrAP for clustering the EGEE job streams. As reported in [11], the performance of the self-adaptive change detection approach (real-time adapted threshold $\lambda_t = 30 * \bar{p}_t$) is compared to the performance of λ adapted by e-greedy and Gaussian Process in [7], and to that of the λ fixed by a given value 40. Meanwhile, we use the *streaming k-centers* as the baseline for the comparison. We develop *Streaming k-centers* based on the same framework of StrAP. The only difference is that we replace Affinity Propagation with k-centers as the clustering algorithm.

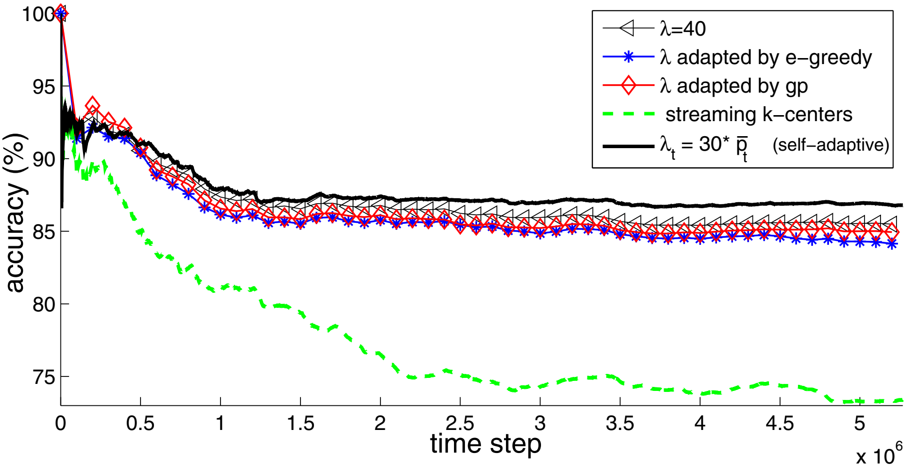


Fig. 3. Performance of StrAP on EGEE jobs in terms of **accuracy**, in comparison among self-adaptive change detection approach $\lambda_t = 30 * \bar{p}_t$, λ adapted by e-greedy and Gaussian Process in [7], and λ fixed by a given value 40

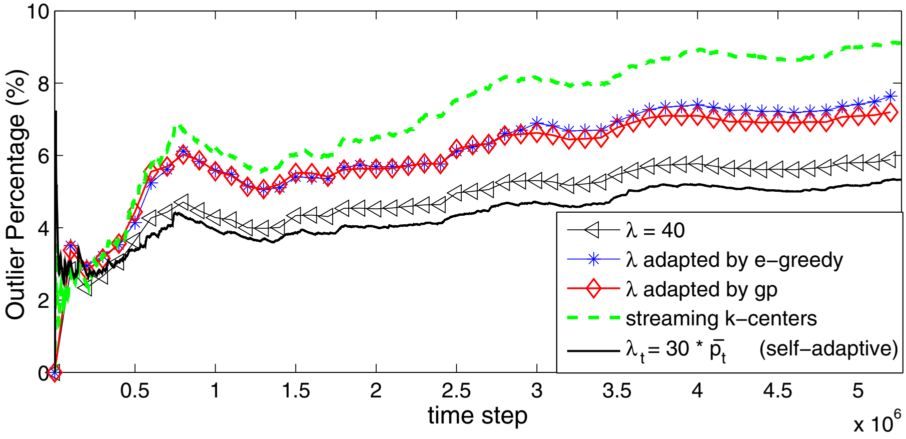


Fig. 4. Performance of StrAP on EGEE jobs in terms of **outlier percentage**, in comparison among self-adaptive change detection approach $\lambda_t = 30 * \bar{p}_t$, λ adapted by e-greedy and Gaussian Process in [7], and λ fixed by a given value 40

In Fig. 4, we show the **outlier percentage** of StrAP for clustering the EGEE job streams. Similarly, we compare the performance of our self-adaptive change detection approach with the other ways of setting threshold λ.

It is observed from Fig. 3 and Fig. 4 that our approach of self-adaptive change detection with real-time adapted threshold λ_t has higher accuracy and lower outlier percentage than all the other approaches, including our previous one adapting λ through optimization in [7].

5 Conclusion

In this paper, we extend our previous work reported in [11] by employing a self-adaptive approach for detecting the changes in data streams with non-stationary distribution. Besides the advantages of reporting the changes for alarming anomalies or failures in [11], our another purpose of detecting changes in this paper is to improve the quality of modeling data streams by timely catching the changes in the distribution. Through focusing on the outliers that deviate from the current model of data streams and through defining the observation object as weighted standard deviation of outliers, we use the Page-Hinkley statistic test to detect the change of distribution in data streams. The threshold λ of PH test for deciding the detection of changes is self-adapted rather than set as a fixed value. The improvement of the quality of modeling data streams is validated on KDDcup 1999 intrusion detection data and on the real-world grid job streams. The experimental results show the better performance achieved by combing the novel self-adaptive approach of change detection and the streaming model StrAP, in terms of higher accuracy and lower percentage of outliers than the other change detection methods.

There are three perspectives for our future work. First, we will apply this approach of self-adaptive change detection on more application fields, e.g., network measurement with network traffic (Netflow data). Second, we will diagnose the different types of changes occurring in the non-stationary data distribution, such as *dissolution*, *coagulation* and *shift*. Finally, we plan to incorporate the approach of self-adapting into other data streaming algorithms.

Acknowledgments. The authors would like to thank European Research Consortium for Informatics and Mathematics (ERCIM) fellowship programme for the support.

References

1. Fan, W., Wang, H., Yu, P.: Active mining of data streams. In: Proceedings of SIAM Conference on Data Mining, SDM (2004)
2. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 81–92 (2003)
3. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: Proceedings of SIAM Conference on Data Mining (SDM), pp. 326–337 (2006)
4. Zhang, X., Furtlehner, C., Sebag, M.: Data streaming with affinity propagation. In: Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD), pp. 628–643 (2008)
5. Muthukrishnan, S.: Data streams: Algorithms and applications. In: Found. Trends Theor. Comput. Sci., vol. 1, pp. 117–236. Now Publishers Inc. (2005)
6. Guha, S., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams. In: IEEE Symposium on Foundations of Computer Science, pp. 359–366 (2000)

7. Zhang, X., Furtlehner, C., Perez, J., Germain-Renaud, C., Sebag, M.: Toward autonomic grids: Analyzing the job flow with affinity streaming. In: KDD 2009: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 987–995 (2009)
8. Arasu, A., Manku, G.S.: Approximate counts and quantiles over sliding windows. In: Proceedings of ACM Symposium Principles of Database Systems(PODS), pp. 286–296 (2004)
9. Zhang, X., Sebag, M., Germain-Renaud, C.: Multi-scale real-time grid monitoring with job stream mining. In: Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid (CCGrid), pp. 420–427 (2009)
10. MIT-Lincoln-Lab: Mit lincoln laboratory, darpa intrusion detection evaluation documentation (1999), http://www.ll.mit.edu/mission/communications/ist/CST/index.html
11. Zhang, X., Germain, C., Sebag, M.: Adaptively detecting changes in autonomic grid computing. In: Proceedings of 11th ACM/IEEE International Conference on Grid Computing (Grid 2010), workshop on Autonomic Computational Science (2010)
12. Page, E.: Continuous inspection schemes. Biometrika 41, 100–115 (1954)
13. Hinkley, D.: Inference about the change-point from cumulative sum tests. Biometrika 58, 509–523 (1971)
14. Dasu, T., Krishnan, S., Venkatasubramanian, S., Yi, K.: An information-theoretic approach to detecting changes in multi-dimensional data streams. In: Proceedings of 38th Symposium on the Interface of Statistics, Computing Science, and Applications, Interface 2006 (2006)
15. Song, X., Wu, M., Jermaine, C., Ranka, S.: Statistical change detection for multidimensional data. In: KDD 2007: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 667–676 (2007)
16. Aggarwal, C.C.: A framework for diagnosing changes in evolving data streams. In: SIGMOD 2003: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp. 575–586 (2003)
17. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics 6, 461–464 (1978)
18. KDDCup: KDD Cup 1999 data (computer network intrusion detection) (1999), http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
19. Lee, W., Stolfo, S.J., Mok, K.W.: A data mining framework for building intrusion detection models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 120–132 (1999)

Anchor Points Seeking of Large Urban Crowd Based on the Mobile Billing Data

Wenhao Huang, Zhengbin Dong, Nan Zhao, Hao Tian, Guojie Song*, Guanhua Chen, Yun Jiang, and Kunqing Xie

Key Laboratory of Machine Perception, Ministry of Education,
Peking University, Beijing, 100871, China
{rubio8741,znzn007,tianhao86}@gmail.com,
{dongzhengbin,gjsong,kunqing}@cis.pku.edu.cn,
{ghchen,jiangyun}@pku.edu.cn

Abstract. In everyday life, people spend most of their time in some routine places such as the living places(origin) and working places(destination). We define these locations as anchor points. The anchor point information is important to the city planning, transportation management and optimization. Traditional methods of anchor points seeking mainly based on the data obtained from the sample survey or link volumes. The defects of these methods such as low sample rate and high cost make it difficult for us to study on the large crowd in the city.In recent years, with the rapid development of wireless communication, mobile phones have becoming more and more popular. In this paper, we proposed a novel approach to obtain the anchor points of the large urban crowd based on the mobile billing data. In addition, we took advantage of the spatial and temporal patterns of people's behavior in the anchor points to improve the simple algorithm.

Keywords: mobile phone, anchor point, OD information, spatial-temporal pattern, large crowd.

1 Introduction

There are some places which we stay most of our time in daily movement and everyday living such as living places (Origin) and working places (Destination). We commute between workplace and home. These places are referred as the anchor points. Obviously everyone has his or her own anchor points. Those anchor points are of significant value to effective traffic network management, public transportation planning, city rational planning and many related social aspects [1]. For example, the travel between living place and working place which is also called commuting is the most basic and common travel for everyone [2]. It has crucial impact on the optimization of urban transportation especially the morning and evening rush [3].

* Corresponding author.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 346–357, 2010.

Traditionally, we obtain the anchor points information by the following two methods: survey method and modeling method [4]. However, though these two methods own their own merits, they are far from satisfaction. In recent years, modern wireless communication especially the mobile phone services have become more and more popular and widespread all over the world. The coverage of mobile phone is larger than 90% to the whole population in European Union [6]. Almost 8 billion people in China own a cell phone. This means a surprisingly full coverage of mobile phone using globally. For the purpose of billing, the billing data recorded by the mobile phone operator often include the location information such as the base station coordinates. This provides us a great opportunity to use these data to seek the anchor points of the large crowd.

In this paper, we proposed a simple algorithm to obtain the anchor points of the large crowd based on the mobile phone billing data. In order to overcome the irregular correlation between the calling patterns and special-temporal patterns of anchor points, we take full advantage of the spatial-temporal stability of the anchor points to improve accuracy and reliability of the algorithm.

2 Related Work

2.1 Anchor Points Estimation Techniques

Researches on the anchor point estimation could be divided into three categories according to the methodology [7].

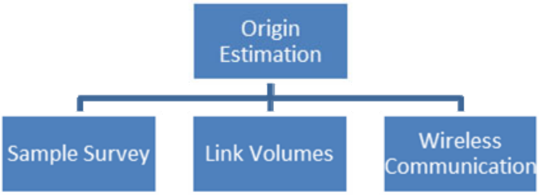


Fig. 1. OD estimation techniques

These techniques all have their own advantages and disadvantages. Sample survey conduct traffic survey by mail or household interview. The data collected by the survey are very detailed and accurate and could be used to many other related works. However, due to the cost and time, the sample rate is quite low. Meanwhile, the survey area is often limited in some near districts or blocks. Furthermore, the error will be increased greatly in the conversion process from the surveyed samples to large crowd.

The estimation by link volumes has large amount of data [8]. But it is only part of the crowd and is constrained coverage area. At the same time, error caused by the assignment model restricted its widespread use.

The wireless communication method has emerged in recent years. It is becoming a hot topic in transportation especially the intelligent transportation systems [9]. Though the data is not very detailed because it is designed for billing not for our purpose, it is sufficient for us to seek the anchor points by the location and time information of the call. In addition, due to the widespread use of the mobile phone, the data we collected are large enough to represent the crowd. Since the data is mainly purposed for billing, it will bring no cost for us to seek the anchor points.

The comparison of the three techniques could be concluded in Table1.

Table 1. The comparison of anchor point estimation method

	Advantage	**Disadvantage**
Survey	Detailed data Accurate	Big cost of time and money Small sample rate
Link Volume	Low cost	Limit area Error of model
Mobile data	Almost no cost Large sample rate	Not detailed data Inaccurate location

In comparison, using mobile phone data to seek the anchor points is the most promising one. It has bright future because of the increasing use of mobile phones.

2.2 Anchor Points Estimation by Mobile Phone

With the development of wireless communications especially the popularity of mobile phones, it is possible to obtain the location information and people's mobility pattern from the mobile phones.

White [10] uses mobile phone location information to originate the traffic OD, in the research mobile phone are used as traffic probes which can record the time and location information. But the data White used in his research are the Location Area (LA) data; it is too big in space to support the further study on the urban transportation system.

The media lab of MIT [11] employs 100 volunteers, who use the designed Nokia phone to obtain the detail information including the call log, bluetooth usage and moving pattern. The data they obtained are as detailed as those got from sample surveys. However, the sample rate is still low because it needs the special designed phones which are not common in our everyday life.

2.3 Datasets

The work of this paper is based on a China Mobile billing dataset of a major city in China within 3 months. Over 2.5 million anonymous mobile phone subscribers and 400 million calls are recorded. For the sake of the limitation of the computer speed, we randomly extracted a sample of 100,000 mobile phone subscribers' phone records as a new dataset after data cleaning and preprocessing.

All the records are preprocessed as the following format: Phone Number (encrypted), Date, Begin Time, End Time, Base ID, Longitude, Latitude, X, Y. The X,Y coordinates are transformed from the longitude and latitude.

For the purpose of verification, we also utilized the datasets collected in the reality mining project from 2004-2005 by the MIT Media Laboratory [12]. Only 94 volunteers took part in the project, however, those volunteers reported their real work and home location. So it is very suitable for the verification of our anchor points seeking algorithm. Though this dataset are not perfectly matching with the billing dataset, we could extract necessary information from the dataset and transform it to the data format as table 2 shows.

3 Simple Algorithm of Anchor Points Seeking

3.1 Algorithm Assumption

Assumption 3.1: There are anchor points in a person's daily life
According to Gonzalez et al's [13] study, we can find that people's mobility is stable from spatial and temporal aspects. Everyone will take rest at home or somewhere else. The place they living is the most important anchor point of people's everyday life. Similarly, working place is also a valuable anchor point.

Assumption 3.2: People have more frequent calls in the anchor point
This assumption may not be established in a short time like one day because of the accidental of mobile phone calls. But if examined in a long history such as three months or half a year, we will find it could be established most of the time. Because of the spatial and temporal stability of people's daily life, people will stay in the anchor points for a long time and have more calls in this period of time. The few exceptions will create only very slight influence which could almost be neglected when the algorithm is applying to large crowd.

3.2 Algorithm Description

According to the assumption 3.2, we can draw the conclusion that the base station where the person has the largest amount of calls in three months has the highest possibility of being his or her anchor point. To most of the people, they stay at home (origin) at night and work in the day time at the work place (destination) which is in accordance to large crowd' living patterns.

The simple algorithm of anchor points seeking can be described as following:

1·Divide the time into two part, 6-20 as the working time, and 0-6, 20-24 as the living time.

2·Sum up all the mobile phone records in three months of the same user to base stations in working time (Tday) and living time (Tnight) respectively.

3·Sort the base stations based on the amount of records from big to small in working time and living time respectively.

4·Choose the first base station during the working time and the living time as the working place (Destination) and living place (Origin). Those points (may be the same) are the anchor points for this person.

```
Algorithm 1: Anchor points seeking simple algorithm
Input: the call data T of each user, it is arranged as the format in 3.1
Output:
for each user $u$ do
    divide the data sets into two part Tday and Tnight according to the calling time
    for Tday and Tnight do
        sum up the calls into base stations
        choose the base station B owns the largest amount
        if (Tday) L(Tday) <- B
        else L(Tnight) <- B
        end if
    end for
    destination <- L(Tday)
    origin <- L(Tnight)
    output destination
    output origin
end for
```

3.3 Experiment and Results

Our experiment is based on the dataset from China Mobile we discussed in the 2.3.

1. Crowd histogram

Figure 2 is the crowd histogram in three different periods of time: Day time, night time and whole day.

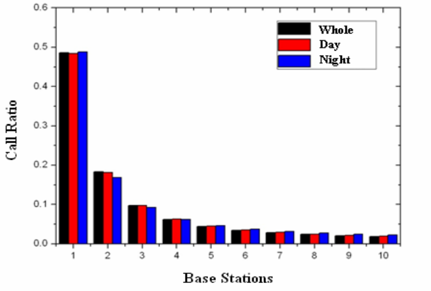


Fig. 2. The call ratio in top 10 Base Stations

The histogram in those three periods of time is very similar. The call ratio of the first base station is near 50%. This means that almost half of the calls of a single person take place at the same base station. This base station is most likely an anchor point of the person. The sum of the call ratio of the first three base stations is over 80%, it provides clear evidence that people's phone call pattern is spatial stable.

2. Different periods of time dividing

Figure 3 gives us the impacts of different working and living time dividing on the crowd histogram. The left figure is the histogram of living time and the right one is the working time.

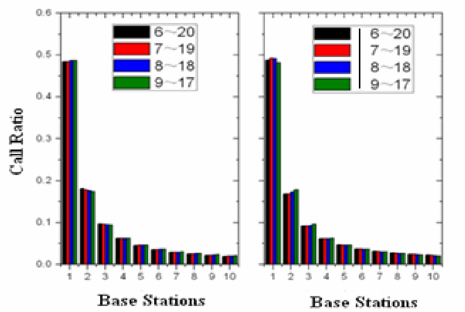


Fig. 3. Correlation with Different Time Dividing and Call Ratio

From the figure, we can safely draw the conclusion that the histogram is almost the same in different dividing. This means that people's phone call patterns are also temporal stable.

3.4 Deficiency of the Simple Anchor Points Seeking Algorithm

The simple algorithm we discussed in the 3.2 is uncomplicated and easy to implement. However, it also has some deficiencies because it neglects some features of mobile phone calls as well as the anchor points.

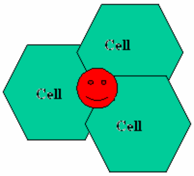


Fig. 4. Oscillation Effect

1. Oscillation Effect

In reality, user's anchor points are not consistent with the base station. As Figure 4 shows, one's anchor point may be covered by three different base stations. One call will only be recorded to one base station. In a long time, the calls in the anchor point may be recorded in three base stations separately [11]. Though this is due to the telecommunication mechanism, we have to handle it appropriately to improve our simple algorithm.

2. Randomness of calls

Mobile phone call is a random event, so it is improper to determine one's anchor points only based on the counts of calls in a base station. For example, it is totally different that one have ten calls in a place in one day because of some emergencies and one have one call in the same place for ten days. Mobile phone call's temporal pattern should also be added as a criterion to improve the simple algorithm.

4 Spatial Pattern to Improve the Anchor Points Seeking Algorithm

4.1 Spatial Pattern

In order to overcome the ambiguity discussed in 3.4.1, we should take advantage of the spatial pattern of mobile calls. As we know, the anchor point is not necessarily one single base station. It could be the combination of two or three near base stations. We could cluster some near base stations to a new area based on a specific rule and then apply the simple algorithm.

4.2 Clustering Rules

The distribution of the base stations is irregular since it is determined by the demand of calls. The coverage area of the base stations ranges from 100 meters of 5000 meters. As a result, different clustering rules will leads differences in the outcome.

1. Based on the distance

The distance between two base stations could be calculated easily. If two base stations are close enough they could be considered as adjacent and could be combined to a larger one.

2. Based on the Voronoi Graph

Voronoi Graph [14] is a common and effective method to determine the cover area of a base station. We could find two base stations are adjacent or not based on the relation in the Voronoi Graph [15]. Figure 5 shows the Voronoi Graph of the base stations.

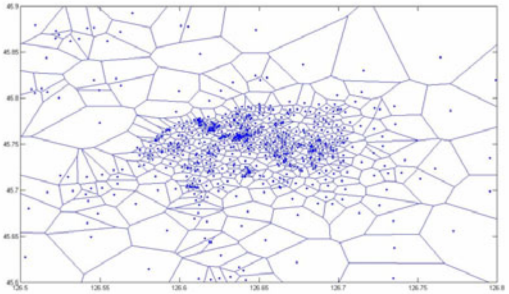


Fig. 5. Voronoi Graph

3. Combination of the distance and Voronoi Graph
Though Voronoi Graph seems more reasonable than simply the method simply based on the distance, it may leads very large clustered area especially in the rural area. So a better way may be the combination of the Voronoi Graph and distance. In other words, two base stations should be adjacent both in distance and Voronoi Graph.

4.3 Experiment and Results

The call ratios of the first ten components are shown in the histogram Figure 6. The outcome varies with different clustering rules. In order to keep the size of the location areas, the clustering rule is the combination of the distance and Voronoi Graph (the red part in Figure 6). It increased from near 0.48 to 0.53. More calls are taken in the first component location area which means that the possibility of being the anchor point increased. As our combination condition is restricted, the result showed that the location areas after clustering are all combined at most 3 base stations. The size of the new location area is still suitable for our further fine grained study on urban transportation and other related applications.

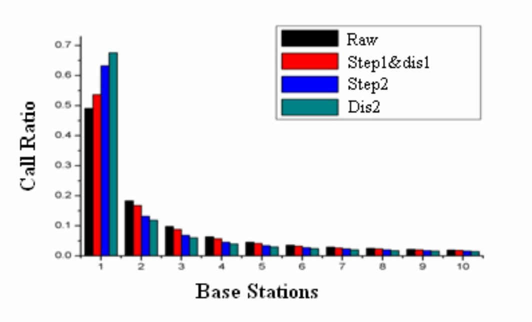


Fig. 6. Call Ratio of Different Combination Strategy

5 Temporal Pattern to Improve the Anchor Points Seeking Algorithm

5.1 Temporal Pattern

It is mentioned in the 3.4.2 that count of calls is not a suitable criterion to anchor points seeking. People's behavior in the anchor points has its obvious temporal pattern: people been to the anchor points regularly and stay in the anchor points for a long period of time. So we take these two factors into account to make mobile phone call such a random event becoming regularly in time.

5.2 Temporal Criterion

Definition 5.2.1 (calling frequency *f*): the ratio of days which people have phone records in a place to the total days.

Calling Frequency reflects people's regularity in one place.

Definition 5.2.2 (minimum calling times *m*): the minimum of calling times of one person in a place one day, it doesn't include those days the person don't have any phone records in the place.

Minimum calling times characterizes the stay time in one place.

Definition 5.2.3 (Calling Occurrence *O*):

*Calling Occurrence = Counts of calls * Calling Frequency * Minimum Calling Times*

Calling Occurrence is a better criterion which takes full advantage of people's temporal pattern in the anchor point.

5.3 Spatial-temporal Improved Algorithm

The spatial-temporal improved algorithm is based on the spatial enhanced algorithm introduced in section 4. It adds the temporal pattern into the spatial enhanced algorithm. It uses the calling occurrence introduced in 5.2.3 as the comparing and selecting criterion.

It could be described as follows:

```
Algorithm 2: Spatial-Temporal improved anchor points seeking algorithm
Input: the call data T of each user, it is arranged as the format in 3.1
Output:
for each user u    do
    divide the data sets into two part Tday and Tnight
    for Tday and Tnight do
        for each base station B do
            sum up the calls
            //spatial improve
              if any base stations adjacent in distance and Voronoi Graph
                  spatial combination
              end if
              //temporal improve
              O <- counts * f * m;
          end for
          find the base station with biggestO
          if (Tday) L(Tday) <- B
          else L(Tnight) <- B
          end if
    end for
    destination <- L(Tday)
    origin <- L(Tnight)
    output destination
    output origin
end for
```

5.4 Experiment and Results

In our dataset, not every subscriber's phone data are rich enough for anchor points seeking. Some only have few calls during three months and someone's calls are centralized only in few days. We set the threshold at one call per day in average. We filtered those who couldn't reach the condition in day and night respectively at first. After filtering, there are 30408 users left.

The call ratio in each location area is clearly reflected in Figure 7. The spatial-temporal pattern increased the ratio of the first component and decreased the ratio of the other components both in day and night. The average call ratio of the first component after applying the spatial-temporal pattern in the algorithm is increased from 0.5 to 0.7.

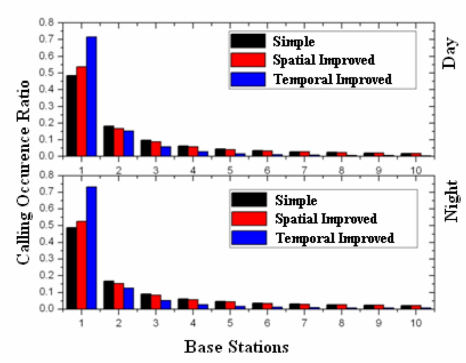


Fig. 7. Calling Occurrence of Spatial-Temporal Improved Algorithm

6 Verification

6.1 Indirect Verification

OD distance is the distance between the living place (O) and the working place (D). The distribution of OD distance is shown as figure 8.

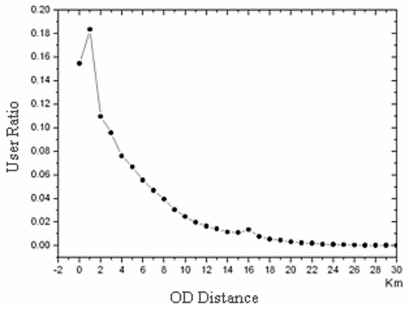


Fig. 8. User Ratio of OD Distance

From the figure, the distribution of the OD distance and the percent of users is in accordance with the power-law distribution. It is very similar to the distribution obtained from the survey data in [16].

6.2 Direct Verification

The billing data is not sufficient for anchor point verification because it doesn't contain real anchor point location information. However, the dataset downloaded from the MIT media lab (introduced in 2.3) perfectly fits the direct verification. It contains the relationship of base station ID and self-reported real location.

We transformed these data to the same format as the preprocessed mobile billing data. To ensure the similarity of the two datasets, we both use three months' phone records. Apply the simple algorithm and the spatial-temporal improved anchor points seeking algorithm on the preprocessed MIT's dataset. Then compare the real home and work place reported by the volunteers in the dataset with the location we obtained using the algorithm. We could get Figure 9 which shows the correctness.

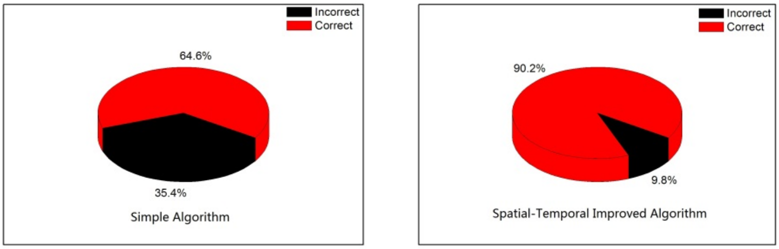


Fig. 9. The correctness of simple and improved algorithm

From the figure we could see that spatial-temporal improved algorithm promoted the accuracy from 66.7% to 91.3% which means a big progress in both algorithm reliability and application effect. With such high reliability, the algorithm could be effective in the anchor points seeking for large urban crowd. When the algorithm is applied to the large urban crowd, we pay attention to the whole crowd but not the single person. So the slight error almost has no effect on the overall performance.

7 Conclusion and Prospects

Modern technology especially the mobile phone has provided us large dataset. The researchers started to focus on the transportation information obtained from the mobile phone data in recent years [17]. In this paper, we proposed a novel approach to obtain the anchor points such as the living place and working place from the mobile billing data. Moreover, we take advantage of the spatial pattern of mobile phone calls. In addition, the calling occurrences substituted the simple phone call counts as the criterion of anchor points selecting. The accuracy of the algorithm is quite high verified by the MIT's real location data.

With the development of the wireless communication as well as the world wide spread of the mobile phones, our approach of anchor points seeking or OD estimation are sure to replace the traditional approach such as sample survey. The rich anchor point information we obtained from the mobile billing data could be used in city planning, urban transportation optimization and analyzing of the mobility patterns of the large crowd.

Acknowledgments. Research was supported by the National Natural Science Foundation of China under Grant No. 60703066 and No.60874082; Supported by Beijing Municipal Natural Science Foundation No. 4102026; supported by the National High Technology Research and Development Program of China (863 Key Program) No.2007AA120502. We sincerely thank our colleague in Key Laboratory of Machine Perception, Ministry of Education, Peking University.

References

1. Ratti, C., Pulselli, R.M., Williams, S., Frenchman, D.: Mobile Landscapes: using location data from cell-phones for urban analysis. Environment and Planning B - Planning and Design (2005)
2. Jungyul, S.: Are commuting patterns a good indicator of urban spatial structure. Journal of Transport geography, 36-42 (2006)
3. Bowman, J.L., Ben-Akiva, M.: Activity-based disaggregates travel demand model system with activity schedules. Transportation Research A 35, 1–28 (2000)
4. Cascetta: Estimation of trip matrices from traffic counts and survey data: A generalized least squares estimator. Transport. Res. 415, 289–299 (1984)
5. Haixiao, M.: Research on Person Trip Characteristics of Chinese Citizens. Doctor dissertation of Beijing Polytechnic University (2005)
6. Townsend, A.M.: Mobile Communications in the 21st Century City. In: Brown, B., Green, N., Harper, R. (eds.) The Wireless World: Social and Interactional Aspects of the Mobile Age. Springer, Berlin (2001)
7. Byeong-Seok, Y., Kyungsoo, C.: Origin-destination estimation using cellular phone as information. Journal of the Eastern Asia Society for Transportation Studies 6, 2574–2588 (2005)
8. Lo, H.P., Chan, C.P.: Simultaneous estimation of an origin–destination matrix and link choice proportions using traffic counts. TR-A 37, 771–788 (1999)
9. Caceres, N., Wideberg, J.P., Benitez, F.G.: Review of traffic data estimations extracted from cellular networks. Intelligent Transport Systems, IET (2008)
10. White, J., Wells, I.: Extracting origin-destination information from mobile phone data (2002) (unpublished working paper)
11. Eagle, N., Pentland, A.: Reality Mining: Sensing Complex Social Systems. Personal and Ubiquitous Computing 10, 255–268 (2006)
12. Eagle, N., Pentland, A., David, L.: Inferring Social Network Structure using Mobile Phone Data. In: Proceedings of the National Academy of Sciences (PNAS), pp. 15274–15278 (2009)
13. González, M.C., Hidalgo, C.A., Barabási, A.L.: Understanding individual human mobility patterns. Nature 6, 779–782 (2008)
14. Franz, A., Rolf, K.: Voronoi diagrams–a survey of a fundamental geometric data structure. ACM Computing Surveys 23, 345–405 (1991)
15. Candia, J., Gonzalez, M.C., et al.: Uncovering individual and collective human dynamics from mobile phone records. Journal of Physics A-Mathematical and Theoretical (2008)
16. Yanwei, C., Zhilin, L.: Spatial and Temporal Structure of Cities in China. Press of Peking University, Beijing (2002)
17. Bolla, R., Davoli, F., Giordano, A.: Estimating road traffic parameters from mobile communications. In: Proceedings 7th World Congress on ITS, Turin, Italy (2000)

Frequent Pattern Trend Analysis in Social Networks

Puteri N.E. Nohuddin[1], Rob Christley[2], Frans Coenen[3], Yogesh Patel[1,3], Christian Setzkorn[2], and Shane Williams[3]

[1] Department of Computer Science, University of Liverpool, UK
{puteri,frans}@liverpool.ac.uk
[2] School of Veterinary Science, University of Liverpool and National Centre for Zoonosis Research, Leahurst, Neston, UK
{robc,c.setzkorn}@liverpool.ac.uk
[3] Deeside Insurance Ltd., Deeside, UK
{yogesh,shane}@deesideinsurance.co.uk

Abstract. This paper describes an approach to identifying and comparing *frequent pattern trends* in social networks. A frequent pattern trend is defined as a sequence of time-stamped occurrence (*support*) values for specific frequent patterns that exist in the data. The trends are generated according to *epochs*. Therefore, trend changes across a sequence epochs can be identified. In many cases, a great many trends are identified and difficult to interpret the result. With a combination of constraints, placed on the frequent patterns, and clustering and cluster analysis techniques, it is argued that analysis of the result is enhanced. Clustering technique uses a *Self Organising Map* approach to produce a sequence of *maps*, one per epoch. These maps can then be compared and the movement of trends identified. This *Frequent Pattern Trend Mining* framework has been evaluated using two non-standard types of social networks, the cattle movement network and the insurance quote network.

Keywords: Social Networks, Pattern Mining, Trend Mining, Trend Analysis.

1 Introduction

This paper introduces a social network Frequent Pattern Trend Mining (FPTM) framework. The framework includes a mechanism for identifying and extracting trends in social network data, grouping those trends and then identifying the "most interesting" trends. The input to the framework is a sequence of n time stamped social networks, partitioned into m sub-sequences, called *epochs*, of n/m time stamps each. A single trend in this context is defined in terms of a sequence of frequency count values associated with a particular pattern. An established frequent pattern mining algorithm, the Total-From-Partial (TFP) algorithm [13] [6] has been adopted to identify the frequent patterns, and modified so as to identify temporal sequences of frequent patterns (i.e. trends). In common with

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 358–369, 2010.

more general frequent pattern mining applications, a great many trends are typically identified. To limit the number of generated trends, constraints are applied to the nature of the frequent patterns. However, this only serves to partially address the issue, in most case a significant number of trends remain. The remaining trends are thus clustered using a Self Organising Map (SOM) approach [1] so that similar trends are grouped. The application of the SOM technique results in a sequence of maps, one per epoch. These maps can then be compared; an "interesting" trend is defined as a trend whose location, with respect to the SOM nodes in which it is located, changes significantly from map to map.

The focus of the work is two specific social networks, the Deeside insurance quote network and the GB cattle movement network. These two networks are exemplars of two generic categories of social network, which the authors refer to as *star networks* and *complex star networks*. Further details of the nature and derivation of the networks are given in Section 3.

A formal definition of the trend mining problem, as envisioned in this paper, is presented in Section 4. A detailed review of the proposed Frequent Pattern Trend Mining (FPTM) framework is given in Section 5 together with a review of the components of the framework. A full evaluation of the framework is given in Section 6 and 7 using the cattle movement and Deeside networks respectively. The evaluation demonstrates that the framework can be used to identify interesting trends in social network data.

2 Previous Work

The proposed FPTM framework is founded on a number of technologies, namely frequent pattern mining and Self Organising Maps (SOMs). A brief review of both of these technologies is therefore provided in this section. The section is concluded with a review of some alternative approaches to trend mining related to the work described in this paper.

Frequent pattern mining is a well established technology that can be traced back to early work on Association Rule Mining (ARM) [4]. The original concept was to find all frequently occurring combinations of a global set attributes (the set of *frequent patterns*) that exist in a binary-valued data set and from these combinations extract relationships defined in terms of *association rules*. In the context of the work described in this paper, the authors are interested in the first part of this process, the identification of frequent patterns. Subsequent to the original ARM algorithm presented in [4] many alternative algorithms have been proposed. Examples include FP-growth [5] and Total-From-Partial [6] [7]. For the FPTM framework, the latter was adopted although similar fast frequent pattern mining algorithms could equally well have been adopted. To limit the number of frequent patterns discovered, a minimum support threshold is usually adopted. A pattern is only recorded as frequent if its frequency count is above this threshold. The lower the threshold, the greater the number of frequent patterns that are discovered.

The concept of Self Organising Maps (SOMs), a type of artificial neural network, was first introduced by Kohonen [2] [3]. A SOM is an effective visualization method to translate high dimensional data into a low dimension grid (map), with $x \times y$ nodes. Initially, the SOM nodes are assigned with weight vectors which have the same dimensions as the input data. The nodes compete with each other to be stimulated to represent input data records according to a distance function[1] and a neighbourhood function[2]. SOM have been used in many areas such as pattern recognition of optic nerve images [12], gene expression patterns [11] and manufacturing processing control [14].

There has been some work on social networks trend analysis. Gloor *et.al.* introduced a novel trend analysis algorithm to generate trends from Web resources [10]. The algorithm calculates the values of temporal *betweeness* of online social network node and link structures to observe and predict trends on the popularity of concepts and topics such as brands, movies and politicians. Research in social networks trend mining has provided advantages for online viral marketing [8], stock market activities [9] and many more. There has been some work on the identification of trends in time stamped sequences of binary valued (ARM) data sets. Early examples can be found in work on Jumping and Emerging Patterns. Emerging Patterns describe patterns with support counts change between time stamps [16]. Jumping Patterns describe patterns whose support counts change drastically (jump) from one time stamp to another. In medical research, jumping and emerging pattern trends have been used to monitor the progress of cancer cells [17]. The concept of frequent pattern trends defined in terms of sequence of frequency counts has also been adopted in [18] in the context of longitudinal patient datasets. In [18], trends are categorised according to pre-defined prototypes, in the authors' work, they are categorised using SOM technology.

3 Target Social Networks

The work described in this paper is directed at two specific social networks. The first is a customer network extracted from an insurance company's database referred to as the Deeside insurance quote network. The second was extracted from the Cattle Tracing System (CTS) database in operation in Great Britain (GB) and is referred to as the GB Cattle Movement Network. The generic nature of these networks is presented (in a styalised form) by the two "network snap shots" given in Figures 1 and 2. With reference to Figure 1, the network is characterised by a single "star shape" with all nodes communicating with one *super-node*, the authors refer to this type of network as a *star network*. Note that, as shown in the figure, not all network nodes will be necessarily communicating (linking) with the super-node at any given time stamp. The generic network snap shot given in Figure 2 is a more complex version of that given in Figure 1, and the authors refer to this as a *complex star network*. The network is characterised by a number of disconnected "star" sub-networks of varying size. Again, not

[1] Euclidean function is the distance function used to calculate shortest distance.

[2] Gaussian function is used to determine the neighbourhood size on the map.

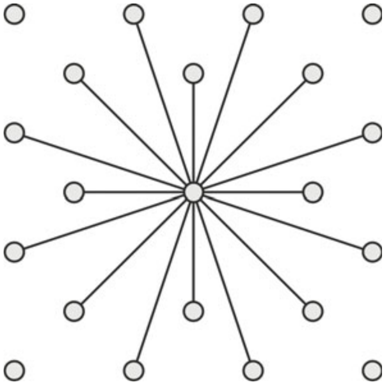


Fig. 1. (Styalised) Star Network

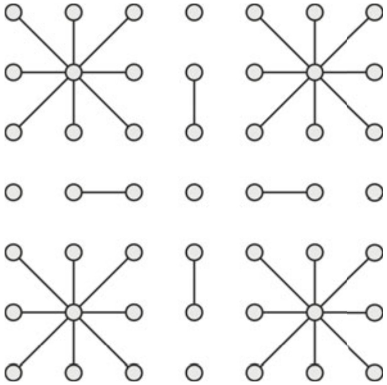


Fig. 2. (Stylaised) Complex Star Network

all network nodes (with respect to the snap-shot time stamp) are necessarily communicating (linking) with any of the other nodes. Note also that, some of the "star" sub-networks comprise only two nodes. The derivation of both networks is briefly described in the rest of this section. In both cases, the authors have applied the FPTM framework to identify the interesting trends that feature in a sequence of time stamped snap shots of these networks.

The Deeside insurance quote network was extracted from a sample of records taken from the customer database operated by Deeside Insurance Ltd. (collaborators on the work described in this paper). Twenty-four months of data were obtained comprising, on average, 400 records per month. In total, the data set comprised 250 records with (after discretisation and normalisation) 314 attributes. The data was processed to produce a sequence of 24 networks, one per month; divided into two epochs comprising 12 months each, 2008 and 2009. In the resulting networks, the nodes comprised postal areas (characterised by the first few digits of UK post/zip codes), and the links are the number of requests for specific types of insurance quotes received for the given time stamp. Thus, although not shown in Figure 1, there can be many links between the super node and the outlying nodes. An example of a type of insurance quote, attached to a link, might be:

$$\{NumberOfQuotes = \{10:20\}, CarType = Vauxhall, EngineSize = \{1500: 1999\}, OffenceCode = SP, Fine = \{200:300\} \ and \ Gender = Male\}$$

where: the value {10:20} indicates the $NumberOfQuotes$ is in a range between 10 and 20; the value {1500:1999} states the $EngineSize$ is within the range 1500 and 1999, the value SP is an $OffenceCode$ indicating exceeding the speed limit, and the value {200:300} indicates a $Fine$ of between 200 and 300. The number of quotes attached to a specific link ranged between 10 and 40. The average number of nodes in a single (one month) network was 120, and the average number of links 200.

The cattle movement network was extracted from the Cattle Tracing System (CTS) database in operation in Great Brittain (GB). The CTS database was introduced in September 1998 and updated in 2001 as a result of a number of outbreaks of bovine diseases. The database is maintained by DEFRA, the Department for Environment, Food and Rural Affairs, a UK government department. The database records all cattle movements in GB, each record describes the movement of a single "beast", identified by a unique ID number, between two *holding locations* (farms, markets, slaughter houses, etc.). However, the database can be interpreted as a social network, where each node represents a geographical location and the links the number of beasts moved between locations. In a similar manner to the Deeside insurance quote network, the links describe specific types of cattle movement, thus there could be more than one link between pairs of nodes. An example of a type of cattle movement that might be attached to a link is:

$$\{NumberOfBeasts = \{50\}, BeastType = Limousin\ Cross, AnimalAge = \{1 : 6\}, PTI = 4\ and\ Gender = Female\}$$

where the attribute label PTI is the Parish Testing Interval which describes the frequency of disease detection testing for each node; the value is between 1 and 4 years. The number of beasts attached to the link is 50. Each node describes a location defined in terms of 100km grid squares. Four years worth of data, from 2003 to 2006, were used to generate the networks giving a sequence of 48 networks, one per month, divided into four epochs of 12 months each. After discretisation and normalisation, the average number of nodes within a single network was 150,000 and the average number of links was 300,000 with 445 attributes. It should be noted that some nodes in the network, represent zoos or "hobby farms", which seldomly move beasts to other locations.

4 Problem Definition

The FPTM framework is directed at finding interesting trends in social network data. A trend is deemed to be interesting if its "shape" changes significantly between epochs. An epoch is defined in terms of a start and an end time stamp. The input to the FPTM framework is a sequence of n time stamped datasets $D = \{d_1, d_2, \ldots, d_n\}$ partitioned into m epochs (note that n should typically be some multiple of m). Each data set d_i comprises a set of records such that each record describes a social network node paring, the description consists of some subset of a global set of attributes A that describes the network (some example records were given in Section 3). There are $2^{|A|} - 1$ patterns that may exist in any given dataset. The *support* (s) for a pattern I in a dataset d_i is the number of occurrences of the pattern in d_i expressed as a percentage of the number of records in d_i. A trend (t) with respect to a pattern I is then a sequence of support values associated with the data sets represented in an epoch, $t = \{s_1, s_2, \ldots, s_{n/m}\}$. Each time stamp ($i$) therefore has a set of trends (T_i) associated with it. The complete set of trends is then given by

$T = \{T_1, T_2, \dots, T_n\}$. The set of trends associated with an epoch E_j (where $0 < j \leq m$) is then defined as $E_j = \{T_{1+k}, T_{1+(k+1)}, \dots, T_{1+(k+(n/m)-1)}\}$ (where $k = (j-1)(n/m)$). The complete set of trends, in a sequence of epochs E, is then given by $E = \{E_1, E_2, \dots E_m\}$. The objective is then to identify interesting individual trends lines that exist across the set E. We have already indicated that "interestingness" is defined according to the distance a trend line moves between SOM maps. A more precise definition of interestingness will be given in the following section, Section 5.

5 The Frequent Pattern Trend Mining (FPTM) Framework

Figure 3 gives a schematic of the FPTM framework. The process commences in the top left of the figure with the collection of datasets $D = \{d_1, d_2, \dots, d_n\}$ describing a sequence of n social networks, each associated with a discrete time stamp of between 1 and n. During preprocessing, constraints may be applied to the nature of the frequent item sets that the authors may be interested in. The constraints take the form of feature-value pairs (attributes) that must be included in a frequent pattern for it to be considered relevant.

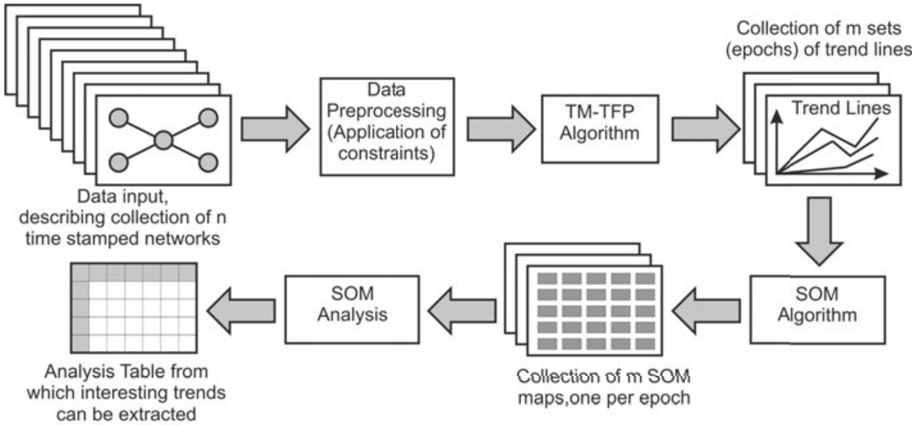

Collection of m sets (epochs) of trend lines
Data Preprocessing (Application of constraints)
TM-TFP Algorithm
Trend Lines
Data input, describing collection of n time stamped networks
SOM Analysis
SOM Algorithm
Analysis Table from which interesting trends can be extracted
Collection of m SOM maps,one per epoch

Fig. 3. Schematic illustrating the operation of the FPTM framework

The next stage is to obtain the set of trends T. This is achieved, as noted above, using an extension of the TFP algorithm called the the TM-TFP (Trend Mining TFP) developed by the authors. The advantage of TFP is that it is very efficient, this is facilitated by the use of two "set enumeration tree" style storage structures, namely the P-tree and the T-tree. The P-tree is used to summarise the input data and, as a result of its construction process, stores partial support counts. The T-tree then stores the frequent item sets contained in the input dataset and is generated from the P-tree. The T-tree can more accurately be described as a *trie*, in that levels in the T-tree branches are actually vectors (arrays). This then facilitates fast "look-up". TM-TFP was built on-top of TFP and includes a TM-tree to store *trend lines* (sequences of support values associated with individual patterns with respect to specific time stamps and consequently specific epochs). The output from TM-TFP is a set of trend lines associated with the given set of epochs $E = \{E_1, E_2, \ldots E_m\}$. In situations where a pattern is sometimes frequent (above the support threshold) and sometimes infrequent, a value of 0 is recorded where the support count falls below the threshold. The authors did experiment with the use of negative borders, as advocated in [15] (amongst others), however it was quickly realised that it was easier to simply use a relatively low support threshold (thresholds of 1, 2, 3, 4 and 5 were used for the experiments reported in this paper).

The next stage is to categorise the trends using a Self Organising Map (SOM) algorithm, one map was created per epoch. The process commenced with the generation (training) of a prototype $x \times y$ node map such that each node would represent a category of trend. The authors experimented with different mechanisms for training the SOM, including: (i) devising specific trends to represent individual nodes; (ii) generating a collection of all trends that are arithmetically possible and training the SOM using this set; and (iii) using some, or all, of the trends in the first epoch to be considered (E_1). The first required prior knowledge of the trend configurations in which the authors might be interested, as the objective was to discover these trends it seemed counter intuitive to first try and guess what they might be. With respect to the second option, it was discovered that this resulted in a map where the majority of nodes, once the maps had been populated, were empty. The third option was therefore adopted, the SOM was trained using the trend lines associated with the first epoch. The resulting *prototype map* was then populated with data for the remaining E_2 to E_m epochs to produce a sequence of m maps.

Temporal trend analysis was then applied to the sequence of m maps. The aim was to compare individual trends so as to discover significant temporal trend changes. Temporal trend changes are defined in terms of the "distance" a trend moves between two subsequent SOM maps. This is encoded in a matrix structure that holds "distances moved" for all trends and all sequences of pairs of SOM maps. Note that, some trends may remain in the same node for the entire sequence of m maps. Some other trends may fluctuate between nodes. While other trends may slowly move across the m maps. The greater the distance moved, the more interesting the trend is deemed to be.

6 Analysis Using the Deeside Insurance Quote Social Network

In this, and the following section, an analysis of the proposed FPTM framework is presented in the context of the Deeside insurance quote network (this section), and the GB cattle movement network (following section). In both cases, the analysis is conducted by considering the output from each stage and the nature of the discovered interesting trends.

Table 1. Number of identified trends using Deeside insurance quote network

Support Threshold (%)	No Constraint	Constraint 1	Constraint 2	Constraint 3
1	830306	8239	5621	3965
2	206219	2163	1431	1595
3	94369	1038	677	863
4	55445	669	401	563
5	37239	469	283	427

For the analysis using the Deeside network, the authors applied a number of pattern constraints so as to reduce the number of patterns to be considered. The experiments used 3 pattern contrainsts with the selected attribute-value pairs:

$$\text{Constraint 1: } \{DriveAge = \{24 : 40\}\}$$
$$\text{Constraint 2: } \{Gender = Male\}$$
$$\text{Constraint 3: } \{PostcodeArea = CH\}$$

Constraint 1 has the effect of insisting that frequent patterns inlcude the attribute $DriveAge = \{24 : 40\}$, while Constraint 2 has the effect of limiting the set of frequent patterns to those where $Gender$ has the value $Male$. Constraint 3 has the effect of restricting patterns to those that include the $PostcodeArea$ "CH" (Chester).

Table 1 gives the number of trends discovered using all the data; and with the application of Constraint 1, Constraint 2 and Constraint 3. With low support

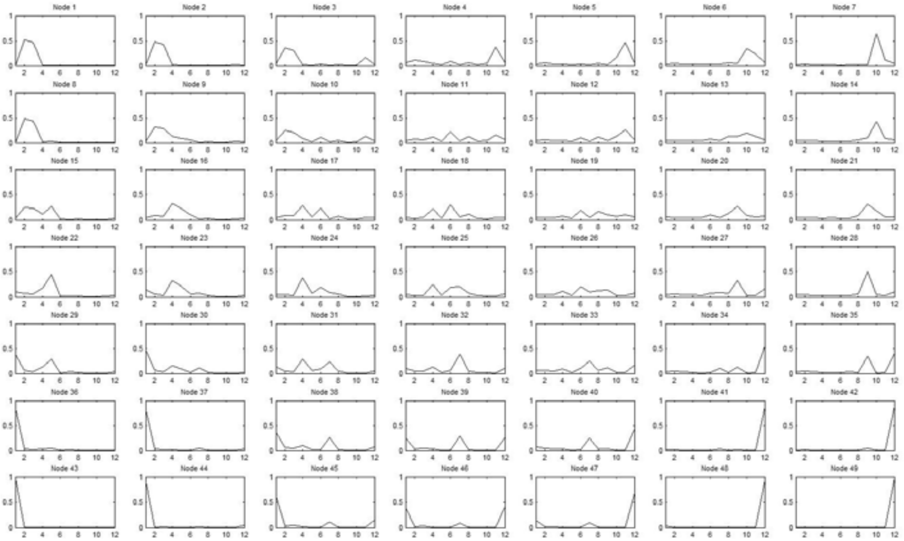


Fig. 4. Deeside Prototype Map for trends with Constraint1

Table 2. Examples of Deeside trends migrating from one SOM node to another (*Dist* = distance value)

Example Trends	Node 2008	*Dist*	Node 2009
$\{EngineSize = \{\leq 1000\}, CarType = Toyota\}$	49	2.0	35
$\{EngineSize = \{\leq 1000\}, CarType = Toyota, DriverAge = \{\geq 51\}\}$	49	3.0	28
$\{EngineSize = \{\leq 1000\}, CarType = Nissan\}$	26	1.41	34
$\{EngineSize = \{\leq 1000\}, CarType = Nissan, DriverAge = \{26 : 30\}\}$	5	6.32	49

thresholds, a large number of trends are generated in each case. When a constraint is imposed, the number of records to be considered is naturally reduced therefore fewer trends are discovered. Also as the support threshold increases, the number of trends decreases. Figure 4 presents the prototype SOM for trends generated using a support threshold 1% with Constraint 1 ($\{DriveAge, \{24 : 40\}\}$). The prototype map displays the characteristics of clusters of trend lines. With reference to the figure, node 1 (top-left) represents trends with high support from January to March, while node 18 (center) portrays trends with fluctuating support values in April, June and August. Note that, the distance between nodes indicates the dissimilarity between nodes; the greatest dissimilarity is thus between nodes at opposite ends of the diagonals. Based on this prototype map, a sequence of m maps, in which all trends for E_1 to E_m are fitted into, are generated.

In the next stage of the FPTM process, the trend and node details from the m maps are transfered into matrix for trend analysis. Table 2 shows examples of trends (representing frequent patterns), with Constraint 1, that migrated from SOM m_1 to m_2. For example, the trend line representing the pattern: $\{EngineSize = \{\leq 1000\}, CarType = Nissan, DriverAge = \{26 : 30\}\}$ was in cluster node 5 (middle top in Figure 4) in 2008, but moved diagonally to node 49 in 2009. The trend shape has changed significantly so it is marked as an interesting trend.

7 Analysis Using GB Cattle Movement Social Network

For the analysis using the GB cattle movement network, the authors also applied several pattern constraints. In the reported experiments, 3 pattern constraints were applied:

$$\text{Constraint 1: } \{BreedType = Beef\}$$
$$\text{Constraint 2: } \{BreedType = Dairy\}$$
$$\text{Constraint 3: } \{SenderLocationType = Algricultural\ holdings, ReceiverLocationType = Market\}$$

Table 3. Number of identified trends using GB cattle movement network

Support Threshold (%)	No Constraint	Constraint 1	Constraint 2	Constraint 3
1	25736	2333	2583	1195
2	8945	1019	1181	535
3	4393	541	715	383
4	2739	349	483	311
5	1928	249	339	267

Table 4. Examples of CTS trends migrating from one SOM node to another (*Dist* = distance value)

Example Trends	Node 2003	*Dist*	Node 2004	*Dist*	Node 2005	*Dist*	Node 2006
{ReceiverArea = 24, SenderLocType = Algricultural holdings, BeastType = Chianina}	47	5.66	15	1.0	8	7.07	48
{ReceiverArea = 24, SenderLocType = Algricultural holdings, BeastType = Lincon Red }	26	0.0	26	4.24	2	2.82	18
{ReceiverArea = 24, SenderLocType = Algricultural holdings, BreedType = Beef}	26	0.0	26	3.60	9	3.60	26
{ReceiverArea = 24, SenderLocType = Algricultural holdings, BreedType = Beef, BeastType = Chianina}	47	5.66	15	1.0	8	7.07	48

Again, Constraint 1 and Constraint 2 filter records with beasts used for *Beef* or *Dairy*. Whereas, Constraint 3 selects movement records from *Algricultural holdings* to the *Markets*.

Table 3 presents the number of trends discovered using all the available data and with 1, 2 and 3 applied respectively constraints. As in the case of the Deeside data, as the support threshold increases, the number of trends decreases. Figure 5 illustrates the prototype SOM for trends generated with support threshold 1% with Constraint 1, ($\{BeastType = Beef\}$). The prototype map displays node clusters of the discovered CTS trends. For example, node 1 (top-left) represents trends that have high support in early summer (May), while node 43 (bottom-left) indicates trend lines with high support in autumn only (October). Again, based on this prototype map, a sequence of m maps, in which all trends from E_1 to E_m are fitted into, was generated.

Table 4 presents some examples of trends, with Constraint 1, that migrated from m_1 to m_4, For example, the trend line representing the frequent pattern: $\{ReceiverArea = 24, SenderLocType = Algricultural\ holdings, BeastType = Chianina\}$ was in cluster node 47 in 2003 and moved to node 15 in 2004, but then migrated to node 8 in 2005 and node 48 in 2006. The greater distance values in m_3 (2005) and m_4 (2006) meant that this trend was deemed to be interesting.

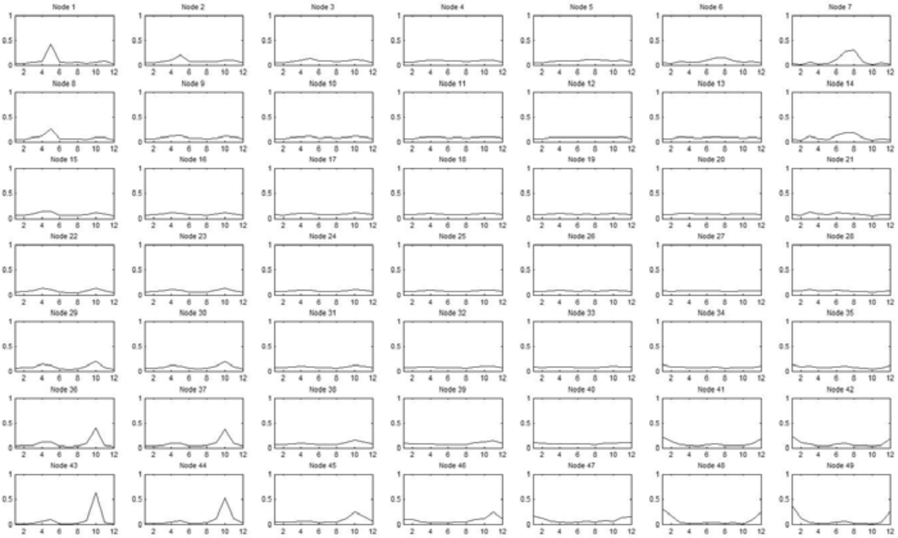


Fig. 5. CTS Prototype maps with Constraint1

8 Conclusions

In this paper, the authors have described the FPTM framework for identifying interesting trends in social network data. The trends are defined in terms of sequences of support counts associated with individual patterns across a sequence of time stamps associated with an *epoch*. Interesting trends are defined as those whose "shape" changes across two or more epochs. Shape changes are captured by plotting identified trends on a sequence of SOM maps, one per epoch, and determining the distance individual trends move between subsequent maps. The greater the distance the more interesting the trend is deemed to be. The operation of the FPTM framework was illustrated using a "star" social network and a"complex star" social network, more specifically the Deeside insurance quote network and the GB cattle movement network which have formed the focus of this work. Future work that the authors intend to undertake is on how to apply the technique to other forms of data than social network data. In particularly, the authors are looking to apply the technique to longitudinal patient data.

References

1. Cottrell, M., Rousset, P.: A powerful Tool for Analyzing and Representing Multidimensional Quantitative and Qualitative Data. In: Cabestany, J., Mira, J., Moreno-Díaz, R. (eds.) IWANN 1997. LNCS, vol. 1240, pp. 861–871. Springer, Heidelberg (1997)
2. Kohonen, T.: The Self Organizing Maps. Neurocomputing 21(1-3), 1–6 (1998)
3. Kohonen, T.: The Self Organizing Maps, 3rd edn. Springer Series in Information Sciences, vol. 30. Springer, Heidelberg (2001)

4. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: Proceedings of ACM SIGMOD Conference, pp. 207–216 (1993)
5. Han, J., Pei, J., Yiwen, Y.: Mining Frequent Patterns Without Candidate Generation. In: Proceedings ACM-SIGMOD International Conference on Management of Data, pp. 1–12. ACM Press, New York (2000)
6. Coenen, F.: The LUCS-KDD TFP Association Rule Mining Algorithm, Department of Computer Science, The University of Liverpool, UK (2004)
7. Coenen, F., Leng, P., Ahmed, S.: Data Structures for association Rule Mining: T-trees and P-trees. IEEE Transactions on Data and Knowledge Engineering 16(6), 774–778 (2004a)
8. Richardson, M., Domingos, P.: Mining Knowledge Sharing Sites for Viral Marketing. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 61–70 (2002)
9. Choudhury, M.D., Sundaram, H., John, A., Seligmann, D.D.: Can blog communication dynamics be correlated with stock market activity? In: Proceedings of the Nineteenth ACM Conference on Hypertext and Hypermedia, pp. 55–60 (2008)
10. Gloor, P.A., Krauss, J.S., Nann, S., Fischbach, K., Schoder, D.: Web Science 2.0: Identifying Trends Through Semantic Social Network Analysis. Social Science Research Network (2008)
11. Wang, J., Delabie, J., Aasheim, H.C., Smel, E., Myklebost, O.: Clustering of the SOM easily reveals distinct gene expression patterns: results of a reanalysis of lymphoma study. BMC Bioinformatics, 3(36) (2002)
12. Yan, S., Abidi, S.R., Artes, P.H.: Analyzing Sub-Classifications of Glaucoma via SOM Based Clustering of Optic Nerve Images. In: Proceedings of MIE 2005 - The XIXth International Congress of the European Federation for Medical Informatics, pp. 483–488 (2005)
13. Coenen, F.P., Goulbourne, G., Leng, P.: Computing Association Rules Using Partial Totals. In: Siebes, A., De Raedt, L. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 54–66. Springer, Heidelberg (2001)
14. Kohonen, T., Oja, E., Simula, O., Visa, A., Kangas, J.: Engineering applications of the Self-Organizing Map. Proceedings of the IEEE 84(10), 1358–1384 (1996)
15. Toivonen, H.: Sampling Large Databases for Association Rules. In: Proceedings of the 22th International Conference on Very Large Data Bases, pp. 134–145 (1996)
16. Dong, G., Li, J.: Efficient Mining of Emerging Patterns: Discovering Trends and Differences. In: Proceeding of Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 43–52 (1999)
17. Yu, L., Chung, F., Chan, S., Yuen, S.: Using Emerging Pattern Based Projected Clustering and Gene Expression Data for Cancer Detection. In: 2nd Asia-Pacific Bioinformatics Conference, New Zealand (2004)
18. Somaraki, V., Broadbent, D., Coenen, F., Harding, S.: Finding temporal patterns in noisy longitudinal data: A study in diabetic retinopathy. In: Perner, P. (ed.) ICDM 2010. LNCS, vol. 6171, pp. 418–431. Springer, Heidelberg (2010)

Efficient Privacy-Preserving Data Mining in Malicious Model

Keita Emura[1], Atsuko Miyaji[2], and Mohammad Shahriar Rahman[2]

[1] Center for Highly Dependable Embedded Systems Technology
[2] School of Information Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa, Japan 923-1292
{k-emura,miyaji,mohammad}@jaist.ac.jp

Abstract. In many distributed data mining settings, disclosure of the original data sets is not acceptable due to privacy concerns. To address such concerns, privacy-preserving data mining has been an active research area in recent years. While confidentiality is a key issue, scalability is also an important aspect to assess the performance of a privacy-preserving data mining algorithms for practical applications. With this in mind, Kantarcioglu et al. proposed secure dot product and secure set-intersection protocols for privacy-preserving data mining in malicious adversarial model using zero knowledge proofs, since the assumption of semi-honest adversary is unrealistic in some settings. Both the computation and communication complexities are linear with the number of data items in the protocols proposed by Kantarcioglu et al. In this paper, we build efficient and secure dot product and set-intersection protocols in malicious model. In our work, the complexity of computation and communication for proof of knowledge is always constant (independent of the number of data items), while the complexity of computation and communication for the encrypted messages remains the same as in Kantarcioglu et al.'s work (linear with the number of data items). Furthermore, we provide the security model in Universal Composability framework.

Keywords: Privacy-preserving Data Mining, Malicious Model, Threshold Two-party Computation, Efficiency.

1 Introduction

1.1 Background

The information age has enabled many organizations to gather large volumes of data through data mining. However, the usefulness of this data is negligible if 'meaningful information' or 'knowledge' cannot be extracted from it. Confidentiality is a key issue that arises in any huge collection of data. The need for privacy is sometimes due to law (e.g., for medical databases) or can be motivated by business interests. However, a key utility of large databases today is scientific or economic research. Despite the potential gain, this is often not

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 370–382, 2010.

possible due to the confidentiality issues which arise, leading to concerns over privacy infringement while performing the data mining operations.To address this problem, several privacy-preserving distributed data mining protocols using cryptographic techniques have been suggested. Depending on the adversarial behavior assumptions, those protocols use different models. Classically, two main categories of adversaries have been considered:

Malicious adversaries: These adversaries may behave arbitrarily and are not bound in any way to follow a specified protocol. Protocols that are secure in the malicious model provide a very strong security guarantee as honest parties are 'protected' irrespective of an adversarial behavior of corrupted parties.

Semi-honest adversaries: These adversaries correctly follow the protocol specification, yet may attempt to learn additional information by analyzing the transcript of messages received during the execution.

Scalability is another important aspect to assess the performance of a privacy-preserving data mining algorithm. In particular, scalability describes the efficiency trends when data sizes increase. Such parameter concerns the increase of both performance and storage requirements as well as the costs of the communications required by a data mining algorithm when it is used to compute some joint data. For this reason, a privacy-preserving data mining algorithm has to be designed and implemented with the capability of handling huge datasets that may still keep growing. Therefore, the scalability measure is very important in determining practical privacy-preserving data mining techniques.

Kantarcioglu et al. [10] showed several protocols on equality, dot product and set-intersection operations that are useful to design privacy-preserving data mining algorithms in malicious model for the first time. To the best of our knowledge, this is the only work on designing dot product and set-intersection protocols in privacy-preserving data mining area that involves malicious model. They utilize homomorphic encryption with Non-Interactive Zero Knowledge (NIZK) proof to realize their protocols. For a dot product, they provide an efficient protocol designed against malicious adversaries assuming that at least one party is semi-honest. They also prove that the given protocols are secure against malicious adversaries. The computation and communication overhead for the NIZK proofs of their secure dot product protocol can be expressed as $O(n)$ where n is the size of the input dataset, and the complexity of their secure set-intersection is $O(D)$ where D is the domain of data items (Given that party P_0 has n items and party P_1 has m items, where both m and n are bigger than $O(\sqrt{D})$, or where n or m equal to total item domain size D). Therefore, the efficiency of the protocol is highly dependent on the size of the input dataset. In their protocols, for n data items n NIZK proofs are computed and sent over the communication channel to other party.

We estimate the size of n in the real world application. Let us consider a practical scenario, where two parties having huge amount of data want to mine their data in a privacy-preserving way through some joint function using Kantarcioglu et al.'s approach. For an example, according to [17],

- The US Library of Congress, among the largest databases of the world, holds over 125 million items.

Now, if organizations having such very large databases want to compute some data mining algorithm (for an example, two such libraries want to perform data mining operations for research on efficient library-management) in a privacy-preserving way using Kantarcioglu et al's model, the computation and communication complexity for the n NIZK proofs of n data items (n= 125 million) will pose a huge burden on both the computation and communication resources. Each NIZK proof of plaintext used in Kantarcioglu et al.'s work requires 4 exponentiations in $\mathbb{Z}_N$. That means, 500 million exponentiations will be required to be computed!!! In other words, the number of NIZK proof grows linearly with the number of data items n in the protocols proposed by Kantarcioglu et al.; such a performance bottleneck must be addressed to design efficient privacy-preserving data mining algorithms for practical implementation.

Applications of Dot Product and Set-Intersection: K-means clustering is a simple and very commonly used clustering algorithm in data mining. It starts with an unclustered dataset with n elements and one attributes and outputs cluster assignments of each data element in the set. It requires prior knowledge of the number of clusters k [8,16,2]. K-means clustering uses dot product and equality protocols as building blocks. Some recent studies [18,19] provide privacy-preserving association rule mining algorithms using vertically partitioned data. These algorithms involve secure dot product computation with inputs of length n, where n can be arbitrarily large. As for the secure set-intersection, to determine which customers appear on a 'do-not-receive-advertisements' list, a store must perform a set-intersection operation between its private customer list and the producer's list.

1.2 Related Work

Cryptographic techniques have been used to design many different distributed privacy-preserving data mining algorithms. In general, there are two types of assumptions on data distribution: vertical and horizontal partitioning. In the case of horizontally partitioned data, different sites collect the same set of information about different entities. For example, different credit card companies may collect credit card transactions of different individuals. Secure distributed protocols have been developed for horizontally partitioned data for mining decision trees [13], k-means clustering [12], k-nn classifiers [9]. In the case of vertically partitioned data, it is assumed that different sites collect information about the same set of entities but they collect different feature sets. For example, both a university and a hospital may collect information about a student. Again, secure protocols for the vertically partitioned case have been developed for mining association rules [18], and k-means clusters [8,16]. All of those previous protocols claimed to be secure only in the semi-honest model (we do not consider the proposals which have not used standard cryptographic notions). In [10], authors present two-party secure protocols in the malicious model for data mining. They follow the generic malicious model definitions from the cryptographic literature, and also focus on the security issues in the malicious model, and provide the malicious versions of the subprotocols commonly used in previous privacy-preserving data mining algorithms. There has been some other works related to secure two-party computation [1,14]. In [1], the protocol has been shown secure assuming that at least one-party behaves in semi-honest model. However, the protocol requires both parties to engage in a 'proof of decryption' ability (where a sender sends a set of ciphertexts to the receiver and checks whether the receiver can decrypt all the ciphertexts or not), which increases the communication overhead. On the other hand, [14] proposed a two-party protocol to securely evaluate a 2DNF (Disjunctive Normal Form) formula using homomorphic encryption from vector decomposition. But this protocol has been shown secure only in the semi-honest adversarial model. Recently, [7] proposed efficient set operations against the malicious adversaries. It is based on oblivious pseudorandom function evaluation in the standard model. They assume no trusted set up or trusted third party for the computation, thus increasing the communication overhead. But for data mining applications, assuming the existence of a trusted set up is not unrealistic in practice (e.g., a government organization acting as a trusted party may want to perform data mining operations on the data of some hospitals who do not have any trust among themselves).

1.3 Our Contribution

Considering the problems mentioned above, we provide sophisticated modifications that lead to bigger increases in efficiency of the privacy-preserving data mining algorithms. In this paper, we build efficient and secure dot product and set-intersection protocols in the malicious model. Our approach is as follows:

- Each party generates the ciphertexts of its private data using Paillier's encryption.
- Product of all the ciphertexts is computed.
- The NIZK proof is computed using the result of the product, not for each of the ciphertexts.
- All the ciphertexts and the proof of knowledge are sent to other party.
- The receiver first computes the product of the received ciphertexts, and then verifies the proof of knowledge.
- Both the parties use Cramer's threshold two-party computation to jointly compute their data.

We apply the homomorphic property of Paillier cryptosystem to reduce the number of NIZK proofs in our protocols. Although the constructions of our protocols do not deviate a lot from that of [10], we say that the effect of our construction for efficient implementation is huge. Our approach reduces the computational and communication complexity of the proof of knowledge drastically. In our work, computation and communication for the NIZK proofs are always constant, i.e., they are independent of number of data items. In other words, to perform data mining operation on n data items, we need only one NIZK proof. However, the computation and communication for the ciphertexts are linear with the number of data items ($O(n)$, similar to that of [10]). We also provide the security model in Universal Composability (UC) framework, since it gives a clear view on the security of the protocol in real world scenario.

2 Cryptographic Primitives

2.1 Security Model: Universal Composability (UC)

Security in the UC framework implies that any adversary in the real-life model can be emulated by an adversary in the ideal model. The advantage of this paradigm is that it is possible to show that anything learned by the real-life adversary during the protocol execution is computationally indistinguishable from what is learned by an ideal model adversary. Since in the ideal model, any adversary can learn at most the final result and what is implied by the final result, proving that the real-life model adversary could be simulated by an ideal model adversary implies that real-life adversary could not learn anything more than the ideal model adversary. In other words, the real protocol execution reveal no more information to an adversary than what is revealed to an ideal model adversary. A detailed discussion on the UC framework can be found in [3].

2.2 Homomorphic Encryption

Let $E_{pk}(.)$ denote the encryption function with public key pk and $D_{sk}(.)$ denote the decryption function with private key sk. A public key cryptosystem is called additive homomorphic if it satisfies the following requirements:

(1) given the encryption of plaintexts m_1 and m_2, $E_{pk}(m_1)$ and $E_{pk}(m_2)$, there exists an efficient algorithm to compute the public key encryption of $m_1 + m_2$, such that $E_{pk}(m_1 + m_2) := E_{pk}(m_1) +_h E_{pk}(m_2)$.
(2) given a constant k and the encryption of m_1, $E_{pk}(m_1)$, there exists an efficient algorithm to compute the public key encryption of $k \cdot m_1$, such that $E_{pk}(k \cdot m_1) := k \times_h E_{pk}(m_1)$.

Paillier cryptosystem [15] based on composite residuosity assumption captures the homomorphic property. A detailed description can be found in [15].

2.3 Threshold Two-Party Computation

Cramer et. al. proposed multi-party computation with threshold homomorphic cryptosystem in [3]. Given the common public key pk, the private key sk corresponding to pk is divided into two pieces sk_0 and sk_1. There exists an efficient, secure protocol $D_{sk_i}(E_{pk}(m))$ that outputs the random share of the decryption result s_i along with the NIZK proof showing that sk_i is used correctly. Those shares can be combined to calculate the decryption result. Also any single share of the private key sk_i cannot be used to decrypt the ciphertext alone. In other words, s_i does not reveal anything about the final decryption result. We use the same special version of a threshold decryption such that only one party learns the decryption result, as shown in [10]. Such a protocol could be easily implemented exploiting the fact that for any given $E_{pk}(m)$, the party that needs to learn the decryption result could generate $E_{pk}(r_1)$ and then both parties jointly decrypt the $E_{pk}(m) +_h E_{pk}(r_1)$. Since only one party knows the r_1, only that party can learn the correct decryption result. To prove that a party P_i knows a plaintext, the party can compute the Proof of Plaintext Knowledge PPK(e_m) if he knows an element m in the domain of valid plaintexts such that $D_{sk}(e_m) = m$. Similarly, to prove a multiplication is correct, a party P_i is given an encryption $E_{pk}(m)$, and it chooses constant c and calculates $E_{pk}(m \cdot c)$. Later on, P_i can give the Proof of Correct Multiplication PCM($e_m, e_c, e_{m \cdot c}$) such that $D_{sk}(e_{m \cdot c}) = D_{sk}(e_c) \cdot D_{sk}(e_m)$.

We use the simulators in the security proofs for the above NIZK proofs due to their security properties. The notion of security is such that the state of the adversary returned by those simulators is statistically indistinguishable from the state of the adversary in the real-life model. A different kind of scheme for proof of knowledge named Public Key Encryption with Non-interactive Openning (PKENO) in standard model [4,5,6,11] has been proposed, which is more efficient than the known NIZK proofs in standard model. For proving the knowledge, a verifier needs to know the (ciphertext, proof, plaintext)-tuple. But for data mining applications, the plaintext can not be revealed to the verifier. In other words, PKENO is not suitable for our purpose. We use the NIZK proof of [3,10] in our scheme for fair comparison with the existing works in data mining area.

3 Our Protocols

3.1 Underlying Idea

Our protocol differs from that of [10] in the process of computing the proof of knowledge. After a party completes all the encryptions of its plaintexts, these ciphertexts are multiplied to get a common ciphertext. Then the NIZK proof is constructed for this common ciphertext.

- Let us assume that we have a set of ciphertexts such that $\{c_1 = E_{pk}(m_1), c_2 = E_{pk}(m_2), \ldots, c_n = E_{pk}(m_n)\}$ corresponding to the set of messages $\{m_1, m_2, \ldots, m_n\}$, where n is the number of messages (the ciphertexts are constructed using Paillier's encryption).

- We compute the product of all the ciphertexts such that $C = \prod_{i=1}^{n} c_i = \prod_{i=1}^{n} E_{pk}(m_i)$, and compute the NIZK proof for the product C as $\mathsf{PPK}(C)$. Instead of computing NIZK proofs for each ciphertext c_i where $1 \leq i \leq n$, we have now reduced the number of NIZK proof to one.

- To evaluate $\mathsf{PPK}(C)$, the verifier must compute $C = \prod_{i=1}^{n} c_i = \prod_{i=1}^{n} E_{pk}(m_i)$ first. Then the verifier should verify the $\mathsf{PPK}(C)$. Note that, the plaintext of C is equal to $M := \sum_{i=1}^{n} m_i = (m_1 + m_2 + \ldots + m_n)$, due to the homomorphic property of Paillier's cryptosystem discussed above. The success probability of an adversary A of cheating is $\frac{1}{N}$,where N is a composite number of two big primes, and the probability is negligible (for Paillier encryption, we assume N is 2048-bit (2^{2048}), so the success probability 2^{-2048} is negligible) (See Lemma 1 for proof).

- Note that, we do not avoid the ciphertext computation and communication cost ($O(n)$) to output all of the n ciphertexts, since all of them are necessary for homomorphic encryption to compute the dot product or intersection. However, for practical implementation, the additional costs (i.e., PPK or PCM) must be reduced as much as possible. In our protocol, this additional cost does not depend on the number of items. Due to the reduced number (constant) of proof of knowledge, the following protocols can offer huge savings when the vectors used for the dot product and the set-intersection have many components.

3.2 Efficient and Secure Dot Product Protocol

In a secure dot product protocol, it is required to check whether the final result is correct. This is possible since at least one of the parties will behave

```
Require: Two parties $P_0$ and $P_1$ with the shares $pr_0$ and $pr_1$ of the private key $pr$
and $n$ bit vectors $x_{ij}$ where $x_{ij}$ belongs to $P_i$, and $1 \leq j \leq n$.
Ensure: Return $res = \sum_{j=1}^{n}(x_{0j} \cdot x_{1j}) + r_1)$ to $P_0$ and $r_1$ to $P_1$
for all $P_i$ do
    $\forall j$, set $c_{ij} \leftarrow e_{x_{ij}} = E_{pk}(x_{ij})$; $C_i = \prod_{j=1}^{n} c_{ij} = \prod_{j=1}^{n} E_{pk}(x_{ij})$;
    Create $\mathsf{PPK}(C_i)$
    if $P_i = P_1$ then
      pick rand $r_1 \in \{0,1\}^*$, set $C_{r_1} \leftarrow e_{r_1} = E_{pk}(r_1)$, $\mathsf{PPK}(C_{r_1})$
    end if
    Send all encryptions $c_{ij}$ and $\mathsf{PPK}(C_i)$ to $P_{1-i}$; when $P_i = P_1$, send $C_{r_1}$ and
    and $\mathsf{PPK}(C_{r_1})$ as well
end for
for $P_0$ do
    Compute $C_1 = \prod_{j=1}^{n} c_{1j}$; Check $\mathsf{PPK}(C_1)$, $\mathsf{PPK}(C_{r_1})$ are correct else ABORT
    Calculate $e_{res^0} = (E_{pk}(x_{11}) \times_h x_{01}) +_h \ldots +_h (E_{pk}(x_{1n}) \times_h x_{0n}) +_h E_{pk}(r_1)$
     $= e_{x_{01} \cdot x_{11}} +_h e_{x_{02} \cdot x_{12}} +_h \cdots +_h e_{x_{0n} . x_{1n}} +_h e_{r_1}$
     (/$\star$ due to homomorphic property of the encryption $\star$/)
end for
for $P_1$ do
    Compute $C_0 = \prod_{j=1}^{n} c_{0j}$; Check $\mathsf{PPK}(C_0)$ is correct else ABORT
    Calculate $e_{res^1} = (E_{pk}(x_{01}) \times_h x_{11}) +_h \ldots +_h (E_{pk}(x_{0n}) \times_h x_{1n}) +_h E_{pk}(r_1)$
     $= e_{x_{01} \cdot x_{11}} +_h e_{x_{02} \cdot x_{12}} +_h \cdots +_h e_{x_{0n} . x_{1n}} +_h e_{r_1}$
     (/$\star$ due to homomorphic property of the encryption $\star$/)
end for
Jointly call decrypt equality protocol to check $D_{pr}(e_{res^1}) = D_{pr}(e_{res^0})$
for all $P_i$ do
    If Secure equality protocol returns true for $D_{pr}(e_{res^1}) = D_{pr}(e_{res^0})$ then
      Jointly call private decrypt function s.t. $P_0$ learns $D_{pr}(e_{res^1})$
     (/$\star$ using threshold two-party computation $\star$/)
    end If
end for
```

Fig. 1. Our proposed protocol that uses threshold multiparty computation with homomorphic encryption to compute secure and efficient dot product

semi-honestly. If both parties are malicious, we do not care whether the privacy of any party is protected or parties get correct results. Assuming that at least one party will behave in a semi-honest fashion (the other party can do any malicious act), an efficient protocol can be constructed. Assuming that at least one party is semi-honest is consistent with the definitions of the malicious model. This assumption was also made in [10]. Such an assumption does not reduce the security guarantees provided by the malicious model.

The result of data mining algorithm $res = \sum_{j=1}^{n}(x_{0j} \cdot x_{1j}) + r_1$ is evaluated correctly by at least one party, assuming that at least one party is semi-honest. Both P_0 and P_1 have enough information to calculate res. If both P_0 and P_1 compute the same res value, then computations must be correct, because at least one of them is semi-honest and calculates correct res. Therefore, if we securely make sure that both parties calculate the same value, then either of the local calculations could be decrypted to reveal res to P_0. In our protocol, each party sends the encrypted inputs along with the PPK to each other, then each party P_i locally computes its respective $e_{res^i} = E_{pk}(res^i)$. Both parties jointly decrypt one of those values to reveal res to P_0, given that those two values are equal. We do not need to send PCM for every multiplication. We provide the details of the efficient secure dot product protocol in fig.1.

A note on secure equality protocol: The whole premise of our constructions is that it leads to bigger increases in efficiency when there are a huge number of data items to be processed. However, a secure equality protocol requires to compute whether two data items are equal or not without revealing these items. Therefore, the efficiency of a secure equality protocol under our approach remains same as that in Kantarcioglu et al.'s work [10]. It is straight forward to construct a secure equality protocol in malicious model, due to [10]. For this reason and due to lack of space, we do not include the equality protocol here.

3.3 Efficient and Secure Set-Intersection Protocol

The main idea of this protocol construction is that we can represent the sets owned by each party as a bit vector of size D, and use secure multiplication property of the homomorphic encryption, and we can then associate PPK/PCM proofs to give secure set protocols in the malicious model. Let us assume that x_{0j} is set to 1 if P_0 has item j in its private set else it is set to 0 (similarly for x_{1j} for P_1). Clearly for calculating set-intersection, we need to calculate $x_{0j} \wedge x_{1j}$ for each j. Similarly, for set union, we need to calculate $x_{0j} \vee x_{1j}$ for all j. This can be rewritten as $\neg(\neg x_{0j} \wedge \neg x_{1j})$. Therefore, the dot product protocol for set union can be used, too. The details of the set-intersection protocol is shown in fig. 2. The same protocol can be used for two-party set union using $\neg x_{0j}$ and $\neg x_{1j}$ as the input values and negating the output bits.

Require: Two parties P_0 and P_1 with the shares pr_0 and pr_1 of the private key pr and input bit vectors of size D where x_{ij} is set to 1 if P_i has item j.
Ensure: Return D bit vector I representing the set-intersection where I_j is set to 1 if item j is in the set-intersection.

```
for $P_0$ do
    $\forall j$, set $c_{0j} \leftarrow E_{pk}(x_{0j})$; $C_0 = \prod_{j=1}^{n} c_{0j} = \prod_{j=1}^{n} E_{pk}(x_{0j})$;
    Create $\mathsf{PPK}(C_0)$ to prove that each $x_{0j}$ is either 0 or 1.
    Send all encryptions $c_{0j}$ and $\mathsf{PPK}(C_0)$ to $P_1$
end for
for $P_1$ do
    Compute $C_0 = \prod_{j=1}^{n} c_{0j}$; Check $\mathsf{PPK}(C_0)$ is correct else ABORT
    $\forall j$ Calculate $c_{1j} \leftarrow E_{pk}(x_{1j})$; $C_1 = \prod_{j=1}^{n} c_{1j} = \prod_{j=1}^{n} E_{pk}(x_{1j})$;
    Create $\mathsf{PPK}(C_1)$
    $\forall j$ Calculate $c_{01j} \leftarrow e_{x_{0j}} \times_h x_{1j}$; Compute $C_{01} = \prod_{j=1}^{n} c_{01j}$;
    Create $\mathsf{PCM}(C_{01})$
    Send all ciphertexts $c_{1j}$, $c_{01j}$, $\mathsf{PPK}(C_1)$, $\mathsf{PCM}(C_{01})$ to $P_0$;
end for
for $P_0$ do
    Compute $C_1 = \prod_{j=1}^{n} c_{1j}$, $C_{01} = \prod_{j=1}^{n} c_{01j}$;
    Check $\mathsf{PPK}(C_1)$, $\mathsf{PCM}(C_{01})$ are correct else ABORT;
end for
Jointly call private decrypt function to learn $D_{pr}(c_{01j})$
Set $I_j$ to $D_{pr}(c_{01j})$
```

Fig. 2. Our proposed protocol that uses threshold multiparty computation with homomorphic encryption to compute secure and efficient set intersection

4 Security Analysis

Lemma 1. An adversary who does not know the input plaintexts of the dot product or set-intersection protocols can successfully cheat using its own input strings with negligible probability.

Proof: Note that we do not have to consider the permutation of messages (e.g., $M = m_1 + m_2 = m_2 + m_1$) in the following proof, since such factorization values are reduced by division process of the probability computations. The number of solution vectors $(t_1, \ldots, t_n)$ satisfying $M = \sum_{j=1}^{n} t_j$ is N^{n-1}, since we can randomly choose $t_j \in \mathbb{Z}_p (j = 1, 2, \ldots, n-1)$, and set $t_n = M - \sum_{j=1}^{n-1} t_j$. Cheating means that an adversary $\mathcal{A}$ can compute a PPK which is accepted by the verifier although there exists a message $m \notin \{m_1, \ldots, m_n\}$. Therefore, the probability of the randomly chosen vector $(t_1, \ldots, t_n)$ satisfying $M = \sum_{j=1}^{n} t_j$ is $N^{n-1}/N^n = 1/N$. This implies that, the probability of a proof of false message satisfying $M = \sum_{j=1}^{n} m_j$ is $N^{n-1}/N^n = 1/N$. This implies that, the probability of a proof of false messages satisfying $M := \sum_{j=1}^{n} m_j$ is $\frac{N^{n-1}-1}{N^n} = \frac{1}{N}(1 - \frac{1}{N^{n-1}})$. Next, we consider $t_j = m_j (j = 1, 2, \ldots, \ell)$, where $\ell < n$ is the number of real messages $\mathcal{A}$ can capture. Let $\ell = n-1$, this means $\mathcal{A}$ has all the messages of M except m_n . Then $t_n = m_n := M - \sum_{j=1}^{n-1} m_j$ holds. This means that $\mathcal{A}$ has valid messages of M, and the proof of M is not a forged proof. Therefore, we set $\ell < n-1$. Then there exist $N^{n-\ell-1}-1$ number of pairs of $(t_{\ell+1},\ldots,t_n)$ such that $(m_1,\ldots,m_\ell,t_{\ell+1},\ldots,t_n)$ satisfies $M = (\sum_{j=1}^{\ell} m_j) + (\sum_{j=\ell+1}^{n} t_j)$ and $(t_{\ell+1},\ldots,t_n) \neq (m_{\ell+1},\ldots,m_n)$. The number of vectors $(t_{\ell+1},\ldots,t_n)$ is $N^{n-\ell}$. Therefore, the probability of a proof made by valid messages of $(m_1,\ldots,m_\ell)$ and forged messages of $(t_{\ell+1},\ldots,t_n)$ satisfying $M = (\sum_{j=1}^{\ell} m_j) + (\sum_{j=\ell+1}^{n} t_j)$ is $\frac{N^{n-\ell-1}-1}{N^{n-\ell}} = \frac{1}{N}(1-\frac{1}{N^{n-\ell-1}}) \leq \frac{1}{N}$. □

Note that, in the malicious model, authors may lie about their input. For example, if the inputs are from $\{0,1\}$ domain, malicious attacker can replace it with 2. In many cases, zero knowledge proofs are needed to protect against such attacks. Completeness and soundness are the two properties that should be achieved by the zero-knowledge proof to prevent the parties from behaving in such malicious way. Completeness requires that an honest verifier is satisfied with the prover, while soundness says that a dishonest party cannot convince an honest party with incorrect input. By using the Lemma 1, we can achieve both the properties considering that achieving such properties will fail with negligible probability. In other words, if the statement (proof of knowledge) is true, the honest verifier (that is, one following the protocol properly) will be convinced of this fact by an honest prover, and if the statement (proof of knowledge) is false, no cheating prover can convince the honest verifier that it is true, except with some small probability. This gives us a sketch on why our approach of ZKP construction works (of course with a negligible probability of failure).

The following two theorems state that our constructions are secure in UC framework. We omit the proofs due to lack of space.

Theorem 1. *The Dot Product protocol is secure in (secure equality, private decrypt)-hybrid model assuming that the non-interactive zero knowledge proofs (PPK) are secure in the presence of malicious adversaries.*

Theorem 2. *The Set-Intersection protocol is secure in (threshold decryption)-hybrid model assuming that the non-interactive zero knowledge proofs are secure in the presence of malicious adversaries.*

5 Efficiency

Secure Dot Product: The complexity of secure dot product protocol proposed in [10] can be expressed as $O(n)$ where n is the size of the input dataset, in other words the size of data items to be processed. Therefore, the efficiency of the protocol is highly dependent on the size of the input dataset. The complexity displayed in Table 1 involves both communication and computation times, and the difference can be explained with the impact of communication and computation overhead that the proof of knowledge brings. While the overhead of ciphertext computation and communication in our protocol are similar to that in [10], the overhead of computation and communication of proof of knowledge

Table 1. Secure Dot Product: Performance Comparison between [10] and our proposed protocol

Schemes	Computation		Communication	
	ciphertext	proof-of-knowledge	ciphertext	proof-of-knowledge
[10]	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Ours	$O(n)$	$O(1)$	$O(n)$	$O(1)$

are constant in our protocol. This is a drastic reduction in computation and communication.

Secure Set-Intersection: Given that P_0 has n items and P_1 has m items, where both m and n are bigger than $O(\sqrt{D})$ or where n or m equal to total item domain size D, we (as well as [10]) suggest using the simple secure set-intersection and set union protocols that are secure in the malicious model. According to Table 2, the complexity (for proof of knowledge) in our protocol does not depend on the size of data items, rather it is constant. Note that, we need PCM besides PPK in set-intersection computation. PCM requires 6 exponentiations in $\mathbb{Z}_N$. Considering the library example, [10] will require to compute 750 million exponentiations, whereas our protocol requires only 6 exponentiations.

Table 2. Secure Set-Intersection: Performance Comparison between [10] and our proposed protocol

Schemes	Computation		Communication	
	ciphertext	proof-of-knowledge	ciphertext	proof-of-knowledge
[10]	$O(D)$	$O(D)$	$O(D)$	$O(D)$
Ours	$O(D)$	$O(1)$	$O(D)$	$O(1)$

6 Conclusion

In this paper, we have proposed efficient and secure dot product and set-intersection protocols in malicious model which are useful for many practical applications. These protocols can be used in various data mining algorithms as building blocks. We provide sophisticated modifications that lead to bigger increases in efficiency of the privacy-preserving data mining algorithms. Although the constructions of our protocols do not deviate a lot from that of [10], the effect of our construction for efficient implementation is huge. We do not avoid the ciphertext computation and communication cost ($O(n)$) to output all of the n ciphertexts that are necessary to compute the dot product or intersection. Our approach drastically reduces the computational and communication complexity of the proof of knowledge. In our work, computation and communication for proof of knowledge is always constant (independent of number of data items), while computation and communication for encrypted messages remains linear with the number of data items. We also provide the security model in UC framework.

References

1. Boneh, D., Goh, E.G., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
2. Bunn, P., Ostrovsky, R.: Secure Two-Party k-Means Clustering. In: ACM CCS 2007, pp. 486–497 (2007)
3. Cramer, R., Damgård, I., Nielsen, J.B.: Multi-party computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–299. Springer, Heidelberg (2001)
4. Damgard, I., Hofheinz, D., Kiltz, E., Thorbek, R.: Public-Key Encryption with Non-interactive Opening. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 239–255. Springer, Heidelberg (2008)
5. Damgard, I., Thorbek, R.: Non-interactive proofs for integer multiplication. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 412–429. Springer, Heidelberg (2007)
6. Galindo, D., Libert, B., Fischlin, M., Fuchsbauer, G., Lehmann, A., Manulis, M., Schroder, D.: Public-Key Encryption with Non-interactive Opening: New Constructions and Stronger Definitions. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 333–350. Springer, Heidelberg (2010)
7. Hazay, C., Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 312–331. Springer, Heidelberg (2010)
8. Jagannathan, G., Wright, R.N.: Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 593–599 (2005)
9. Kantarcıoğlu, M., Clifton, C.: Privately computing a distributed k-nn classifier. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 279–290. Springer, Heidelberg (2004)
10. Kantarcioglu, M., Kardes, O.: Privacy-preserving data mining in the malicious model. International Journal of Information and Computer Security 2(4), 353–375 (2008)
11. Lai, J., Deng, R.H., Liu, S., Kou, W.: Efficient CCA-Secure PKE from Identity-Based Techniques. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 132–147. Springer, Heidelberg (2010)
12. Lin, X., Clifton, C., Zhu, M.: Privacy-preserving clustering with distributed EM mixture modeling. Knowledge and Information Systems 8(1), 68–81 (2005)
13. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000)
14. Okamoto, T., Takashima, K.: Homomorphic Encryption and Signatures from Vector Decomposition. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 57–74. Springer, Heidelberg (2008)
15. Paillier, P.: Public-key cryptosystems based on composite degree residue classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
16. Su, C., Bao, F., Zhou, J., Takagi, T., Sakurai, K.: Security and Correctness Analysis on Privacy-Preserving k-Means Clustering Schemes. IEICE Trans. Fundamentals E92-A(4), 1246–1250 (2009)

17. Top 10 Largest Databases in the World, http://www.worldsbiggests.com/2010/02/top-10-largest-databases-in-world.html
18. Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 639–644 (2002)
19. Yang, Z., Wright, R.N.: Privacy-preserving computation of Bayesian networks on vertically partitioned data. IEEE Transactions on Knowledge and Data Engineering 18(9), 1253–1264 (2006)

Analyze the Wild Birds' Migration Tracks by MPI-Based Parallel Clustering Algorithm

HaiMing Zhang, YuanChun Zhou, JianHui Li, XueZhi Wang, and BaoPing Yan

Computer Network Information Center, Chinese Academy of Sciences
100190 Beijing
{hai,zyc,lijh,wxz,ybp}@cnic.cn

Abstract. Aiming at the avian influenza outbreak in Qinghai Lake area, the satellite tracking of migratory birds in Qinghai Lake is studied to analyze the relationship between bird migration, virus spread and ecological environment. These biological problems have been converted into computational studies in previous studies in which spatial clustering is the key factor. A bird migration data analysis system based on DBSCAN algorithm was designed in previous work, by which data can be systematically analyzed, and knowledge patterns are subsequently available for deep biological studies. As the GPS (Global Positioning System) raw data grows rapidly which is large scale with high complexity, DBSCAN takes long time (several minutes) to get the result. In this paper, parallel STING (statistical information grid) algorithm is designed and implemented based on MPI (message passing interface) for spatial clustering. By using parallel STING algorithm, it only takes several seconds to get the result.

Keywords: Clustering, Parallel, MPI, STING, Bird Migration, Scientific data, Tracks, Qinghai Lake.

1 Introduction

The Asian outbreak of highly pathogenic avian influenza H5N1 disease in poultry in 2003 and 2004 was unprecedented in its geographical extent, and its transmission to human beings showed an ominous sign of life-threatening infection [1]. Research findings indicate that the domestic ducks in southern China played a central role in the reproduction and maintenance of this virus, and wild birds may have contributed to the wide spread of the virus. According to this assumption, finding the habitat and migration distance and time of wild birds is important for H5N1 disease research. It is very interesting for biologists to acquire the knowledge about the wetland use of migratory bird species during the annual life circle, as it is critically important in many decision-making processes such as for conservation site construction and avian influenza control [1].

Recent advances in satellite tracking technology have enabled researchers to continuously track the movements of individual birds over a broad spatial scale without conducting extensive field observations after the birds have been equipped with

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 383–393, 2010.

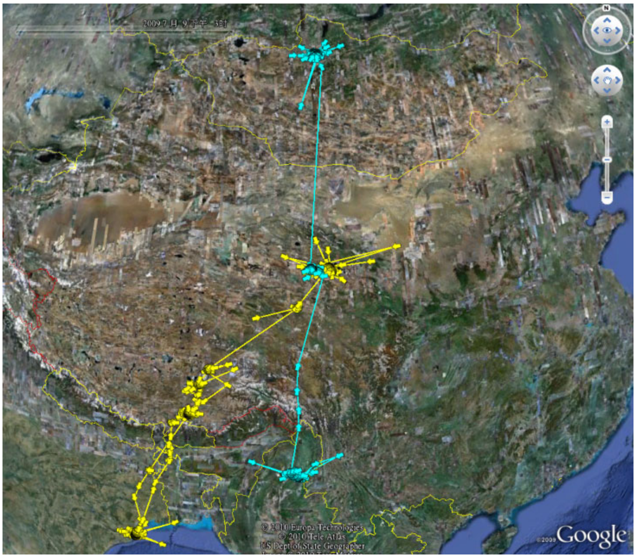


Fig. 1. Two birds' migration route shown on Google earth

satellite transmitters. The applications of satellite tracking to bird migration studies have enabled considerable progress to be made with regard to elucidating the migration routes and stay sites of various migratory bird species, with important implications, for example, for conservation as shown in figure 1. Traditionally, most of biologists have to count those location plots in a certain area and then utilize kernel model to calculate the home range of bird species [5]. Until recently, Hiroto Shimazaki1 et al. [6] proposed a method to examine the location data points based on the idea of clustering. At the first step, their method groups the location points with similar characteristics in approaching speed and departure speed by using the ISODATA algorithm [7]. And then the extent of stay sites is determined by specifying the area attainable by a bird moving speed. At last, they evaluate the site connectedness between stopover sites. However they do not make full use of the bird tracking data—features such as latitude and longitude have not been used to get the habitat range. The identification of the migration routes has not been touched either. As shown in the previous studies that satellite tracking is a powerful to monitor birds' migration behavior, and the data is valuable to make significant contribution to biological research, yet, to the best of our knowledge, it has been long lack of a data mining system capable of conducting systematic migration data analysis. The spatial data analysis on the specie's transmission coordinates together with their layered maps can be conducted by GIS (Geographic Information System) including ESRI'S ARC\INFO 7.1.2 and ArcView 3.1 (Research Institute, Inc., Redlands, California, U.S.A.) [5].

2 Background and Related Work

Spatial clustering can help to find the wild birds' habitats and migration characteristics. Based on the clustering algorithms, the data mining system for Qinghai lake bird

Table 1. Relational representation of bird migration data

Animal	Date&Time	Latitude	Longitude	LC94	Speed
BH07_67695	2008-03-02 03:27:10	29.275	88.731	LZ	32km/h
BH08_82081	2009-04-03 00:14:13	28.814	90.494	L1	43km/h

migration is designed and implemented [2]. In that system, preprocessed data is stored in relation database as shown in table1.

2.1 Overview to Clustering Algorithms

Clustering is an extensively studied topic in the machine learning and data mining field. Clustering, as applied to large datasets, is the process of creating a group of objects organized on some similarity among the members. A clustering algorithm refers to a method that subgroups a set of data points according to a distance or density metrics. There are a number of different methods proposed for performing clustering, but the three main divisions are partitional clustering, hierarchical clustering, and locality-based clustering. In spatial data sets, clustering permits a generalization of the spatial component that allows for successful data mining [9]. In our former system, DBSCAN algorithm is used.

2.2 DBSCAN Algorithm

DBSCAN (Density Based Spatial Clustering of Applications with Noise) is a locality-based algorithm, relying on a density-based notion of clustering. The density-based notion of clustering states that within each cluster, the density of the points is significantly higher than the density of points outside the cluster. DBSCAN is designed to handle the issue of noise and is successful in ignoring outliers. The algorithm can handle large amounts of data, and the order of processing does not affect the shape of the clusters. However, the runtime is O(NlogN) for DBSCAN. In addition, the algorithm is not designed to handle higher dimensional data [9].

Table 2. The statistical information of the birds' GPS raw data

Bird Name	Number of Tracked Birds	Record Start Time	Record End Time	Record Number
Bar headed goose	29	2007-03-21	2010-01-06	973649
Ruddy Shelduck	20	2007-03-21	2010-05-24	393072
Great black headed gull	10	2007-06-21	2008-06-07	37242
Total	59	2007-03-21	2010-05-24	1403963

With the continuous growing of the GPS raw data, the dataset is very large as shown in table 2. It takes several minutes to get the result by DBSCAN algorithm. In order to conduct an online mining system for biologists to analyze the data, the efficiency of clustering needs to be improved.

3 Parallel STING Algorithm Research and Implementation

3.1 STING Algorithm Description

STING (Statistical Information Grid-based method) exploits the clustering properties of index structures. It divides the spatial area into rectangular grid cells using, for example, longitude and latitude. This makes STING order independent, i.e. the clusters created in the next step of the algorithm are not dependent on the order in which the values are placed in the grid. A hierarchical structure is used to manipulate the grid. Each cell at level i is partitioned into a fixed number k of cells at the next level. This is similar to spatial index structures.

Clustering operations are performed using a top-down method, starting with the root. The relevant cells are determined using the statistical information, and only the paths from those cells down the tree are followed. Once the leaf cells are reached, the clusters are formed using a breadth-first search, by merging cells based on their proximity and whether the average density of the area is greater than some specified threshold.

Hierarchical statistical information grid based approach for spatial data mining has much less computational cost than other approaches especially when the data set is very large [9]. The runtime complexity for STING is O(K) where K is the number of cells at the bottom layer assume that K << N. In addition, it offers an opportunity for parallelism (STING is trivially parallelizable) [3]. Parallel STING algorithm is designed and implemented in this paper based on MPI.

The GPS data with 50-meter accuracy is recorded by latitude and longitude coordinates. The latitude range of wild birds' activity area is from about 15°N to 50°N. In this area, the latitude and longitude coordinates can be used directly to divide the spatial area into approximate rectangle cells without coordinate systems transformation for simplicity as shown in figure 2. In this paper, the size of the leaf level cells is set to 0.001 degree in latitude and longitude coordinates. Nine cells form a higher level cell and there are totally 4 levels in the hierarchical structure.

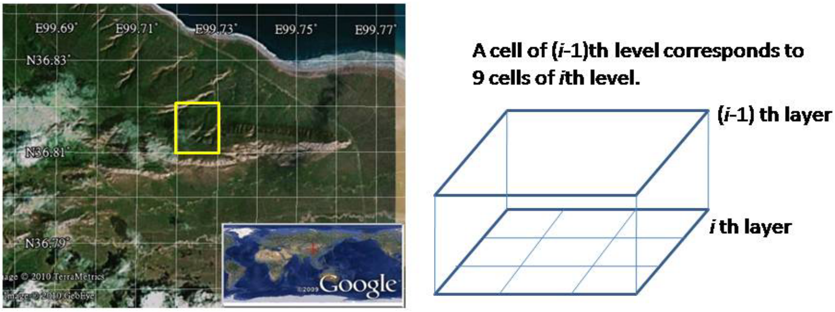


Fig. 2. Divide the spatial area into approximate rectangle cells

For each cell, we have attribute-dependent and attribute-independent parameters. In this paper, the attribute-independent parameter is:

- count: Number of GPS points in this cell

As for the attribute-dependent parameters, we have the following three parameters for each cell:

- m: Mean of all the speed values in this cell
- min: The minimum speed value in this cell
- max: The maximum speed value in this cell

Parameters of higher level cells can be easily calculated from parameters of lower level cell. Let count, m, min and max be parameters of current cell and $count_i$, m_i, min_i and max_i be parameters of corresponding lower level cells, respectively. The count, m, s, min, and max can be calculated as follows.

$$count=\sum count_i$$

$$m=\sum m_i*count_i/count$$

$$min= minimum\ (min_i)$$

$$max= maximum\ (max_i)$$

In order to get the birds' habitat areas, query must be made after the hierarchical structure is constructed. The parameter m of one cell is the key value. The parameters count, min and max are used as auxiliary values. For example, the habitat areas can be queried by the following SQL style query.

Example: Select the maximal regions that have at least 3 GPS points per unit area and the mean of the speed values are no more than 8km/h.

```
SELECT REGION FROM GPS-map
WHERE DENSITY IN (3, ∞)
AND m RANGE (0, 8)
```

Starting with the root, we calculate the likelihood that this cell is relevant to the query at some confidence level using the parameters of this cell [3]. When we finish examining the current layer, we proceed to the next lower level of cells and repeat the same process. Instead of going through all cells, only those cells that are children of the relevant cells of the previous layer are proceeded. This procedure continues until we finish examining the lowest level layer. These relevant cells and their associated statistical information are enough to give a satisfactory result to the query.

3.2 Parallel STING Algorithm

The STING algorithm discussed above can be parallelized to get higher performance. Generate the hierarchical structure and query from the hierarchical structure are two relatively independent parts. Both of them can be implemented parallel. In this section, parallel STING algorithm is discussed.

The parallel STING algorithm consists of two sub-algorithms. Algorithm 1 is used to construct the hierarchical structure from the GPS data. Algorithm 2 is used to get the migratory birds' habitats from the hierarchical structure. The two sub-algorithms are described as follow.

Algorithm 1 (Generate the hierarchical structure of grid cells parallel)
Input: A relational database of bird migration data.
Output: A hierarchical structure of grid cells stored in global database.
Method:

1. Sort the bird migration data by latitude and longitude;
2. Decide the tasks number n and divide the bird migration database into n blocks equally according to the order in step 1;
3. Create 4 separate tables for 4 level layers in global database to store the information of cells in each layer;
4. For each task[i] ($1 \leq i \leq n$), parallel do
   a) For each record in block[i], with all the 4 level layers, calculate which cell it belongs to and update the parameters of the cells by this record [8].
   b) Combine all the cells of all the layers in current task to the global database.
5. Wait until all the tasks finished;
6. Return the hierarchical structure of grid cells stored in the global database.

Note that in step 4 of Algorithm 1, there are two cases when combining one cell to the global database. One case is that there is no record of this cell stored in the database. In this case, we should just insert the cell as a new record to the database. The other case is that there is already one record of this cell in the database. In this case, the parameters of this cell must be combined first and then the record in the global database should be updated. It is easy to get the combined parameters' value as follow.

$$count=count_1 + count_2$$

$$m=(m_1*count_1 + m_2*count_2) /count$$

$$min= minimum (min_1 , min_2)$$

$$max= maximum (max_1 , max_2)$$

With the hierarchical structure of grid cells on hand, we can use a top-down approach to answer spatial data mining queries. For each query, we begin by examining cells on top level layer. The algorithm for parallel querying from the hierarchical structure is given in Algorithm 2.

Algorithm 2 (Query from the hierarchical structure of grid cells parallel)
Input: A hierarchical structure of grid cells generated in algorithm 1 and several threshold values.
Output: The migratory birds' habitats.
Method:

1. Find the relevant cells in top level by the threshold values;
2. Decide the tasks number n and divide the m relevant cells in top level layer into n parts(each part has m/n cells);
3. For each task[i] ($1 \leq i \leq n$), parallel do
   a) Get the cells in part[i].

   b) With each relevant cell, proceed to its next lower level cells. Determine whether the lower level cell is relevant. Label them as relevant or not relevant.
   c) Repeat the same process in (b) until we finish examining the lowest level layer (bottom layer).

4. Wait until all the tasks finished;
5. Gather all the relevant cells in bottom layer;
6. Find the wild birds' habitat regions from all the relevant cells in bottom layer [3];
7. Return the migratory birds' habitats.

Note that in step 6 of Algorithm 2, because the side length of bottom layer cell is carefully set up to fit the clustering, we just need to examine neighboring cells and find regions that are formed by connected cells.

3.3 Algorithm Implementation

The two sub-algorithms are coded in C++ and run in open source MPICH2 environment which is a high-performance and widely portable implementation of the message passing interface (MPI) standard. Both of the two sub-algorithms are designed to avoid multiple communications among tasks. So the network load is low when the program is running.

4 Algorithm Analysis and Experiment Results

Algorithm analysis: The number of the tasks can be determined by the number of CPUs. In parallel computing, communications among tasks usually consume a lot of additional time which significantly reduces the performance. The parallel STING algorithm is designed to avoid a large number of communications among tasks. All tasks are relatively independent in sub-algorithm 1 and 2. After the work for each task is determined, there is no need for the tasks to communicate with others. Note that it is necessary for each task to access the global database in high-speed. Algorithm 1 is incremental. With the new coming GPS data, what we needs to do is just updating all the cells that the new data belong to in the global database. This process is similar to the combining process.

The detailed processing analysis of bar headed goose's tracks is carried out. After the hierarchical structure of grid cells are generated, the relevant cells in top level layer are selected and shown in figure 3. The relevant cells in top level layer are divided into several parts according to the number of tasks. Each task processes its own part. After all the tasks are finished, the relevant cells in bottom level layer are gathered and shown in figure 4. After we have labeled all cells as relevant or not relevant, we can easily find all the regions that satisfy the density specified by a breadth-first search [3]. In order to get regions from relevant cells in high-speed, the side length of bottom layer cell is carefully set up and just neighboring cells need to be examined. The regions are formed by connected cells. The single relevant cells and small regions are excluded in our implementation.

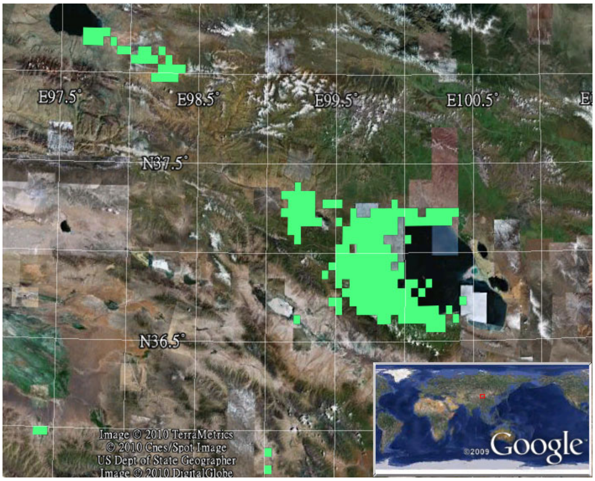

E97.5°
E98.5°
E99.5°
E100.5°
N37.5°
N36.5°
Image © 2010 TerraMetrics
© 2010 Cnes/Spot Image
US Dept of State Geographer
Image © 2010 DigitalGlobe
©2009 Google

Fig. 3. The relevant cells in top level layer

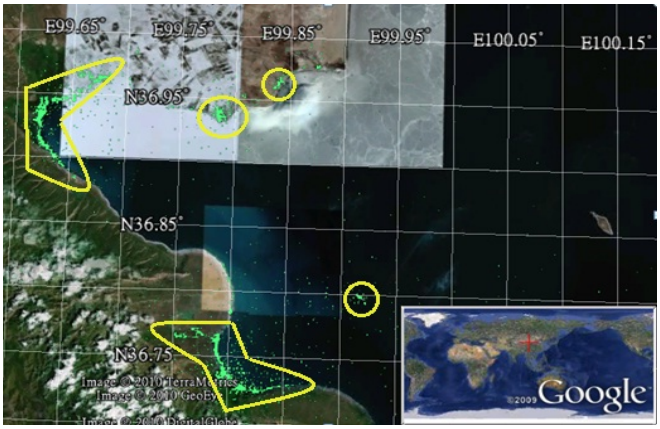


Fig. 4. The relevant cells in bottom level layer and the habitat areas

The habitats discovered by using parallel STING algorithm are show on Google earth. There are five main habitat areas which are marked out by yellow line in figure 4. Field observations were conducted in these areas. The result is proved to be correct and accurate. In order to compare the cluster area spatial distribution between different years, time-related parameters are added to the STING cell. This can help to discover the habitat changing trend of bird species. The cluster results in different years match greatly, which indicates that some certain habitats, such as the Qinghai Lake, Daling Lake and the Tibet river valley, are of vital importance for some species. The result is useful for scientists to discover the way of highly pathogenic avian influenza dispersing with some other research, such as virus, plant and environment quality survey.

5 Performance Evaluation

We run several tests to evaluate the performance of parallel STING. The following tests are run on 4 X86_64 machines with Linux Fedora13 operating system (8 CPU cores of 2.4GHz, 32GB memory). Besides, an independent oracle server is used as the high-performance global database.

In order to verify the performance improvements of parallel algorithm, tests with different number of tasks was carried out. The tasks number was from 1 to 10. All the data of 1403963 records were processed. The time consuming and parallel acceleration rates are shown in figure 5. When only one task was executed, the time consuming was 9 seconds and the parallel acceleration rate was 1.

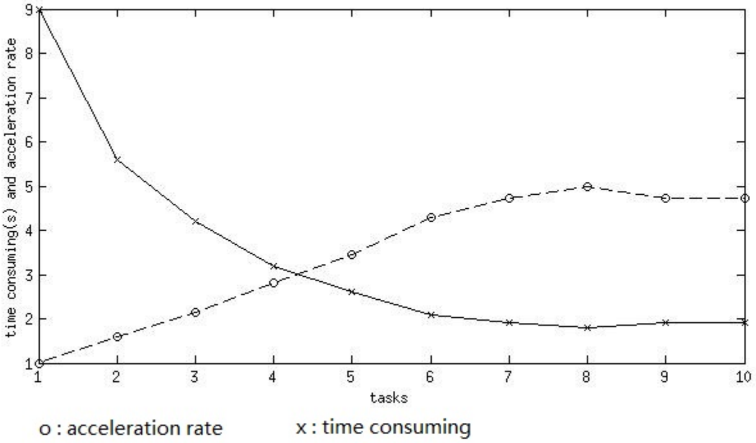


Fig. 5. Tests carried out with different number of tasks

It shows in figure 5 that the best parallel acceleration rate of parallel STING algorithm is 5 when 8 tasks are executed on different CPUs. The parallel acceleration rate doesn't improve when the tasks number is over 8. By real-time monitoring we found that the global database limited the parallel speedup as every task need to access the database in both of the two sub-algorithms, and the database became the bottleneck. In order to get the liner speedup ratio, replacing the global database with parallel file system is a good solution.

The performances of DBSCAN and STING algorithms are compared as shown in table 3. In parallel query STING test, 8 tasks are executed which are evenly distributed in 4 computers. All consuming times are in units of seconds. The dataset of all the 1403963 GPS records are preceded in these tests.

DBSCAN algorithm needs 318 seconds to get the result while parallel STING algorithm only needs 105+1.8 seconds. It should be noted that the hierarchical structure of grid cells only need to be generated once for each dataset while different queries can be carried out in STING algorithm. Logically STING needs extra storage space in database to save the hierarchical structure. So actually it takes only several seconds to finish a query in STING. Parallel STING outperforms DBSCAN by a very large margin.

Table 3. The performances of DBSCAN and STING algorithms

Algorithm	DBSCAN	STING (query)	STING (parallel query)	STING (parallel generation)
Time consuming(s)	318	9	1.8	105

6 Conclusion and Future Work

The satellite tracking has been used successfully to record the migration routes and stopover sites of a number of birds. In order to find the useful information such as new habitats from the large number of the GPS data for biologists to do further research, a bird migration data mining system was designed and implemented [2]. In the previous system, it takes many minutes to get one result that the user can't do analysis efficiently. In this paper, parallel STING algorithm is used to improve the clustering's performance. The parallel query operation needs no more than 2 seconds to get the habitat areas from the STING hierarchical structure. An online processing system is developed based on this algorithm to help biologists to analyze the GPS data of the wild birds' migration tracks. With the continued growth of the raw data, parallel file system is intended to be used in the mining system. We also intend to extend our analysis to other species in the Qinghai Lake to identify the cross habitat for different species. Satellites browse images and other data such as amount of precipitation and human activity information will also be analyzed.

Acknowledgments. The work is granted by the Natural Science Foundation of China (NSFC) under grant number 61003138, the Knowledge Innovation Program of the Chinese Academy of Sciences, No. CNIC_QN_09009, Special Project of Informatization of Chinese Academy of Sciences in "the Eleventh Five-Year Plan", E-Science Application of Research on Resources, Disease Monitoring and Risk Assessment of Important Wild Birds in Qinghai Lake Region under Grant No. INFO-115-D02, Special Project of Informatization of Chinese Academy of Sciences in "the Eleventh Five-Year Plan", Basic Databases of Joint Research Center of Chinese Academy of Sciences and Qinghai Lake National Nature Reserve under Grant No. INFO-115-C01-SDB2-02. The authors would like to thank the researchers in Western Ecological Research Center, Patuxent Wildlife Research Center and United States Geological Survey for sharing the data sets.

References

1. Liu, J., et al.: Highly pathogenic H5N1 influenza virus infection in migratory birds. Science 309, 1206 (2005)
2. Tang, M., Zhou, Y., Cui, P., Wang, W., Li, J., Zhang, H., Hou, Y., Yan, B.: Discovery of Migration Habitats and Routes of Wild Bird Species by Clustering and Association Analysis. In: Huang, R., Yang, Q., Pei, J., Gama, J., Meng, X., Li, X. (eds.) ADMA 2009. LNCS, vol. 5678, pp. 288–301. Springer, Heidelberg (2009)

3. Wang, W., Yang, J., Muntz, R.: STING: A Statistical Information Grid Approach to Spatial Data Mining. In: Proceedings of the 23rd International Conference on Very Large Data Bases, pp. 186–195 (1997)
4. Yutaka, K., et al.: Discovery of breeding grounds of a Siberian Crane Grus leucogeranus flock that winter sin Iran, via satellite telemetry. In: Bird Conservation International, pp. 327–333 (2002)
5. Mathevet, R., Tamisier, A.: Creation of a nature reserve, its effects on hunting management and waterfowl distribution in the Camargue (southern France). In: Biodiversity and Conservation, pp. 509–519 (2002)
6. Shimazaki, H., et al.: Migration routes and important stopover sites of endangered oriental white storks (Ciconia boyciana) as revealed by satellite tracking. Mem. Natl Inst. Polar Res., Spec. Issue 58, 162–178 (2004)
7. Ball, G.H., Hall, D.J.: ISODATA: a novel method of data analysis and pattern classification. Technical Report of Stanford Research Institute, Menlo Park, CA, Stanford Research Institute, p. 66 (1965)
8. Wang, W., Yang, J., Muntz, R.: STING+: An Approach to Active Spatial Data Mining. In: Proceedings of the International Conference on Data Engineering (ICDE 1999), Sydney, pp. 116–125 (1999)
9. Berkhin, P.: A survey of clustering data mining techniques. In: Jacob, K., Charles, N., Marc, T. (eds.) Grouping Multidimensional Data: Recent Advances in Clustering, pp. 25–71. Springer, Heidelberg (2006b)

Formal Concept Analysis Based Clustering for Blog Network Visualization

Jing Gao and Wei Lai

Faculty of Information and Communication Technologies
Swinburne University of Technology
PO Box 218 Hawthorn Victoria 3122, Australia
{jgao,wlai}@groupwise.swin.edu.au

Abstract. Blog network is growing explosively recently. As a result, the network becomes huge and dynamic. People have an urge to have effective ways to explore and retrieve related information. Blog analysis has been investigated for several years. There are still some improvement space exist. In this paper, we provide a formal concept analysis based clustering visualization to help people find information easily. Especially it is easy for them to find hot topics and their related information. Our approach has several steps such as extracting keywords from individual blog entries, formal concept analysis (FCA) based clustering and user interactions. Compare with other applications, the main difference is using FCA to analysis the content of individual entries so that group similar entries into one community. Experiments results are provided to show the advantages of our approach.

Keywords: Clustering, FCA, Information Visualization.

1 Introduction

Blog is a popular and flexible way to publish information and express feelings, especially for private use. With the enormous growth of Internet, it has become more and more difficult for people to discover useful information. Most of the current search engines just provide a ranked list based on users' queries. Normally, the size of the list is over a thousand. This searching process is inefficiency. Most people just review the top ranked results in the first page and hence some relevant information could be missed out. Moreover, the ranking is for all users without any customized option. Most queries are very short, so there are not clear for search engines. As the result, the ranked list might has few relevant information for users.

To facilitate retrieving information, some search engines organize web pages into different categories or directories. The blogshpere can also be classified into distinct groups. Pedersen and Hearst [1] showed that similar documents tend to have relevant information, thus clustering these documents would help users find relevant information. However, there is a gap running through the semantic treatments of categories and concepts [2].

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 394–404, 2010.

In this study, we propose a new content based clustering using Formal Concept Analysis (FCA). FCA is used to construct a concept lattice for the extracted keywords from individual entries. A heuristic method is proposed to select the most relevant entries based on the generated concepts and the user query. Concept ranking and clustering are part of the process. The result is presented as a graph to the user. We treat this approach as Blog Concept Clustering (BCC).

The remainder of this paper is organized as follows. We will give some related work in section 2, followed by some basic definitions in section 3. Section 4 and section 5 will present our main clustering method and experiment results. Conclusion and future work will be provided in section 6.

2 Related Work

2.1 Background

Publishing information online is much easier than ever before. People could express their feelings and comment on some events or news. The most popular place is blog. Most people have personal blogs and some companies also have their blogs. Based on the statistics from Blogpulse, the total identified blogs are over one hundred millions, and the growing is fast since it has over 50,000 blogs created within 24 hours. Blogs subjects frequently change and express a wide variety of opinions, because they are written by different kinds of people at different time with different knowledge level. Having observed this huge network, some researchers have applied social network analysis methods to the blogsphere. They are trying to find out how people share information and make own network. Touchgraph [3] and Vizster [4] are some examples.

Also for each blog, it has three ways to interactive with other people such as blogroll links, citation links and comments. At the moment, people tend to group blogs based on part or all of these three criteria. People did some work to analysis the linkage information inside the blog so that to find out some networks for the blog. People did the work based on the mutual awareness which is based on the exist linkage information. Except the linking based clustering, researchers start using the tags to group blogs. Some blog softwares provide tag option during the creation process such as [5,6].

2.2 Our Motivation

The existing clustering methods have some disadvantages. First of all, the structure based clustering which is normally using the linking information ignores the content inside the blogs. When people look for relevant information, they want to get similar documents together. However, the structure based clustering only considers the linking information without any content consideration. Xiaodi et.al [7] gave a structure based clustering method, which only consider the linking information. Well Jianhua et. al [8] and Vassiliki et.al [9] also gave the structure based clustering. But, they gave little information related to the content of the page. For example, people always put their friends' blog on the blogroll but they discuss different topics. In order to group related information together, we have to analysis the content of the blogs. Secondly, the tags using in the blogs are provided by individual user. However, there are problems using free-form tagging. There is no relationship and formal inclusion criteria between terms. And all the tags are without formal definitions. It is hard to group similar content only based on tags, such as bags and belts.

We want to improve the blog clustering by considering the content aspect. Catch all the information on each blog entry and then analysis the content relationship among all the entries. Using FCA to group blogs into different topics and finally give the visualization. Our approach is useful in various scenarios. Because large number of blogs are created everyday, it can be used to provide a cluster view of blogs. It reduces the information overload by automatically generating keywords of each cluster of blogs it identifies. Also it can be used as an aid to identify the "trends" of blogs, which keywords are popular and which cluster is being most discussed by individuals.

3 Formal Concept Analysis

Formal Concept Analysis is a theory of data analysis which identifies conceptual structures among data sets. It was introduced by Rudolf Wille in 1984 [10] and has since then grown rapidly. It can be used for investigating and processing explicitly given information. The data are organized into units which reflect concepts of human thought allowing meaningful comprehensible interpretation.

3.1 Basic Definitions

Formal Concept Analysis takes as input a matrix specifying a set of objects and a set of keywords, and find the different clusters. We refer the matrix as the formal context, the set of objects as O and the set of keywords as K.

Definition 1. ***Formal Context:*** *A formal context is a triple (O, K, R), where O is a set of objects, K is a set of keywords and $R \subseteq O \times K$ is a binary relation between O and K. The formal context forms as a matrix, which columns are keywords, rows are objects and the incidence relationship is represented in each cell. We use $(o, k) \in R$ to indicate the object o has the keyword k.*

Table 1 gives an example formal context (O, K, R). $O = U1, U2, ..., U5$, $K = K1, K2, ..., K8$ and the cross signs indicate the relationships between objects and keywords. For $U \subseteq O$, we have $U' := \{k \in K | \forall o \in U : (o, k) \in R\}$ and dually for $T \subseteq K : T' := \{o \in O | \forall k \in T : (o, k) \in R\}$. We say U' is the set of all common keywords shared by the objects of U, while T' is the set of all objects have keywords in T

Definition 2. ***Formal Concept:*** *Let the formal context be $C = (O, K, R)$. A formal concept is a pair (A,B) with $A \subseteq O$, $B \subseteq K$ and $A' = B \wedge B' = A$. The set of objects A is referred to as the extent of the concept (A,B) and the set of keywords B is referred to as the intent of the concept (A,B). The set of all formal concepts of the formal context C is denoted by B(C).*

Table 1. Formal Context

No.	Objects	K1	K2	K3	K4	K5	K6	K7	K8
1	U1			X					
2	U2	X	X	X	X	X			
3	U3	X	X				X	X	
4	U4	X						X	X
5	U5							X	X

In table 1, $(\{U1, U2\}, \{K3\})$ is a formal concept of the formal context. $\{U1, U2\}$ are the concept's extent, while $\{K3\}$ is the concept's intent.

To organize all the concepts for a formal context, a partial ordering can be obtained over the concepts. We write the partial order relationship as $\leq$, which represents "is a subconcept of" relation. If we have a formal context $C = (O, K, R)$ and two formal concepts $C_1 = (O_1, K_1) \in B(C)$, $C_2 = (O_2, K_2) \in B(C)$, also the concept relation is given by $(O_1, K_1) \leq (O_2, K_2) \Leftrightarrow O_1 \subseteq O_2 (\Leftrightarrow K_2 \subseteq K_1)$. A concept $C_1 = (O_1, K_1)$ is a subconcept of concept $C_2 = (O_2, K_2)$ only if the set of C_1's objects is a subset of C_2's objects. Equivalently, the set of keywords in C_2 should be the subset of C_1's keywords. That is, a subconcept contains fewer objects and more keywords than the superconcept.

The set of the concepts with the partial order is always a complete lattice, which is called concept lattice and written as $L := (B(C), \leq)$. Fig. 1 shows the concept lattice for the context in table 1. The lattice is normally represented as a Hasse Diagram. It has nodes and links. Each node is a concept with its extent and intent. Links are the connection for the subconcept and superconcept among the concepts. Objects propagate to the top of the diagram and keywords propagate along the links to the bottom of the diagram. We could see that the more abstract nodes exist at higher level, whereas the more specific ones stay at the lower level in the diagram.

3.2 Conceptual Scaling in FCA

In some cases, they are not always binary relationships. FCA has to apply to the contexts, which objects have keywords with values. For example in table 1, K_1 as a keyword exists in U_2, U_3 and U_4. However, for these three objects, they might have different percentage related to K_1, such as 2 for U_2, 20 for U_3 and 70 for U_4. Because of the different percentages, K_1 plays different roles into the three objects. The more percentage the keyword has, the more important it is in the object. We call this kind of context as many value context rather than the normal binary context.

Definition 3. ***Multi-Valued Context:*** *A Multi-Valued Context* (O, K, V, R) *is a set of objects (O) has different values (V) on a set of keywords (K) followed*

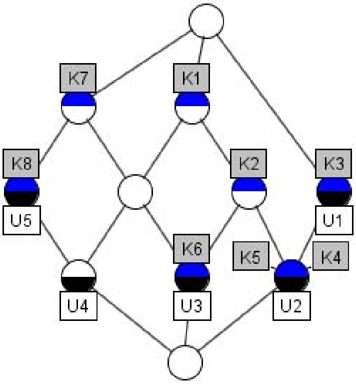


Fig. 1. Concept Lattice for Table 1

by a relationship $(R \subseteq O \times K \times V3)$*, where the following holds:* $(o, k, v_1) \in R \wedge (o, k, v_2) \in R \Rightarrow (v_1 = v_2)$*.*

Like the single valued context, the multi-valued context can be represented as a matrix as well. Such columns are the keywords, rows are the objects and each cell has value according to the relationship between the object and the keyword. We call this matrix as the indicator matrix. Table 2 gives a multi-valued context.

To get the concept lattice for the multi-valued context, we have to analyze the values and the corresponding keywords and then transfer the context into a single valued context. This process is called conceptual scaling in FCA. But the

Table 2. A Multi-Valued Formal Context

Objects	Type	Price	Color
Item 1	Grape	\$3.99	Green
Item 2	Apple	\$1.99	Green
Item 3	Apple	\$2.99	Red
Item 4	Grapefruit	\$5.99	Yellow

Table 3. An Example Rule of Scale Contexts

S_{price}	Cheap	Average	Expensive
$0 \leq price \leq 2$	X		
$2.01 \leq price \leq 4$		X	
$4.01 \leq price \leq 6$			X

Table 4. Derived Context for Price

C_{price}	Cheap	Average	Expensive
Item 1		X	
Item 2	X		
Item 3		X	
Item 4			X

indicator matrix is not unique. It depends on the transformation rules. We have to construct a rule (scaling) for each attribute. This scaling is used to form a single valued context. Take the table 2 as an example. To make it clear, we only show a scaling for the "price". Table 3 shows the rules for scaling. Table 4 gives the scaled context of the original one. Fig. 2 shows the concept lattice for the multi-valued context. The whole process can be achieved by using ConExp [11].

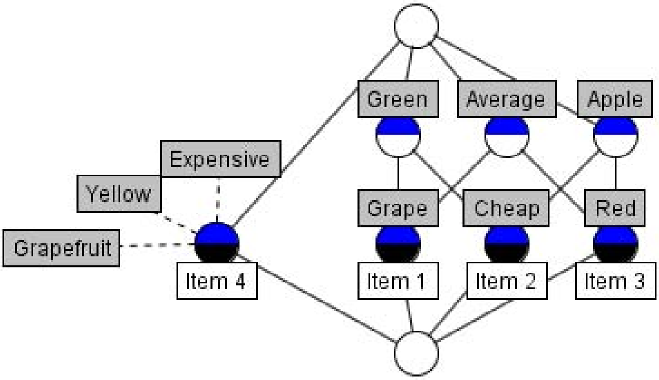


Fig. 2. Concept Lattice for Multi-Valued Context in Table 2

4 Find Clusters from Concept Lattice

By looking at the concept lattice, we could find that some concepts are irrelevant for users' queries. To help users quickly find relevant information, we only select those concepts whose intend contain part or full of the user queries. By calculating the importance and similarity of these concepts, we will find clusters about different keywords. To help users navigate and browse the blog entries, we put all the clusters as a graph in an interactive interface. The detailed process will be shown in the following subsections.

4.1 Concept Ranking

The concept ranking (CR) is a process to find out how relevant a concept is compared with the query. Each concept has extent and intent. The more objects are in the extent of a concept, the more times the keywords of the concept occur in the objects. In the lattice, we have subconcept-superconcept relationship. Subconcept always contains more specific keywords than the superconcept, the opposite applies to the objects such that subconcept contains fewer objects than the superconcept. From the keywords point of view, subconcept gives support to the superconcept. We borrow the famous PageRank [12] from information retrieval and give CR as follows:

$$CR(C_i) = (1-p) + p \sum_{C_j \in B(C)} \frac{CR(C_j)}{N(C_j)} . \tag{1}$$

where $B(C)$ is the whole set of concepts, $N(C_j)$ is the number of superconcept C_j has and p is a damp factor which we set as 0.85 here. We use Eq(1) to calculate the CR value of each concept and then sort them in descending order.

4.2 Concept Similarity

Concept similarity (denoted by CS) shows how close two concepts are. To calculate the similarity, we need to create the concepts and keywords matrix.

Definition 4. ***Concept Vector:*** *Let $B(C) = (IN, E)$ be a concept with $|IN| = n$ and context $C = (O, K, R)$ has $|O| = m$ $|K| = q$, the concept vector is defined as $C = (V_{k_1}, V_{k_2}, ..., V_{k_q})$. V_{k_i} is obtained as follows:.*

$$\left[V_{k_i} = \begin{cases} \frac{1}{n} \sum_{j=1}^{n} R(in_j, k_i) & \text{if } k_i \in E \\ \frac{1}{m} \sum_{j=1}^{m} R(o_j, k_i) & \text{if } k_i \notin E \end{cases} \right] . \tag{2}$$

where $in \in IN$, $o \in O$ and $k \in K$.

The method to calculate each keyword within a concept is different. It depends on whether the keyword is in the intent of each concept or not. The concept vector is the base for getting the similarity between concepts. This vector is obtained from the context based on the intent and extent of each concept without any information loss.

Definition 5. ***Concept Similarity:*** *Concept Similarity (CS) is calculated based on the concept vector using traditional Euclidean Distance. For concept $C_1 = (V_{k_1}, V_{k_2}, ..., V_{k_q})$ and $C_2 = (V_{k_1}, V_{k_2}, ..., V_{k_q})$, we have the Euclidean Distance as:*

$$CS(C_1, C_2) = \sqrt{(V_{1k_1} - V_{2k_1})^2 + (V_{1k_2} - V_{2k_2})^2 + ... + (V_{1k_q} - V_{2k_q})^2} . \tag{3}$$

For any pair of concepts, they have the similarity as:

$$CS(C_m, C_n) = \sqrt{\sum_{i=1}^{q} (V_{mk_i} - V_{nk_i})^2} . \tag{4}$$

Based on the similarity between concepts, we could easily find close concepts. This is the idea for clustering.

4.3 Clusters from Lattice

In order to form clusters, we need to go through the following steps:

1. Filter concepts based on user queries.
2. Calculate Concept Ranking for remained concepts and sort in descending order.
3. Compute Concept Similarity and filter the similarity score based on a threshold.
4. Generate different clusters.

We have given the methods about calculating the concept ranking and concept similarity. In this section, we only explain how we organize the clusters. As we have the concept ranking and concept similarity, we have two ways to form clusters. The simplest way is only considering the concept ranking as the criteria to allocate clusters. Like the PageRank, we return top 20 concepts as 20 clusters. For each cluster, the objects in the extend of the concept are the members and the keywords in the intent are the label for the cluster. It automatically extracts the label for individual cluster and makes browsing easily. The second way is to combine the ranking and similarity. We get the top 10 concepts first, then for each concept we will find the 3 nearest concepts based on the concept similarity. At last, combine the original concepts and their closer neighbors to form the clusters. The following program is the procedure to construct clusters based on formal concept analysis.

```
Input: All Concepts B(C) from Formal Context C,
partial concept set: M<-\phi
Output: Clusters C_{i} (i=1,2,...,n)
for each Concept M_{i}\in B(C) (i=1,2,...,n)
    Calculate CR(M_{i})
    Calculate CS(M_{i}, M_{j})
    {M_{j}|M_{j}\in B(C) && M_{j}\neq M_{i}, j=1,2,...,n}
End for
Option1: Based on CR(M_{i})
   Assign Top 20 Concepts as 20 Clusters
   For each cluster C_{i}(i=1,2,...,n)
   The objects are the corresponding concept's extent
   The cluster's label is the corresponding concept's intent
   End for
Option2: Based on CR(M_{i}) extract top 10 concepts
   For each concept N_{i} (i=1,2,...,10)
       Find the nearest 3 concepts
       N_{j},N_{k},N_{m} (i\neq j\neq k\neq m=1,2,...,n)
       M\leftarrow M \cup {N_{i},N_{j},N_{k},N_{m}}
   End for
   Apply K-Means to M
```

Based on this approach, we could finally form different groups for user navigating and browsing. Next section, we will show our experiment and the comparison with other people's work.

5 Experiment

In this section, we discuss our experiments using the formal concept analysis for discovering different clusters from the blog space. In order to evaluate our proposed approach, we design an interface to show the visualization and interactions.

5.1 Test Data Set

We collected the test data set by writing a program which can automatically extract the content of all blog entries with initial blog entry provided between 5th August and 10th August. In order to get proper result, we skip the entries which only contain 30% words in total. After getting the raw context of the entries, we have examined several problems. First of all, words in the blog entries tend to have special meanings such as 4 is for and u means you. Secondly, there are lots of synonyms. Also the stop words are not useful for clustering. And some words are actually belonged to the same original word. Finally, we found only the top-ranked keywords can reperesent the main content of the blog entry. In order to clean the raw data set, we have set up the following steps:

1. Stopword removing: A list of common used stopwords is provided to eliminate the content length.
2. Word stemming: This step helps to reduce the actually indexed keywords. We apply the famous stemming algorithm in our approach.
3. Extract top-ranked keywords: In this step, we use the classical TF-IDF to rank the keywords. And finally only extract top 20 keywords which stand for each blog entry.

5.2 Construct Concept Lattice and Clustering

After extracting the keywords for each entry, we will create the formal context. Because keywords have different values in each entry, we have to scale the original context. The correspond lattice for part of the test data set is showing in Fig 3. Because we have keywords' values in different entries, we could calculate the concept similarity and concept ranking based on the equations in section 4.

We have all the relevant measurement for generating clusters. We now can select the top-ranked concepts as initial cluster centers and then apply K-Means to form the final clusters.

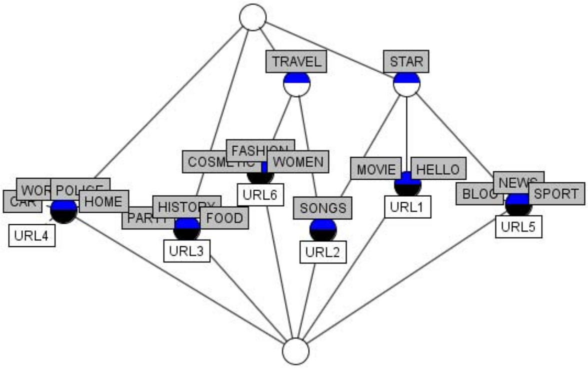


Fig. 3. The Lattice for the Test Data Set

5.3 Visualization and Advantages

All clusters are generated for users' exploring and browsing. To make this process easy, we put the clusters in an interactive interface to reduce information overload for users. Variety interactions can be implemented. Fig 4 is the interface. The "travel" and "star" are the two clusters generated from the constructed lattice.

By analyzing the labels for each cluster, we could see the difference between our method and others. Others tend to use phrase to name clusters. However, we use one or more keywords as the cluster label. This is the main feature for concept lattice based clustering. Each label contains the given query, which is not assigned artificially, but generated automatically by the lattice. As a result, all the clusters are query related and satisfy user need. Our proposed approach gives an effective way to select query related concepts and focus on the blog entries relevant to the interested areas.

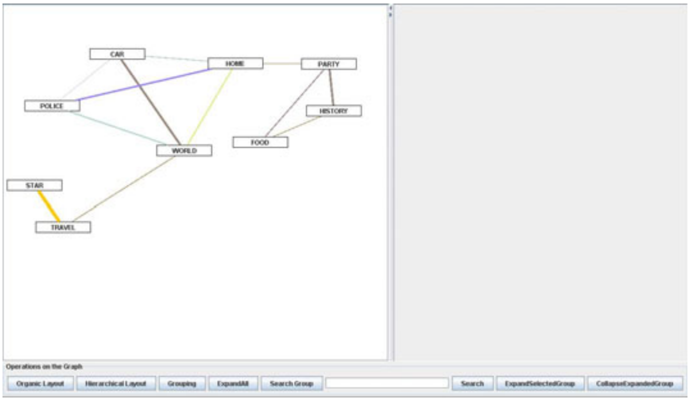


Fig. 4. The Visualization of FCA based Clustering

6 Conclusions

In this work, we propose a formal concept analysis based method to cluster the blogshpere. Initial content of each blog entry is extracted using designed program. Then some operations are conducted on the raw data. Followed by creating formal context and concept lattice, the concept similarity and concept ranking are obtained. With the above data, we do the clustering and visualization process. We also compare our work with others. We see some advantages for our method. The results show that our method could improve the efficiency of users' browsing and searching.

In future, we want to make the separated keywords into meaningful phrases. Also we should contain some semantic aspect for analyzing the content, i.e. considering WordNet during the analysis.

References

1. Hearst, M.A., Pedersen, J.O.: Reexamining the cluster hypothesis: scatter/gather on retrieval results. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 76–84. ACM Press, New York (1996)
2. Laurence, S., Margolis, E.: Concepts and Cognitive Science. MIT Press, Cambridge (1999)
3. TouchGraph, http://www.touchgraph.com
4. Jeffrey, H., Danah, B.: Vizster: Visualizing Online Social Networks. In: Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization. IEEE Computer Society, Washington (2005)
5. Blogger, https://www.blogger.com/start
6. Livejournal, http://www.livejournal.com/
7. Huang, X.D., Lai, W.: Clustering graphs for visualization via node similarities. Journal of Visual Language Computing 17(3), 225–253 (2006)
8. Li, J.H., Laleh, B., Blair, S.: A Structure Based Clustering Algorithm with Applications to VLSI Physical Design. In: Proceedings of the Fifth International Workshop on System-on-Chip for Real-Time Applications, pp. 270–274. IEEE Computer Society, Washington (2005)
9. Vassiliki, A.K., Vakali, A., Mpalasas, A., Valavanis, M.: A Structure-Based Clustering on LDAP Directory Information. In: An, A., Matwin, S., Raś, Z.W., Ślezak, D. (eds.) Foundations of Intelligent Systems. LNCS (LNAI), vol. 4994, pp. 121–130. Springer, Heidelberg (2008)
10. http://en.wikipedia.org/wiki/Formal_concept_analysis
11. ConExp, http://sourceforge.net/projects/conexp
12. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical report. Computer Science Department, Stanford University, California (1998)

Finding Frequent Subgraphs in Longitudinal Social Network Data Using a Weighted Graph Mining Approach

Chuntao Jiang, Frans Coenen, and Michele Zito

Department of Computer Science,
University of Liverpool, Ashton Building,
Ashton Street, L69 3BX Liverpool, UK
{c.jiang,coenen,zito}@liverpool.ac.uk

Abstract. The mining of social networks entails a high degree of computational complexity. This complexity is exacerbate when considering longitudinal social network data. To address this complexity issue three weighting schemes are proposed in this paper. The fundamental idea is to reduce the complexity by considering only the most significant nodes and links. The proposed weighting schemes have been incorporated into the weighted variations and extensions of the well established gSpan frequent subgraph mining algorithm. The focus of the work is the cattle movement network found in Great Britain. A complete evaluation of the proposed approaches is presented using this network. In addition, the utility of the discovered patterns is illustrated by constructing a sequential data set to which a sequential mining algorithm can be applied to capturing the changes in "behavior" represented by a network.

Keywords: Frequent subgraph mining, Weighted graph mining, Social network mining, Longitudinal data.

1 Introduction

Social Network Mining (SNM) [1] is a sub-field of research within the context of Knowledge Discovery in Data (KDD). A social network comprises a set of nodes that represent entities and links that represent some form of communication between such entities. The entire network is not necessarily connected and usually includes some *super-nodes* to which many other nodes are connected. Popular examples of social networks include social networking sites such as FaceBook[1] and Flickr[2]. However, there are many other forms of social networks such as information sharing networks and co-authoring networks. Many of these networks, such as co-authoring networks, are derived from tabular data. In the case of co-authoring networks [2] these can be derived from web applications such as DBLP[3] and CiteSeer[4]. The focus of the work described in this paper is the

[1] http://www.facebook.com

[2] http://www.flickr.com

[3] http://www.informatik.uni-trier.de/ ley/db/

[4] http://citeseerx.ist.psu.edu/

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 405–416, 2010.

Cattle Tracking System (CTS) database, in operation in Great Britain, from which a cattle movement network can be extracted. However, the techniques described are generally applicable.

SNM is directed at the identification of patterns within social networks networks. The nature of the patterns can take many forms, a common type of SNM is the identification of clusters of frequently "corresponding" nodes. SNM is usually applied in a static context, i.e. a "snap shot" of the network is taken to which SNM is then applied. In this paper we are interested in applying SNM techniques to sequences of such snap shots where each snap shot is time stamped in someway. We refer to these sequences as *longitudinal social networks*, in that the sequences may be compared to longitudinal data collections such as those found in medical applications [3]. More specifically we are interested in finding frequently occurring subgraphs in longitudinal social network data.

Social network data is often substantial, usually comprising many nodes and links. Consequently, longitudinal social network data tends to be even more substantial, typically an order of magnitude relative to the number of time stamps to be considered. Standard frequent subgraph mining algorithms, such as gSpan [4], are thus unable to process such large longitudinal networks in a realistic manner. In recognition that some links (and/or nodes) may be considered to be more important than others, in this paper weighted longitudinal social network mining is proposed. Weightings can be applied in a number of manners, three weighting schemes are proposed and evaluated in this paper.

The rest of this paper is structured as follows. Some previous work and a problem definition are presented in Sects 2 and 3 respectively. Three proposed weighting schemes to achieve longitudinal SNM are presented in Sect 4. The suggested weighting schemes are evaluated and compared in Sect 5. Some conclusions and main findings are presented in Sect 6.

2 Previous Work

Examples of research work on applying frequent subgraphs to analyzing social networks include: community pattern mining [5], targeted advertising [6], and structural prediction [7]. Frequent subgraph mining [8,9,4] entails two significant overheads: candidate set generation and (sub)graph isomorphism checking. However, these overheads are exacerbated when the size of the graph data is substantial and the support threshold is low. Weighted frequent subgraph mining [10] advocates the use of *weighted support counts* to identify weighted frequent subgraphs. Hence, the "computational burden" of subgraph mining can be considerably alleviated by generating a set of weighted frequent subgraphs.

3 Problem Definition

This section provides the necessary longitudinal social network representation and mining definitions

Definition 1. *A longitudinal social network is comprised of a sequence of graphs $N_G = \{G_1, G_2, \cdots, G_T\}$, where G_t is a graph corresponding to the social network representation at time period of $t \in [1, T]$. Let $V = \bigcup_{t=1}^{T} V_t$ denote the whole set of entities for the network. Each entity $v \in V$ is uniquely labeled, and each v can appear only once at each time-step.*

Definition 2. *For any arbitrary subgraph $g = (V', E')$ such that $V' \subseteq V$, the support set of g is defined as $\delta(g) = \{t | g \subseteq G_t\}$, e.g., the set of time-steps for which g is a subgraph of G_t. The support of g with respect to N_G, $sup(g)$, is defined to be the cardinality of the support set, $|\delta(g)|$.*

Definition 3. *A subgraph g is frequent with respect to N_G if $|\delta(g)| \geq \tau \times T$, where $0 < \tau \leq 1$ is a minimum support threshold. The frequent subgraph mining problem is thus to find all frequent subgraphs in N_G.*

4 Weighted Frequent Subgraph Mining

Most research work in frequent subgraph mining [8,9,4], assumes each discovered frequent subgraph is equally important. Because of this, a lot of redundant and repetitive frequent patterns exist in the resultant set. In addition, if the size of the graphs is substantial and the minimum support is low, a typical frequent subgraph mining task will not terminate within a reasonable period of time due to the exponential computation incurred by subgraph isomorphism testing. If we put emphasis on differentiating each discovered frequent subgraph by their importance as defined by the user, or as derived from the application domain, a reduced computational complexity can be achieved without compromising the effectiveness of the discovery process.

Therefore, the graphs in N_G are assumed to have weights associated with their nodes or links. Let $W(g)$ be a weighting function that assigns a weight to each discovered subgraph g. The *weighted support* of g with respect to N_G, is then defined as $wsup(g) = W(g) \times |\delta(g)|$.

When a weighting function is integrated into the process of mining weighted frequent subgraphs, the well-known *anti-monotone property*[5], which is often used to prune the search space of the patterns, is not satisfied anymore. There are two general solutions to this dilemma: (i) design a weighting function that keeps the property; (ii) utilize some heuristics to reduce the computation incurred by not maintaining the property.

In the context of weighted frequent subgraph mining, the weighting function associated with a subgraph pattern g can be defined in various manners. Three example approaches are proposed in this paper: (i) Average Mutual Information Based Weighting (AMW), (ii) Affinity Weighting (AW), and (iii) Utility Based Weighting (UBW). The first two approaches satisfy the anti-monotone property while the last one adopts an alternative pruning heuristic. The last two approaches employ two parameters to control the mining result while the first one uses one parameter only. Each approach is discussed in further detail in the following three sub-sections.

[5] If a graph is infrequent, then all its supergraphs are infrequent.

4.1 Average Mutual Information Based Weighting (AMW)

In the AMW approach, the weight for a subgraph g is calculated by dividing the sum of the average weights in graphs that contain g with the sum of the average weights across the entire data set T. Thus:

Definition 4. *Given a node weighted graph g with node weights $\{w_1,w_2,\cdots,w_k\}$, the average weight associated with g is defined as $W_{avg}(g) = \frac{\sum_{i=1}^{k} w_i}{k}$.*

Where w_i can be user defined or calculated by some weighting methods. In this paper, a weighting method to generate the node weight, is introduced as follows:

Definition 5. *Given a subgraph $g = \{e_1, e_2, \cdots, e_k\}$, for each link e_i with two connecting nodes v_a and v_b, their corresponding support values are $sup(e_i)$, $sup(v_a)$, and $sup(v_b)$. The mutual information between two nodes, $PMI(e_i)$, is then defined as $PMI(e_i) = \log\left(\frac{sup(e_i)}{sup(v_a) \times sup(v_b)}\right)$. $PMI(e_i) = 0$, when $sup(v_a) = 0$ or $sup(v_b) = 0$.*

Definition 6. *Given a subgraph g, with $V(g) = \{v_1, v_2, \cdots, v_m\}$ and $E(g) = \{e_1, e_2, \cdots, e_k\}$, the weight for node v_i, w_i is defined as:*

$$w_i = \frac{\sum_{i=1}^{k} PMI(e_i)}{deg(v_i) - 1} \quad . \tag{1}$$

Where $deg(v_i)$ denotes the number of links incident to v_i, if $deg(v_i) = 1$ or 0, $w_i = 0$.

Definition 7. *Given a set of graphs $N_G = \{G_1, G_2, \cdots, G_T\}$, the total weight of this set of graphs is defined as $W_{sum}(N_G) = \sum_{i=1}^{T} W_{avg}(G_i)$.*

Definition 8. *Given an arbitrary subgraph g with its support set $\delta(g)$, the weighting function of g with respect to N_G, $W_{N_G}(g)$, is defined as:*

$$W(g) = \frac{\sum_{G_i \in \delta(g)} W_{avg}(G_i)}{W_{sum}(N_G)} \quad . \tag{2}$$

Definition 9. *A subgraph g is weighted frequent with respect to G_N, if $|\delta(g)| \times W(g) \geq \tau \times T$, where $0 < \tau \leq 1$ is a minimum support threshold.*

From the above it can be easily inferred that the function $W(g)$, as defined by (2), satisfies the anti-monotone property. Therefore, if a k-subgraph candidate is not frequent, then any of its $(k+1)$-supergraph candidates can be safely pruned from this branch in the search space lattice during the $k+1$ candidate generation process. It should be noted, however, that the approach will tend to bias large transaction graphs over smaller transaction graphs, thus is best applied to graph sets where the individual graphs are of a similar size.

4.2 Affinity Weighting (AW)

The AW approach is founded on two elements to restrict the growth of the search space: (i) a graph distance measure, and (ii) a weighting ratio. For a subgraph g to be frequent both elements must be greater than specified user thresholds. The graph distance measure is calculated using an appropriately defined support weighting function, $W(g)$. This is defined as follows. Let g be a candidate pattern for a database $N_G = \{G_1, G_2, \cdots, G_T\}$. In the context of AW we define:

$$W(g) = \frac{1}{|V(g)|} \sum_{G_i \in \delta(g)} \frac{|V(G_i)| - |V(g)|}{|V(G_i)|} \quad . \tag{3}$$

Where $V(G_i)$ is the set of nodes in transaction graph G_i and $V(g)$ is the set of nodes in the subgraph g. Observe that $W_T(g)$ satisfies:

$$W(g) = \frac{|\delta(g)|}{|V(g)|} - \sum_{G_i \in \delta(g)} \frac{1}{|V(G_i)|} \quad . \tag{4}$$

It should be noted that adding nodes to g can only reduce the value of the above expression because the support($|\delta(g)|$) cannot be increased; the sum contains as many terms as $|\delta(g)|$ and each of these cannot be larger than $1/|V(g)|$. Thus $W(g)$ as defined above, insures that the weighted support of g is non-increasing (i.e. anti-monotone) in $|V(g)|$.

The graph distance measure is directed at the number of nodes contained in a graph, the weighting ratio concerned with the link weights. The weighting ratio of an link-weighted graph g is a function $c(g)$ returning a value between zero and one which is decreasing in the number of links of g. Given an link weighted subgraph g with link weights $W = \{w_1, w_2, \cdots, w_k\}$ the weighting ratio function, $c(g)$, is defined as follows:

$$c(g) = \frac{MIN_{w_i \in W}\{w_i\}}{MAX_{w_j \in W}\{w_j\}} \quad . \tag{5}$$

Definition 10. *An link-weighted graph g is a weighted frequent (i.e. weighted affinity) pattern within a data set $N_G = \{G_1, G_2, \cdots, G_T\}$, with respect to a support threshold $\tau > 0$ and weighting ratio threshold $\gamma \in [0, 1]$, if the following two conditions (C1 and C2) are satisfied:*

$$\textbf{(C1)}\ wsup(g) \geq \tau \times T, \qquad \textit{and} \qquad \textbf{(C2)}\ c(g) \geq \gamma \quad .$$

Definition 10 leads to an alternative pruning strategy which, may be used as part of any frequent subgraph mining algorithms. During the candidate selection phase, the mining will keep track of the weighted support and weighting ratio of all candidates and discard all those candidates that do not satisfy at least one of **(C1)** and **(C2)**.

4.3 Utility Based Weighting (UBW)

The previous two approaches both satisfy the anti-monotone property. In this section an alternative weighting scheme which does not feature the property is proposed, instead an alternative mechanism for limiting growth is adopted. The UBW scheme is influenced by ideas suggested in [11]. As in the case of the AW scheme, the UBW scheme is founded on two elements: (i) weighted support and (ii) the share (SH) of a subgraph. Thus:

Definition 11. *Given a subgraph g with links $E(g) = \{e_1, e_2, \cdots, e_k\}$. For each $e_i \in E(g)$, two nodes connecting e_i are v_1 and v_2. Their associated support sets are given as $\delta(v_1)$ and $\delta(v_2)$. The Jaccard similarity coefficient between the two nodes is defined as $jC(e_i) = |\delta(v_1) \cap \delta(v_2)| / |\delta(v_1) \cup \delta(v_2)|$. The weighting function of g, $W(g)$, is then defined as*

$$W(g) = \frac{1}{\sum_{e_i \in E(g)} jC(e_i)} \ . \tag{6}$$

From the above it is clear that $W(g)$ satisfies the anti-monotone property.

Definition 12. *Given an link weighted graph set $N_G = (G_1, \ldots, G_T)$ with link weights $\{w_1, w_2, \cdots, w_k\}$ for each transaction graph G_t and a subgraph g. Let $g \subseteq G_t$, the weight of g denoted as $W(g, G_t)$, is the sum of the weights of the links which occurred in G_t. That is, $W(g, G_t) = \sum_{e_i \in g, g \subseteq G_t} w_i$. The total weight of N_G, denoted as $TW(N_G)$, represents the sum of link weights in N_G, where $TW(N_G) = \sum_{G_t \in N_G} \sum_{e_i \in G_t} w_i$. The total weight of $\delta(g)$, is defined as $TW(\delta(g)) = \sum_{G_t \in \delta(g)} \sum_{e_i \in G_t} w_i$.*

Definition 13. *The graph weight of g with respect to N_G, denoted as $GW(g)$, is the sum of the weight of the g in each transaction graph $G_t \in \delta(g)$. That is, $GW(g) = \sum_{G_t \in \delta(g)} W(g, G_t)$.*

Definition 14. *The share of a subgraph g, denoted as $SH(g)$, is the ratio of the graph weight of g with respect to N_G to the total weight of N_G. Thus:*

$$SH(g) = \frac{GW(g)}{TW(N_G)} \ . \tag{7}$$

Given a share threshold λ, a subgraph g is SH-frequent if $SH(g) \geq \lambda$; otherwise, g is SH-infrequent.

Theorem 1. *Given a $N_G = (G_1, \ldots, G_T)$, a subgraph g, and a threshold λ, if $TW(\delta(g)) < \lambda \times TW(N_G)$, all supergraphs of g are SH-infrequent.*

Proof. Let h be an arbitrary supergraph of g. Clearly, $GW(h) \leq TW(\delta(h)) \leq TW(\delta(g))$. If $TW(\delta(g)) < \lambda \times TW(N_G)$ holds, $GW(h) < \lambda \times TW(N_G)$. That is, $SH(h) = GW(h)/TW(N_G) < \lambda$. Therefore, h is SH-infrequent. □

By Theorem 1, if $TW(\delta(g)) < \lambda \times TW(N_G)$, all supergraphs of g and g are SH-infrequent and can be pruned; otherwise, g is a candidate subgraph.

Definition 15. *An link-weighted graph g is a weighted frequent pattern for a graph set $N_G = (G_1, \ldots, G_T)$ with respect to a support threshold $\tau > 0$ and share threshold $\lambda \in (0, 1]$ if the following two conditions are satisfied.*

$$\textbf{(D1)}\ wsup(g) \geq \tau \times T, \qquad \textit{and} \qquad \textbf{(D2)}\ SH(g) \geq \lambda \quad .$$

5 Evaluation

To evaluate the proposed weighting schemes experiments were conducted using a projection of the Cattle Tracking System (CTS) database in operation in Great Britain (GB). The proposed weighting schemes were incorporated into weighted variations and extensions of the well established gSpan frequent subgraph mining algorithm [4]. For comparison purposes we also derived variation of gSpan, *extGspan*; that did not feature weightings, but could handle directed graphs, self-cycles and multiple links. Results from the experiments are presented in the following sub-sections.

Table 1. The statistics of graph data generated by the projection of the CTS database

	Derbyshire	Lancashire	*GB*
# graphs	53	53	53
Max # links	179	394	30107
Average # links	137	297	23055
Max # nodes	227	396	23660
Average # nodes	172	318	18749

5.1 The Cattle Tracking System Database

The Cattle Tracking System (CTS) database in operation in Great Britain (GB), which forms the focus of the research described in this paper, is maintained by the Department for Environment, Food and Rural Affairs (DEFRA) as part of the Rapid Analysis and Detection of Animal-related Risks (RADAR) initiative[6]. The CTS database records all cattle movements in GB, each record describes the movement of a single animal, identified by a unique ID number, between two *holding locations* (e.g. agriculture holdings, markets, etc). Social network can be extracted from this database such that each node represents a geographical location and the links the number of animals moved between locations. By considering the time stamps associated with movements, temporal sequences of networks can be extracted (i.e. longitudinal social networks).

For the experimental analysis three distinct longitudinal social network data sets were extracted from the CTS database using data from 1 January 2005 to 31 December 2005. The first two data sets, Derbyshire[7] and Lancashire[8].

[6] http://www.defra.gov.uk/foodfarm/farmanimal/diseases/vetsurveillance/radar

[7] *Derbyshire* is a county in the East Midlands of England.

[8] *Lancashire* is a county in the North West of England.

The third data set represented GB in its entirety. The data divided into 7-day "episodes" (there is a 6-day movement restriction that applies to agriculture holdings in GB), giving longitudinal sequences of 52 episodes. The links were annotated with a weight, indicating the number of animals that were moved, and a label, indicating the type of movement (e.g. farmToFarm, farmToMarket, etc). Some statistics for each of the data sets is presented in Table 1. Note that the *GB* database is significantly larger than the *Lancashire* data set, which in turn was larger than the *Derbyshire* data set. Note that all the graphs featured "self-cycles" and "multiple links".

5.2 Comparison between Weighted and Non-weighted Approaches

In this sub-section the proposed weighting schemes (AMW-gSpan, AW-gSpan, and UBW-gSpan) are compared with the extended gSpan algorithm in terms of efficiency (runtime and the number of frequent subgraphs generated). For AW-gSpan, $\gamma = 0.6$ was chosen as the weighgting ratio threshold, and $\lambda = 8\%$ was used as the share threshold for UBW-gSpan. The justification for these γ and λ values is given in Sect 5.3 below.

Figure 1 shows the performance of the weighting schemes and extGspan on the *Derbyshire* and *Lancashire* data sets. It can be clearly seen from the figure that all four algorithms display a similar behavior when the support value is between 12% to 20%, however the number of patterns generated by the extGspan algorithm increases abruptly when the support value is decreased to below 12%. In the figure, it can be observed that: (i) significantly more frequent subgraphs are found using UBW algorithm than using any of the others, indicating that UBW algorithm can not show its advantage against the non-weighted extGspan

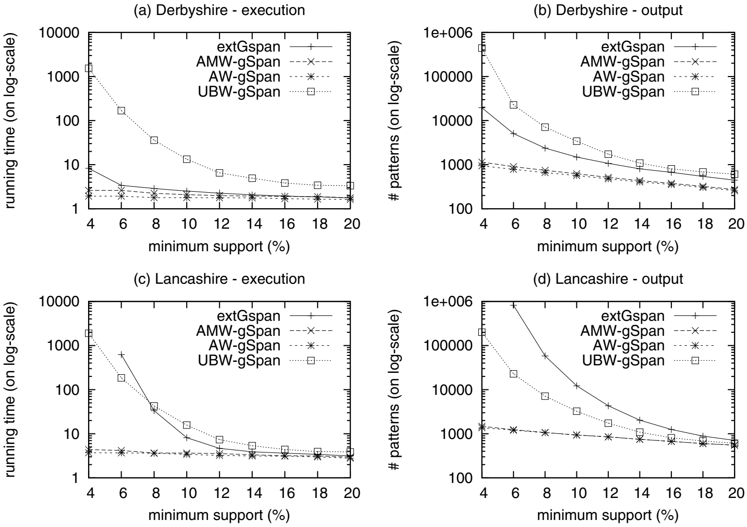


Fig. 1. Performance comparison of weighting schemes vs. extGspan on two graph sets

for a smaller data set, (ii) Two weighting schemes: AW and AMW algorithms achieves better performance than extGspan and UBW algorithms. The reason for the better performance of AMW and AW schemes is that the pruning technique adopted by UBW scheme is not sufficiently effective in reducing the search space when compared to the anti-monotone method used by the AMW and AW schemes.

Experiments (not shown) using extGspan and the *GB* data set failed to produce any results (because of memory errors) unless the support threshold was set to 30% or above, a threshold at which only one node size subgraphs are discovered. Thus it was not possible to conduct any meaningful comparison between the weighted frequent subgraph mining algorithms and a non-weighted approach using the *GB* data set.

5.3 Comparison of Weighting Schemes

In this sub-section the three proposed weighting schemes are compared with one another using the large *GB* data set. As above, γ was initially set to 0.6 and λ to 8% for use with AW-gSpan and UBW-gSpan algorithms. Figure 2 shows the performance of the weighting schemes on the *GB* data set. In Fig. 2 (a), each curve depicts the number of patterns generated against the minimum support value used. From the figure it can be seen that UBW-gSpan produces the least number of patterns while AW-gSpan produces the most. Figure 2 (b) indicates the "run time" for the approaches using the same sequence of support threshold values. From the figure it can be seen that UBW-gSpan is the most "expensive", indicating that the cost of finding a minimum number of patterns is higher compared to the other two mechanisms. AMW-gSpan is the most economical. It is interesting to note in Fig. 2 (b) that as the support threshold is reduced the effect on run-time is much smaller for AMW-gSpan than the other two weighting schemes.

Figure 3 displays the effect on performance of different values for the weighting ratio threshold (γ) used in conjunction with AW-gSpan, and the share threshold (λ) used with UBW-gSpan, for a range of support threshold values from 4% to 12%. From Fig. 3 (a) and (c) it can be seen that the run time increased as the γ value is decreased, while a marginal increase in the number of patterns is

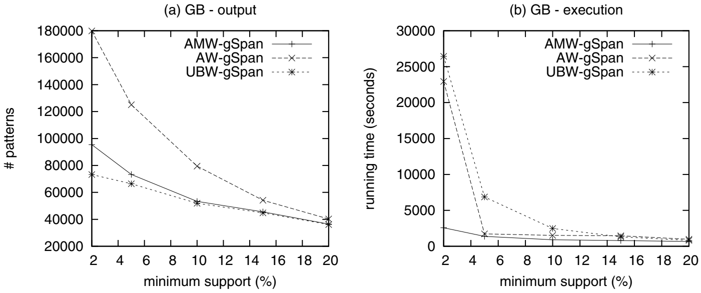


Fig. 2. Performance comparison of using three weighting schemes on the *GB* data set

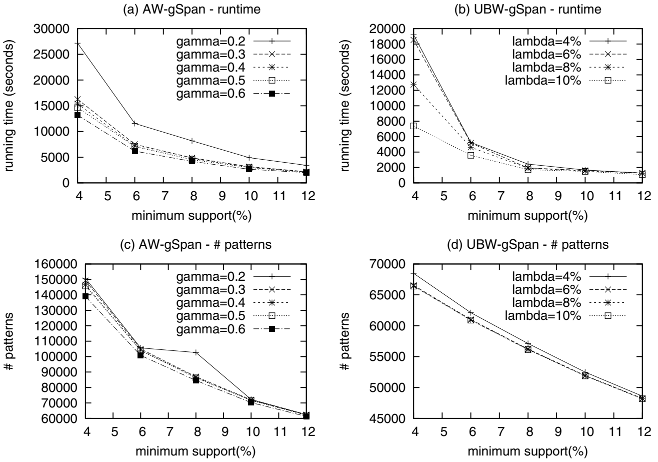


Fig. 3. Analysis of the performance of AW-gSpan and UBW-gSpan mining algorithms

witnessed. With respect to Fig. 3 (b) and (d) it can be seen that the run time increases as the λ value is decreased, while a small corresponding increase in the number of identified patterns is witnessed. However, increasing the λ value beyond 8% seems to have very little effect on the number of patterns. Overall it was found that a γ value of 0.6 and a λ value of 0.8% was the most appropriate.

5.4 Subgraph Pattern Analysis

To demonstrate that the the utility of the subgraphs that have been discovered this sub-section briefly discusses an application of the approach. The frequent subgraphs identified by the above mining algorithms can be further used to construct a sequential database where each item of the sequence is a frequent subgraph. Formally, each sequential transaction is extracted using the following identity (8).

$$S_{t,t+ts} = \{f_i | f_i \in (FS(G_t) \cup FS(G_{t+ts}) - FS(G_t) \cap FS(G_{t+ts}))\} \ . \tag{8}$$

where $S_{t,t+ts}$ represents the sequence of frequent subgraphs in a ts time period, f_i represents a frequent subgraph and $FS(G_t)$ represents all frequent subgraphs of the graph G_t, where G_t represents the graph at a time instance t.

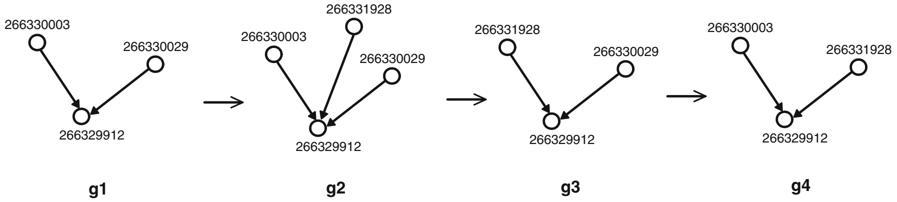


Fig. 4. An example of the sequential patterns extracted using AMW-gSpan algorithm

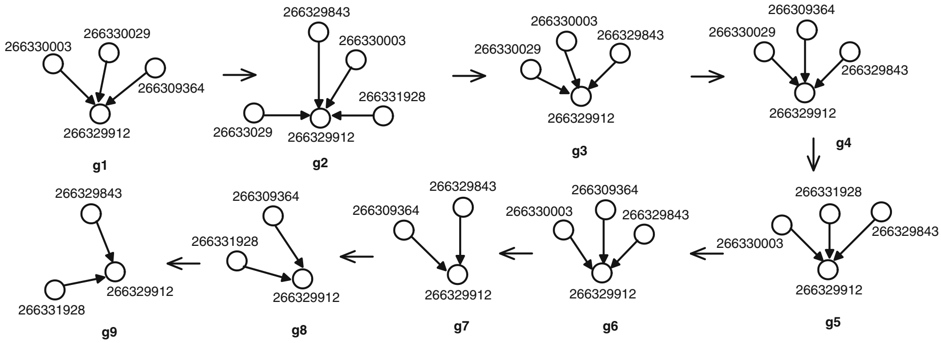


Fig. 5. Another example of a longer sequential pattern extracted using AMW-gSpan

Using the AMW-gSpan algorithm as an example; the output when applied to the *GB* data set, with a support of 10%, was utilized to create a sequential data collection with a time-step value of 1. A sequential mining algorithm, PrefixSpan [12], was then applied to this database with a support threshold of 60%. One example of the maximal-size sequential patterns is displayed in Fig. 4. The patterns in the figure indicate these four subgraphs occurred together on 32 occasions out of 52 in the order showed in the figure. In the figure, each node denotes the location of the agriculture holding, and the number next to the node denotes the unique identification number. It can be seen that $g1$, $g3$, and $g4$ are all subgraphs of $g2$. All the movements are pointed to the location "266329912", and $g1$ always occurs before $g2$, while $g3$ and $g4$ always occur after $g2$. If a smaller support threshold of 30% was used, a longer sequence consisting of 9 patterns were extracted as illustrated in Fig. 5. Figure 5 features additional movements to those given in Fig. 4, and includes new locations. In the figure, the movement was still centered on the location "266329912", however two new locations "266329843", and "266309364" were added into the sequence, and each pattern in the figure contains either one of them or both.

6 Conclusions

A weighted approach to longitudinal social network mining is described. The approach allows large longitudinal networks, such as the GB network used to illustrate this paper, to be mined where this was not possible using more conventional approaches. Three weighting mechanisms were proposed to reduce the overall computational complexity. Reported experiments comparing the operation of the weighting schemes with each other and a non-weighted version of gSpan demonstrated that many fewer patterns are derived. The reported experiments also indicated that UBW-gSpan finds the least number of patterns while requiring the largest amount of run-time. AMW-gSpan provided the best compromise, a limited number of patterns found in reasonable time (especially at low support threshold values). To illustrate that the utility of the subgraphs that were discovered; further analysis was conducted to capturing changes in "behavior" within the network structures.

References

1. Wasserman, S., Faust, K.: Social Network Analysis, Method and Applications. Cambridge University Press, New York (1994)
2. Barabsi, A.L., Jeong, H., Nda, Z., Ravasz, E., Schubert, A., Vicsek, T.: Evolution of the Social Network of Scientific Collaborations. Physica A: Statistical Mechanics and Its Applications 311, 590–614 (2002)
3. Somaraki, V., Broadbent, D., Coenen, F., Harding, S.: Finding Temporal Patterns in Noisy Longitudinal Data: A Study in Diabetic Retinopathy. In: Proceedings of the 10th Industrial Conference on Data Mining, Berlin, pp. 418–431 (2010)
4. Yan, X., Han, J.: gSpan:Graph-based Substructure Pattern Mining. In: Proceedings of 2002 International Conference on Data Mining (2002)
5. Mukherjee, M., Holder, L.B.: Graph-based Data Mining on Social Networks. In: Proceedings of the ACM KDD Workshop on Link Analysis and Group Detection (2004)
6. Yang, W., Dia, J., Cheng, H., Lin, H.: Mining Social Networks for Targeted Advertising. In: Proceedings of the 39th Annual Hawaii International Conference on System Science (2006)
7. Lahiri, M., Berger-Wolf, T.Y.: Structure Prediction in Temporal Networks using Frequent Subgraphs. In: Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining, Hawaii, pp. 35–42 (2007)
8. Inokuchi, A., Washio, T., Motoda, H.: An Apriori-based Algorithm for Mining Frequent Substructures from Graph Data. In: Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (2000)
9. Kuramochi, M., Karypis, G.: Frequent Subgraph Discovery. In: Proceedings of IEEE International Conference on Data Mining (2001)
10. Jiang, C., Coenen, F., Zito, M.: Frequent Subgraph Mining on Edge Weighted Graphs. In: Proceedings of the 12th International Conference on Data Warehousing and Knowledge Discovery (2010)
11. Carter, C.L., Hamilton, H.J., Cercone, N.: Share based Measures for Itemsets. In: Komorowski, J., Żytkow, J.M. (eds.) PKDD 1997. LNCS, vol. 1263, pp. 14–24. Springer, Heidelberg (1997)
12. Pei, J., Han, J., Asl, M.B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Patterns Growth. In: Proceedings of the 17th International Conference on Data Engineering (2001)

Weigted-FP-Tree Based XML Query Pattern Mining

Mi Sug Gu[1], Jeong Hee Hwang[2], and Keun Ho Ryu[1]

[1] Database/BioInformatics Laboratory, Chungbuk National University
{gumisug,khryu}@dblab.chungbuk.ac.kr
[2] Department of Computer Science, Namseoul University
jhhwang@nsu.ac.kr

Abstract. According as XML data have been prevailing in many areas such as internet and public documentation, we need to research data mining algorithm to XML data. And many kinds of techniques have been researched to speed up the query performance about XML data. In this paper, therefore, as the method for speeding up the query performance we analyze the XML query pattern and propose Weighted- FP-growth algorithm extracting the similar XML query pattern fast. The proposed method is applied to XML query subtrees. And we experimented our method compared with the existing algorithm. And we showed the proposed method outperform the other methods and give the fast query result to the repeatedly occurring queries.

Keywords: XML, XML Query, Frequent Pattern, XML Mining.

1 Introduction

Recently, the importance of storing the data such as XML data is increasing and indexing and query processing of XML documents have become the core research areas according to the growth of the data volume. XML data change dynamically while processing the query[1,2,6,7].

Many kinds of techniques have been proposed to perform the efficient queries about XML documents till now. There have been many kinds of research to extract frequent structure and similar structure from the XML documents by not only XML indexing technique but also data mining methods[1,5,7]. And also researches have been conducted to find out the similar structure pattern from XML query and then speed up the query performance about XML documents [2,6,13].

When a lot of users pose queries to XML documents, because usually the query structures have the similar patterns, it is efficient to provide the fast query processing through searching for the rules of the frequent structure patterns[1,3,7,9,10,13]. Therefore, in this paper, we propose the method that provides the efficient results to the users' queries after finding out the frequent structure pattern rules. That is, we propose the data mining method which analyzes XML query patterns, generates the frequent query pattern rules by the users, and extracts the structure patterns of the similar queries in the structures of XML data.

In this paper, to speed up the query performance of XML data, XML query patterns are analyzed and then the rules that have the similar repeated query patterns

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 417–428, 2010.

are found. To find out the method that provides the fast results about the users'queries, using the similar query patterns, we applied the data mining method. To speed up the query processing of the similar XML query patterns and to search for the frequent query patterns, we constructed Weighted-FP-Tree based on FP-growth algorithm[4]. That is, while constructing the tree, it is the method to improve the precision and the efficiency of the data retrieval, giving the weight to the useful subtrees constructed by the frequent patterns found frequently in the users' queries in the XML structure documents.

Because the Apriori technique is using the breadth-first method, it must perform tree joining to all the items in the database. Therefore, because the Apriori method needs to retrieve the database frequently to generate a lot of candidate sets, it requires a lot of time. Compared with Apriori, FP-growth algorithm with depth-first technique doesn't need to scan the database for generating candidate set, reducing time complexity and space complexity. So this method performs more efficiently than the existing data mining technique. Therefore in this paper we applied the algorithm based on FP-growth method to find out the similar XML query patterns[1,3,7,13].

We propose the Weighted-FP-Tree method, the extended form of FP-growth. We gave the weight, 1, adding (+) in front of the id, if the frequencies of the sutbrees generated frequently by FP-growth are bigger than the minimum frequency given by the user. And if the frequencies are smaller than the minimum frequency, we gave the weight, 0, adding (-) in front of the id and then we deleted them from the frequent candidate items. So this method performs better than FP-growth algorithm. Giving the weight to find out the users' frequent query patterns, the precision of the users' queries is improved compared with the existing query pattern mining methods.

The rest of this paper is organized as follows. Chapter 2 will explain the related work, and chapter 3 will describe the processing procedure of the frequent XML query pattern structures. And chapter 4 will represent XML query pattern mining procedure. Experiment result and conclusion will be represented in chapter 5 and 6 respectively.

2 Related Work

XML data which have been used in the internet and the public documentations feature changing dynamically in their structures and values. [1,7] used FP-growth algorithm to find out the sequence patterns in the XML structures and introduced the algorithm to search for the FCSPs(Frequently Changing Subtree Patterns), giving the weight. Compared with the existing methods, it is the improved one in that the candidate sets are reduced in the data mining procedure for the frequent structure.

Because XML data are prevalent in lots of areas, [2] introduced the algorithm that finds out the frequent query pattern to speed up the query performance. They introduced XQPMiner and XQPMinerTID using the existing Apriori and AprioriTID, and proposed the method that finds out the frequent query pattern in XML queries. However, because this method uses the existing Apriori method, it has the limitation of the waste of time for the candidate sets. And also [3] introduced FP-growth algorithm using the FP-tree, the extended the prefix tree. Compared with Apriori algorithm, this algorithm has the advantage that the procedure time is reduced.

[5] explained how they stored XML data versions changing dynamically over time and how they extracted the useful information. They introduced a new algorithm that extracted the information of XML documents according to time changes. [6] described the XML query pattern mining algorithm, SOLARIA. This algorithm features that it does not have candidate sets and scan the tree subsumption which takes too long time and cost. [9] introduced the extended FP-growth method which gives the weight to find WIP(Weighted Interesting Patterns). However, these methods for finding out the useful patterns are applied to the general sequential data, not to XML data.

[10] defines the range of the weight and the limitations of the minimum weight and provides the various weights in the range of the weight of each item. Here, WFIM method is used for generating the frequent items with the weight in the large volume of database. However, this method is applied to static data, not to dynamic data that changes over time. And [11] explains the procedures of data mining that gives the various weights in the specified range to the items after giving the range of the weights for the general data. In the database of sequence, the maximum weight is used for removing the infrequent sequential patterns. Adjusting the range of the weight, it features less generating the useful patterns with weight in the large volume of database.

[12] proposes the method for finding out the useful sequential structures, WIS(Weighted Interesting Sequential) pattern, giving the weight, compared with the existing method. This method finds out the sequential patterns with the similar level of support and weight, using sequential s-confidence and w-confidence as a new estimate. It is the method that keeps the balance with the support and the weight and considers the association with each item among the sequential patterns of general data.

Even though we overviewed a few of data mining algorithms for speeding up the query, the existing query pattern mining techniques usually use Apriori algorithm based on association algorithms and general sequential data. However, in this paper we constructed the extended Weighted-FP-Tree based on FP-growth to speed up the performance of XML data, and the proposed tree extracts fast the useful query patterns frequently generated by the users with the similar structures, giving the weight.

3 XML Query Pattern Tree Matching

In this section, the basic definition for performing XML query pattern mining will be described. We used XML data represented by hierarchical structure as a database, and we regard the user query pattern as a transaction. User query pattern is represented by XML structure which has a root, ancestor and descendent nodes. Here, we will take an example. Suppose that there are given the following query, and then we will explain the procedure for **searching** for the answer about the query.

Q1: “Search for the title of the movie of which the genre is action movie, the playing year is 1994, and the main actor is Jean Reno.”

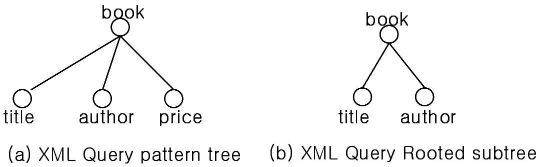
Based on the query we must represent the path of the data such as genre, year, and actor, and the value that we want to find out. That is, the path is represented as //year=”1994”, //genre=”action”, and actor=”Jean Reno”. While this kind of path is used as a transaction, we perform the work that finds out the similar pattern with users' query pattern by using FP-growth algorithm.

Next, we describe the definition for performing XML query pattern mining.

Definition 1. XML Query Pattern Tree: XML Query Pattern Tree is a rooted subtree and is composed of XQPT = <N, E>. Here, N is a vertex set and E is an edge set respectively. Root is represented as Root(XQPT). Each edge E is represented as < Np, Nc>, here Np is a parent of Nc. Each vertex N has a label of {*, //, tagSet}. Here tagSet means the set of all the elements and attributes in DTD.

Definition 2. XML Query Rooted Subtree: Given XML Query Pattern Tree, XQPT = <N, E>, XML Query Rooted Subtree, XQRST = <N', E'> will be the subtree of XQPT, if it has the following conditions. That is, it has the relation such as Root(XQRST) = Root(XQPT), and the subsumption relation such as N'$\subseteq$N, E'$\subseteq$E.

That is, after searching for the frequent useful subtree structures, they are defined as the subtree of XML query. And then based on the structures, the similar structures are found out. The trees are called XML query pattern tree. (a) and (b) in Fig. 1 are shown the examples of XML query pattern tree and the rooted subtree.



(a) XML Query pattern tree (b) XML Query Rooted subtree

Fig. 1. XML Query Pattern Tree

We make the query set into query pattern tree and then store it to construct the database of query pattern tree, XQT_DB = {XQPT$_1$,…,XQPT$_n$}. XML Query Pattern mining means that we find out all the frequent XML rooted subtree with minimum support level from the database, XQT_DB. The number of frequent XML rooted subtree in XQT_DB is represented as FRQ(XQRST). Support is calculated using the equation, SUP(XQRST)=FRQ(XQRST)/| XQT_DB |.

Given constant value σ, if XQRST has the condition, SUP(XQRST)≥σ, XQRST is in the frequent set in the database, XQT_DB. As an example, Fig. 2 shows three kinds of XML query pattern trees. XQRST can be found in XQPT1 and XQPT2. Therefore, the frequency FRQ(XQRST) is 2 and SUP(XQRST) = 2/3.

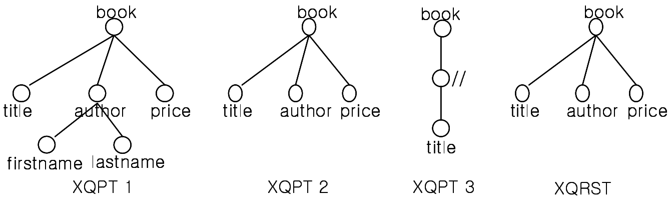


Fig. 2. XML Query Pattern Tree and XML Query Rooted Subtree

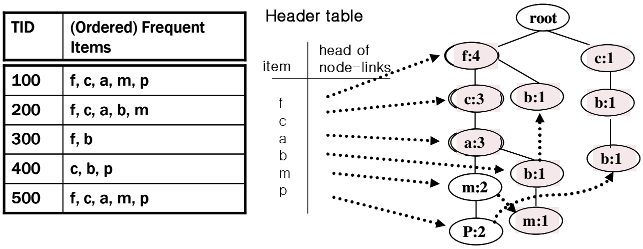
Mining XML query pattern is to search the similar query patterns for speeding up the query performance. We need XML query pattern matching to find out whether the query pattern corresponds with each other similarly. We explain the definition about XML Query Pattern Matching as follows.

Definition 3. XML Query Pattern Tree Matching: Given XML Query Pattern Tree, XQPT, and XML Query Rooted Subtree, XQRST, if it meets the following conditions, XQRST is subsumed by XQPT. If XQRST has the below conditions, XQRST and XQPT have the subsumption relation and XML query tree matching

1. Root node of XQRST and XQPT shares the same label.
2. If nodes have the relation such as w∈XQRST and v∈XQPT, they have the relation such as w.label≤v.label and each subtree of w is subsumed by the subtree of XQPT.

4 Weighted-FP Tree Based Mining

XML query pattern mining is performed as constructing FP-Tree and giving the weight to the useful subtree. To perform the frequent query pattern mining, we need preprocessing to construct FP-tree to perform the data mining for the frequent XML query pattern using FP-growth algorithm as follows[1,7].



TID	(Ordered) Frequent Items
100	f, c, a, m, p
200	f, c, a, b, m
300	f, b
400	c, b, p
500	f, c, a, m, p

Fig. 3. Transaction database and FP-tree

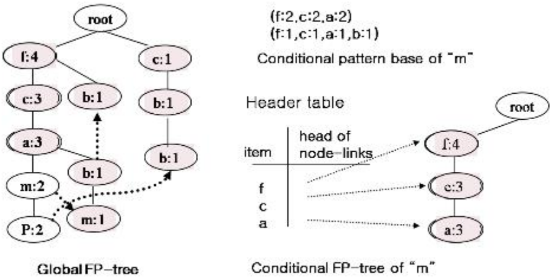


Fig. 4. Conditional FP-TRee

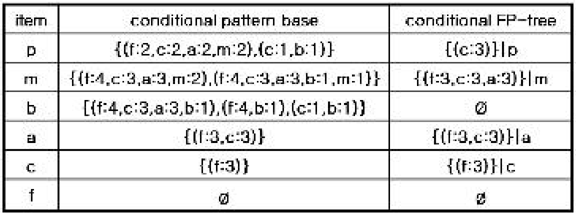
First, we regard XML query subtree as a transaction and then examine the frequency of the items through first scanning the database which is made of transactions. Here the frequency means the support value of an element item set. Second, we sort the items in the transactions by descending order according to the frequency. Third, we prune the items sorted by descending order if the items are below the user defined minimum support. Fourth, after pruning the infrequent items in the transaction database, we construct FP-tree with frequent items. Here the FP-tree is the prefix tree and each path represents the transactions sharing the prefix. And each node means an item.

We perform the above preprocessing to construct the FP-tree. Fig. 3 represents the constructed FP-tree and each transaction sharing the prefix. FP-tree described the database in the compressed form. We can see the node link of each item and pointer stored in the header table. After constructing FP-Tree, the data mining procedure for searching the frequent XML query pattern is performed using FP-Growth algorithm. The items that meet the minimum support are selected, while the items that are less than the minimum support are pruned.

Fig. 4 shows the procedure of constructing the conditional FP-Tree of an item, m, in the procedure of data mining of FP-Growth algorithm. Constructing the conditional tree for each item, the procedure searching for the conditional patterns is performed repeatedly.

And Table 1 shows the results of the conditional patterns, which are the results after performing the procedure repeatedly searching for the conditional patterns.

Table 1. Conditional patterns after performing FP-Growth algorithm



item	conditional pattern base	conditional FP-tree
p	{(f:2,c:2,a:2,m:2),(c:1,b:1)}	{(c:3)}|p
m	{(f:4,c:3,a:3,m:2),(f:4,c:3,a:3,b:1,m:1)}	{(f:3,c:3,a:3)}|m
b	[(f:4,c:3,a:3,b:1),(f:4,b:1),(c:1,b:1)]	Ø
a	{(f:3,c:3)}	{(f:3,c:3)}|a
c	{(f:3)}	{(f:3)}|c
f	Ø	Ø

Next we will explain the query pattern mining using Weighted-FP-Tree, the procedure giving the weight in the procedure of data mining for the useful frequent query pattern structures to the users' queries.

Assume that there is XML Query based on the XQuery Syntax. That is, "Retrieve the title, the author, and the price of the books which were written by the name "Kim"."

```
Q: for $b in document(book.XML)/book
where some $a in $b/author
        satisfies $a/lastname/data()="Kim"
return <result>
        <book>{$b/title,$b/author,$b/price}</book>
        </result>
```

The result about the above query is like this:

{resultPattern={/book/author,/book/title,/book/price}, predicates={/book/author/lastname/data()="Kim"}, documents={book.XML}}

Here, resultPattern is the schema pattern about the query result, the predicates are the filtering conditions, and documents are the related XML file. Fig. 5 shows Book DTD Tree of XML document based on the above XQuery and DTD shows the form of transaction database.

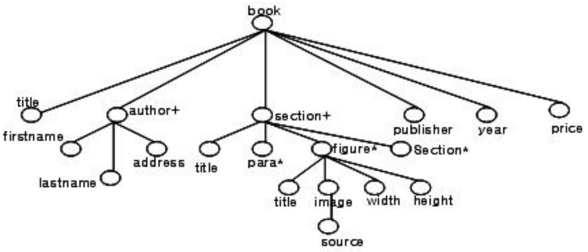


Fig. 5. Book DTD tree

To perform the frequent pattern mining for XML query, we first construct the database which is XML query pattern tree as a transaction. We examine each query pattern tree and construct the transaction database. We first find out the XML rooted subtrees and then mine to search for the similar pattern with them. Based on the transactions which are composed of each XML query pattern, we construct the structure of FP-tree. Based on the constructed FP-tree, we perform the data mining using FP-Growth technique to search for the frequent XML query patterns. We perform the data mining procedure for the frequent pattern tree and find out the maximal frequent subtrees. We define the tree subsumption that we need for the mining as follows.

Definition 4. Tree subsumption: Given two subtree sets S and S1, they have the relation with $S=S1 \cup \{t1,t2,\ldots,tn\}$. Here, if there is the relation with $\forall i(1 \le i \le n)$, $\exists tj \in S1$ and $tj<ti$, S1 is subsumed by S. And the relation is represented as $S1<S$.

Definition 5. Maximal Frequent Subtree: If the subtree set is a frequent subtree and there aren't any subsumption relations of any other frequent subtree, the subtree is the maximal frequent subtree. And it has the relation like $\{\text{maximal FSP}\} \subseteq \{\text{FSP}\}$.

The method that searches similar query pattern about XML data using FP-growth algorithm is more efficient than that by Apriori algorithm. However, giving the weight to the frequent subtree and the infrequent subtree differently, we extended the FP-growth algorithm to prune the infrequent subtrees.

Using the method that gives the weight to the items and calculates the values, we give a certain weight to each item. When using the weight and the support, we can get the results which are different from the existing support. Because the existing pattern mining using just support is useful but not considers each item, the useful patterns are not searched. However, the method that gives the weight to each item and searches the patterns has the advantage that can search all the useful patterns. Therefore, in this paper we carried out data mining to XML data using the method which extends FP-growth algorithm to search for the interesting frequent patterns. Here we define the weight of subtree.

Definition 6. Weight of Subtree: If the frequency of the frequent subtree among the subtrees is greater than the user defined minimal frequency, we give the weight to the frequency as 1. Otherwise, we give the weight to it as -1. So if the frequency of the subtree is less than the user defined minimal frequency, we give (-) label to the infrequent subtree. And then we give the weight as 0 to the subtree sets related to the subtree with (-).

In addition, we will describe the method which gives the weight to perform the efficient query based on Weighted-FP-Tree and FP-Growth technique used for speeding up the performance of the query of XML data in this paper. Table 2 shows the items and the items with preference. And Table 3 shows the procedure for giving the weight to each item. This weight is calculated a new support multiplied with the support. Table 4 shows the order of the items after sorting by a new support multiplied the weight and the support.

Table 2. Data and the frequent items

TID	Set of Items	Frequent Item List
100	a, c, d, f, m, r	c, d, f, m, r
200	a, c, d, f, i, m	c, d, f, m, r
300	b, c, f, m, p	c, f, m, p
400	b, d, f, m, p, r	d, f, m, p, r

Table 3. Providing the support and the weight to each item

Item(min_sup 2)	a, b, c, d, f, i, m, p, r
Support	2, 2, 3, 3, 4, 1, 4, 2, 2
Weight(0.2<wr<0.7)	0.5, 0.3, 0.6, 0.4, 0.7, 0.3, 0.5, 0.2, 0.7
Weight(0.8<wr<1.3)	1.1, 1.0, 0.9, 1.0, 0.7, 0.9, 1.2, 0.8, 1.3

Table 4. The frequent items with the weights

TID	WFI(0.2<wr<0.7)	WFI(0.8<wr<1.3)
100	d, f, m	c, d, f, m, r
200	d, f, m	c, d, f, m
300	f, m	c, f, p, m
400	d, f, m	d, f, m, p, r

We will explain specifically the data mining procedure Weighted-FP-Growth algorithm with Weighted-FP-Tree for efficient query of XML data in this paper.

First, we construct FP-tree using the subtrees which are composed of query patterns about XML data. And then we sort the subtree list in the header table of the constructed FP-tree in reverse order of depth-first traversal. If the maximal subtree pattern subsumes the rooted subtree in the upper level node of XML tree, it must be generated earlier than the maximal frequent subtree pattern which has the rooted subtree in the lower level node.

Fig. 6 shows the ordered subtree in the database using the reverse order of depth-first traversal and bottom-up such as {t5, t4, t3, t2, t1}. We arrange the subtree, and construct the tree that has a root in the top, and shares the prefix using FP-Growth algorithm. And if the subtrees in the database have the same label, we increment the number of each node of the subtree. And then we finally construct the global FP-tree which is made of XML query pattern subtree.

Second, from the global FP-tree constructed by XML query patterns, we generate repeatedly the conditional FP-tree which has each node of subtree as a condition.

DID	SID
1	t1,-t2,-t3,t4,t5
2	t1,t2,-t3,t4,t5
3	t1,-t2,t4,-t5
4	-t1,t2,-t3
5	-t1,-t2,t3

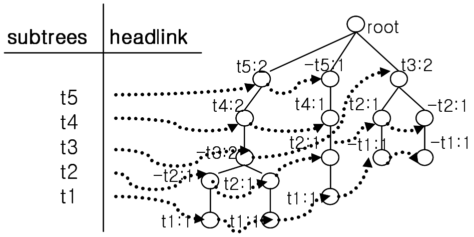


Fig. 6. FP-tree for XML query subtree

Fig. 7 shows the conditional FP-tree about the condition t1 and all the subtree patterns related to t1. In this way, we generate the conditional FP-tree for not only t1 but also all the subtree pattern related t2 and t3 and so on except t1, and then perform the data mining to search for the frequent XML query patterns. We perform constructing the conditional FP-tree about each subtree repeatedly and then find out the maximal frequent XML query pattern tree.

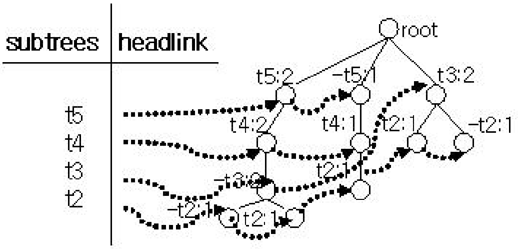


Fig. 7. Conditional FP-tree of the subtree t1

Here we used (-) labeling for the subtree identifier of the infrequent subtrees according to the weight explained previously. We labeled (-) to the infrequent subtrees and then didn't traverse them. We continue this procedure repeatedly and then we can generate the conditional FP-tree. We can confirm this procedure in Fig. 8. Constructing the conditional Weighted-FP-tree for each subtree repeatedly, the procedure to search for the frequent query pattern tree is carried out. And then we can find out the maximal frequent subtree pattern among the similar user query patterns.

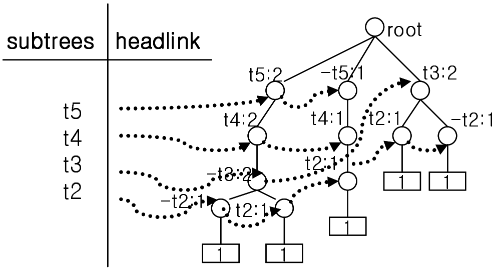


Fig. 8. Conditional FP-tree labeling for subtree t1

5 Experimental Results

We experimented the efficient data mining procedure for the useful XML query pattern, constructing Weighted-FP-Tree. While performing the query for XML data, the frequent pattern structures by users' queries are constructed and then the frequent patterns with the similar structure are found. To evaluate the efficiency and the precision of the proposed method, the query pattern mining was performed using the existing Apriori algorithm, FP-Growth algorithm, and Weighted-FP-Growth algorithm which are used for finding XML query patterns. The experiments were about the relations between the minimum support and the runtime, and the relations between the minimum support and the number of the frequent pattern.

To compare and evaluate three kinds of algorithms, we used XML data from the site, http://dblap.uni-trier.de/xml/[8]. And we performed the preprocessing of XML data for experiments as follows. First, we stored the XML data from the above site. Second, we parsed the XML data using DOM parser. Third, we extracted the tags from the XML query pattern subtrees. Fourth, we extracted the tags and made the mapping table numbering the tags to perform the FP-growth algorithm. Fig. 9 shows the variation of the runtime for the frequent XML query pattern, comparing with the method using Apriori algorithm, FP-growth algorithm and Weighted-FP-Growth algorithm according to support threshold. We can see that the runtime of FP-growth algorithm and Weighted-FP-growth algorithm is decreasing than that of Apriori algorithm according to growing the support threshold. We can see the runtime of FP-growth and Weighted-growth method is getting lower according as the support threshold is getting higher.

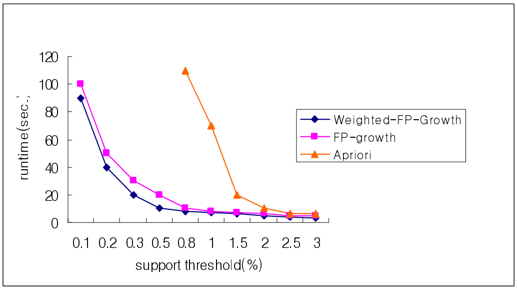


Fig. 9. Apriori, FP-growth, Weighted-FP-growth in runtime by support threshold

Fig. 10 shows the results of the experiment comparing the rate of the number of XML query pattern according to the minimum support. Given the minimum support, we can see the variation of the number of the extracted frequent pattern structures of three kinds of data mining methods. However, as increasing the minimum support, the whole efficiency of the performance is decreasing. Therefore, while keeping the minimum support steadily, the efficient frequent patterns can be confirmed to be acquired.

Through this kind of experiment, compared with the existing data mining method, the proposed method in this paper, Weighted-FP-Tree method saves the time for generating candidate set more than the existing data mining methods. Therefore, the time for retrieving the data can be saved. And as we can see the results through the experiment according to the minimum support, the method that gives the weight to the frequent pattern items is more efficient than the existing method in retrieving the frequent query pattern. However, even though it is more efficient for the small amount of data, it has the disadvantage that it is not efficient for the large amount of data because of the complexity of the data. To compensate for the restrictions we need the additional researches forward.

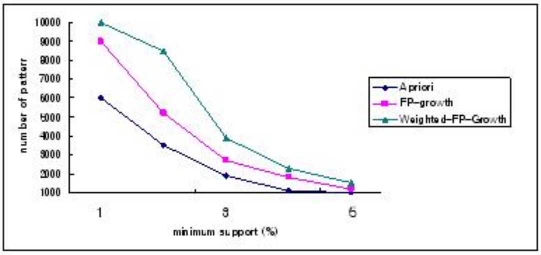


Fig. 10. Changes of the pattern according to minimum support

6 Conclusion

Currently, the representation of data using XML data is increasing in a variety of areas. Accordingly, researches to speed up the performance of the query for XML documents are being tried. In this paper to speed up the query performance for the XML data, we proposed the method that finds out the exact and efficient XML query patterns, constructing the Weighted-FP-growth algorithm considering the weight based on the existing FP-Growth algorithm. And we showed the improved performance of the information retrieval.

Because XML data have the characteristics that they are changing dynamically according to the changes of the time, to improve the performance of the retrieval for the stream data such as web data that change so frequently, the proposed data mining method will be efficient for applying to the data. To retrieve the data that the users want to search for from the frequently changing data in the ubiquitous environment, using data mining for the stream data, we need to extend the research. Therefore, in the future, the researches for improving the query performance and data mining technique for XML stream data need to be studied consistently.

Acknowledgement. This work was supported by the grant of the Korean Ministry of Education, Science and Technology(The Regional Core Research Program / Chungbuk BIT Research-Oriented University Consortium) and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF No. 2010-0001732).

References

1. Chen, L., Bhowmick, S.S., Chia, L.T.: Mining Maximal Frequently Changing Subtree Patternsfrom XML Documents. In: Kambayashi, Y., Mohania, M., Wöß, W. (eds.) DaWaK 2004. LNCS, vol. 3181, pp. 68–76. Springer, Heidelberg (2004)
2. Yang, L.H., Lee, M.L., Hsu, W., Acharya, S.: Mining Frequent Query Patterns from XML Queries. In: Proceedings of the Eighth International Conference on Database Systems for Advanced Applications, pp. 7695–1895 (2003)
3. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns without Candidate Generation. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (2000)
4. Borgelt, C.: An Implementation of the FP-growth Algorithm. In: OSDM 2005, Chicago, Illinoise, USA, August 21 (2005)
5. Rusu, L.H., Rahayu, W., Taniar, D.: Mining Changes from Versions of Dynamic XML Documents. In: Nayak, R., Zaki, M.J. (eds.) KDXD 2006. LNCS, vol. 3915, pp. 3–12. Springer, Heidelberg (2006)
6. Feng, J., Qian, Q., Wang, J., Zhou, L.: Exploit Sequencing to Accelerate Hot XML Query Pattern Mining. In: SAC 2006, April 23–27 (2006)
7. Chen, L., Bhowmick, S.S., Chia, L.T.: FRACTURE-Mining: Mining Frequently and Concurrently Mutating Structures from Historical XML Documents. Elsevier Science Journal: Data & Knowledge Engineering 59, 320–347 (2006)
8. `http://dblp.uni-trier.de/xml`
9. Yun, U., Leggett, J.J.: WIP:mining Weighted Interesting Patterns with a strong weight and/or support affinity. In: Proceedings of the Sixth SIAM International Conference on Data Mining, Bethesda, MD, USA, pp. 20–22. SIAM, Philadelphia (2006)
10. Yun, U., Leggett, J.J.: WFIM: Weighted Frequent Itemset Mining with a weight range and a minimum weight. In: Jonker, W., Petković, M. (eds.) SDM 2005. LNCS, vol. 3674. Springer, Heidelberg (2005)
11. Yun, U., Leggett, J.J.: WSpan: Weighted Sequential Pattern Mining in Large Sequence Databases. In: Proc. of the Third Int'l Conf. on IEEE Intelligent Systems, pp. 512–517 (September 2006)
12. Yun, U.: WIS: Weighted interesting sequential pattern mining with a similar level of support and/or weight. ETRI Journal 2007 29, 336–352 (2007)
13. Hwang, J.H., Ryu, K.H.: A weighted common structure based clustering technique for XML documents. Journal of Systems and Software 2010 83, 1267–1274 (2010)

Privacy-Preserving Data Mining in Presence of Covert Adversaries

Atsuko Miyaji and Mohammad Shahriar Rahman

School of Information Science, Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa, Japan 923-1292
{miyaji,mohammad}@jaist.ac.jp

Abstract. Disclosure of the original data sets is not acceptable due to privacy concerns in many distributed data mining settings. To address such concerns, privacy-preserving data mining has been an active research area in recent years. All the recent works on privacy-preserving data mining have considered either semi-honest or malicious adversarial models, whereby an adversary is assumed to follow or arbitrarily deviate from the protocol, respectively. While semi-honest model provides weak security requiring small amount of computation and malicious model provides strong security requiring expensive computations like Non-Interactive Zero Knowledge proofs, we envisage the need for 'covert' adversarial model that performs in between the semi-honest and malicious models, both in terms of security guarantee and computational cost. In this paper, for the first time in data-mining area, we build efficient and secure dot product and set-intersection protocols in covert adversarial model. We use homomorphic property of Paillier encryption scheme and two-party computation of Aumann et al. to construct our protocols. Furthermore, our protocols are secure in Universal Composability framework.

Keywords: Privacy-preserving Data Mining, Covert Adversary, Efficiency, Multi Party Computation.

1 Introduction

1.1 Background

Recent advances in information technology has empowered many organizations to collect large volumes of data through data mining. A key utility of large databases today is scientific or economic research. However, this data is not useful if worthwhile information cannot be extracted from it. Privacy is a key issue that arises in any huge collection of data. The need for privacy is sometimes due to personal interests, law (e.g., for medical databases), or business interests. However, despite the potential gain, this is often not possible to utilize large databases for scientific or economic research due to the concerns over privacy infringement while performing the data mining operations. To address this problem, several privacy-preserving distributed data mining protocols using

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 429–440, 2010.

cryptographic techniques have been suggested. Depending on the adversarial behavior assumptions, those protocols can be categorized into two main classes of adversaries:

Malicious adversaries: These adversaries may behave arbitrarily and are not assumed to follow a specified protocol. Protocols that are secure in the malicious model provide a very strong security guarantee as honest parties are 'protected' irrespective of an adversarial behavior of corrupted parties.

Semi-honest adversaries: These adversaries correctly follow the protocol specification, yet may attempt to learn additional information by analyzing the transcript of messages received during the execution.

The assumption of semi-honest behavior may be unrealistic in some settings. In such cases, participating parties may prefer to use a protocol that is secure against malicious behavior. It is clear that the protocols secure in the malicious model offer more security. Regarding malicious adversaries, it has been shown that, under suitable cryptographic assumptions, any multiparty probabilistic polynomial time functionality can be securely computed for any number of malicious corrupted parties. However, these are not efficient enough to be used in practice. Most of these constructions use general zero-knowledge proofs for fully malicious multi party computation protocols. The zero-knowledge proofs have inefficient constructions. These constructions make a non-black-box use of the underlying cryptographic primitives through the use of trapdoor permutations[8]. Assuming a trapdoor permutation takes one second to compute, its circuit implementation contains trillions of gates, thereby requiring the protocol trillions of second to run. Some middle type of adversary model that accurately models adversarial behavior in the real world efficiently are thus to be introduced in data mining.

In this work, we introduce a new adversary model that lies between the semi-honest and malicious models. To the best of our knowledge, this is the first attempt to introduce such a model for data mining applications. In many real-world settings, parties are willing to actively cheat (not semi-honest), but only if they are not caught (not arbitrarily malicious). This is natural in many business, financial, and diplomatic settings, where honest behavior cannot be assumed, but where the companies, institutions and individuals involved cannot afford the embarrassment, loss of reputation associated with being caught cheating. This type of covert adversarial behavior explicitly models the probability of catching adversarial behavior. In particular, it is not assumed that adversaries are only willing to risk being caught with negligible probability, but rather allow for much higher probabilities.

Applications of Dot Product and Set-Intersection: K-means clustering is a simple and very commonly used clustering algorithm in data mining. It starts with an unclustered dataset with n elements and one attributes and outputs cluster assignments of each data element in the set. It requires prior knowledge of the number of clusters k [10,20,3]. K-means clustering uses dot product and equality protocols as building blocks. Some recent studies [22,23] provide privacy-preserving association rule mining algorithms using vertically partitioned data. These algorithms involve secure dot product computation with inputs of length n, where n can be arbitrarily large. As for the scure set-intersection, to determine which customers appear on a 'do-not-receive-advertisements' list, a store must perform a set-intersection operation between its private customer list and the producer's list.

1.2 Related Work

Cryptographic techniques have been used to design many different distributed privacy-preserving data mining algorithms. In general, there are two types of assumptions on data distribution: vertical and horizontal partitioning. In the case of horizontally partitioned data, different sites collect the same set of information about different entities. For example, different credit card companies may collect credit card transactions of different individuals. Secure distributed protocols have been developed for horizontally partitioned data for mining decision trees [17], k-means clustering [15], k-nn classifiers [12]. In the case of vertically partitioned data, it is assumed that different sites collect information about the same set of entities but they collect different feature sets. For example, both a university and a hospital may collect information about a student. Again, secure protocols for the vertically partitioned case have been developed for mining association rules [22], and k-means clusters [10,20]. All of those previous protocols claimed to be secure only in the semi-honest model (we do not consider the proposals which have not used standard cryptographic notions). In [13], authors present two-party secure protocols in the malicious model for data mining. They follow the generic malicious model definitions from the cryptographic literature, and also focus on the security issues in the malicious model, and provide the malicious versions of the subprotocols commonly used in previous privacy-preserving data mining algorithms. They use threshold homomorphic encryption for malicious adversaries, presented by Cramer et.al. [4]. [13] shows that there is a positive linear relationship between the overall complexity of the subprotocols and the input size. There has been some other works related to secure two-party computation [2,18]. In [2], the protocol has been shown secure assuming that at least one-party behaves in semi-honest model. However, the protocol requires both parties to engage in a 'proof of decryption' ability (where a sender sends a set of ciphertexts to the receiver and checks whether the receiver can decrypt all the ciphertexts or not), which increases the communication overhead. On the other hand, [18] proposed a two-party protocol to securely evaluate a 2DNF formula using homomorphic encryption from vector decomposition. But this protocol has been shown secure only in the semi-honest adversarial model. Recently, [9] proposed efficient set operations against the malicious adversaries. It is based on oblivious pseudorandom function evaluation in the standard model. They assume no trusted set up or trusted third party for the multiparty computation, thus increasing the communication overhead.

1.3 Our Contribution

Considering the problems mentioned above, for the first time in data mining area, we provide efficient dot product and set-intersection protocols that are secure in presence of the covert adversaries. While semi-honest model provides weak security requiring small amount of computation and malicious model provides strong security requiring expensive computations like Non-Interactive Zero Knowledge proofs, we envisage the need for 'covert' adversarial model that performs in between the semi-honest and malicious models, both in terms of security guarantee and computational cost. We use homoprphic property of Paillier encryption scheme and two-party computation of Aumann et al. to construct our protocols. Furthermore, our protocols are secure in Universal Composability framework.

2 Cryptographic Primitives

2.1 Security Model: Universal Composability (UC)

Security in the UC framework implies that any adversary in the real-life model can be emulated by an adversary in the ideal model. The advantage of this paradigm is that it is possible to show that anything learned by the real-life adversary during the protocol execution is computationally indistinguishable from what is learned by an ideal model adversary. Since in the ideal model, any adversary can learn at most the final result and what is implied by the final result, proving that the real-life model adversary could be simulated by an ideal model adversary implies that real-life adversary could not learn anything more than the ideal model adversary. We briefly visit the universal composability model of [4], a detailed description can be found there.

Ideal Model: Let the set of parties be P_1, P_2 and $I \in \{0, 1\}$ denote the indices of the corrupted parties, controlled by an adversary A. An ideal execution proceeds as follows:

Each party obtains an input; the ith partys input is denoted x_i. The adversary A receives an auxiliary input denoted z. Any honest party P_j sends its received input x_j to the trusted party. The corrupted parties controlled by A may either abort, send their received input, or send some other input of the same length to the trusted party. This decision is made by A and may depend on the values x_i for $i \in I$ and its auxiliary input z. Denote the vector of inputs sent to the trusted party by w. If the trusted party does not receive 2 valid inputs, it replies to all parties with a special symbol and the ideal execution terminates. Otherwise, The trusted party computes $(f_1(w), f_2(w))$ and sends $f_i(w)$ to party P_i, for all $i \in I$ (i.e., to all corrupted parties). A sends either continue or halt to the trusted party. If it sends continue, the trusted party sends $f_j(w)$ to party P_j , for all $j \notin I$ (i.e., to all honest parties). Otherwise, the trusted party sends to all parties. An honest party always outputs the message it obtained from the trusted party. The ideal execution of f on inputs x, auxiliary input z to A and security parameter

n, denoted $\text{IDEAL}_{f,A(z),I}(x,n)$, is defined as the output vector of the honest parties and the adversary A from the above ideal execution.

Real-life Model: The adversary A sends all messages in place of the corrupted parties, and may follow an arbitrary polynomial-time strategy. In contrast, the honest parties follow the instructions of π.

Let f be as above and let π be an two-party protocol for computing f. Furthermore, let A be a non-uniform PPT machine and let I be the set of corrupted parties. Then, the real execution of π on inputs x, auxiliary input z to A and security parameter n, denoted REAL$= \pi, A(z), I(x,n)$, is defined as the output vector of the honest parties and the adversary A from the real execution of pi.

Definition 1. *Let f and π be as above. Protocol π is said to securely compute f with abort in the presence of malicious adversaries if for every non-uniform PPT adversary A for the real model, there exists a non-uniform PPT adversary S for the ideal model, such that for every $I \subseteq [m]$, every balanced vector $x \in (\{0,1\}^*)^2$, and every auxiliary input $z \in \{0,1\}^*$: $\{\mathsf{IDEAL}_{f,S(z),I}(x,n)\}_{n\in\mathbb{N}} \overset{c}{\approx} \{\mathsf{REAL}_{\pi,A(z),I}(x,n)\}_{n\in\mathbb{N}}$, where $\overset{c}{\approx}$ indicates computational indistinguishability.*

2.2 Homomorphic Encryption

Let $E_{pk}(.)$ denote the encryption function with public key pk and $D_{sk}(.)$ denote the decryption function with private key sk. A public key cryptosystem is called additive homomorphic if it satisfies the following requirements:
(1) given the encryption of plaintexts m_1 and m_2, $E_{pk}(m_1)$ and $E_{pk}(m_2)$, there exists an efficient algorithm to compute the public key encryption of $m_1 + m_2$, such that $E_{pk}(m_1 + m_2) := E_{pk}(m_1) +_h E_{pk}(m_2)$.
(2) given a constant k and the encryption of m_1, $E_{pk}(m_1)$, there exists an efficient algorithm to compute the public key encryption of $k{\cdot}m_1$, such that $E_{pk}(k{\cdot}m_1) := k \times_h E_{pk}(m_1)$.

We briefly state the Paillier cryptosystem [19] based on composite residuosity assumption here. A more detailed description can be found in [19].

- **Key generation:** Let p and q be prime numbers where $p < q$ and $p \nmid q-1$. Set the public key pk to N where $N = p \cdot q$, and private key sk to (λ, N), where $\lambda = LCM(p-1, q-1)$.
- **Encryption with the public key:** Given n, plaintext m, and a random number $r \in [1, \ldots, N-1]$, encryption of the message m: $E_{pk}(m) = (1 + N)^m \cdot r^N \pmod{N^2}$. Given any encrypted message, a different encryption can be computed by multiplying it with some random r^N.
- **Decryption with the private key:** Given N, the ciphertext $c = E_{pk}(m)$, decrypt as follows: $m = \frac{(c^\lambda \pmod{N^2})-1}{N}\lambda^{-1}$, where λ^{-1} is the inverse of λ in modulo N.
- **Adding two ciphertexts:** Given the encryptions $E_{pk}(m_1)$ and $E_{pk}(m_2)$ where $\forall m_1, m_2 \in \mathbb{Z}_N$, $E_{pk}(m_1 + m_2)$ can be computed as follows: $E_{pk}(m_1) \cdot E_{pk}(m_2) \bmod N^2 = ((1{+}N)^{m_1} r_1^N){\cdot}((1{+}N)^{m_2} r_2^N) \bmod N^2 = ((1{+}N)^{m_1+m_2} \cdot (r_1 r_2)^N) \bmod N^2 = E_{pk}(m_1 + m_2) \bmod N$.

- **Multiplying a ciphertext with a constant:** Given a constant $k \in \mathbb{N}$ and the encrypted value $E_{pk}(m_1)$, $k \times E_pk(m_1)$ can be computed as: $k \times E_{pk}(m_1) = E_{pk}(m_1)^k \bmod N^2 = ((1{+}N)^{m_1} \cdot r^N)^k \bmod N^2 = (1{+}N)^{k \cdot m_1} \cdot r_1^{kN} \bmod N^2 = E_{pk}(k \cdot m_1)$.

Definition 2. *Assume the existence of semantically secure Paillier encryption scheme with errorless decryption. Then, for any $k = poly(n)$ there exists a secure protocol for computing the parallel string oblivious transfer functionality $((x_1^0, x_1^1), \ldots, (x_n^0, x_1^n), (\sigma_1, \ldots, \sigma_n)) \mapsto (\lambda, (x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}))$ in the presence of covert adversaries with ϵ-deterrent for $\epsilon = 1 - 1/k$.*

2.3 Two-Party Computation

We use secure multi-party protocol in covert adversarial model proposed in [1]. A two-party protocol problem is cast by specifying a random process that maps sets of inputs to sets of outputs (one for each party). We refer to such a process as a functionality and denote it $f : (\{0,1\}^*)^2 \rightarrow (\{0,1\}^*)^2$, where $f = (f_1, f_2)$. The oblivious transfer functionality is denoted by $((x_0, x_1), \sigma \rightarrow (\lambda, x_\sigma)$, where (x_0, x_1) is the first party's input, σ is the second party's input, and λ denotes the empty string (meaning that the first party has no output).

We use the simulators in the security proofs due to their security properties. The notion of security is such that the state of the adversary returned by those simulators is statistically indistinguishable from the state of the adversary in the real-life model. In the case of an attempted cheat, if the trusted party sends $corrupted_i$ to the honest party and the adversary (an event which happens with probability ϵ), then the adversary does not obtain the honest party's inputs. Thus, if cheating is detected, the adversary does not learn anything and the result is essentially the same as a regular abort. In other words, the adversary learns nothing when it is detected. Since it is always detected, this means that full security is achieved. We denote the resultant ideal model by $\mathsf{IDEALSC}^{\epsilon}_{f,S(z),I}(x,n)$ and have the following definition:

Definition 3. *Let f, π be as in Definition 1, and $\epsilon : \mathbb{N} \rightarrow [0,1]$. Protocol π is said to securely compute f in the presence of covert adversaries with ϵ-deterrent if for every non-uniform PPT adversary A for the real model, there exists a non-uniform PPT adversary S for the ideal model such that for every $I \subseteq [2]$, every balanced vector $x \in (\{0,1\}^*)^2$, and every auxiliary input $z \in \{0,1\}^*$:*

$\{\mathsf{IDEALSC}^{\epsilon}_{f,S(z),I}(x,n)\}_{n \in \mathbb{N}} \overset{c}{\approx} \{\mathsf{REAL}_{\pi,A(z),I}(x,n)\}_{n \in \mathbb{N}}$, where $\overset{c}{\approx}$ indicates computational indistinguishability.

A detailed discussion on the relationship among semi-honest model, malicious model, and and ϵ-deterrent covert adversarial model can be found in [1].

3 Our Protocols

3.1 Underlying Idea

In the protocol, P_1 sends P_2 a number of garbled circuits; denote this number by l. Then, P_2 asks P_1 to open all but one of the circuits (chosen at random) in order to check that they are correctly constructed. This opening takes place before P_1 sends the keys corresponding to its input, so nothing is revealed by opening the circuits. If the unopened circuit is correct, then this will constitute a secure execution that can be simulated. With probability $1 - 1/l$ party P_1 will have been caught cheating and so P_2 will output $corrupted_1$ if the unopened circuit is not correct. To do so, it is crucial that the oblivious transfers are run before the garbled circuits are sent by P_1 to P_2. This is due to the fact that the simulator may send a corrupted P_2 a fake garbled circuit that evaluates to the exact output received from the trusted party (and only this output). However, in order for the simulator to receive the output from the trusted party, it must first send it the input used by the corrupted P_2. This is achieved by first running the oblivious transfers, from which the simulator is able to extract the corrupted P_2's input. Moreover, let us consider a corrupted P_1 that behaves exactly like an honest P_1 except that in the oblivious transfers, it inputs an invalid key in the place of the key associated with 0 as the first bit of P_2. The result is that if the first bit of P_2's input is 1, then the protocol succeeds and no problem arises. However, if the first bit of P_2's input is 0, then the protocol will always fail and P_2 will always detect cheating. Thus, P_1's decision to cheat may depend on P_2's private input. In order to solve this problem, a circuit that computes the function $g(x_1, x_2^1, \ldots, x_2^m) = f(x_1; \oplus_{i=1}^m x_2^i)$, instead of a circuit that directly computes f. This makes every bit of P_2's input uniform when considering any proper subset of $x_2^1, \ldots x_2^m$. This helps because as long as P_1 does not provide invalid keys for all m shares of x_2, the probability of failure is independent of P_2's actual input. This method has been used in [1].

3.2 Efficient and Secure Dot Product Protocol

In a secure dot product protocol, it is required to check whether the final result is correct. If both parties are trying to cheat, we do not care whether the privacy of any party is protected or parties get correct results. Assuming that any party may want to actively cheat without being caught, an efficient protocol can be constructed.

Require: Two parties P_1 and P_2 with n bit vector inputs x_{lj} where x_{lj} belongs to P_l, and $1 \leq j \leq n$, $l \in \{1, 2\}$.
Auxiliary input: Both parties have the description of a circuit C for inputs of length n that computes the function f. The input wires associated with x_1 and x_2 are $w_1, \ldots, w_n$ and $w_{n+1}, \ldots, w_{2n}$, respectively.
Ensure: Return $res = \sum_{j=1}^n (x_{1j} \cdot x_{2j}))$ to P_1.

- Parties P_1 and P_2 define a new circuit C' that receives $n+1$ inputs $x_1, x_2^1, \ldots,$ x_2^n each of length n, and computes the function $f(x_1, \oplus_{i=1}^n x_2^i)$. Denote the input wires associated with x_1 by $w_1, \ldots, w_n$, and the input wires associated with x_2^i by $w_{in+1}, \ldots, w_{(i+1)n}$.
- Party P_2 chooses $n-1$ random strings $x_2^1, \ldots, x_2^{n-1} \in_R \{0, 1\}^n$ and defines $x_2^n = (\oplus_{i=1}^{n-1} x_2^i) \oplus x_2$. The value $z_2 = x_2^1, \ldots, x_2^n$ serves as P_2's new input of length n^2 to C'.

- Party P_1 chooses two sets of $2n^2$ random keys by running $G(1^n)$, the key generator for the encryption scheme: $(\hat{k}^0_{n+1}, \ldots, \hat{k}^0_{n^2+n})$,$(\tilde{k}^0_{n+1}, \ldots, \tilde{k}^0_{n^2+n})$, $(\hat{k}^1_{n+1}, \ldots, \hat{k}^1_{n^2+n})$,$(\tilde{k}^1_{n+1}, \ldots, \tilde{k}^1_{n^2+n})$.

- P_1 and P_2 run n^2 executions of an oblivious transfer protocol, as follows. In the ith execution, party P_1 inputs the pair $((\hat{k}^0_{n+1}, \tilde{k}^0_{n+1})$, $(\hat{k}^1_{n+1}, \tilde{k}^1_{n+1}))$ and party P_2 inputs the bit z^i_2. The executions are run using a parallel oblivious transfer functionality, as in Definition 2. If a party receives a $corrupted_i$ or $abort_i$ message as output from the oblivious transfer, it outputs it and halts.

- Party P_1 constructs two garbled circuits $G(C')_0$ and $G(C')_1$ using independent randomness. The keys to the input wires $w_{n+1}, \ldots, w_{n^2+n}$ in the garbled circuits are taken from above. Let $\hat{k}^0_1, \hat{k}^1_1, \ldots, \hat{k}^0_n, \hat{k}^1_n$ be the keys associated with P_1's input in $G(C')_0$ and $G(C')_1$ has analogous keys. Then, for every $i \in \{1, \ldots, n\}$ and $b \in \{0, 1\}$, party P_1 computes $\hat{c}^b_i = Com(\hat{k}^b_i; \hat{r}^b_i)$ and $\tilde{c}^b_i = Com(\tilde{k}^b_i; \tilde{r}^b_i)$, where Com is a perfectly-binding commitment scheme. P_1 sends the garbled circuits to P_2 together with all of the above commitments. The commitments are sent as two vectors of pairs; in the first vector the ith pair is $\{\hat{c}^0_i, \hat{c}^1_i\}$ in a random order, and in the second vector the ith pair is $\{\tilde{c}^0_i, \tilde{c}^1_i\}$ in a random order.

- Party P_2 chooses a random bit $b \in_R \{0, 1\}$ and sends b to P_1. The following steps are a single step: P_1 sends a message to P_2, and P_2 carries out a computation.

- P_1 sends P_2 all of the keys for the inputs wires $w_1, \ldots, w_{n^2+n}$ of the garbled circuit $G(C')_b$, together with the associated mappings and the decommitment values.

- P_2 checks the decommitments to the keys associated with $w_1, \ldots, w_n$, decrypts the entire circuit using the keys and mappings that it received, and checks that it is exactly the circuit C' derived from the auxiliary input circuit C. In addition, it checks that the keys that it received in the oblivious transfers match the correct keys that it received in the opening. If all the checks pass, it proceeds to the next step. If not or it does not get any message, it outputs $corrupted_1$ and halts.

- If $b = 0$, then P_1 sends P_2 the keys and decommitment values $(\tilde{k}^{x^1_1}_1, \tilde{r}^{x^1_1}_1), \ldots,$ $(\tilde{k}^{x^1_n}_1, \tilde{r}^{x^n_1}_1)$ to P_2. Otherwise, P_2 sends the $(\hat{k}^{x^1_1}_1, \hat{r}^{x^1_1}_1), \ldots, (\hat{k}^{x^1_n}_1, \hat{r}^{x^n_1}_1)$.

- P_2 checks that the values received are valid decommitments to the commitments received above. If not, it outputs $abort_1$. If yes, it uses the keys to compute $C'(x_1, z_2) = C'(x_1, x^1_2, \ldots, x^n_2) = C(x_1, x_2)$, such that $C(x_1, x_2) = f = e_{res^2} = (E_{pk}(x_{21}) \times_h x_{21}) +_h \ldots +_h (E_{pk}(x_{1n}) \times_h x_{2n})$.

- P_1 calls decrypt protocol to learn $D_{sk}(e_{res^2}) = res = \sum_{j=1}^{n}(x_{1j} \cdot x_{2j}))$.

The result of data mining algorithm $res = \sum_{j=1}^{n}(x_{1j} \cdot x_{2j})$ is evaluated correctly by at least one party, assuming that at least one party tries to cheat. Both P_1 and P_2 have enough information to calculate res. If P_2 computes the same res value as P_1 expects, then computations must be correct, because none of them calculates incorrect res. Therefore, if we securely make sure that any party fails to cheat successfully with incorrect output value, then the local calculation can be decrypted to reveal res to P_1 correctly. In our protocol, party P_1 sends the encrypted inputs along with necessary keys to P_2, then party P_2 locally computes its respective $e_{res^2} = E_{pk}(res^2)$. For example, P_1 generates a homomorphic key pair and sends the pk to P_2 along with the encrypted vector $E_{pk}(x_{1j}) = E_{pk}(x_{11}), (E_{pk}(x_{12}), \ldots, E_{pk}(x_{1n}))$. Given pk, P_2 calculates $e_{x_{1j} \cdot x_{2j}} = (E_{pk}(x_{11}) \times_h x_{21}) +_h (E_{pk}(x_{12}) \times_h x_{22}) +_h \ldots +_h (E_{pk}(x_{1n}) \times_h x_{2n})$ where x_{1j} is the P_1's string's j-th component, and sends e_{res^2} to P_1. For the decryption, the P_1 and P_2 jointly decrypts to reveal res to P_1, given that the computated values are correct.

A note on secure equality protocol: A secure equality protocol requires to compute whether two data items are equal or not without revealing these items. It is thus straight forward to construct a secure equality protocol in covert model. For this reason and due to lack of space, we do not include the equality protocol here.

3.3 Efficient and Secure Set-Intersection Protocol

The main idea of this protocol construction is that we can represent the sets owned by each party as a bit vector of size D, and use secure multiplication property of the homomorphic encryption to give secure set protocols in the covert model. Let us assume that x_{0j} is set to 1 if P_0 has item j in its private set else it is set to 0 (similarly for x_{1j} for P_1). Clearly for calculating set-intersection, we need to calculate $x_{0j} \wedge x_{1j}$ for each j. Similarly, for set union, we need to calculate $x_{0j} \vee x_{1j}$ for all j. This can be rewritten as $\neg(\neg x_{0j} \wedge \neg x_{1j})$. Therefore, the dot product protocol for set union can be used, too. One important thing here to note that, unlike the dot product protocol, each parties need to perform the whole protocol. In other words, each party needs to send its garbled circuits and the necessary keys to the other party before they compute the joint function. We put the detailes of the protocol in the full version due to lack of space. The same protocol can be used for two-party set union negating the input and output bits.

4 Security Analysis

Theorem 1. *Let l and m be parameters in the protocol that are both upper-bound by poly(n), and set $\epsilon = (1 - 1/l)(1 - 2^{m+1})$. Let f be any PPT function. Assume that the Paillier encryption scheme used to generate the garbled circuits has indistinguishable encryptions, and that the oblivious transfer protocol used is secure in the presence of covert adversaries with ϵ-deterrent according to Definition 2. Then, our protocols securely compute dot product and set-intersection in the presence of covert adversaries with ϵ-deterrent.*

Proof (Sketch): We breifly sketch the idea on the proof due to page limitation. Our analysis of the security of the protocol is in the (OT, ϵ)-hybrid model, where the parties are assumed to have access to a trusted party computing the oblivious transfer functionality following the ideal model of our definition. Thus the simulator will play the trusted party in the oblivious transfer, when simulating for the adversary. We separately consider the different corruption cases (when no parties are corrupted, and when either one of the parties is corrupted). In the case that no parties are corrupted, the security reduces to the semi-honest case.

Party P2 is corrupted: Intuitively, the security in this case relies on the fact that P_2 can only learn a single set of keys in the oblivious transfers and thus can decrypt the garbled circuit to only a single value as required. In other words, A is the PPT adversary who controls P_2. The simulator S fixes A's random-tape to a uniformly distributed tape. S meets the requirements of our Definition 3. S only needs to send $cheat_2$ due to the oblivious transfer. Thus, if a"fully secure" oblivious transfer protocol were to be used, the protocol would meet the standard definition of security for malicious adversaries for the case that P_2 is corrupted.

Party P1 is corrupted: The proof of security in this corruption case is considerably more complex. Intuitively, security relies on the fact that if P_1 does not construct the circuits correctly or does not provide the same keys in the oblivious transfers and circuit openings, then it will be caught with probability at least ϵ. In contrast, if it does construct the circuits correctly and provide the same keys, then its behavior is effectively the same as an honest party and so security is preserved. □

5 Efficiency

The efficiency of our protocol is better compared to the best known results for the malicious adversary model. Our protocol requires only a constant number of rounds, a single oblivious transfer for each input bit, and has communication complexity $O(n|C|)$ where n is the security parameter and $|C|$ is the size of the circuit being computed. Two efficient protocols for general two-party computation in the presence of malicious adversaries has been presented in [11,16] recently. The protocol of [11] achieves universal composability under the decisional composite residuosity and strong RSA assumptions under a common reference string. The protocol of [16] has been constructed under more general assumptions and is secure in the plain model. [11] requires $O(|C|)$ public-key operations and bandwidth of $O(n \cdot |C|)$. [16] requires symmetric operations and bandwidth of the order of $O(sn|C|+s^2k)$ where k is the input length, n is the computational security parameter, and s is a statistical security parameter. Thus, our protocol in covert adversarial model is much more efficient for circuits that are not very small. On the other hand, it is sufficient for the oblivious transfer protocol to be secure in the presence of covert adversaries. Hence, a protocol for general two-party computation with $\epsilon = 1/2$ is only a constant factor slower than the original protocol of Yao that is only secure for semi-honest adversaries.

6 Conclusion

In this paper, we have proposed efficient and secure dot product and set-intersection protocols in covert adversarial model for the first time which are useful for many practical applications. These protocols can be used in various data mining algorithms as building blocks. We provide sophisticated modifications that lead to greater efficiency of the privacy-preserving data mining algorithms in more realistic settings. The effect of our construction for efficient implementation is huge. Our protocols are much efiicient than the protocols in malicious models without requiring expensive zero knowledge proofs, and are slightly expensinve than the semi-honest models. However, the security of our protocol is much stronger than that in semi-honest models. We also provide the security model in UC framework. Applying the covert adversarial model in a more efficient way for specific data mining applications under reasonable assumptions are the open problems.

References

1. Aumann, Y., Lindell, Y.: Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 137–156. Springer, Heidelberg (2007)
2. Boneh, D., Goh, E.G., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
3. Bunn, P., Ostrovsky, R.: Secure Two-Party k-Means Clustering. In: ACM CCS 2007, pp. 486–497 (2007)
4. Cramer, R., Damgard, I., Nielsen, J.B.: Multi-party computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–299. Springer, Heidelberg (2001)
5. Damgard, I., Hofheinz, D., Kiltz, E., Thorbek, R.: Public-Key Encryption with Non-interactive Opening. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 239–255. Springer, Heidelberg (2008)
6. Damgard, I., Thorbek, R.: Non-interactive proofs for integer multiplication. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 412–429. Springer, Heidelberg (2007)
7. Galindo, D., Libert, B., Fischlin, M., Fuchsbauer, G., Lehmann, A., Manulis, M., Schroder, D.: Public-Key Encryption with Non-interactive Opening: New Constructions and Stronger Definitions. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 333–350. Springer, Heidelberg (2010)
8. Goyal, V., Mohassel, P., Smith, A.: Efficient Two Party and Multi Party Computation Against Covert Adversaries. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 289–306. Springer, Heidelberg (2008)
9. Hazay, C., Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 312–331. Springer, Heidelberg (2010)
10. Jagannathan, G., Wright, R.N.: Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 593–599 (2005)

11. Jarecki, S., Shmatikov, V.: Efficient Two-Party Secure Computation on Committed Inputs. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 97–114. Springer, Heidelberg (2007)
12. Kantarcioglu, M., Clifton, C.: Privately computing a distributed k-nn classifier. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 279–290. Springer, Heidelberg (2004)
13. Kantarcioglu, M., Kardes, O.: Privacy-preserving data mining in the malicious model. International Journal of Information and Computer Security 2(4), 353–375 (2008)
14. Lai, J., Deng, R.H., Liu, S., Kou, W.: Efficient CCA-Secure PKE from Identity-Based Techniques. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 132–147. Springer, Heidelberg (2010)
15. Lin, X., Clifton, C., Zhu, M.: Privacy-preserving clustering with distributed EM mixture modeling. Knowledge and Information Systems 8(1), 68–81 (2005)
16. Lindell, Y., Pinkas, B.: An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 52–78. Springer, Heidelberg (2007)
17. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000)
18. Okamoto, T., Takashima, K.: Homomorphic Encryption and Signatures from Vector Decomposition. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 57–74. Springer, Heidelberg (2008)
19. Paillier, P.: Public-key cryptosystems based on composite degree residue classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
20. Su, C., Bao, F., Zhou, J., Takagi, T., Sakurai, K.: Security and Correctness Analysis on Privacy-Preserving k-Means Clustering Schemes. IEICE Trans. Fundamentals E92-A(4), 1246–1250 (2009)
21. Top 10 Largest Databases in the World, http://www.worldsbiggests.com/2010/02/top-10-largest-databases-in-world.html
22. Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 639–644 (2002)
23. Yang, Z., Wright, R.N.: Privacy-preserving computation of Bayesian networks on vertically partitioned data. IEEE Transactions on Knowledge and Data Engineering 18(9), 1253–1264 (2006)
24. Yao, A.: How to Generate and Exchange Secrets. In: FOCS 1986, pp. 162–167 (1986)

Multiple Level Views on the Adherent Cohesive Subgraphs in Massive Temporal Call Graphs

Qi Ye, Bin Wu, and Bai Wang

Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia
Beijing University of Posts and Telecommunications, Beijing, China, 100876
yeqibupt@gmail.com, {wubin,wangbai}@bupt.edu.cn

Abstract. In this paper, we present a multi-level empirical study of locality structures of several temporal call graphs containing both mobile and fixed-line call graphs emphasizing on comparing the patterns of these two types of call graphs. We investigate the topological patterns of the cohesive subgraphs in a mesoscopic scale, and we find several novel patterns in these call graphs. We study the correlation between the link weights and their localities. To our surprise, we can find there is nearly no correlation between link weights and their edge betweenness values. We also find that in the network of communities, communities' sizes and betweenness centralities are highly positively correlated, which indicates large communities tend to be in the center of the call networks. Our analysis also suggests that 'small-world' phenomenon still exists in the community-based networks. We believe that our analysis results will help Telecom operators have better understanding of their customers.

Keywords: Data Mining, Social Network Analysis, Community Detection, Visual Analytics.

1 Introduction

Recently, massive data sets of social networks are accumulating at a tremendous pace in various fields. Telecom companies have call records among their customers over years, and these data can give us comprehensive pictures of different levels of social communication patterns. However, the structure and functions in these temporal call graphs have not been well studied yet. For such reasons, it is important to have a good knowledge of not only the graph structure characteristics on macroscopic level, but also the properties of cohesive subgraphs or communities in multi-level views. First, We explore the basic statistical characteristics of the call graphs in macroscopic sense. Second, we study the properties of the cohesive subgraphs and the structure of the statistically significant communities in it. Through our empirical study, we can find several interesting phenomena in these call graphs which have never been reported before. Our main contributions in the paper can be summarized as follows:

- To best of our knowledge, we propose one of the largest-scale detailed studies of structural properties temporal fixed-line and mobile call graphs. We

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 441–452, 2010.

compare the statical patterns between the fixed-line graphs and the mobile call graphs.

- We study the correlation between the weights of links and their localities. To our surprise, we can find there is nearly no correlation between link weights and their edge betweenness centralities.
- We find that in the network of communities, communities' sizes and betweenness centralities are highly positively correlated, which indicates large communities tend to be in the center of the call networks.

This paper is organized as follows: Section 2 surveys the related work. In Section 3, we show the macroscopic structure characteristics of call graphs. In Section 4, we zoom into the cohesive subgraphs to study the local structures of the call graphs. In Section 5, we explore the cohesive subgraphs based on the community level views and study the structure characteristics of these communities. Section 6 concludes this paper.

2 Related Work

Recently, lots of studies have been proposed to show the macroscopic statistical patterns in real-world networks, such as 'small-world' phenomenon [1], the 'scale-free' property [2], etc. Nanavati et al. [3] [4] study a broad set of structural properties in massive telecom call graphs. Onnela et al. [5] observe a coupling between interaction strengths and network's local structure. Leskovec et al. [6] study the the network community profile plots of a lot of large sparse real-world networks and find that most networks may be views as having a large expander-like "core" with no obvious underlying geometry, and there are a large number of small "whiskers" connect to the "core". Currently, many community detecting algorithms have been proposed such as divisive clustering algorithm [7], clique percolation [8], label propagation algorithm [9] [10], etc. Girvan and Newman [7] have introduced a divisive algorithm based on the values of "edge betweenness". They also propose a measurement of the goodness of communities called modularity Q for the extracted communities [11]. There are lots of algorithms are based on the global modularity optimization [12] [13]. However, the modularity is not a scale-invariant measurement, and the modularity optimization algorithms may fail to identify communities smaller than a certain scale [14]. Kumpula et al. [15] even show a single global optimization criteria do not seem capable for detecting all communities if their size distribution is broad. They believe that in large networks, local community detection methods seem to perform better from the point of view of resolution. Currently, lots of efficient local community detection algorithms have also been proposed. Blondel et al. [16] have introduced a greedy agglomerative clustering algorithm (BGLL algorithm) based on the local modularity optimization. They argue that the resolution limit problem seems to be circumvented by the intrinsic local nature of the algorithm. Recently, Fortunato [17] gives an excellent survey of community detection algorithms.

3 Macro-level Statistics

3.1 Data Set

Call pairs in the call graphs are gained from CDR (Call Detail Records) in the billing systems of several telecom service providers in 4 cities called city A, B, C and D: July 2008 to February 2009 in city A, October 2005 to March 2006 in city B, October 2005 to March 2006 in city C and October 2007 to February 2008 in city D. The call graphs in city A are fixed-line call graphs, and others call graphs are mobile call graphs. The data set has the following characteristics: the study is for intra-region calls, and does not include long distance or international calls; a Call Detail Record (CDR) contains the details of a call such as the time, duration phone number, origin phone number, destination, etc. We form these data sets into undirected call graphs; we connect two phones with an undirected edge (non-multiple edges) if there is at least one reciprocated pair of phone calls between them. To show the temporal information of these call graphs, we form two kinds of call graphs: the first kind graphs are formed monthly, and we call the i^{th} month call graph in city A as G_{Ai}, similarly for city B, C and D; the other kind call graphs contain all the call links over the observed months, and we call this total cumulative graphs in the 4 cities as G_{At}, G_{Bt}, G_{Ct} and G_{Dt}, respectively. Table 1 shows the basic structure properties of these call graphs.

Table 1. Data set of the temporal call graphs

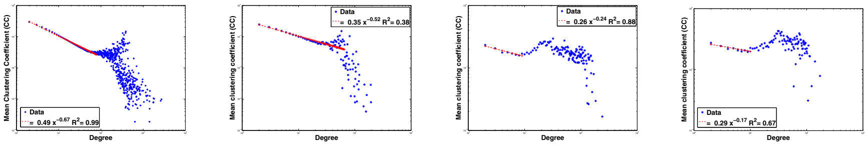
Graph	$\lvert V\rvert$	$\lvert E\rvert$	CC	$\lvert GC\rvert(Percentage)$	$\bar{l}$	$\langle k\rangle$	$\langle k_{GC}\rangle$	r	$\lvert MC_{k\geqslant 3}\rvert$
G_{A1}	1,045,464	3,134,230	0.102	1,036,923 (99.2%)	8.51	6.00	6.04	0.055	447,598
G_{A2}	1,032,906	2,940,745	0.101	1,023,369 (99.1%)	8.67	5.69	5.74	0.054	400,658
G_{A3}	1,009,604	2,613,374	0.093	996,659 (98.7%)	9.05	5.18	5.22	0.062	326,054
G_{A4}	1,011,108	2,729,659	0.094	999,754 (98.9%)	8.89	5.40	5.44	0.062	352,401
G_{A5}	1,005,421	2,674,010	0.095	993,400 (98.8%)	8.97	5.32	5.37	0.067	343,670
G_{A6}	1,009,162	2,722,225	0.096	998,291 (98.9%)	8.87	5.40	5.44	0.066	356,894
G_{A7}	1,033,344	3,065,945	0.106	1,029,662 (99.6%)	8.70	5.96	5.99	0.061	433,365
G_{A8}	999,552	2,580,878	0.095	987,810 (98.8%)	9.08	5.16	5.12	0.073	323,133
G_{At}	1,188,230	9,251,228	0.108	1,187,530 (99.9%)	6.39	15.57	15.58	0.044	2,953,008
G_{B1}	128,115	180,863	0.103	106,850 (83.4%)	11.01	2.82	3.13	0.229	19,312
G_{B2}	129,323	180,974	0.104	107,052 (82.8%)	11.17	2.80	3.11	0.244	19,413
G_{B3}	132,322	192,589	0.105	111,659 (84.4%)	10.96	2.90	3.20	0.257	21,156
G_{B4}	130,875	186,537	0.104	109,111 (83.3%)	10.92	2.85	3.17	0.257	20,458
G_{B5}	133,477	175,315	0.101	106,267 (79.6%)	11.63	2.63	2.96	0.235	17,882
G_{B6}	133,775	170,829	0.100	105,336 (78.7%)	12.59	2.55	2.89	0.276	17,140
G_{Bt}	215,080	544,304	0.129	205,626 (95.6%)	8.04	5.06	5.24	0.142	85,053
G_{C1}	25,676	40,529	0.123	20,906 (81.4%)	10.01	3.16	2.76	0.542	7,829
G_{C2}	25,178	40,214	0.124	20,026 (79.5%)	9.80	3.19	2.64	0.544	8,374
G_{C3}	24,805	40,002	0.127	20,020 (80.7%)	9.75	3.23	2.75	0.530	8,155
G_{C4}	25,012	39,879	0.127	19,980 (79.9%)	9.50	3.19	2.65	0.530	8,159
G_{C5}	23,933	34,943	0.124	17,721 (74.0%)	9.73	2.92	2.72	0.465	6,466
G_{C6}	23,658	35,016	0.121	17,803 (75.3%)	9.83	2.96	2.62	0.507	6,621
G_{Ct}	39,226	102,574	0.143	36,988 (94.3%)	7.32	5.23	2.43	0.466	34,716
G_{D1}	10,499	18,510	0.148	7,973 (75.9%)	8.81	6.00	6.04	0.521	4,577
G_{D2}	10,390	18,444	0.147	7,859 (75.6%)	8.56	5.69	5.74	0.518	4,536
G_{D3}	10,382	18,684	0.146	8,074 (77.8%)	8.64	5.18	5.22	0.519	4,890
G_{D4}	10,545	19,008	0.148	8,177 (77.5%)	8.56	5.40	5.44	0.508	5,061
G_{D5}	10,447	17,267	0.149	7,600 (72.8%)	8.77	5.32	5.37	0.540	3,962
G_{Dt}	15,858	41,450	0.167	14,395 (90.8%)	7.36	5.23	5.64	0.470	17,178

3.2 Degrees

A basic network characteristic is the distribution of degrees which is the well known 'scale-free' phenomenon [2]. As the phenomenon has been well studied and the degree distribution is not the key point of this paper, we will not show the distributions of degrees in these networks. We also calculate the average degrees of the nodes denoted as $\langle k \rangle$, and the average degrees of the nodes in the giant components which are denoted as $\langle k_{GC} \rangle$. We also calculate the correlations between the degrees of different adjacent nodes in these graph, which is characterized by correlation coefficient r [18]. As shown in Table 1, the correlation coefficients are all positive. In the fixed-line phone call graphs the values are rather small, while in the mobile call graphs the values are much larger. The result shows all these call graphs studied are assortative mixing, and the structure of the fixed-line call graphs seems more random.

3.3 Clustering Coefficient

Many real-world networks have been found to be highly transitive. The clustering coefficient CC_i of node i [1] has been used as a measure of transitive of node i in the network. The clustering coefficient for the whole network is $CC = \sum_i CC_i/|V|$, where $|V|$ is number of nodes in the graph. The average clustering coefficients CC for all the temporal graphs are about 0.1 as shown in Table 1. The result indicates that local clustering in the call graphs is much higher than expected in the random graphs, and customers with common friends tend to communicate with each other. Fig. 1 shows the distribution of the mean clustering coefficient versus degree of all the cumulative call graphs. We fit the data with power-law distributions $P(k) \sim k^{-\gamma}$. In Fig. 1, we find the mean clustering coefficients start to disperse with the high degrees. We also notice that in all the cumulative call graphs there are some nodes with large degrees have high mean values of clustering coefficients. This result runs counter to previous result reported in the MSN instant-messaging network which shows that the clustering coefficients decay much faster with high degrees [19]. This phenomenon may indicate that there are many large social groups still prefer to communicate with each other by phones, while many large degree nodes in instant-message networks tend to spam messages to other none-community nodes.



(a) Call graph G_{At} (b) Call graph G_{Bt} (c) Call graph G_{Ct} (d) Call graph G_{Dt}

Fig. 1. Distribution of mean clustering coefficient versus degree in the cumulative graphs

3.4 Components and Shortest Path Length

We denote the giant connected component in each graph as GC and denote the number of nodes in it as $|GC|$. As shown in Table 1, all these call graphs are well connected, and most nodes in these call graphs are in the giant connected components. The average shortest path length between all nodes is often used as a measure of network efficiency. As the call graphs of city A is too large, to approximate the average shortest path lengths between nodes in call graphs of city A, we randomly sample 10% nodes in the giant components and calculate the average shortest path length for each sampled nodes to all others in the call graphs of city A. We denote the average shortest path length in each call graph as $\bar{l}$, and get all the average shortest path lengths of the mobile call graphs in other 3 cities. The values of average shortest path lengths in the giant components of all the call graphs are shown in Table 1. As shown in Table 1, we find the average shortest path lengths in these cumulative call graph are rather small, and there are about "7-degrees of separation" among the users in the cumulative call graphs in these cities.

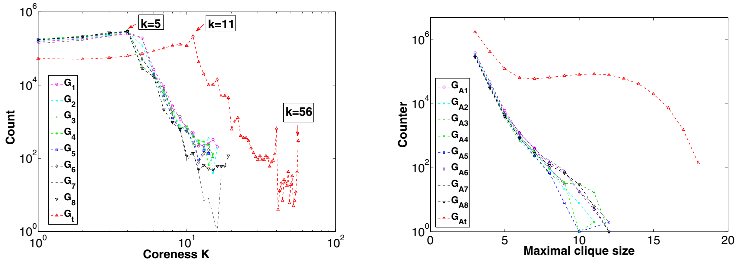
4 Local Cohesive Subgraph

4.1 Graph Transformation and Symbols

To explore massive graphs at different scales, we define the following symbols. An undirected graph G is a triple consisting of a vertex set V, an edge set E, and a relation that associates with each edge two vertices. Suppose the some nodes of graph G are divided into m clusters or communities (overlapping or non-overlapping) that is $c_1, c_2, \cdots, c_m$, and let $C = \{c_1, c_2, \cdots, c_m\}$ be the community set which contains all the communities. In the community quotient graph $G_C^{\div}$ of a community set C, each node i corresponds to a community $c_i \in C$, and there is an edge $e_{i,j}$ between community node i and community node j in $G_C^{\div}$, if and only if there is at least an edge between a node in community c_i and a node in community c_j in the original graph. To get a subgraph of the cohesive communities, we also define the community addition subgraph G_C^+ of a set of node clusters C. The community addition subgraph G_C^+ is the union of all the induced subgraphs of each cluster c_i in C that is $G_C^+ = \bigcup_{c_i \in C} G[c_i]$. The subgraph G_C^{ind} is an induced subgraph of all the nodes in the cluster set C that is $G_C^{ind} = G[\bigcup_{c_i \in C} c_i]$. So in the induced subgraph G_C^{ind} of the community set C, we can not only keep the links in the communities but also keep the links between the communities. Note that G_C^+ and G_C^{ind} are subgraphs of the original G and the community quotient graph $G_C^{\div}$ is an abstract graph of communities derived from G. All the figures of networks in this paper are drawn by our tool *TeleComVis* [20].

4.2 k-Cores and Maximal Cliques

The distribution of coreness and maximal cliques gives us ideas of how quickly the network shrinks as we explore the cohesive subgraphs of networks. We find



(a) Coreness in G_A.

(b) k-Max Clique in G_A.

Fig. 2. Distribution of coreness and k-maximal cliques in city A. The distribution of the number of nodes' coreness is in log-log scale, and the distribution of k-maximal cliques is in semi-log scale.

all these cities have similar distributions of the coreness and maximal cliques, so we just show the distributions in city A. The coreness of a vertex is the highest order of a k-core containing the vertex [21]. Fig. 2(a) shows the distribution of coreness in the call graphs of city A in log-log scale. We note the distributions are remarkably stable when the coreness are at low values. After that, the distribution rapidly drops with their sizes. The k-cores in the cumulative call graph are much larger than those formed in each month. The maximal coreness in G_{A1} is only 16, however, in G_{At} it is 56.

A maximal clique is a complete subgraph that is not contained in any other complete subgraph. The maximal cliques are a more strict strategy to find out the cohesive subgraphs than the method of k-core. It is easy to prove that in a graph, all the maximal cliques whose sizes are at least $k + 1$ must be contained in the k-core. We denote the set of maximal cliques of size k as MC_k, and $|MC_k|$ indicates the number of maximal cliques in MC_k. We show the distributions of maximal cliques in Fig. 2(b), and it is in semi-logarithmic scale. As shown in Fig. 2(b), the number of the maximal cliques drops dramatically as the growing of the size of maximal cliques. We also find that the both of the cores and maximal cliques in the cumulative call graphs are much larger than those in monthly generated call graphs. We hypothesize that the reason for this is that increasing the sizes of graphs in the cumulative leads to increasing connectivity of between the cohesive subgraphs. The result also may indicate that there are many local community links are not very active, so in a single month only some local links appear.

Cohesive Subgraph Link Strength. In this part, we now focus on the relations between the link strength and locality of edges. It is supposed that the high value betweenness edges act as bridges between community [22], so we check the relations between edge betweenness values and their weights. We get the values of edge betweenness in call graph G_{Bt}, G_{Ct} and G_{Dt}, and the fixed-line call graph G_{At} is too large to compute the values of edge betweenness. We calculate the correlations between the edge betweenness values and their weights. To our surprise, we find that there is nearly no correlations between the link weights and their edge betweenness, and the correlation coefficients are 0, -0.047 and 0, respectively. Recently, Kwak et al. [22] study the correlation between edge betweenness and the community pairwise membership probability, and they find there is no correlation between them. Our result coincides with the their finding on the other hand, and we can not find obvious relations between the edge betweenness and their weights.

5 Community Characteristics

In this section, we will extract the communities in the cohesive subgraphs and study the structures and temporal patterns of the communities.

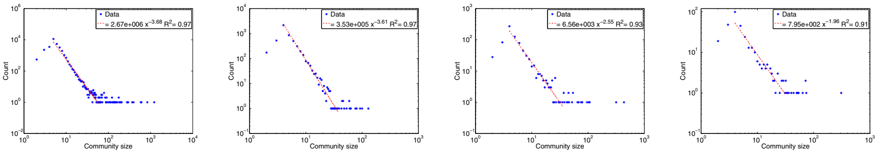
5.1 Community Detection and Overview

As social network data is often incomplete and full of noise and it is unnecessary to divide all the nodes into communities in such massive networks in real-world application, in this empirical study, we are only interested in finding out and studying the statistically significant communities. As maximal cliques are evident from comparison with random graphs, and are robust to noise in the data [23]. There are already some community extracting methods based on maximal cliques [23] [8]. Palla et al. [8] [24] regard that if a node belongs to a k-clique-community then it must in a clique whose size is at least k, otherwise the node will not belong to any community. However, clique overlapping can present serious problems in terms of interpreting the structure of a networks [25] [24]. However, we find the k-clique-community is too slow for detecting communities in these massive network, even worse, in the cumulative graphs the clique overlapping phenomenon can give rise to too many overlapping communities which makes it difficult to explain the community detection results in the coherent subgraphs.

We first use maximal cliques to filter the less statistically significant cohesive subgraphs. After that we use recent high quality non-overlapping community detection algorithm to divide these cohesive subgraphs into different communities. Specifically, in the fixed-line call graphs, we enumerate all the maximal cliques whose sizes are at least 5, 6, 7 and 8 in G_{At}, respectively. In the mobile call graphs, we enumerate all the maximal cliques whose sizes are at leat 4 in G_{Bt}, G_{Ct} and G_{Dt}. We denote these subgraphs composed by maximal cliques whose size are at least $\mathcal{K}$ as $G^{+}_{MC_{k \geqslant \mathcal{K}}}$, and denote the induced subgraph of the nodes in the maximal clique set $MC_{k \geqslant \mathcal{K}}$ as $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$. In the following parts, we will focus on the communities' characteristics in the subgraphs of $G^{+}_{MC_{k \geqslant \mathcal{K}}}$ and $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$ in all the call graphs.

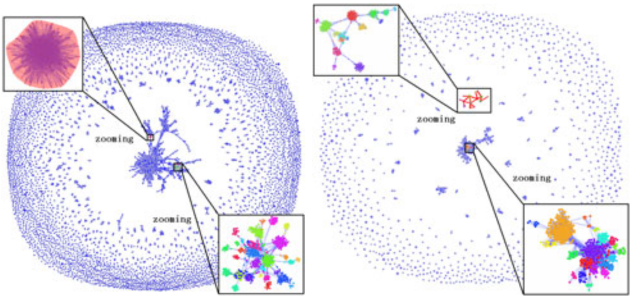
5.2 Communities in Cohesive Subgraphs

After we get the subgraphs composed by these maximal cliques which are denoted as $G^{+}_{MC_{k \geqslant \mathcal{K}}}$ in the cumulative call graphs, we use the BGLL algorithm



(a) $G^{+}_{MC_{k\geqslant 5}}$ of G_{At} (b) $G^{+}_{MC_{k\geqslant 4}}$ of G_{Bt} (c) $G^{+}_{MC_{k\geqslant 4}}$ of G_{Ct} (d) $G^{+}_{MC_{k\geqslant 4}}$ of G_{Dt}

Fig. 3. Distribution of the community sizes in the subgraph graphs $G^{+}_{MC_{k\geqslant \mathcal{K}}}$ of the cumulative call graphs. We get the communities by the BGLL algorithm in this subgraphs after filtering maximal cliques whose sizes are less than $\mathcal{K}$.



(a) $G^{+}_{MC_{k\geqslant 6}}$ of G_{At} (b) $G^{+}_{MC_{k\geqslant 7}}$ of G_{At}

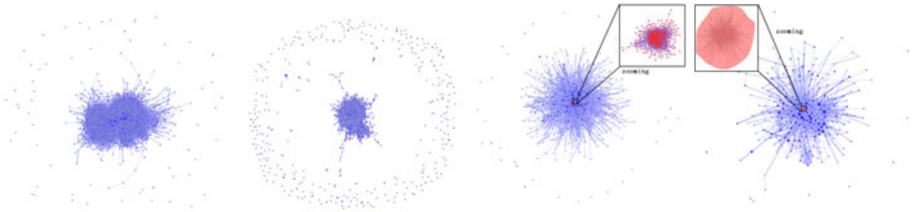
Fig. 4. The community quotient graphs $CG_{k\geqslant 6}$ and $CG_{k\geqslant 7}$ of maximal clique addition graphs $G^{+}_{MC_{k\geqslant 6}}$ and $G^{+}_{MC_{k\geqslant 7}}$ in the cumulative call graphs G_{At}. The communities are extracted by the BGLL algorithm.

to extract communities. To avoid the famous resolution limit problem, as mentioned by Blondel et al. [16], we get the communities obtained at the first level partition. By comparing with the results extracted by other famous algorithms such as GN algorithm, CNM algorithm and LPA algorithm, we find the quality of the communities detected is better as measured by the modularity and the size of largest community which is usually much smaller than the one found by other algorithms. In G_{At}, the modularity values of extracted communities in subgraphs $G^{+}_{MC_{k\geqslant 5}}$, $G^{+}_{MC_{k\geqslant 6}}$, $G^{+}_{MC_{k\geqslant 7}}$ and $G^{+}_{MC_{k\geqslant 8}}$ are 0.835, 0.911, 0.885 and 0.808, respectively. In the subgraphs $G^{+}_{MC_{k\geqslant 4}}$ of G_{Bt}, G_{Ct} and G_{Dt}, the modularity values of extracted communities are 0.875, 0.721 and 0.770, respectively. The modularity values show that there are obvious community structures in these clique addition subgraphs. Fig. 3 shows the distributions of the community sizes in these clique addition subgraphs, and we can not find very large “monster” communities in the extracted communities.

To show the original relations between communities, we define community quotient graph $CG_{k\geqslant \mathcal{K}} = G^{\div}_{C}$ of the community set C extracted in these clique addition subgraphs. In the community quotient graph $CG_{k\geqslant \mathcal{K}} = G^{\div}_{C}$ of community set C, nodes are communities extracted from subgraph $G^{+}_{MC_{k\geqslant \mathcal{K}}}$, and there is an edge between two community nodes if and only if there are the links between the two communities in $G^{+}_{MC_{k\geqslant \mathcal{K}}}$. Fig. 4 shows the community quotient graph $CG_{k \geqslant 5}$ and $CG_{k \geqslant 6}$ of G_{At}. As shown in Fig. 4, although many communities are not connected in the maximal clique addition subgraphs $G^{+}_{MC_{k \geqslant \mathcal{K}}}$, we can find out the core-periphery pattern of the communities in the call graphs. In the subgraphs, through many maximal cliques are isolated, however, we can still find a large connected component in each community quotient networks.

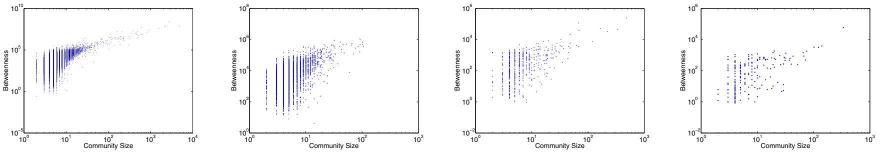
5.3 Community Centrality and Periphery

To study the relationships between different communities in the cohesive subgraphs, we get another kind of community quotient network based on maximal clique induced subgraph $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$. After we extract the communities C in the induced graph $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$, we can also get a quotient graph $IG_{k \geqslant \mathcal{K}} = G^{\div}_{C}$ based on the community set C extracted in the subgraph $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$. Fig. 5 shows the community quotient graph of $G^{ind}_{MC_{k \geqslant 6}}$ in the cumulative graph G_{At}, and the community quotient graphs of $G^{ind}_{MC_{k \geqslant 4}}$ in other 3 cumulative graphs G_{Bt}, G_{Ct} and G_{Dt}, respectively. The node sizes in the the quotient graphs are proportional to the sizes of communities. As shown in Fig. 5, we find that there is a well connected giant component in each community quotient graph and many large communities are in the center of the quotient graphs. We calculate the relations between the positions and sizes of these communities in the community quotient network. Betweenness centrality based on shortest paths is a standard measure of node control in numerous studies for network analysis [26]. The community



(a) $G^{ind}_{MC_{k \geqslant 6}}$ of G_{At} (b) $G^{ind}_{MC_{k \geqslant 4}}$ of G_{Bt} (c) $G^{ind}_{MC_{k \geqslant 4}}$ of G_{Ct} (d) $G^{ind}_{MC_{k \geqslant 4}}$ of G_{Dt}

Fig. 5. The community quotient graphs in the induced subgraph $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$. The communities are extracted by the BGLL algorithm.



(a) $G^{ind}_{MC_{k \geqslant 5}}$ in G_{At} (b) $G^{ind}_{MC_{k \geqslant 4}}$ in G_{Bt} (c) $G^{ind}_{MC_{k \geqslant 4}}$ in G_{Ct} (d) $G^{ind}_{MC_{k \geqslant 4}}$ in G_{Dt}

Fig. 6. Community size and community betweenness in the community quotient network $G^{\div}_{C}$ of the community set C in induced subgraph $G^{ind}_{MC_{k \geqslant \mathcal{K}}}$

Table 2. Properties of community quotient graph $IG_{k\geqslant\mathcal{K}}$ in $G^{ind}_{MC_{k\geqslant\mathcal{K}}}$

Graph	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert GC\rvert(Percentage)$	$\bar{l}$	$\bar{w}_{ct}$	$\bar{w}_{cd}(min)$	$\bar{w}_{it}$	$\bar{w}_{id}(min)$
$IG_{k\geqslant5}$ of G_{At}	27012	404080	26995(99.9%)	4.26	63.28	101.67	19.73	33.74
$IG_{k\geqslant6}$ of G_{At}	6334	47356	6274(99.5%)	4.68	65.14	94.51	19.00	27.88
$IG_{k\geqslant7}$ of G_{At}	1247	5663	1180(94.6%)	4.39	59.25	73.29	17.62	22.68
$IG_{k\geqslant8}$ of G_{At}	306	1344	277(90.5%)	3.93	50.31	55.18	15.26	18.63
$IG_{k\geqslant4}$ of G_{Bt}	4984	19770	4647(93.2%)	5.42	42.56	47.76	12.95	16.93
$IG_{k\geqslant4}$ of G_{Ct}	744	2358	710(95.4%)	3.81	31.72	35.91	10.79	16.17
$IG_{k\geqslant4}$ of G_{Dt}	283	871	273(96.5%)	3.49	37.03	34.48	13.54	14.88

sizes versus their betweenness values in the community quotient graphs are plotted in log-log scales in Fig. 6. By using the betweenness algorithm proposed by Brandes [26], we find there is a strong linear correlation between community size and community betweenness in the community quotient network. In quotient graphs of $G^{ind}_{MC_{k\geqslant5}}$, $G^{ind}_{MC_{k\geqslant6}}$, $G^{ind}_{MC_{k\geqslant7}}$ and $G^{ind}_{MC_{k\geqslant8}}$ in G_{At}, the values of correlation between community size and community betweenness are 0.759, 0.807, 0.867 and 0.794, respectively. In quotient graphs based on communities in $G^{ind}_{MC_{k\geqslant4}}$ in G_{Bt}, G_{Ct} and G_{Dt}, the values of correlation are 0.700, 0.823 and 0.839, respectively. This results indicate that the larger a community is, the more likely it will be in the center of the call graph. We also extract communities by the CNM algorithm [13] and LPA algorithm [10], and we can observe the same results as the communities extracted by BGLL algorithm.

We also calculate the sizes of the giant components and the average shortest path lengths of these community quotient graphs. The basic properties of the community quotient graphs are shown in Table 2. To compare the strengths of the inter-community links and the intra-community links, we calculate the mean weights of these edges. We denote the mean of intra-community call times and durations (minutes) as $\bar{w}_{ct}$ and $\bar{w}_{cd}$, and denote inter-community call times and durations (minutes) as $\bar{w}_{it}$ and $\bar{w}_{id}$. As shown in Table 2, the mean link strength in these communities is much stronger than the mean link strength between them. The results show that there are many weak inter-community links existed between these local communities and the intra-community link strength is usual much more stronger than that of the inter-community link; The result also indicates that users in the same community still tend to communicate with each other. We can also find the average shortest path lengths in the giant components of the community quotient networks are rather small, and there are about "4-degrees of separation" among the users in the community quotients graphs. The result indicates that we get even the 'small-world' phenomenon from community quotients graphs, and it also shows these weak links are often the bridges between local cohesive communities or subgraphs.

6 Conclusion

In this paper, we give an empirical study on several temporal call graphs in multilevel views. We first explore the basic statistical characters of the call graphs. After that we focus on the properties of the local cohesive subgraphs, we filter less cohesive subgraphs and study the properties of the statistically significant communities. To fully understand the properties of communities, with the help of visual analytics, we first get an overview of the community network, and then we zoom into certain interesting communities on a microscopic scale to get more details about their structures and evolution trends and check the pattern observed. We find the 'small-world' phenomenon in the community quotient graph, and the large communities are very likely to be in the cores of the call graphs. We believe that our study has revealed some structure in different kinds of call graphs and provide a valuable platform for theoretical modeling and further analysis. We also believe that our analysis results will help Telecom operators have better understanding of their customer communities and provide insights for designing better market strategies for different customer communities.

Acknowledgments. This work is supported by the National Science Foundation of China (No. 90924029, 60905025, 61074128). It is also supported by the National Hightech R&D Program of China (No.2009AA04Z136) and the National Key Technology R&D Program of China (No.2006BAH03B05).

References

1. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
2. Barabási, A.L., Albert, R.: Emergence of Scaling in Random Networks. Science 286, 509–512 (1999)
3. Nanavati, A.A., Gurumurthy, S., et al.: On the Structural Properties of Massive Telecom Call Graphs: Finding and Implication. In: Proceedings of CIKM, pp. 435–444 (2006)
4. Nanavati, A.A., Singh, R., et al.: Analyzing the Structure and Evolution of Massive Telecom Graphs. IEEE Transactions on Knowledge and Data Engineering 50, 703–718 (2008)
5. Onnela, J.P., Saramäki, J., et al.: Structure and tie Strengths in mobile communication networks. Proc. Natl. Acad. Sci. 104, 7332–7336 (2007)
6. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Statistical properties of community structure in large social and information networks. In: Proceeding of the 17th International Conference on World Wide Web, pp. 695–704 (2008)
7. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. J. Proc. Natl. Acad. Sci. 12, 7821–7826 (2002)
8. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature 435, 814–817 (2005)
9. Leung, I.X.Y., Hui, P., Liò, P., Crowcroft, J.: Towards real-time community detection in large networks. Phys. Rev. E. 79, 66107 (2009)
10. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E. 76, 36106 (2007)
11. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E. 69, 26113 (2004)
12. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Phys. Rev. E. 69, 66133 (2004)
13. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E. 70, 66111 (2004)

14. Fortunato, S., Barthélemy, M.: Resolution limit in community detection. Proc. Natl. Acad. Sci. 104, 36–41 (2007)
15. Kumpula, J.M., Saramäki, J., Kaski, K., Kertész, J.: Limited resolution in complex network community detection with Potts model approach. European Physical Journal B 56, 41–45 (2007)
16. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. 10008 (2008)
17. Fortunato, S.: Community detection in graphs. Physics Reports 486, 75–174 (2010)
18. Newman, M.E.J.: Assortative Mixing in Networks. Phys. Rev. Lett. 89, 208701 (2002)
19. Leskovec, J., Horvitz, E.: Planetary-scale views on a large instant-messaging network. In: Proceeding of the 17th International Conference on World Wide Web, pp. 915–924 (2008)
20. Ye, Q., Wu, B., et al.: TeleComVis: Exploring Temporal Communities in Telecom Networks. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, pp. 755–758. Springer, Bled Slovenia (2009)
21. Vladimir, B., Matjaž, Z.: An O(m) Algorithm for Cores Decomposition of Networks. CoRR arXiv.org/cs.DS/0310049 (2003)
22. Kwak, H., Choi, Y., et al.: Mining communities in networks: a solution for consistency and its evaluation. In: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference, pp. 301–314. ACM, New York (2009)
23. Spirin, V., Mirny, L.A.: Protein complexes and function modules in molecular networks. Proc. Natl. Acad. Sci. 100, 12123–12128 (2003)
24. Palla, G., Barabasi, A.L., Vicsek, T.: Quantifying social group evolution. Nature 446, 664–667 (2007)
25. Everett, M.G., Borgatti, S.P.: Analyzing Clique Overlap. Connections 21, 49–61 (1998)
26. Brandes, U.: A Faster Algorithm for Betweenness Centrality. Journal of Mathematical Sociology 25, 163–177 (2001)

Combating Link Spam by Noisy Link Analysis

Yitong Wang, Xiaofei Chen, and Xiaojun Feng

School of Computer Science, Fudan University, Shanghai, China
{yitongw,082024056,072021144}@fudan.edu.cn

Abstract. Link Spam has indentified as one of the major obstacles for link-based ranking algorithms of modern search engine since it intently constructs hyperlink structure to help some poor-content pages obtaining undeserved high rank. This problem is even worse with the advent of wikis, blogs and forum that are rich in links. Existing works on link spam are mainly focused on link spam detection by extracting some special link structures (e.g. clique, tight bipartite etc.). However, link spam structures could have many variations and easily make the existing detection methods ineffective. In this paper, we tackle the problem of link spam from a more fundamental viewpoint—"noisy link" analysis. First of all, how "non-voting" hyperlinks affect the quality of ranking is investigated, and then based on this investigation, an approach to detect and process "noisy link" both effectively and automatically is proposed. We also compare our work with two other related works (TrustRank and Site-level Noise removal) on two real web datasets. The experimental results demonstrate that the proposed "noisy link" analysis is very effective on both spam page filtering and final ranking improvement.

Keywords: Link Spam, Noisy link, Down-weight, PageRank.

1 Introduction

Nearly 90% of the traffic to most web sites is found by using a search engine, and most people are only interested in top k search results. Due to a huge number of business opportunities brought by popular web pages, everyone wants his sites to show up as high as possible on the search results. To achieve that goal, some will provide good contents and services, while others will simply pull dirty tricks. Web spam refers to any deliberate actions to help some pages obtaining undeserved high rank. It has been identified as one of the most difficult challenges for web search in future decades [1].

Hyperlinks between pages have been proven to be valuable to identify the relationship between source and target pages. Link-based ranking algorithms even brought up the new generation of web search engine [2] [3]. Due to its success, link spam has appeared as a major web spam technique through intently-constructed tight link structure. Combating link spam thus becomes a challenging web mining task.

Most existing link spam detection algorithm is mainly focused on identifying some special link structures (clique, bipartite etc) or computing link-spamicity score [4]. Nevertheless, the results are still far from satisfaction. Link spam structures have many variations and can easily make the existing detection algorithms ineffective.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 453–464, 2010.

Moreover, whether one page is a spam page or not is a subjective judgment to different people. Some pages are totally meaningless or are simply junk pages, while others are related to the topic but are of poor-content. It is very difficult to tell the differences between the two situations. Link spam detection is not the final goal for improving the ranking; the final goal is to have a high quality web search engine, which is robust to various spam techniques.

It is worth noting that although spammers' tricks ([1], [5], and [6]) sometimes made existing algorithms difficult to distinguish them, they cannot conceive their ultimate goal: to "push" the ranking of target pages with some intently constructed links, which is usually made up of useless or non-voting links. In this paper, we try to combat link spam by noisy link analysis. Our main contributions in this paper are 1) proposed a flexible algorithm to automatically detect "noisy link"; 2) discussed three methods to down-weight noisy links automatically; 3) revealed the relationship between noisy links and links spam and demonstrated by experimental results that by noisy link analysis, we could combat link spam effectively; 4)compared the proposed approach with two other related works (TrustRank [8]and Site Level Noise Removal[9]) to show its effectiveness on spam page filtering as well as final ranking improvement.

The rest of the paper is organized as follows: Section 2 gives background knowledge and presents priori related work from two aspects: link spam and noisy link analysis. In section 3, after a brief investigation on how non-voting links affects search quality, we describe core ideas of our approach in detail. Three down-weighted methods to deal with noisy link are also discussed. Experiments and comparisons are conducted and evaluated in section 4. Finally, in Section 5 we conclude our paper with future work discussion.

2 Related Work

In this section, we review related works from two aspects: link spam and noisy link analysis.

2.1 Link Spam

Spam has appeared much earlier than the advent of link-based ranking algorithms. Link Spam, which is targeting at link-based ranking algorithm by artificially created link structure, is an important web spam technique and difficult to identify. This issue is so important because it is very difficult to tell the quality of hyperlinks and fake scores could quickly propagate to other pages and thus affect many pages.

Most existing works on link spam detection are starting from theoretical analysis of PageRank formula[10]. It is found that "link farm" [10] (many template-based, auto-generated pages with common out-link pointed to a page), or clique are the commonly used structure for link spam. TrustRank[8] is proposed to combat spam pages by selecting a set of good pages as seeds with high trust score assigned and then a biased version of PageRank is used to propagate these trust values along out-links throughout the entire web. Finally, a page with high trust score is believed to be a good page. Oppositely, Anti-Trust Rank [11], SpamRank [12] are proposed with bad pages as seeds and propagating the bad score. However, TrustRank is very sensitive and highly dependent on the choice of seeds. For a given seeds set, TrustRank is also biased towards large communities. Zhou et al. [4] proposed to use spamicity to measure how likely a web page is a spam by assigning a spamicity score to each page.

BCC (bi-connected components) structure [12], complete bipartite graph [13], clique [14] are all studied to indicate that they are prone to be attacked by spammers. The work in [15] and [16] proved that collusion (including clique, star and ring structure) can make a page obtain new PageRank value up to 7 times as its original value. These approaches all aim at certain particular structures and exact mapping is required, which make the proposed approaches inefficient if the spammers make some minor changes to the link structure. Although link spam is mainly composed of "non-voting links", to our best knowledge, no work so far studies the relationships between noisy link and link spam.

2.2 Noisy Link Analysis

The Concept of "noisy link" was first referred in [9] as non-voting links. They enumerated three kinds of site-level link structures as models of noisy links. If link structures match one of the three models, they will be identified as noisy links and then removed. The core idea of these models is: if most of one-site's in-link pages are from the same site (based on some pre-defined threshold), all in-links are considered as "noisy" links and removed. This work is most related to our work in this paper. However, the three models proposed in [9] are too rigid and could easily be made ineffective when spammers change the link structure. Moreover, totally removing the so-called "noisy link" is too arbitrary and could easily remove some normal pages mistakenly. Final results are also very sensitive to filtering threshold. Our approach, however, is on page-level and could be adaptable to site-level situations studied in [9] without these limitations. It is in [17] that qualified links are defined and identified. It defined qualified links as the links that actually contribute to the true recommendation or "vote" for the target pages. The paper proposed a classifier which could tell such "qualified link" automatically by measuring contents-similarity between the two end pages of the link. However, we argue that this presumption is too arbitrary and will narrow the results greatly since similar pages actually do not always link to each other and some qualified links sometimes connect two high-quality pages but with very different contents.

3 Noisy Link Analysis

3.1 Motivation

There was not a very clear definition of "noisy link" in [9] where the concept of "noisy link" was first referred. We define 'noisy link" as non-voting link or link does not imply any "vote" or "support" for the target page it points to. While this definition is a little bit mislead since even a qualified link could be "non-voting" due to its other function, our definition here is mainly ranking-oriented. We first briefly investigate how non-voting links actually affect the results of PageRank.

The experiment is based on query “cell phone”. Topic-related neighbor graph is constructed as that in [3] with top 100 search results as seed set and the seed set is expanded by two steps along forward links and backward links to form the final web graph. We obtain a web graph with 24,441 nodes and 105,589 links and then run PageRank algorithm on the neighbor graph. The top 15 ranking results are shown in Table 1. Among top 15, most results are not relevant with topic “cell phone” at all but dominated by URLs of some big companies. Eight out of the top 15 results (marked as bold font) are dominated by company about.com. Top 2-3 results are dominated by company “howstuffworks.com”. The reason is that pages belong to one company are usually connected and co-cited each other very strongly.

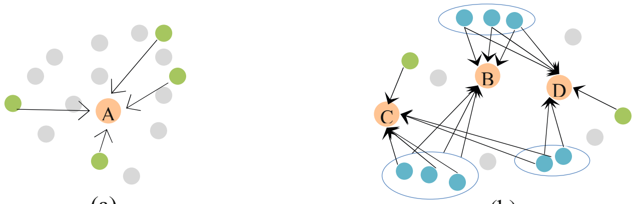
Table 1. Top 15 PageRank Results for “Cell Phone”

Ran	URL	Rank	URL
1	**http://about.com/**	9	**http://spiderbites.about.com/sitemap.htm**
2	*http://www.howstuffworks.com/*	*10*	http://www.nytco.com/
3	*http://mobiltravelguide.howstuffworks.com/*	*11*	http://www.cellhire.fr/
4	*http://auto.consumerguide.com/*	*12*	**http://advertise.about.com/reprint.html**
5	http://www.cellhire.com/	13	**http://beanadvertiser.about.com/news.html**
6	**http://beanadvertiser.about.com/**	14	**http://ourstory.about.com/**
7	http://www.cellhire.co.uk/	15	**http://jobs.about.com/**
8	**http://w.about.com/w3c/p3p.xml**		

From Table 1, we observe a kind of “collusive” phenomenon among in-link pages as that of node *B* shown in Fig.1 (b).When we say “collusion”, we mean these in-link pages usually belong to the same “big company” or are strongly “related” to each other. Moreover, we found many in-links actually do not imply “voting” no matter this “collusion” is intentional or accidental.

Motivated by this observation, we argue that if one page’s in-link pages demonstrate some kind of “collusion” like that of node *B* depicted in Fig .1(b), it is with high probability that node *B* is a spam page or low-quality page and links from circled groups to node *B* are “noisy links”. Obviously, in-link distribution depicted in Fig.1 (a) is normal for a high quality page *A*. Further investigation to support this assumption will be shown in section 4.3.

According to the above analysis, we propose that if one page’s in-links demonstrate a kind of “collusion” phenomenon, then it is of high probability that the page is a spam page and those in-links are defined as “noisy links”. For some pages, “collusive” behaviors among their in-link pages could be accidental. Our approach of noisy link analysis includes three steps: 1) clustering; 2) noisy links detection; 3) penalty processing.



(a) (b)

Fig. 1. In-link distribution for pages with different qualities

3.2 STEP 1: Clustering

Different clustering approaches could be incorporated here in the case that they are effective to group “collusive” pages in the same cluster. Our clustering method is based on the assumption: if two pages share many out-links or in-links, they must be related in some respect. This “relation” could imply many cases: similar in contents, having similar interests, created for the same purpose, belonging to the same mother company etc.

3.2.1 Similarity Measurement

$$S(p,q)=\frac{1}{2}\frac{I(p)\cap I(q)}{I(p)\cup I(q)}+\frac{1}{2}\frac{O(p)\cap O(q)}{O(p)\cup O(q)}. \tag{1}$$

Above formula gives detailed similarity measurement during clustering process. *S (p, q)* represents similarity between page *p* and *q*, *O(p)* and *I (p)* represent out-links and in-links of page *p* respectively. Obviously, the bigger *S(p,q)*, more related *p* and *q*.

3.2.2 Clustering Based on Modified K-Means

A modified K-means clustering method is implemented in our approach. Like K-means, all nodes are clustered progressively by assigning each node to its nearest cluster. Due to the inherent limitations of standard K-means, K is not specified explicitly but is generated automatically during clustering process according to pre-defined similarity threshold. We define the centroid of cluster *C* as the node that makes the diameter of *C* minimum. The diameter of one cluster C is defined as the average similarity between the centroid to all other nodes in C. If cluster *C* is a singleton cluster with only one node member *p*, *p* is also the centroid of *C*. As a result of clustering, some pages will be singleton clusters since they could not be grouped together with any other pages. We think it is reasonable for our analysis. Similarity threshold is important for clustering and we set it as 0.25 based on empirical values. Our clustering method is described in table 2.

Table 2. Clustering Method

1	Define similarity threshold and filter pages with few links
2	Assign each page *p* progressively to its nearest cluster *C* based on the similarity (if above the similarity threshold) between page *p* and centroid of cluster *C*
3	The page will be one cluster itself if no existing cluster meets step 3
4	Recompute the centroid of cluster *C* if its cluster members changed
5	Repeat Step 2 to step 5 until all pages are assigned and all centroids do not change anymore

3.3 STEP 2: Noisy Links Detection

We believe that if a page is indeed valuable to a topic or field, its in-link distribution should be “decentralized” instead of “collusive”. This understanding motivates the main idea of the paper, that is: if in-links of a page are mainly from one particular cluster or a few clusters, we define these links as “noisy links” and the cluster(s) as “support cluster(s)”. We define “noisy link ratio”, *ratio (p, X)* as the percentage of page *p’s* in-links from cluster *X*.

$$ratio(p,X)=\frac{|X\cap In(p)|}{|In(p)|},|In(p)|>1, 0\le ratio(p,X)\le 1\,. \tag{2}$$

If *ratio (p, X)* is above certain pre-defined threshold, we define links from cluster *X* to *p* as “noisy links” and *X* as “support cluster”. Obviously, there could be more than one support cluster for a given page *p*.

3.4 STEP 3: Penalty Processing

Once “noisy links” are identified, we have to penalize them accordingly. One way is to filter/remove all “noisy links” as the PageRank0 penalty given in BadRank [7] or in Slab [9] and the other way is to down-weight the link as in [17]. Three down-weighted methods are considered here: *constant method*, *linear method* and *logarithm method*. At last, we chose the Liner Method

Liner Method**:** PageRank value propagated along out-link from page *q* in cluster *X* to page *p (q→p)* is down-weighted as *PR* (*q)* d* (*d* is *1-ratio (p, X)*)

4 Experiments

In this part, we conduct thorough experiments to evaluate the effectiveness of noisy link analysis proposed on ranking improvement as well as spam page filtering. Firstly, we compare three down-weighted methods introduced in section 3.4 to study their effects on improvement of final ranking performance; secondly, we compare our work in this paper with two other related works TrustRank [8] and Site-level noise removal [9] in terms of ranking as well as spam page filtering. Meanwhile, we also investigate the relationship between link spam and noisy links detected.

4.1 Datasets

4.1.1 Query-Oriented Dataset

In March 2008, we collected our first dataset using Yahoo! API [3]. By submitting queries to Yahoo!, we obtain top 100 search results for each query as a seed set and expand the seed set with 100 in-links of each page in the seed set. As a result, for each query, we obtain a web sub-graph with 100+100×100=10,100 nodes. Top 21 hottest queries to Yahoo! search engine are chosen for our experiments. The queries cover various topics including entertainment, IT, real estate, sports etc.

4.1.2 Real Spam Datasets: WEBSPAM-UK2006

We use another datasets released by the Search Engine Spam Project at Yahoo! Research Barcelona, namely WebSpam -UK2006. A team of volunteers were asked to classify this set of pages as “normal”, “spam” or “borderline”. Moreover, the project organizer added two kinds of special votes: all the UK pages mentioned in the open directory project (http://www.dmoz.org) are voted “normal” since it is believed that the pages in those domains are rarely spams. It is interesting to note that on the average, each volunteer spends 5 minutes at each marked site. We interpret this as that the mark (normal, spam) is given after serious consideration instead of just out of intuition. For WebSpam-UK2006, there are totally 11402 hosts. More than 70% of all pages are checked and 1942 hosts are labeled as "spam".

4.2 Clustering Results

Now let's check what actually are grouped together in one cluster? Table 3 lists two examples of clusters obtained. For topic "apple", we find that blog websites all belong to the same mother company Weblogs Inc. They not only share common out-links but also link each other tightly. Actually, there are more than 30 such websites under the same mother company but in different domains. By our clustering method, they are all clustered together. This is also in accordance with our expectation of clustering. For UK2006 dataset, pages in different domains but controlled by the same company (mysite.wanadoo-members.co.uk) are all grouped together.

Table 3. Example of one cluster on "Apple" dataset and one on UK2006 dataset

"Apple" Dataset	UK-2006 Dataset
http://www.adjab.com/	a.w.hunt.mysite.wanadoo-members.co.uk
http://www.autoblog.com/	aandpdistribution.mysite.wanadoo-members.co.uk
http://www.autobloggreen.com/	abridgeflowers.mysite.wanadoo-members.co.uk
http://www.bloggingstocks.com/	absolutetreecare.mysite.wanadoo-members.co.uk
http://www.cinematical.com/	acadiabuns.mysite.wanadoo-members.co.uk
http://www.downloadsquad.com/	aeromedia.co.uk
http://www.dsfanboy.com/	afcrompers.mysite.wanadoo-members.co.uk
http://www.luxist.com/	africanwildlife.mysite.wanadoo-members.co.uk
	agivey.mysite.wanadoo-members.co.uk
	aidcall-alarms.co.uk

4.3 Statistic Analysis

In this section, detailed statistic analysis is made on WEBSPAM-UK2006 dataset to give more convincing support to our assumption about noisy link. We try to clarify three points: 1) Whether in-links of spam pages do demonstrate a kind of "collusive" behavior? 2) How spam pages are related each other? 3) Whether link spam is an extreme situation of noisy links?

There are totally 11402 hosts and among them 1924 hosts are labeled "spam". Among all pages, 6748 pages are clustered in 843 clusters (cluster size is bigger than 3). Of 843 clusters, only 307 clusters include spam pages and among all 1568 spam pages in clusters, around 68.4% (1073 spam pages) are actually gathered in 81 clusters. Table 4 conveys a very clear message: most spam pages are strongly related to each other.

Table 4. Spam page distributions in clusters

	Total	Including Spam	#Spam >5
#Cluster (size>3)	843	307	81
#Page	6748	3498	1699
#Spam page	1568	1568	1073
Spam Perc.	23.2%	44.8%	68.4%

Table 5. In-link distribution of Spam from Spam

Percent of in-links from Spam	100%	90%	80%	70%	60%	50%
Total	80	109	284	706	1398	1755
Percentage	4.15%	5.66%	14.8%	36.7%	72.7%	91.2%

Table 6. In-link Distribution of Spam pages from Clusters

Per. Of In-link	From Largest Cluster		From 3 clusters	
	# spam pages	Percent	#spam pages	Percent
100%	105	5.46%	262	13.6%
≥90%	142	7.38%	348	18.1%
≥80%	190	9.86%	565	29.4%
≥70%	260	13.5%	890	46.3%
≥60%	422	21.9%	1325	68.9%
≥50%	683	**35.5%**	1637	**85.1%**

We further in vestigate in-link distribution of spam pages and statistic results are shown in Table 5 and Table 6. Table 5 gives how likely in-links of spam pages are from spam pages. According to in-link distribution of spam pages, it is very clear that in-link pages of spam pages do demonstrate a kind of "collusive" behavior in order to "push" spam pages to get undeserved high rank and this "collusive" behavior supports our previous assumption.

4.4 Noisy Links Analysis

According to clustering results and down-weighted methods discussed in section 3.4, it is quite easy to detect "noisy links" and handle them accordingly. We check the effect of three down-weight methods by conducting experiments on query-oriented datasets and compare the results with the original one (PageRank without "noisy link analysis). The top 20 URLs are checked based on PageRank value manually to determine their relevance to the topic in terms of number of relevant results. For constant method, we vary t_N with 0, 0.25 and 0.5 vary d with 0.2, 0.5 and 0.8. Linear and logarithm methods down-weighted noisy link automatically based on "noisy link ratio". Clustering threshold is set based on empirical value 0.25.

Of 21 topics, the results of 12 topics are presented in Table 7. The average value is based on total 21 topics. Max-Rel-Imp (maximum relative improvement) means the ratio of maximum extra relevant results obtained by noisy-link analysis with the original one and Max-Abs-Imp (maximum absolutes improvement) means the ratio of maximum extra relevant results obtained with 20. (E.g. based on the result in 1st line of Table 7 max-rel-imp is 40% (14-10)/10); max-abs-imp is 20% ((14-10)/20). It is very clear that when d is 0.2 and t_N is 0 to 0.25 for constant method, as well as linear method achieve the best performance. For some specific query such as apple, blackberry and fashion, the maximum absolute improvement even exceeds 50%.

In the following discussion, if there is no explicit explanation, we use linear down-weight method in our approach.

Table 7. Effects of Noisy link Analysis on ranking performance (t_C is 0.25)

	Orig	Cons.									Lin.	Log	Max. Rel. Imp.	Max Abs. Imp
t_N		0			0.25			0.5						
D		0.2	0.5	0.8	0.2	0.5	0.8	0.2	0.5	0.8				
a)	10	13	12	11	13	13	12	13	**14**	13	12	12	40.0%	20%
b)	6	**19**	17	8	18	17	9	**19**	17	8	**19**	**19**	217%	**65%**
c)	10	18	19	14	**20**	**20**	15	**20**	**20**	18	**20**	**20**	100%	**50%**
d)	10	18	16	15	18	18	14	19	15	12	**20**	18	100%	**50%**
e)	12	**20**	19	13	17	17	13	14	14	15	16	16	66.7%	40%
f)	12	16	16	12	16	16	12	16	18	12	**19**	**19**	58.3%	35%
g)	13	**16**	**16**	**16**	15	**16**	**16**	15	14	**16**	15	14	23.1%	15%
h)	15	**20**	**20**	15	**20**	**20**	15	**20**	**20**	15	**20**	**20**	33.3%	25%
i)	17	**18**	16	16	16	16	15	13	12	15	**18**	**18**	5.9%	5%
j)	14	15	12	14	**17**	15	14	**17**	**17**	14	**17**	16	21.4%	15%
k)	11	**18**	15	13	**18**	16	13	**18**	16	13	17	17	63.6%	35%
l)	13	**17**	16	14	16	15	13	13	13	16	16	14	30.8%	20%
Avg.	11.8	**16.4**	15.1	13	**16.1**	15.2	13.1	15.3	14.9	13.2	**16.3**	15.8	59.7%	28%

a) American Idol, b) apple, c) Blackberry, d) fashion, e)iphone, f) Jaguar, g) Juno, h) Lexus,i) mouse, i) office, k) soccer, l) Web Hosting

4.5 Comparing with Other Related Works

In this section, we compare noisy link analysis proposed with two other related works TrustRank and Site-level Noise Removal in terms of ranking improvement as well as spam page filtering.

4.5.1 Ranking Improvement

Site-level noise removal [9] is in the site-level instead of page-level, we also need to run our approach on site-level in order for comparison. We convert query-oriented data set from page-level to site-level by re-establishing link connection between two websites if there are more than 5 links between two websites (pages of two sites). In this way, we obtained experimental dataset in site-level. Totally, there are 114,529 websites and 115,480 links between sites. Our comparison includes three approaches:

1) *Slab* : As in [9], 2% is used as filtering threshold. After Slab filtering, we run PageRank on the filtered graph and the results rank is based on Pagerank value.

2) *TrustRank* : We choose top 10 hosts from original query results as seeds.

3) *Our approach*: We first detect and down-weight noisy links between websites as proposed in Section 4 and then run PageRank on the processed site-level graph to rank each site based on its PageRank value.

Table 8. Comparisons based on Precision for "office" and "fashion"

Dataset		office			fashion	
Algo.	Slab	Trust	Ours	Slab	Trust	Ours
P@3	0	0	0.333	0.333	0	0.667
P@5	0.2	0.2	0.4	0.6	0	0.8
P@10	0.5	0.4	0.4	0.4	0.4	0.7
P@15	0.467	0.333	0.533	0.4	0.533	0.533

Table 9. Average Comparisons based on Precision for twenty-one topics

Average	P@3	P@5	P@ 10	P@ 15
Slab	0.467	0.52	0.54	0.633
Ours	0.60	0.56	0.60	0.733
Trust	0.333	0.40	0.52	0.583

It could be seen from Table 8 that our approach is stable for different topics and outperforms Slab and TrustRank on site-level ranking. Even given very good seeds, performance of TrustRank is not stable for difference topics. Table 9 gives average precision comparisons on total 21 topics. It is very clear that our approach gives the most stable and best performance on top k ranking, which is usually the most important. We think the reason why our approach could beat slab at site-level is that slab could miss some spam pages and at the same time remove some high-quality results.

4.5.2 Spam Filtering

We also present comparisons between our approach and other two related works in terms of spam page filtering on UK2006. We have to note that not all spam pages are marked. We mainly concern about the rank of first spam page in search results and how many spam pages appeared in top K. Here, K is set to 800.

Table 10. Distribution of Spam pages in Top K ranking for UK2006

Rank Rank-	PageRank	Slab	TrustRank	Our App.
1-100	0	0	0	0
101-200	**4**	**2**	**1**	0
201-300	1	1		0
301-400	1	0	0	0
401-500	11	0	0	0
501-600	1	0	0	**1**
601-800	3	0	0	0

Table 10 presents ranking results from another angle: top K pages. It could be seen that spam pages appear among the top 100-200 in the above three approaches, while our approach can sink the rank of the first spam page to more than 500 at least. We believe this improvement is very remarkable since for web search, top k (k is usually 100 or 200) ranking is much more important.

5 Conclusion and Discussion

In this paper, we proposed using noisy link analysis to combat link spam and further improve web search ranking. Link spam is a very serious problem for web search. Our main idea is to identify “noisy links”, which was defined as “non-voting links”. “Non-voting link” is the main cause for the degeneration of the performance of link-based ranking. We proposed that if in-links of one page are mainly from some peculiar cluster or a few clusters, it is with very high probability that the page is spam page and in-links are noisy links. By noisy link analysis, we could effectively identify noisy links as well as noisy link support pages. Instead of removing all noisy links simply, we down-weighted them automatically after checking the effect of three possible methods. Comparison with two other related works TrustRank and SLAB are also conducted on two real datasets to evaluate the performance of our work. Experimental results indicate that our approach is very effective on spam page filtering for top k ranking. Since users are only interested in top k ranking, this improvement is very remarkable. We would like to further investigate the function of links in different situations. Moreover, how to exploit the functions of links effectively to support various web services is also another interesting research topic.

References

1. Gyongyi, Z., Garcia-Molina, H.: Web spam taxonomy. In: First International Workshop on Adversarial Information Retrieval on the Web (2005)
2. Sergey, B., Lawrence, P.: The anatomy of a large-scale hypertextual Web search engine. In: Proceeding of WWW 1998, pp. 107–117 (1998)
3. Kleinberg Jon, M.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
4. Zhou, B., Pei, J., Tang, Z.: A Spamicity Approach to Web Spam Detection. In: SIAM International Conference on Data Mining (2008)
5. Gyongyi, Z., Garcia-Molina, H.: Link spam alliances. In: Proceedings of the 31st International Conference on Very large Data Bases, pp. 517–528 (2005)
6. Collins, G.: Latest search engine spam techniques (August 2004), `http://www.itepoint.com/article/search-enginespam-techniques`
7. Wu, B., Goel, V., et al.: Propagating Trust and Distrust to demote Web Spam. In: Proceedings of MTW, a Workshop at the 15th International WWW Conference (2006)
8. Gyongyi, Z., Garcia-Molina, H., et al.: Web spam with TrustRank. In: Proceedings of the 30th International Conference on Very Large Data Bases, pp. 271–279 (2004)
9. Luiz da Costa Carvalho, A., et al.: Site Level Noise Removal for Search Engines. In: Proceedings of the 15th International Conference on World Wide Web, pp. 73—82 (2006)
10. Krishnan, V., Raj, R.: Web Spam Detection with Anti-Trust Rank. In: The 2nd International Workshop on Adversarial Information Retrieval on the Web, AIRWeb (2006)
11. Benczur, A.A., Csalogany, K., Sarlos, T., Uher, M.: Spamrank-fully automatic link spam detection. In: Proceedings of the First International Workshop on AIRWeb (2005)
12. Metaxas, P.T., DeStefano, J.: Web Spam, Propaganda and Trust. In: 1st International Workshop on Adversarial Information Retrieval on the Web (2005)

13. Wu, B., Davison, B.D.: Undue influence: eliminating the impact of link plagiarism on web search rankings. In: Proceedings of the 2006 ACM Symposium on Applied computing, pp. 1099–1104 (2006)
14. Ono, H., Toyoda, M., Kitsuregawa, M.: Identifying Web Spam by Densely Connected Sites and its Statistics in a JapaneseWeb Snapshot. In: Proceedings of the 22nd International Conference on Data Engineering Workshops (2006)
15. Zhang, H., Goel, A., Govindan, R., Mason, K., Roy, B.V.: Making eigenvector-based reputation systems robust to collusion. In: Third Workshop on Algorithms and Models for the Web Graph (2004)
16. Baeza-Yates, R., Castillo, C., Lopez, V.: Pagerank increase under different collusion topologies. In: First International Workshop on Adversarial Information Retrieval on the Web (2005)
17. Xiaoguang, Q., Lan, N., Brian, D.: Davison Measuring Similarity to Detect Qualified links. In: Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web, pp. 49–56 (2007)

High Dimensional Image Categorization

François Poulet[1] and Nguyen-Khang Pham[2]

[1] University of Rennes I - IRISA, Campus de Beaulieu,
35042 Rennes Cedex, France
francois.poulet@irisa.fr
[2] Cantho University, 1, Ly Tu Trong street, Cantho, Vietnam
pnkhang@cit.ctu.edu.vn

Abstract. We are interested in varying the vocabulary size in the image categorization task with a bag-of-visual-words to investigate its influence on the classification accuracy in two cases: in the first one, both the test-set and the training set contains the same objects (with only different view points in the test-set) and the second one where objects in the test-set do not appear at all in the training set (only other objects from the same category appear). In order to perform these tasks, we need to scale-up the algorithms used to deal with millions data points in hundred of thousand dimensions. We present k-means (used in the quantization step) and SVM (used in the classification step) algorithms extended to deal with very large datasets. These new incremental and parallel algorithms can be used on various distributed architectures, like multi-thread computer, cluster or GPU (graphics processing units). The efficiency of the approach is shown with the categorization of the 3D-Dataset from Savarese and Fei-Fei containing about 6700 images of 3D objects from 10 different classes. The obtained incremental and parallel SVM algorithm is several orders of magnitude faster than usual ones (like lib-SVM, SVM-perf or CB-SVM) and the incremental and parallel k-means is at least one order of magnitude faster than usual implementations.

Keywords: High dimensional classification, parallel algorithms, image categorization, GPU-based parallel algorithms.

1 Introduction

More and more images are stored in various databases, for example FlickR today has more than 4 billion images, it has been estimated that people having digital camera will take around 100 000 images during their whole life.

Image categorization is a very challenging task today. The most successful approach used in recent years is the bag-of-visual-words model [19] with local descriptors. The bag-of-visual-words comes from the text classification (or text categorization) area. In text categorization, we have a set of documents, each document containing a set of words. The bag of words model is the number of occurrence of each word in each document. Most of existing algorithms cannot deal easily with

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 465–476, 2010.

large number of words, that is why they use a pre-processing step to keep "interesting words". The vocabulary size is a major problem for this kind of approach. In the bag-of-visual-words, we first compute low level descriptors in particular points of image (for example SIFT descriptors [11]) and then a vector quantization is performed on these SIFTs (for example with a clustering algorithm like k-means [12], each cluster is then considered as a visual-word) to finally get the distribution of the SIFT of each image in the set of clusters obtained. Once this bag-of-visual-words is computed, machine learning algorithm is used to learn a model from this bag-of-visual words and predict the class of unlabeled image instances.

This is a high performance method for image categorization, but its main drawback is the computational cost of both parts of the method. The clustering algorithm has a complexity O(t.k.n), *k* is the number of clusters, *n* is the number of points and *t* is the number of iterations performed by the k-means algorithm. The learning algorithm can have a higher computational cost, for example if we use an SVM algorithm [23], the standard one requires solving a quadratic or linear program so the computational cost is at least O(n^2), *n* being the number of points and large number of dimensions (the number of clusters obtained from the k-means, we call it the vocabulary size) is often a problem.

In order to deal with very large image datasets, we need to scale-up both algorithms, the k-means and the SVM. Several solutions can be foreseen. Starting from a sequential software program or algorithm, the easiest way is to perform a SIMD (Single Instruction Multiple Data) parallelization. This can be achieved in several ways: on a cluster of processors, this can be a cluster of CPUs (the most usual case) or a cluster a GPU cores (i.e. a GPU card). The other solution is to use a grid system (like Grid 5000 [9]) to distribute both software program and data on a computer grid. We'll focus in this paper on the first two solutions.

The remainder of the paper is organized as follows: in section 2 we describe the two algorithms (k-means and SVM) and their extensions to deal with very large datasets, section 3 describes the large image dataset categorization and the results obtained by the algorithms before the conclusion and future work.

2 Scaling-Up Algorithms

2.1 Scaling-Up the k-Means Algorithm

2.1.1 The Original k-Means Algorithm

The k-means algorithm can be summarized as shown in Table 1. In each k-means iteration, we compute for each point, the distance from this point to each cluster center. The point is then allocated to the nearest cluster. Once each point has been treated, we compute the new cluster centers. This process is repeated until there is no more variation of the points from one cluster to another or a maximum number of iterations is reached.

Table 1. Pseudo-code of the k-means algorithm

```
Input: k (the cluster number), x (data-points)
for it = 0 to nb_it do
    for i = 0 to n do
        min_dist=d(x[i],c[0]); min_cluster=0
        for j = 1 to k-1 do
            dist=d(x[i], c[j])
            if (dist<min_dist) then
                min_dist = dist; min_cluster=j
            endif
        endfor
        assign x[i] to cluster min_cluster
    endfor
    compute the new cluster centers
endfor
```

For each point x_i we need to compute the distance to each cluster center c_j:

$$d(x_i, c_j)= \sqrt{}(x_i-c_j)^2 \quad (1)$$

In input of the algorithm we have X the data-point matrix of size n x d, C the cluster center matrix of size d x k and in output we have D the distance matrix of size n x k.

If we want to deal with very large datasets, we can suppose the input data point matrix cannot be loaded in memory and the distance matrix will need a very long time to be computed. We have to solve two problems: the space complexity problem to deal with very large datasets and the time complexity problem to get the result in a reasonable time.

2.1.2 Space Complexity Problem

To address the space complexity problem, the main idea is to split the whole dataset into B blocks of rows as shown in the figure 1. So we load successively the data blocks Xi into the main memory, we perform the multiplication Xi.C to get the corresponding block of distances Di and we repeat this process from X1 to XB.

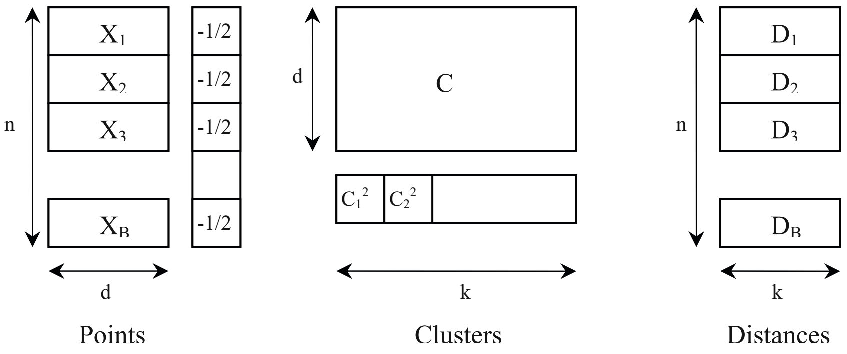


Fig. 1. Incremental k-means algorithm

With such an algorithm, we can deal with arbitrarily large datasets, whatever the dataset size is, we can split it into blocks that can be loaded into main memory to compute the corresponding distances.

2.1.3 Time Complexity Problem

We have seen in the previous sub-section we can deal with arbitrarily large datasets. Here we are interested in performing the computation in a reasonable running time. We have seen the k-means algorithm is an iterative one. We need the result of outer-loop it to perform the computation of the it+1 iteration. So the only way to perform efficient computation of the k-means algorithm is to parallelize the inner loop.

First of all, we have seen in subsection 2.1.2 we need to compute the distance between any data point x_i and each cluster center c_j (cf. equation 1). But this formulation is not the most appropriate, it is better to compute it with equation 2.

$$d(x_i, c_j)= \sqrt{}(x_i-c_j)^2 = \sqrt{}(x_i^2+c_j^2-2x_ic_j) . \quad (2)$$

As we are only interested in finding the minimum value, we can only compute for a given x_i, $\min(c_j^2-2x_ic_j)$ or $\max(x_ic_j-1/2c_j^2)$ it will be much more efficient than the initial formula in equation 1.

To get the result in a reasonable computing time, we need now to solve the time complexity problem. We need to compute the distance between each point and each cluster center. The computation of the distance between x_i and c_j is independent of the computation of the distance $x_{i'}$ and c_j. The computation of the distance between the points and the cluster c_j can be performed in parallel in a SIMD (Single Instruction Multiple Data) way. We have split the data into blocks, we can compute the distance of each row or sub-blocks of rows in a parallel explicit way.

To perform the matrix multiplication X.C, we use the BLAS library [1], for the CPU-based version and CUBLAS library for the GPU-based one. BLAS has a multi-thread capability and CUBLAS uses the whole set of available processors of the GPU. With this library, a implicit parallelization of the algorithm is performed. The main advantage is its ease of use and its high performance. We have checked during the evaluation process of the CPU-based algorithm, the CPU use is almost always 100%, so the library uses the full machine computation capability.

2.2 Scaling-Up the SVM Algorithm

2.2.1 The Original SVM Algorithm

In spite of the prominent properties of SVMs, current SVM algorithms cannot easily deal with very large datasets. A standard SVM algorithm requires solving a quadratic or linear program; so its computational cost is at least O(m2), where m is the number of training data points and the memory requirement of SVM frequently make it intractable. There is a need to scale up these learning algorithms for dealing with massive datasets. Efficient heuristic methods to improve SVM learning time divide the original quadratic program into series of small problems [2], [16]. Incremental learning methods [3], [5], [7], [8], [17], [21] improve memory performance for massive datasets by updating solutions in a growing training set without needing to load the entire dataset into memory at once. Parallel and distributed algorithms [7], [17] improve learning performance for large datasets by dividing the problem into components that are executed on large numbers of networked personal computers (PCs). Active learning algorithms [22] choose interesting data point subsets (active sets) to construct models, instead of using the whole dataset.

The starting point of our work is the LS-SVM classifiers proposed by [20]. Consider the linear binary classification task depicted in figure 2, with *m* data points x_i (i=1..m) in the n-dimensional input space R^n. It is represented by the [mxn] matrix *A*, having corresponding labels $y_i = \pm1$, denoted by the [mxm] diagonal matrix *D* of ±1 (where D[i,i] = 1 if x_i is in class +1 and D[i,i] = -1 if x_i is in class -1). For this problem, a SVM algorithm tries to find the best separating plane, i.e. the one farthest from both class +1 and class -1. Therefore, SVMs simultaneously maximize the distance between two parallel supporting planes for each class and minimize the errors.

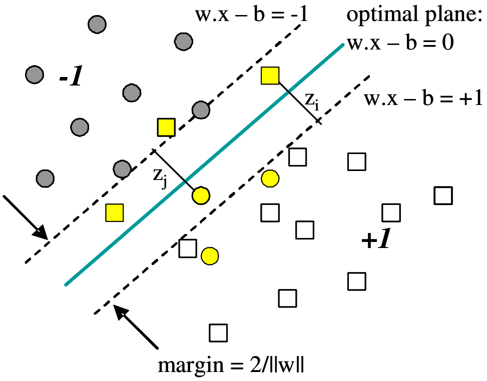


Fig. 2. Linear separation of the data points into two classes

For the linear binary classification task, classical SVMs pursue these goals with the quadratic program (3):

$$\min \Psi(w, b, z) = (1/2)\,\|w\|^2 + cz\,, \qquad \text{s.t.: } D(Aw - eb) + z \geq e\,, \tag{3}$$

where the slack variable $z \geq 0$ and the constant $c > 0$ is used to tune errors and margin size.

The plane (w, b) is obtained by the solution of the quadratic program (3). Then, the classification function of a new data point *x* based on the plane is: predict(x) = sign(w.x – b).

Unfortunately, the computational cost requirements of the SVM solutions in (3) are at least $O(m^2)$, where *m* is the number of training data points, making classical SVM intractable for large datasets. The LS-SVM proposed by Suykens and Vandewalle has used equality instead of the inequality constraints in the optimization problem (4) with a least squares 2-norm error into the objective function Ψ as follows:

- minimizing the errors by $(c/2)\|z\|^2$
- using the equality constraints D(Aw – eb) + z = e.

Thus substituting for *z* from the constraint in terms *w* and *b* into the objective function Ψ of the quadratic program (3), we get an unconstraint problem (4):

$$\min \Psi\ (\mathrm{w}, \mathrm{b}) = (1/2)\|\mathrm{w}\|^2 + (\mathrm{c}/2)\|\mathrm{e} - \mathrm{D}(\mathrm{Aw} - \mathrm{eb})\|^2 \tag{4}$$

In the optimal solution of (4), the gradient with respect to *w* and *b* will be 0. This yields the linear equation system of (n+1) variables (w_1, w_2, …, w_n, b) as follows:

$$\Psi'(\mathrm{w}) = \mathrm{cA}^\mathrm{T}(\mathrm{Aw} - \mathrm{eb} - \mathrm{De}) + \mathrm{w} = 0\ , \tag{5}$$

$$\Psi'(\mathrm{b}) = \mathrm{ce}^\mathrm{T}(\text{-}\mathrm{Aw} + \mathrm{eb} + \mathrm{De}) = 0\ , \tag{6}$$

(5) and (6) are rewritten by the linear equation system (7):

$$[w_1 w_2 w_3 ... w_n b]^T = \left(\frac{1}{c} I^o + E^T E\right)^{-1} E^T De\ , \tag{7}$$

where E = [A -e], I^o denotes the (n+1)x(n+1) diagonal matrix whose (n+1)th diagonal entry is zero and the other diagonal entries are 1.

The LS-SVM formulation (7) requires thus only the solution of linear equations of (n+1) variables (w_1, w_2, …, w_n, b) instead of the quadratic program (3).

To be able to deal with very large datasets, here again we need to solve the space and time complexity problems. We extend the LS-SVM algorithm into to ways to solve these problems.

2.2.2 Space Complexity Problem

To solve the space complexity problem, we use exactly the same incremental mechanism as for the k-means algorithm. We split the dataset into blocks of rows to fit into main memory. The linear equation system (7) can then be computed with sum on the row-based blocks (8):

$$[w_1 w_2 ... w_n b]^T = \left(\frac{1}{c} I^o + \sum_{i=1}^{k} E_i^T E_i\right)^{-1} \sum_{i=1}^{k} E_i^T D_i e_i\ , \tag{8}$$

E is almost the data point matrix, it is split in row blocks, D is the diagonal matrix with class of each data-point

Consequently, the incremental LS-SVM algorithm presented in table 2 can handle massive datasets on a PC. The accuracy of the incremental algorithm is exactly the same as the original one. Even if there are billions data points, the incremental LS-SVM algorithm is able to classify them on a simple PC. The algorithm only needs to store a small (n+1)x(n+1) matrix and two (n+1)x1 vectors in memory between two successive steps. The numerical test has shown the incremental LS-SVM algorithm can classify one billion data points in 20-dimensional input into two classes in 21 minutes and 10 seconds (except the time needed to read data from disk) on a PC (Pentium-IV 3 GHz, 512 MB RAM), we have solved the space complexity problem.

Table 2. Incremental LS-SVM algorithm

```
Input:
training dataset split in k blocks: A1, D1, ..., Ak, Dk
constant c to tune errors and margin size
Training:
init: $E^TE=0$, $d=E^TDe=0$
for i = 1 to k do
    load Ai and Di
    compute $E^TE=E^TE+E_i^TE_i$ and $d=d+d_i$ ($d_i=E_i^TD_ie_i$)
end for
solve the linear equation system (8)
get the optimal plane $(w,b)=w_1, w_2, ..., w_n, b$
Classify new data-point x according to f(x)=sign(w.x-b)
```

2.2.3 Time Complexity Problem

We have an incremental SVM algorithm able to deal with arbitrarily large datasets, now we are interested in performing the classification in a reasonable running time. We have developed different parallel versions of this algorithm, but the main idea is always the same, it is the parallel computation of the data blocks. It can be performed on CPU (to use on cluster for example) or on GPU. We give the details here of the fastest implementation we have developed. This implementation is based on GPU.

During the last decade, GPUs described in [24] have developed as highly specialized processors for the acceleration of raster graphics. The GPU has several advantages over CPU architectures for highly parallel, compute intensive workloads, including higher memory bandwidth, significantly higher floating-point, and thousands of hardware thread contexts with hundreds of parallel compute pipelines executing programs in a single instruction multiple data (SIMD) mode. The GPU can be an alternative to CPU clusters in high performance computing environments. Recent GPUs have added programmability and been used for general-purpose computation, i.e. non-graphics computation, including physics simulation, signal processing, computational geometry, database management, computational biology or data mining.

NVIDIA has introduced a new GPU, the GeForce GTX 280 and a C-language programming API called CUDA [14] (Compute Unified Device Architecture). A block diagram of the NVIDIA GeForce GTX 280 architecture is made of 3 groups of 10 multiprocessors. Each multiprocessor has 8 streaming processors for a total of 240. Each group of 8 streaming processors shares one L1 data cache. A streaming processor contains a scalar ALU (Arithmetic Logic Unit) and can perform floating point operations. Instructions are executed in SIMD mode. The NVIDIA GeForce GTX 280 has 1 GB of graphics memory, with a peak observed performance of 933 GFLOPS and 142 GB/s peak memory bandwidth. This specialized architecture can sufficiently meet the needs of many massively data-parallel computations. In addition, NVIDIA CUDA also provides a C-language API to program the GPU for general-purpose applications. In CUDA, the GPU is a device that can execute multiple concurrent threads. The CUDA software package includes a hardware driver, an API, its runtime and higher-level mathematical libraries of common usage, an implementation of Basic Linear Algebra Subprograms (CUBLAS [15]). The CUBLAS library allows access to the computational resources of NVIDIA GPUs. The basic model by which applications use the CUBLAS library is to create matrix and vector objects in GPU memory space, fill them with data, call a sequence of CUBLAS functions and finally, upload the results from GPU memory space back to the host. Furthermore, the data transfer rate between GPU and CPU memory is about 2 GB/s.

Thus, we developed a parallel version of incremental LS-SVM algorithm based on GPUs to gain high performance at low cost. The parallel incremental implementation in table 2 using the CUBLAS library performs matrix computations on the GPU massively parallel computing architecture. It can be used on any CUDA/CUBLAS compatible GPU (today more than 200 different ones, all from NVidia). Note that in CUDA/CUBLAS, the GPU can execute multiple concurrent threads. Therefore, parallel computations are done in an implicit way.

First, we split a large dataset A, D into small blocks of rows A_i, D_i. For each incremental step, a data block A_i, D_i is loaded into the CPU memory; a data transfer task copies A_i, D_i from CPU to GPU memory and then GPU computes the sums of $E_i^TE_i$ and $d_i = E_i^TD_ie_i$ in a parallel way. Finally, the results $E_i^TE_i$ and $d_i = E_i^TD_ie_i$ are uploaded from GPU memory space back to the CPU memory to solve the linear equation system (8). The accuracy of the new algorithm is exactly the same as the original one. More details about this implementation and results on large datasets can be found in [6]. To summarize we can say the execution time of linear LS-SVM is two order of magnitude faster on GPU than on CPU and three to four order of magnitude faster than standard SVMs like libSVM [4], CB-SVM [25] or SVM-Perf [10] with almost the same accuracy.

3 Large Image Dataset Categorization

Now, we have efficient algorithms to compute the k-means and perform the supervised classification with SVM, we will use these algorithms for high dimensional image datasets. Let us briefly summarize here the full process. The starting point is an image dataset. The one we will use in our experiment is from [18], it is made of 6675 images of objects from ten different classes. In each classe, there are almost ten different objects (for example, ten different cars) and each object is taken from various camera positions: 8 different view angles, 3 different scales and 3 different heights.

For each image of the dataset we compute interest points (hessian-affine [13]), then the SIFT (Scale Invariant Feature Transform) [11], to be partially invariant to some affine transformations. The resulting vectors are 128 dimensional float vectors, on the 3D-Dataset image database, we have obtained 5.791.568 SIFTs. In order to perform the supervised classification task, we need to separate the dataset into a training-set and a test-set. Being given the number of SIFTs obtained, a reasonable choice was to choose almost ten percent of the SIFTs for the test-set.

This has been done into two different ways:

- the first way was to simply randomly select 10% of the database SIFTs to form the test-set, in this case, we can assume, there are examples of all objects both in the training-set and in the test-set (with different view-points), the results are in the first two columns of table 3,

- the second way was to remove all the SIFTs corresponding to one object (whatever the view-point was) to form the test-set, so in each training-set all the views from one object are missing, the results are in the last three columns of table 3.

We have chosen the number of SIFTs in the test-set for the first way so that it is equal to the number of SIFTs in the second one and this value has no influence on the results.

Once the data are separated into training-set and test-set, we apply the k-means algorithm with 50 iterations and various cluster numbers to evaluate the influence of the bag-of-visual-words vocabulary size in the classification step with SVM (both with linear and RBF kernel).

Several conclusions can be drawn from these experiments: the first one is when both the training-set and the test-set contain samples of all objects (even with different view-points in the test-set) the larger vocabulary size, the better results. Another interesting fact is the linear classifier performs similarly as RBF one in this case. This means we can use very fast linear SVM classifier to perform the classification task. In these experiments we do not report the linear classifier execution time, its GPU running time is always less than 1 sec., most of the user time is for reading data and writing results (these values are highly dependent on the hardware and file-system used and are not valuable information here).

When the test-set contains unknown objects in the training-set, the results are different: from almost 2k to 50k (in the linear case) and 3k to 100k (in the RBF case) the accuracy is almost the same. So the vocabulary size has really less influence on the accuracy when some occurrences of the objects are not in the training set. The column "RBF no tuning" is only to show people who are not familiar with RBF classification with non linear kernels, they absolutely need to tune the SVM parameters to get high quality results usually reported with this kind of algorithm (the accuracy can go from 11% to 69% on the same dataset without or with parameter tuning).

Table 3. Accuracy results for SVM algorithms

Vocabulary size	Linear	RBF	Linear	RBF no tuning	RBF with tuning
1k			57.83%	51.88%	62.75%
2k	74.29%	76.36%	64.64%	66.67%	66.38%
3k	79.91%	78.02%	63.62%	68.12%	68.12%
4k	77.12%	78.65%	65.80%	68.55%	68.12%
5k	78.87%	79.10%	67.10%	68.70%	69.28%
6k	79.06%	79.28%	64.06%	67.54%	67.39%
7k	79.87%	80.67%	67.10%	67.39%	67.83%
8k	79.33%	79.96%	67.10%	67.25%	69.86%
9k	79.37%	79.87%	66.23%	66.81%	68.12%
10k	80.63%	80.90%	68.12%	66.23%	69.57%
20k	82.65%	82.29%	67.25%	57.97%	69.57%
25k	83.33%	82.97%	65.94%	49.71%	68.84%
50k	84.09%	83.96%	65.65%	20.89%	73.91%
100k	84.81%	84.76%	57.10%	11.30%	68.84%
200k	85.39%	85.75%	53.62%	14.49%	66.96%
250k			50%	15.07%	65.07%
500k			40%	15.51%	61.45%

These first results are about accuracy in the classification task, about the running time, we have already said the incremental and parallel linear SVM runs in less than 1sec, for the RBF classification, we have used libSVM, it requires between 20 and 30 min for each step of the parameter tuning process (so it needs to be multiplied by the number of parameter values to be tuned, in our case between 400 and 600 different values so about 12,000 mn or 200 hours). But the main bottleneck is the k-means algorithm. 1k vocabulary size needs 6mn to be computed on two GPUs GTX-280, 10k needs 37 mn and 100k requires more than 6 hours. On CPU, even with a 16 cores PC with 100GB RAM the running time for computing 500k vocabulary is more than one week (user time) or 16 weeks or four months (cpu time). This means our choice was not the good one for the implementation of the parallelized version of the k-means, it was only the easiest one to implement. The good choice would have been to develop a version for larger cluster or grid systems.

Furthermore, the k-means is the clustering algorithm usually used in image classification, but it is known to have several drawbacks, the first one is its sensibility to the initialization step, but here we cannot run several executions of the algorithm, another one is it can only find convex clusters and we cannot assume SIFT clusters are convex in image categorization...

4 Conclusion and Future Work

Before to conclude, some words about the results reported in [18], their average accuracy was 36.9% with four objects out of the training set, so it is difficult to compare with our results. These results are only obtained on one dataset, they need to be shown on more than this.

We have applied the same incremental and parallel method for two different algorithms, it was very efficient for the SVM algorithm (3 or 4 order of magnitude faster than state of the art algorithms like libSVM, CB-SVM or SVM-Perf) and much less efficient for the k-means one. The SVM algorithm only needs a "simple" matrix multiplication to be solved while the K-means needs several ones to be solved one by one in an iterative way to get its final result. This last one is much more difficult to efficiently be parallelized. In this case, a good choice could be to choose the most available possible cpu power. The size of the dataset used with SVM is far from the maximum the algorithm can deal with.

The k-means algorithm is the most frequently used in image classification despite its drawbacks. It may be because it is simple to use and implement and has a quite low complexity compared to other algorithms, so another clustering algorithm could give better results, we will investigate this possibility.

The results we have obtained are really different according to the type of test-set we used. When the test-set is made of random images with all object types, the accuracy is rather a good one, but when one object is not in the training set, the accuracy is much lower. May be the SIFT descriptor is efficient for finding occurrences of the same object with various view points (it has been designed for this) but not so efficient to find different objects…

Finally the SVM speed is good enough (cpu time is less than one second) an obvious improvement will be to use binary files in input/output of the algorithm to reduce the user computation time.

References

1. Blackford, L.S., Demmel, J., Dongarra, J., Duff, I., Hammarling, S., Henry, G., Heroux, M., Kaufman, L., Lumsdaine, A., Petitet, A., Pozo, R., Remington, K., Whaley, R.C.: An Updated Set of Basic Linear Algebra Subprograms (BLAS). ACM Trans. Math. Soft. 28(2), 135–151 (2002)
2. Boser, B., Guyon, I., Vapnik, V.: A Training Algorithm for Optimal Margin Classifiers. In: 5th ACM Annual Workshop on Computational Learning Theory, Pittsburgh, Pennsylvania, pp. 144–152 (1992)
3. Cauwenberghs, G., Poggio, T.: Incremental and Decremental Support Vector Machine Learning. In: Advances in Neural Information Processing Systems, vol. 13, pp. 409–415. MIT Press, Cambridge (2001)
4. Chang, C.C., Lin, C.J.: LIBSVM: a Library for Support Vector Machines (2001), Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
5. Do, T.N., Fekete, J.D.: Large Scale Classification with Support Vector Machine Algorithms. In: 6th International Conference on Machine Learning and Applications, ICMLA 2007, pp. 7–12. IEEE Press, Ohio (2007)
6. Do, T.N., Pham, N.K., Poulet, F.: GPU-based Parallel SVM Algorithm. Journal of Frontiers of Computer Science and Technoloy 3(4), 368–377 (2009)
7. Do, T.N., Poulet, F.: Classifying one Billion Data with a New Distributed SVM Algorithm. In: 4th IEEE International Conference on Computer Science, Research, Innovation and Vision for the Future, RIVF 2006, Ho Chi Minh, Vietnam, pp. 59–66 (2006)
8. Fung, G., Mangasarian, O.: Incremental Support Vector Machine Classification. In: The 2nd SIAM Int. Conf. on Data Mining, SDM 2002, Arlington, Virginia, USA (2002)
9. Grid5000, `https://www.grid5000.fr/mediawiki/index.php/Grid5000:Home`
10. Joachims, T.: A Support Vector Method for Multivariate Performance Measures. In: The International Conference on Machine Learning, ICML (2005)
11. Lowe, D.: Object Recognition from Local Scale-Invariant Features. In: The 7th International Conference on Computer Vision, ICCV 1999, vol. 2, pp. 1150–1157 (1999)
12. McQueen, J.: Some Methods for classification and Analysis of Multivariate Observations. In: The 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press, Berkeley (1967)
13. Mikolajczyk, K., Schmid, C.: Scale and Affine invariant interest point detectors. International Journal of Computer Vision 60(1), 63–86 (2004)
14. NVIDIA® CUDA^{TM}, CUDA Programming Guide 1.1 (2007)
15. NVIDIA® CUDA^{TM}, CUDA CUBLAS Library 1.1 (2007)
16. Platt, J.: Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In: Advances in Kernel Methods – Support Vector Learning, pp. 185–208 (1999)
17. Poulet, F., Do, T.N.: Mining Very Large Datasets with Support Vector Machine Algorithms. In: Enterprise Information Systems, pp. 177–184. Kluwer Academic Publishers, Dordrecht (2004)
18. Savarese, S.L.: Fei-Fei.:3D generic object categorization, localization and pose estimation. In: International Conference on Computer Vision (2007)
19. Sivic, J., Zisserman, A.: Video Google: A Text Retrieval Approach to Object Matching in Videos. In: The International Conference on Computer Vision, pp. 1470–1477 (2003)

20. Suykens, J., Vandewalle, J.: Least Squares Support Vector Machines Classifiers. Neural Processing Letters 9(3), 293–300 (1999)
21. Syed, N., Liu, H., Sung, K.: Incremental Learning with Support Vector Machines. In: The Workshop on Support Vector Machines at the International Joint Conference on Artificial Intelligence, Stockholm, Sweden (1999)
22. Tong, S., Koller, D.: Support Vector Machine Active Learning with Applications to Text Classification. In: ICML 2000, The 17th Int. Conf. on Machine Learning, Stanford, USA, pp. 999–1006 (2000)
23. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
24. Wasson, S.: Nvidia's GeForce 8800 graphics processor. Technical report, PC Hardware Explored (2006)
25. Yu, H., Yang, J., Han, J.:Classifying Large Data Sets Using SVM with Hierarchical Clusters.In: The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 306–315(2003)

Efficiently Mining Co-Location Rules on Interval Data

Lizhen Wang, Hongmei Chen*, Lihong Zhao, and Lihua Zhou

Department of Computer Science and Engineering,
School of Information Science and Engineering, Yunnan University,
Kunming 650091, China
hmchen@ynu.edu.cn

Abstract. Spatial co-location rules represent subsets of spatial features whose instances are frequently located together. This paper studies co-location rule mining on interval data and achieves the following goals: 1) defining the semantic proximity between instances, getting fuzzy equivalent classes of instances and grouping instances in a fuzzy equivalent class into a semantic proximity neighborhood, so that the proximity neighborhood on interval data can be rapidly computed and adjusted; 2) defining new related concepts with co-location rules based on the semantic proximity neighborhood; 3) designing an algorithm to mine the above co-location rules efficiently; 4) verifying the efficiency of the method by experiments on synthetic datasets and the plant dataset of "Three Parallel Rivers of Yunnan Protected Areas".

Keywords: Spatial data mining, Co-location rule, Interval data, Fuzzy equivalent class, Semantic proximity neighborhood.

1 Introduction

Spatial co-location rule mining is an important spatial data mining task. Co-location patterns represent subsets of spatial features whose instances are frequently located together. Spatial features describe object types. Examples of them include plant species, animal species, business types, mobile service requests, disease, crime, climate, etc. Instances describe the presence or absence of spatial features at different locations in a two-dimensional or three-dimensional space. Co-location rules can yield important insights for many applications. For example, symbiotic plant species "Picea Brachytyla", "Picea Likiangensis" and "Tsuga Dumosa" frequently grow together in the alpine terrain of the "Three Parallel Rivers of Yunnan Protected Areas". Co-location rule "Picea Brachytyla → Picea Likiangensis and Tsuga Dumosa" can be obtained and applied to predict that "Picea Likiangensis" and "Tsuga Dumosa" may appear in the areas where "Picea Brachytyla" presents.

In real life, values of variables are often not precise, due to missing information or ill-defined concepts. Interval data can be used to represent values of quantities in uncertain situations [1, 2]. For example, meteorological data, such as daily temperatures and rainfall densities, are registered as minimal and maximum values. As

[*] Corresponding author.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 477–488, 2010.

statistical characters of uncertain variables, interval data are also widely used in economics, engineering and many other real-world applications.

The locations of instances, such as plants, have often been expressed as interval data as well. Since they occur in tufts, the location of a tuft of plant is not a point but an area which can be described by two or three interval data corresponding to a two-dimensional or three-dimensional space.

How can we mine co-location rules efficiently on interval data?

First, we define the semantic proximity between instances based on a conceptual hierarchy tree of locations described by interval data, and so get fuzzy equivalent classes of instances based on the fuzzy similarity matrix of the semantic proximity as well as a threshold by fuzzy equivalence partition. This groups instances in a fuzzy equivalent class into a semantic proximity neighborhood. Then, the proximity neighborhood over interval data can be rapidly computed and adjusted by changing the threshold when adopting the fuzzy equivalence partition method in [15].

Second, based on the semantic proximity neighborhood, we define new related concepts concerning co-location rules including co-location, row instance, table instance, conditional probability, and participation index.

Third, we use an algorithm for mining the above co-location rules efficiently. In the algorithm we have designed, we mainly discuss the sort-merge join method that generates table instances of candidate co-locations, and the prefix-based method that decomposes a large spatial feature set into smaller pieces, which can be handled independently.

The rest of the paper is organized as follows: Section 2 introduces related works. Section 3 presents basic concepts. In Section 4, the algorithm for efficiently mining co-location rules on interval data is given. Experimental evaluations on a synthetic dataset and on the plant dataset of “Three Parallel Rivers of Yunnan Protected Areas” are reported in Section 5. Finally, we summarize our work and discuss future directions in Section 6.

2 Related Work

Morimoto [3] defined distance-based patterns as k-neighboring class sets. The number of instances in a pattern was used as a prevalence measure, which does not possess any anti-monotone properties. However, Morimoto used a non-overlapping instance constraint to get the anti-monotone property for this measure. Shekhar and Huang [4] developed an event centric model, which does away with the non-overlapping instance constraint. Also, a new prevalence measure called the participation index was defined. This measure possesses the desirable anti-monotone property. At the same time, Huang, Shekhar and Xiong [5] proposed a general approach for mining co-location rules, the join-based approach. It works well on sparse datasets, but it is inefficient when dealing with dense datasets, because the computation time of join grows as the table instances and co-location rules get larger. To alleviate this problem, Yoo and Shekhar [6, 7] proposed two improved algorithms, the partial-join approach and the join-less approach. Wang [8] also presented a new join-less algorithm based on the CPI-tree.

Huang, Pei and Xiong [9] addressed co-location pattern mining on rare datasets. A new measure called the maximal participation ratio was introduced and a weak monotone property was identified. The concept and method of mining maximal co-location patterns were presented in [10]. It proposed a novel order-clique-based approach that mines maximal co-location patterns without storing excessive table instances.

Differing from the research mentioned above, this paper discusses discovering co-location rules from interval data sets.

3 Basic Concepts

Fig. 1 gives an example to illustrate basic concepts. In this figure, an instance is uniquely identified by *e.i*, where *e* is a spatial feature and *i* is the id of an instance of *e*. For example, A.2 represents the second instance of spatial feature A. Notice that the location of an instance is not a point but an area.

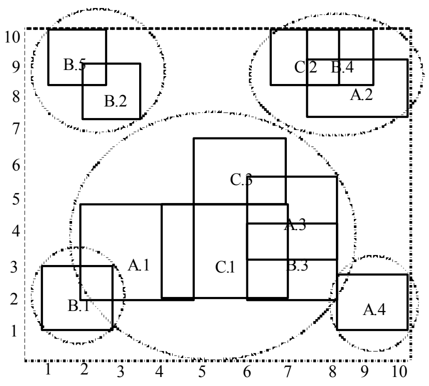


Fig. 1. An example of instances of spatial features

Table 1. An example of instances of spatial features (Instances ordered by spatial features and id)

instance-id	spatial feature	location
1	A	([2-5], [2-5])
2	A	([7-10], [7-9])
3	A	([6-8], [3-6])
4	A	([8-10],[1-3])
1	B	([1-3], [1-3])
2	B	([2-4],[7-9])
3	B	([6-8], [2-4])
4	B	([7-9] ,[8-10])
5	B	([1-3], [8-10])
1	C	([4-7], [2-5])
2	C	([6-8],[8-10])
3	C	([5-7], [5-7])

An instance can be described as a vector <*instance-id*, *spatial feature*, *location*>, where the location of an instance can be described by two interval data. Then the data of Fig. 1 can be stored in Table 1.

3.1 Fuzzy Equivalent Classes of Instances

To efficiently handle locations described by interval data, we compute fuzzy equivalent classes of instances.

First, we construct a conceptual hierarchy tree of locations according to the domain knowledge (shown in Fig. 2), and define the semantic proximity between instances based on it.

The conceptual hierarchy trees can be obtained from domain experts or using a method based on entropy [11].

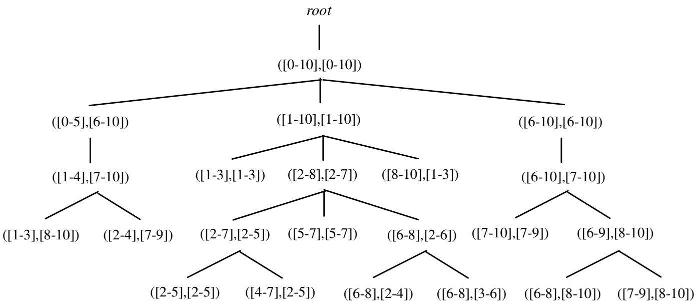


Fig. 2. The concept hierarchy tree of locations in Table 1

Definition 1. The depth $d(nd)$ of a node nd in a concept hierarchy tree T is the length of the path from the root to nd. The height $h(nd)$ is the length of the longest path from nd to leaves in the sub-tree whose root is nd. The height $h(T)$ of a concept hierarchy tree T is the height of the root. The level $l(nd)$ is given by $l(nd)= h(T) - d(nd)$.

Definition 2. The *semantic proximity* $sp(i_1, i_2)$ between two instances i_1 and i_2 can be defined as

$$sp(i_1,i_2)=\begin{cases}1 & i_1=i_2\\ 1-l(p(i_1.location,i_2.location))/h(T) & i_1\neq i_2\end{cases} \tag{1}$$

where $p(i_1.location,\ i_2.location)$ returns the nearest common father of $i_1.location$, $i_2.loaction$, $0\leq sp(i_1, i_2)\leq 1$.

As we know all semantic proximities between all instances, we can form a fuzzy similarity matrix utilizing Definition 2.

At the second stage we obtain fuzzy equivalent classes of instances based on the fuzzy similarity matrix and a threshold λ by fuzzy equivalence partition.

There are several fuzzy equivalence partition methods. A traditional method computes the fuzzy equivalence matrix by self-multiplication of the fuzzy similarity matrix repeatedly. This is completed in $O(n^3)$ [11, 12]. Using an extension of Tarjan's set union algorithm, the fuzzy equivalence matrix can be computed in $O(n^2\log_2 n)$ [13, 14]. Using a binary partition tree, the fuzzy equivalence matrix can be computed in $O(m^2)$, where m is the number of nonzero elements in the upper-triangle of the fuzzy similarity matrix [11]. The fuzzy equivalence matrix can be computed in $O(n^2)$ by constructing a maximum weighted spanning tree [15]. This last method is adopted in this paper.

After getting the maximum weighted spanning tree, we can easily get fuzzy equivalent classes of instances by deleting edges whose weights are smaller than a threshold λ.

The maximum weighted spanning tree is shown in Fig. 3. When the threshold λ is 0.5, fuzzy equivalent classes of instances are circled by dashed lines in Fig. 3 and Fig. 1.

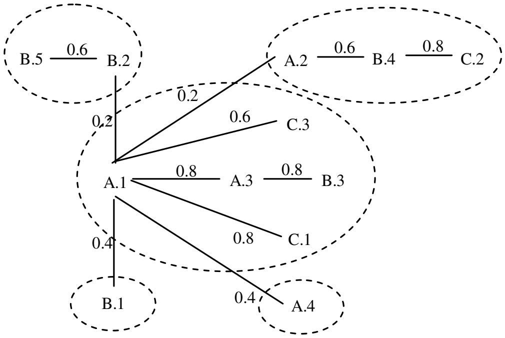


Fig. 3. Fuzzy equivalent classes of instances in Fig. 1

3.2 Co-location Rules Based on Fuzzy Equivalent Classes

Definition 3. Given an instance set *I* of spatial feature set *E*, an instance set $I' \subseteq I$ is a *semantic proximity neighborhood* if it is a subset of a fuzzy equivalent class.

Definition 4. A *co-location c* is a subset of spatial feature set *E*. A *co-location rule* is a rule $c_1 \rightarrow c_2(p, cp)$, where c_1 and c_2 are disjoint co-locations, *p* is the prevalence measure, and *cp* is the conditional probability.

Definition 5. A *semantic proximity neighborhood* I' is a *row instance* (denoted by row-instance (*c*)) of a co-location *c* if I' contains instances of all spatial features in *c* and no proper subset of $I^{'}$ does so. A *table instance*, table-instance (*c*), of a co-location *c* is the collection of all row instances of *c*.

In Fig. 1, the semantic proximity neighborhood {A.3, B.3} is a row instance of the co-location {A, B}; {A.3, C.1, C.3} is not a row instance of {A, C} because its subset {A.3, C.1} or {A.3, C.3} contains instances of all spatial features in {A, C}; the table instance of the co-location {B, C} has 3 row instances {B.3, C.1}, {B.3, C.3} and {B.4, C.2}.

Fuzzy equivalent classes of instances can be rapidly adjusted by changing the threshold λ after getting the maximum weighted spanning tree. Thus the semantic proximity neighborhood can also be rapidly adjusted and the cost of generating the table instances in traditional co-location mining is reduced because of the fuzzy equivalence partition.

Definition 6. The *conditional probability* $cp(c_1 \rightarrow c_2)$ of a co-location rule $c_1 \rightarrow c_2$ is the probability of finding an instance of c_2 in the semantic proximity neighborhood of an instance of c_1. It is estimated as

$$\frac{\left|\pi_{c_1}(\text{table}-\text{instance}(c_1 \cup c_2))\right|}{\left|\text{table}-\text{instance}(c_1)\right|} \tag{2}$$

where π is the relational projection operation.

Definition 7. The *participation index pi*(*c*) is used as the prevalence measure of a co-location *c*. It is defined as

$$\min_{e_i \in c}\{pr(c, e_i)\} \tag{3}$$

where *pr*(*c*, *e*$_i$), called the ***participation ratio*** of e_i in *c*, is the fraction of instances of feature e_i which participate in any instance of *c*, and is estimated as

$$\frac{|\pi_{e_i}(\text{table} - \text{instance}(c))|}{|\text{table} - \text{instance}(\{e_i\})|} \tag{4}$$

In Fig. 1, for the co-location *c*={B, C}, *pi*(*c*) =min{*pr*(*c*, B), *pr*(*c*, C)}=min{2/5, 3/3}=2/5.

In co-location mining, we are interested in co-location rules whose participation index and conditional probability are higher than a minimum prevalence threshold and a minimum conditional probability threshold. We observe that the participation ratio and the participation index are monotonic. So, the participation index can be used to effectively prune the search space.

4. An Algorithm for Mining Co-location Rules

```
Algorithm: mining co-location rules
Input: a set $E$ of $K$ spatial features, a set $I$ of $n$ instances, a fuzzy equivalence partition threshold $\lambda$, a minimum prevalence threshold $min\_prev$, and a minimum conditional probability threshold $min\_cond\_prob$
Output: co-location rules
Steps:

1) Computing the semantic proximity between instances in $I$ based on the conceptual hierarchy tree of locations, getting the fuzzy similarity matrix;
2) Getting fuzzy equivalent classes $EPC$ based on the fuzzy similarity matrix and $\lambda$;
3) $k$:=1; $C_1$:=$E$; $P_1$:=$E$;
4) $T_1$=gen_table-instance ($C_1$, $EPC$);
5) While (($P_k$ is not empty) and ($k<K$)) do {
6) $C_{k+1}$=gen_candidate_co-location ($P_k$);
7) $T_{k+1}$=gen_table_instance ($C_{k+1}$, $T_k$);
8) $P_{k+1}$=select_prevalent_co-location($min\_prev$, $C_{k+1}$, $T_{k+1}$);
9) $R_{k+1}$=gen_co-location_rule($min\_cond\_prob$, $P_{k+1}$, $T_{k+1}$);
10) $k$:=$k$+1; }
11) Return $R_2 \cup \cdots \cup R_{K+1}$;
```

We note that C_1 and P_1, the set of candidate co-locations and the set of prevalent co-locations whose size is 1, are initialized to *E* because the participation index is 1 for all co-locations whose size is 1. T_1 is created by ordering instances by the class-id and the instance-id. An example is shown in Fig. 4. The While loop iteratively performs four basic tasks: generating candidate co-locations whose size are all *k*, generating table instances of candidate co-locations, selecting prevalent co-locations, and generating co-location rules.

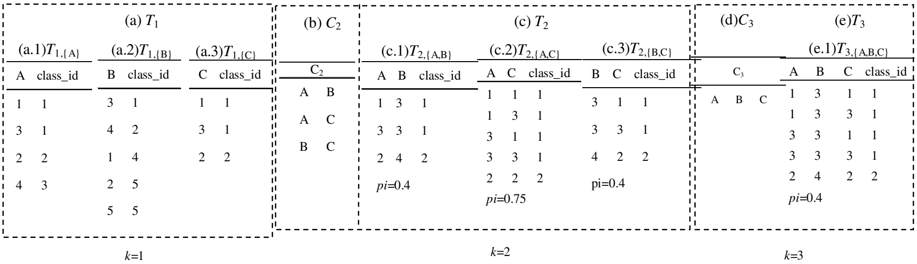
4.1 Sort-Merge Join Method

In the Algorithm, the key step is to generate table instances of candidate co-locations. We adopt the sort-merge join method that is similar to the Apriori algorithm. It can be described as the following join query.

```
Foreach $c \in C_{k+1}$
    Insert into $T_{k+1,c}$ /*$T_{k+1,c}$ is the table instance of $c$ */
    Select $p.e_1$, … , $p.e_{k-1}$, $p.e_k$, $q.e_k$, $p.class\_id$
    From $T_{k,c-\{e_{k+1}\}}$ $p$, $T_{k,c-\{e_k\}}$ $q$
    Where $p.e_1$=$q.e_1$, … , $p.e_{k-1}$=$q.e_{k-1}$, $p.class\_id$=$q.class\_id$
End
```

The sort-merge join method can be well performed because row instances are ordered based on the class-id and instance-id, and the order can be kept in the next iteration.

Fig. 4 gives main mining steps of the example in Fig. 1. {1, 3, 1} in (c.1) and {1, 1, 1} in (c.2) are joined to generate row instance {1, 3, 1, 1} of co-location {A, B, C} in (e.1). Because A.1 and B.1 do not belong to the same fuzzy equivalent class, {1, 1} in (a.1) and {1, 4} in (a.2) they fail to generate a row instance of co-location {A, B}.



(a) T_1

(a.1)$T_{1,\{A\}}$

A	class_id
1	1
3	1
2	2
4	3

(a.2)$T_{1,\{B\}}$

B	class_id
3	1
4	2
1	4
2	5
5	5

(a.3)$T_{1,\{C\}}$

C	class_id
1	1
3	1
2	2

k=1

(b) C_2

C_2	
A	B
A	C
B	C

(c) T_2

(c.1)$T_{2,\{A,B\}}$

A	B	class_id
1	3	1
3	3	1
2	4	2

pi=0.4

(c.2)$T_{2,\{A,C\}}$

A	C	class_id
1	1	1
1	3	1
3	1	1
3	3	1
2	2	2

pi=0.75

(c.3)$T_{2,\{B,C\}}$

B	C	class_id
3	1	1
3	3	1
4	2	2

pi=0.4

k=2

(d)C_3

C_3		
A	B	C

(e)T_3

(e.1)$T_{3,\{A,B,C\}}$

A	B	C	class_id
1	3	1	1
1	3	3	1
3	3	1	1
3	3	3	1
2	4	2	2

pi=0.4

k=3

Fig. 4. An Example of the co-location rule mining algorithm

4.2 Prefix-Based Method

When the number of spatial features is too large to be handled in a limited main-memory, can the spatial feature set be decomposed into smaller pieces so that each can be handled independently in a limited main-memory? The prefix-based method is presented to solve this problem.

The power set P(E) of the spatial feature set E is a complete lattice [16]. Fig. 5 shows the lattice of the power set P({A, B, C, D}).

First, we define a function f to get the k length prefix of a spatial feature subset X:

$$f : \mathrm{P}(E) \times \mathrm{N} \rightarrow \mathrm{P}(E) \tag{5}$$

i.e., $f(X,k)=X[1{:}k]$.

Second, we define a relation θ_k on P(E):

$$\forall X,Y \in \mathrm{P}(E), X \equiv_{\theta_k} Y \Leftrightarrow f(X,k)=f(Y,k) \tag{6}$$

That is, two spatial feature subsets satisfy θ_k if their k length prefixes are same. We therefore call θ_k a prefix-based relation.

In fact, θ_k is an equivalence relation because it is a reflexive, symmetric and transitive relation. So, θ_k can partition P(E) into equivalent classes. The equivalent class of a $X \in P(E)$ is denoted by $[X]_k = \{Y \in P(E) \mid X \equiv_{\theta_k} Y\}$.

Fig. 5 shows equivalent classes induced by θ_1 on P({A, B, C, D}). The set of equivalent classes is {[A]$_1$, [B]$_1$, [C]$_1$, [D]$_1$,[{}]$_1$}.

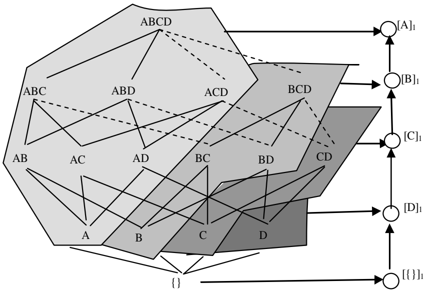


Fig. 5. The lattice of the power set P({A, B, C, D}), and equivalence classes induced by θ_1

Lemma 1. Each equivalent class [X]$_k$ induced by the equivalence relation θ_k is a sub-lattice of the lattice of the power set P(E).

Proof. Let $U,V \in [X]_k$,

$\because$ $f(U \vee V,k) = f(U \cup V,k) = f(X,k)$, $f(U \wedge V,k) = f(U \cap V,k) = f(X,k)$

$\therefore$ $U \vee V \in [X]_k$, $U \wedge V \in [X]_k$

Therefore, [X]$_k$ is a sub-lattice of the lattice of P(E). □

[X]$_k$ can be independently handled because the table instances can be independently generated in [X]$_k$ according to the sort-merge join method. In Fig. 5, real links denote the join operation for generating table instances in [X]$_1$.

In the prefix-based method, we recursively partition spatial features into small equivalent classes by θ_1, θ_2, … until each class is small enough to be handled in a limited main-memory. To effectively prune the search space, we have to process the equivalent classes from bottom to top, corresponding to the reverse lexicographic order. For example, in Fig. 5, the order of processing is [D]$_1$, [C]$_1$, [B]$_1$, and [A]$_1$.

5 Experimental Evaluation

In this section, experiments on synthetic datasets and the plant dataset of "Three Parallel Rivers of Yunnan Protected Areas" will be evaluated. Synthetic datasets are generated by applying a method similar to that in [5]. The plant dataset contains distribution information of plant species in the “Three Parallel Rivers of Yunnan Protected Areas”.

Experiments were performed on a Celeron computer with a 2.40 GHz CPU and 376 Mbytes memory running the Windows XP operating system. All programs are written in Java.

5.1 Performance Study

Computing fuzzy equivalent classes of instances is an important step in the Algorithm. Therefore, two methods are evaluated in the performance study. One uses the traditional method denoted as TCG (Traditional Co-location Generation). Another uses the maximum weighted spanning tree method denoted as OCG (Optimized Co-location Generation).

Parameters used in experiments are as follows: the spatial framework is 250×250, the number of spatial features is 10, and the location of an instance is described by two interval data whose average area is 25×25.

Effect of data density in the spatial framework: The effect of data density in the spatial framework was evaluated on synthetic datasets in which the number of instances of a spatial feature varies from 100 to 700 stepped by 200. Fig. 6 shows the execution time of TCG and OCG when the fuzzy equivalence partition threshold λ and the minimum prevalence threshold *min_prev* are 0.1. As data density increases, the execution time dramatically increases. For further investigation, Fig. 7 shows a comparison between the fuzzy equivalence partition step and the other steps in the algorithm. The fuzzy equivalent partition step appears costly.

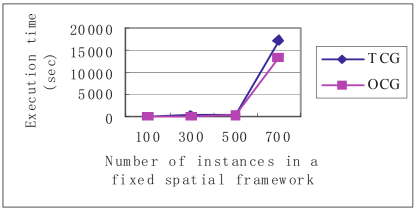


Fig. 6. Effect of data density

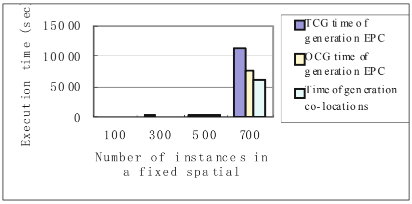
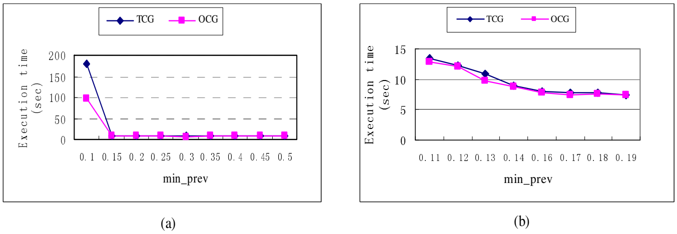


Fig. 7. Comparison between the fuzzy equivalence partition step and the other step

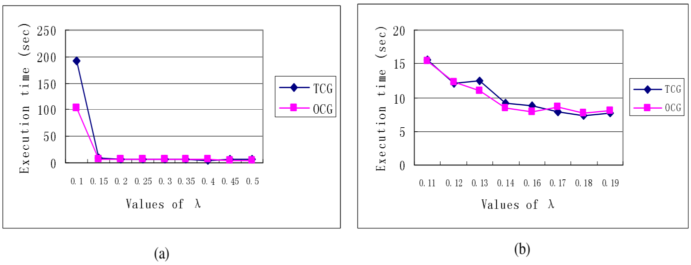
Effect of the minimum prevalence threshold *min-prev*: The effect, as *min-prev* increases, is given in Fig. 8. The experiment is conducted on the above dataset in which the number of instances of a spatial feature is 300, and λ is 0.2. The effect of *min-prev* for TCG and OCG is almost same. However, we can see the execution time decreases with the increase of *min-prev* in Fig. 8 (b) which is an enlargement of part of Fig. 8 (a). The cause is the decrease in the number of joins of instances due to the efficient pruning. In addition, the execution time is much less than the first time because the fuzzy similarity matrix is computed only once in algorithms.



(a) (b)

Fig. 8. Effect of the minimum prevalence threshold

Effect of the fuzzy equivalence partition threshold λ : The effect of changing λ is evaluated on the synthetic dataset in which the number of instances of a spatial feature is 300, and *Min-prev* is 0.2. Fig. 9 shows the execution time when changing λ. Analysis for Fig. 9 (a) and (b) is similar to Fig. 8 (a) and (b).



(a) (b)

Fig. 9. Effect of the fuzzy equivalence partition threshold

5.2 Experiments on the Plant Dataset

The Algorithm is evaluated on the plant dataset of “Three Parallel Rivers of Yunnan Protected Areas”. The number of plant species (spatial features) is 29. The number of instances is 3908. When the fuzzy equivalence partition threshold λ is 0.09, the number of fuzzy equivalent classes is 252. When the minimum prevalence threshold *Min_prev* is 0.1, the maximum size of co-locations is 9. Some co-location rules are shown in Table 2. A few plant species, corresponding to the plant ids in Table 2, are listed in table 3. Plant experts have verified that these rules are helpful.

Table 2. Some co-location rules on the plant dataset

Rule_id	The left-hand side of the rules	The right-hand side of the rules	Cond_Prob
1	(1)	(3)(7)(14)(15)	0.92
2	(1)(7)	(3)(14)(15)	0.95
3	(2)	(4)(5)(9)	0.86
4	(2)(3)	(6)(8)(10)(11)(12)(13)	0.96
…	…	…	…

Table 3. Plant species corresponding to plant ids

Plant_id	Plant species
(1)	Kobresia tunicata
(2)	Kobresia stiebritziana
(3)	Primula serratifolia
(4)	Vertrilla baillonii
…	…

In the experiments on the plant dataset, we observed that the semantic proximity neighborhood based on fuzzy equivalent classes of instances can be a plant community even if λ is very low. The key to making these discoveries is to be able to easily alter the fuzzy equivalent partition threshold λ. In traditional co-location mining, the proximity neighborhood must be recomputed when the distance threshold is changed, making such discoveries harder and the results more difficult to understand.

6 Conclusions and Future Work

This paper discusses co-location rule mining on interval data. Based on a semantic proximity neighborhood defined by fuzzy equivalent classes of instances, new related concepts with co-location rules are presented. The semantic proximity neighborhood on interval data can be rapidly computed and adjusted due to a fuzzy equivalence partition. An algorithm for efficiently mining co-location rules is given. The algorithm uses the sort-merge join method to generate table instances of candidate co-locations, and a prefix-based method to decompose a large spatial feature set into smaller pieces that can be handled independently in a limited main-memory. The paper also verifies that this method is efficient by experiments on synthetic datasets and the plant dataset of "Three Parallel Rivers of Yunnan Protected Areas".

In the future, we will study ways to obtain the fuzzy semantic proximity neighborhood and the fuzzy participation index from the fuzzy similarity matrix. We will also explore co-location mining on other uncertain data. Furthermore, we plan to collaborate with domain experts to further investigate co-location rules found in our experiments.

Acknowledgments. Authors would like to thank Yun Xiong and Yuzhen Bao for their help in the experiments. This research is supported by the National Natural Science Foundation of China (No. 61063008), the Research Foundation of the Educational Department of Yunnan Province (No. 09Y0048) and the Research Foundation of Yunnan University (No. 2009F29Q).

References

1. Bock, H.H., Diday, E. (eds.): Analysis of Symbolic Data, Exploratory Methods for Extracting Statistical Information from Complex Data. Studies in Classification, Data Analysis, and Knowledge Organization. Springer, Berlin (2000)

2. Gioia, F., Lauro, C.N.: Principal component analysis on interval data. Computational Statistics 21(2), 343–363 (2006)
3. Morimoto, Y.: Mining Frequent Neighboring Class Sets in Spatial Databases. In: The 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 353–358. ACM Press, New York (2001)
4. 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. 236–240. Springer, Heidelberg (2001)
5. Huang, Y., Shekhar, S., Xiong, H.: Discovering Colocation Patterns from Spatial Data Sets: A General Approach. IEEE Transactions on Knowledge and Data Engineering 16(12), 1472–1485 (2004)
6. Yoo, J.S., Shekhar, S.: A partial Join Approach for Mining Co-location Patterns. In: the 12th Annual ACM International Workshop on Geographic Information Systems, pp. 241–249. ACM Press, New York (2004)
7. Yoo, J.S., Shekhar, S., Celik, M.: A Join-Less Approach for Co-Location Pattern Mining: A Summary of Results. In: The 5th IEEE International Conference on Data Mining, ICDM 2005, pp. 813–816. IEEE Press, New York (2005)
8. Wang, L., Bao, Y., Lu, J., Yip, J.: A New Join-less Approach for Co-location Pattern Mining. In: The 8th IEEE International Conference on Computer and Information Technology (CIT 2008), pp. 197–202. IEEE Computer Society Press, Los Alamitos (2008)
9. Huang, Y., Pei, J., Xiong, H.: Mining Co-location Patterns with Rare Events from Spatial Data Sets. Geoinformatica 10, 239–260 (2006)
10. Wang, L., Zhou, L., Lu, J., Yip, J.: An order-clique-based approach for mining maximal co-locations. Information Sciences 179, 3370–3382 (2009)
11. Wang, L.: A method of the abstract generalization on the bases of the semantic proximity. Chinese Journal Computers 23(10), 1114–1121 (2000) (in Chinese)
12. Huo, Z.: Fuzzy Mathematics and its Applications. Tianjin Science and Technology Press, Tianjin (1989) (in Chinese)
13. Larsen, H.L., Yager, R.R.: Efficient computing of transitive closures. Fuzzy Sets and Systems 38(1), 81–90 (1990)
14. Tarjan, R.E., Leeuwen, J.: Worst-case analysis of set union algorithms. Journal ACM 31(2), 245–281 (1984)
15. Wang, X., Shi, W., Wang, S.: Processing in fuzzy spatial information, pp. 71–74. Wuhan University Press, Wuhan (2003) (in Chinese)
16. Zaki, M.J.: Scalable Algorithms for Association Mining. IEEE Transactions on Knowledge and Data Engineering 12(3), 372–390 (2000)

Multiple Attribute Frequent Mining-Based for Dengue Outbreak

Zalizah Awang Long[1], Azuraliza Abu Bakar[1],
Abdul Razak Hamdan[1], and Mazrura Sahani[2]

[1] Center for Artificial Intelligence Technology
Faculty of Information Science and Technology
[2] Faculty of Allied Health Science
Universiti Kebangsaan Malaysia
Bangi, Selangor, Malaysia
zalizah@miit.unikl.edu.my, aab@ftsm.ukm.my,
arh@ftsm.ukm.my, mazrura@gmail.com

Abstract. Dengue fever (DF) and dengue hemorrhagic fever (DHF) are vector borne disease which is notifiable diseases in Malaysia since 1974. Early notification is essential for control measures as delayed notification will lead to further occurrences of outbreak cases. In this study we identify the number of attributes to be used in determining outbreaks rather than using only case counts. The experiment is conducted using multiple attribute value based on Apriori concept. The outcomes are promising when we can identify more than one attributes showing similar graph in vector-borne diseases outbreaks. Our methods also outperform in term of detection rate, false positive rate and overall performance. We prove through our experiment that more than one attributes can be used to better detect outbreaks.

Keywords: Frequent mining, outbreak, dengue.

1 Introduction

Dengue outbreak is one of the critical communicable diseases and becoming more serious in Malaysia. The first dengue outbreak reported in Malaysia was recorded in 1901 in Penang, more than century ago. Quoted from [1], the dengue hemorrhagic fever (DHF) was first detected in 1962 and gradually increases with the development of the country. Dengue fever (DF) and dengue hemorrhagic fever (DHF) have been continuously becoming a public health related issues in Malaysia and growing pandemic as reported by World Health Organization (WHO).It is estimated that there are 50 million dengue fever cases with 500,000 people with DHF requiring hospitalization each year. The dengue outbreaks are increasing not only in Malaysia but also in Thailand, Vietnam and Singapore. As in Malaysia there are 24 dengue hotspots and most are of them in Selangor state [2].

Surveillance for DF and DHF outbreaks in Malaysia is based on Laboratory-based surveillance system. In most cases of dengue surveillance system are considered as

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 489–496, 2010.

passive surveillance system where it depends on cases reported from physicians through the routine reporting system. In Malaysia, mandated by law under Seksyen 10(2) Akta 342, all communicable diseases must be reported to ministry of health [3]. In order to permit action on DF and DHF or control dengue epidemic a surveillance system must capture related information either clinical or non-clinical data. The nature of passive surveillance system may not incorporate the capability to determine the potential combination related to non-clinical data in generating outbreak particularly in dengue cases. Thus, to improve early detection of the dengue outbreak is to look at the historical data from the passive surveillance system to identify the potential of collected attribute or data to be use to early detect dengue outbreak. It may improve public health surveillance system particularly in Malaysia to ensure the effectiveness of the actions taken by public health officers to control such epidemics and mitigate the impact to the nations.

In order to determine early detection of dengue outbreak or prediction of potential dengue outbreak are insufficiently being discussed. Analyzing dengue outbreak is based on vector-borne diseases epidemic curve. The analysis considers the increasing, peak and declining to determine the outbreak. Most of the studies focus on determining outbreak based on case counts over a period of time to predict the dengue outbreak. Therefore, in this study we focus on other potential attributes to be used to determine the potential dengue outbreak predominantly in Malaysia.

In this paper, we apply the method called Multiple Attribute Value (MAV), which employs an Apriori concept for frequent mining. We use the MAV to detect outbreaks and we calculate the algorithm performance based on detection rate, false positive rate and overall performance. Our techniques are able to identify combinations attribute *(k-length)* to be used to detect outbreaks. Our evaluations are based on real data set.

2 Related Study

The frequent items problem is one of the interesting and popular studies in data mining. The problem is interesting due to its simplicity to state, and its interest and values of associating between the items. Typically this will involve formalizing as to find whose frequency exceeded the specified fractions of the total numbers of items and also generating combine items or candidate items. The variation of problem becomes larger and larger such as the frequent value can be used to in some real life purposes such as outbreak detection. This is because of the number of candidate itemset is exponentially increases as the minimum support decreases. The abstraction of the above problem is viewed as the passive surveillance system particularly in the dengue context. The vast amount of collected data can be represented as a set of transaction, which contains multiple attribute, can be viewed as the items which may store more than one value.

Originally, Apriori [4],[5] implemented Apriori algorithm to mine one-dimensional Boolean association rule from transactional database [6]. [4] was the first to introduce the frequent pattern mining for the market basket analysis in a form of association mining. The concept of present and absent of the data in the transaction, with data representation in form of binary. However in the dengue outbreak detection we need to analyse data in the categorical representation which consist of multiple values within the attributes. We view dataset in a form of non-binary and not in transaction format. Based on classical Apriori we develop a new algorithm named Multiple Attribute Value Function (MAV) to calculate the frequency of each attribute value within a set of database [7].

The verification of dengue outbreaks are based on the data collected for the dengue cases.Reported [8] quoted by [9] dengue outbreaks mean incidence of notified dengue cases more than 1SD above average. While [10] define outbreaks as number of cases with 2 SD above mean baseline during non-epidemic week. Outbreaks being defined in many ways depending on the diseases and also vehicle of the diseases. Widely accepted definition based on CDC "The occurrence of more cases of disease than is expected in a given area over a particular period of time. While epidemic often implies a large number of cases a wide geographic area. Cluster refers to an aggregate of cases in a given area over a particular period of time without regard whether number of cases is greater than expected"[11]. We will use the definition of dengue outbreak based on definition by [12] the dengue outbreaks is the occurrence of more than one case in the same locality, where the date of onset between the cases is less than 14 days.

There are number of researches conducted in identifying the dengue outbreak cases. [12] was used geospatial modeling to detect dengue outbreak. The researcher focus on identifying the relation of population density and rainfall that contributed to dengue outbreak. [9] Dengue analysis aims at estimating the expected monthly values in each province, to define a significant statistical threshold, and then to identify periods differing from that basic value. While [10] tried to find correlation of different indicators to access usefulness in dengue outbreaks and the result indicated that the negative malaria diagnosis is an indicator for dengue. Proposed by [8] the uses of different indicator should be investigated in different settings particularly in dengue outbreaks.

Based on the recommendations quoted by [8] our research focuses on finding a new way of defining outbreaks with dengue as our case study. Our focus is on the usages of data mining techniques precisely association rules mining (ARM) in identifying the potential component within the data surveillance to be used as indicator of dengue outbreaks.

We mined frequent itemset based on attribute value using Apriori based concepts. Our experiment will identify the possible number of items to be considered in determining outbreaks based on detection rate, false positive rate and overall performance. Our experiment also indicate that the number of cases are not critical in determining the dengue outbreak.

3 Experiment Setting

We run our algorithm based on real dataset on dengue cases. We compare our technique based on the detection rate, false positive rate and overall performance in detecting dengue outbreak with CUSUM technique. The data is obtained from the Unit Kawalan Vektor, Pusat Kesihatan Hulu Langat. Detail pre-processing data are discussed below.

3.1 Data Pre-processing

Due to the nature of data, we have to conduct extensive data cleaning, data transformation and data reduction. In this experiment, we focus on non-clinical dataset. Dataset consists information on year and epic week (week1 to week 52), age, sex, races, address, nature of work, type of dengue, incubation period, epidemic type, recurrent cases and dead code. We focus on demographic effect to the recurrent cases and incubation period from the onset toward to confirm diagnose.

Approximately 0.14% data is reduced through the pre-processing stage. We try to maintain closely to real sets of the original dataset in analyzing the real dengue dataset.

3.2 Multiple Attribute Frequent Mining

We introduce the frequent mining analysis to retrieve normal behavior as the baseline. We implement and introduce Multiple Attribute Value (MAV) to calculate the frequent attribution within the surveillance data based on Apriori-based algorithm. Assume the attribute is an item. Let P = set of items {P_1, P_2, P_3 .... P_n} denotes as items/attribute. For each P there exist multiple values, P = {p_{n1}, p_{n2}, p_{n3}...... p_{nm},}. Let T_i has p_{nj} items, so we can write a transaction as t_i = { P_1, P_2, ... P_n} and P_n = { t_1, t_2, t_3 ..... t_i}. Following the above reason an indication to calculate frequency for each attribute value can be defined as:

$$\forall\ \mathrm{P_{ij}} = \mathrm{P_{kj}}\ \text{where}\ i \neq k\,,\ \exists\ \mathrm{MAV_{ij}} = \frac{\sum P_{ij}}{P_{ij}}\ . \tag{1}$$

We calculate the outbreak based on the definitions of the diseases. As in this case, we use dengue outbreak definitions.

4 Result and Discussion

The experiment is divided into two main objectives; the first objective is to analyze the important of numbers attribute @ k-length to be considered in determining the outbreak. While the second objective is to identify the number of records to be considered in analyzing the outbreak using multiple attribute frequent mining. The result discussions are based on fig1 and also table 1 and table 2.

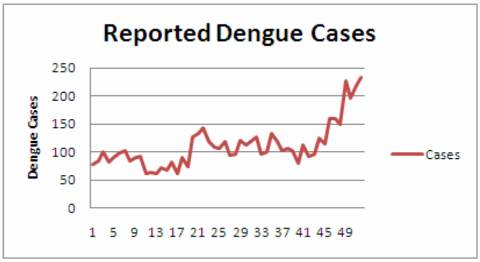


Fig. 1. Number of records

In identifying dengue outbreak, the reported cases are plotted based on the definition quoted from [12]. In Fig 1, it indicate the cases for Dengue fever (DF) and dengue hemorrhagic fever (DHF) from epic 1 toward epic week 52. There are 39.6% of the cases reported shown the above average number of cases. Mostly cases reported are towards week 44. Based on the graph, it indicate in fig 1, there are few peak situations throughout the 52 weeks. Dramatically a change is at week 49. There are number of outbreak cases detected using the number of reported cases. We identify outbreak at week 3, week 5-7, week 14, week 16, week 18, week 20-22, week 26 week 28-29, week 31-32, week 34-35, week 38, week 41, week43-44, week 46-47, week 49 and week 51-52. The question is which is the case throughout the year is the true outbreak? Our research is able to identify true outbreaks using our techniques. The discussions on the performance of our technique are based on table2. While in table 1 we analyze our technique in correlation of number of records and maximum length can be produced with speed and frequent items detected.

Table 1. MAV results for dengue dataset

WEEK	*RECORDS*	*MAX_LENGTH*	*SPEEDS*	*FREQUENT_ITEMS*
34	101	5	0.125	50
13	62	5	0.203	93
12	63	5	0.125	63
52	232	4	0.234	56
49	226	4	0.187	52
21	132	4	0.11	37
19	74	4	0.11	44

The results in table1 are collected using the dengue data from week 1 towards week 52 using MAV algorithm. Due to limited space we illustrate a few results for discussion purpose as in table 1. Based on table 1, we find that the number of records or cases do not indicate the execution time. As recorded execution time in week 12 and week 34 is 0.125sec with 63 records and 103 records while in week 21 and 19 the records are 132 and 74 with the same execution time 0.11sec. Again the same pattern of execution time in week 49 and week 13 the execution time for 62 records is 0.203sec and 226 records is 0.187 seconds. The execution time is recorded for several attempts of experiment with 0.001 second changes. Our experiment indicates that the fluctuation between numbers of records and execution times was not representing number of records. We believe that the execution times recorded are based on the complexity of the attribute values.

Our experiment also identify maximum length produced by the algorithm in dengue dataset. We identify most of the longest length are appearing in the beginning of the epic week, even though the number of records is small compared to the end of epic week. We believe this is also related to the complexity of the attribute values. We try to analyze whether the important number of records have any significance towards generating the combination of potential attributes in determining the outbreaks. We find that number of records will not have significant effect on the frequent items and the execution time recorded. Our experiment also indicates the higher number of frequent items and maximum length in the week 13 with 62 records produce higher number of frequent items. In contrast our experiment has shown that with the highest number on record 232 at week 52, the frequent items less 25%.

Table 2. Performance in detecting outbreak

MEASURE	CUSUM	MULTIPLE ATTRIBUTE VALUE(MAV)				
		MAV-1l	**MAV-2l**	**MAV-3l**	**MAV-4l**	**MAVFI**
DR	70.8%	58.8%	57.9%	65.0%	72.2%	74.1%
FPR	28.0%	28.0%	32.0%	28.0%	20.0%	28.0%
OP	67.3%	53.8%	53.8%	59.6%	63.5%	73.1%

The measurement used to compare the proposed technique as in table 2. The calculations are based on table3. We compare our algorithm with CUSUM [12], [13] in detecting outbreak. We also show in table 2 our detection rate (DR), false positive rate (FPR) and overall performance (OP) based on various length produced by our algorithm. Our result on detection manage to outperform the CUSUM with full length and with 4-length. In detecting the false positive rate our algorithm outperform CUSUM with 4-length while in full length we produce the same result. In the overall performance detecting outbreak, our algorithm manages to outperform CUSUM.

Table 3. Calculation matrix for detection rate (DR) False positive rate (FPR) and overall performance (OP) adopted from [14]

System detected	Actual cases		
	Outbreak	**No outbreak**	
Outbreak	True positive (TP)	False positive (FP)	**TP+FP**
No outbreak	False negative (FN)	True negative (TN)	**FN+TN**
	TP+FN	**FP+TN**	**TOTAL**

5 Conclusion and Future Work

Dengue is a mosquito borne viral disease with high capacity for epidemic outbreaks. Infection can be asymptomatic or can present with symptoms ranging from mild, self-limiting, febrile illness to severe, life-threatening disease. Two clinical pictures are recognized: (a) dengue fever (DF) and (b) dengue hemorrhagic fever (DHF) or dengue shock syndrome (DSS). In detecting outbreak, there are various techniques being applied ranging from statistics such as Cumulative Sum (CuSUM) related [13], [15], Space-time scan statistic [16] just to name a few. Extensive literature on detection techniques are from [17], [18]. In most literature, the development of analytical algorithms to detect anomalies is to reduce the outbreak curve are based on number of reported cases. Our study is to identify the combination of attributes to be used in determining the outbreak focusing on vector borne diseases. Our experiment shows that more than one attributes can be used as projected in frequent items. The experiment is conducted using Apriori concept for frequent mining. We find that using maximum item length shows better performance in detecting outbreak based on detection graph in vector-borne diseases. We also find out that high volumes of records are not critical since the complexity of the attribute value will determine the potential dengue outbreaks. Our next experiment will focus on determine the outbreak using frequent mining with outlier concept.

Acknowledgement

We would like to thank the Health District Officer, Ministry of Health (MOH) who has provided the dengue database(Vekpro) and Dr. Zainuddin Mohd Ali for his information and support.

References

1. Choy, E.A., Asmahani, A., Mazrura, S.: Perubahan Iklim dan Kesihatan Manusia: Metodologi dan Senario Penyakit Bawaan Vektor (unpublished)
2. New Strait Time (NST) online, Dengue Alert, `http://www.nst.com.my/Current_News/NST/articles/6dent/Article/`
3. Seksyen Penyakit Berjangkit, Bahagian Kawalan Penyakit, Jabatan Kesihatan Awam, Kementerian Kesihatan Malaysia, `http://www.moh.gov.my`
4. Agrawal, R., et al.: Mining association rules between sets of items in large databases. J. ACM SIGMOD Record. 22, 207–216 (1993)
5. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, vol. 1215, pp. 487–499 (1994)
6. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco (2001)
7. Zalizah, A.L., Azuraliza, A.B., Abdul-Razak, H.: Mining Multiple Attribute Values for Frequent Itemset Generation in Non-Binary Search Space (2009)
8. Runge-Ranzinger, S., Horstick, O., Marx, M., Kroeger, A.: What does dengue disease surveillance contribute to predicting and detecting outbreaks and describing trends? J. Tropical Medicine & International Health 13, 1022–1041 (2008)
9. Barbazan, P., Yoksan, S., Gonzalez, J.P.: Dengue hemorrhagic fever epidemiology in Thailand: description and forecasting of epidemics. J. Microbes and infection 4, 699–705 (2002)
10. Talarmin, A., Peneau, C., Dussart, P., Pfaff, F., Courcier, M., de Rocca-Serra, B., Sarthou, J.L.: Surveillance of dengue fever in French Guiana by monitoring the results of negative malaria diagnoses. J. Epidemiology and Infection 125, 189–193 (2000)

11. Excite, `http://www.cdc.gov/excite/classroom/outbreak/objectives.htm`
12. Seng, S.B., Chong, A.K., Moore, A.: Geostatistical modelling, analysis and mapping of epidemiology of Dengue fever in Johor State, Malaysia (2005)
13. Shmueli, G.: Current and Potential Statistical Methods for Anomaly Detection in Modern Time Series Data: The Case of Biosurveillance. Data Mining Methods for Anomaly Detection (2005)
14. German, R.R., Armstrong, G., Birkhead, G.S., Horan, J.M., Herrera, G.: Updated guidelines for evaluating public health surveillance systems. MMWR Recomm. Rep. 50, 1–35 (2001)
15. Watkins, R.E., Eagleson, S., Veenendaal, B., Wright, G., Plant, A.J.: Applying cusum-based methods for the detection of outbreaks of Ross River virus disease in Western Australia. J. BMC Medical Informatics and Decision Making 8, 37 (2008)
16. Kulldorff, M., Heffernan, R., Hartman, J., Assuncao, R., Mostashari, F.: A Space-Time Permutation Scan Statistic for Disease Outbreak Detection. J. Plos Medicine 2, 216 (2005)
17. Buckeridge, D.L., Burkom, H., Campbell, M., Hogan, W.R., Moore, A.W.: Algorithms for rapid outbreak detection: a research synthesis. Journal of Biomedical Informatics 38, 99–113 (2005)
18. Hutwagner, L., Browne, T., Seeman, G.M., Fleischauer, A.T.: Comparing aberration detection methods with simulated data. J. Emerging Infectious Diseases 11, 314–316 (2005)

A Top-Down Approach for Hierarchical Cluster Exploration by Visualization*

Ke-Bing Zhang, Mehmet A. Orgun, Peter A. Busch, and Abhaya C. Nayak

Departmen of Computing, Macquarie University, Sydney, NSW 2109, Australia
{kebing.zhang,mehmet.orgun,peter.busch,abhaya.nayak}@mq.edu.au

Abstract. With the much increased capability of data collection and storage in the past decade, data miners have to deal with much larger datasets in knowledge discovery tasks. Very large observations may cause traditional clustering methods to break down and not be able to cope with such large volumes of data. To enable data miners effectively detect the hierarchical cluster structure of a very large dataset, we introduce a visualization technique HOV^3 to plot the dataset into clear and meaningful subsets by using its statistical summaries. Therefore, data miners can focus on investigating a relatively smaller-sized subset and its nested clusters. In such a way, data miners can explore clusters of any subset and its offspring subsets in a top-down fashion. As a consequence, HOV^3 provides data miners an effective method on the exploration of clusters in a hierarchy by visualization.

Keywords: Top-down data analysis, hierarchical cluster exploration, visualization.

1 Introduction

Clustering aims at finding natural groups of data by a given similarity measure. Clustering approaches are roughly divided into partitioning and hierarchical clustering [2]. Compared to partitioning approaches, hierarchical approaches are more versatile, but sensitive to noise and outliers, and also suffer from higher computational complexity. As a consequence, hierarchical clustering algorithms cannot easily correct possible previous misclassifications for very large-sized datasets [17]. Clustering is a multiple run exploratory process under the guidance of data miners to find the optimum (or an optimal) cluster structure for a particular application. However, automated hierarchical clustering algorithms may exclude data miners' intuition and knowledge in the intermediate process of clustering. As a result, hierarchical clustering algorithms are not always effective in dealing with a very large number of observations in real world applications. [2]

Visualization is an effective approach to utilize the advantage of humans on decision making in the process of data analysis. The result of hierarchical clustering is often visualized as a *dendrogram*. However, dendrograms are only suitable to present a small number of observations [12]. Several techniques have been proposed to visualize clusters in a tree-like structure, but they do not allow data miners to participate

* This research has been supported in part by a Macquarie University Safety Net Grant.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 497–508, 2010.

in the process of hierarchical cluster exploration. These challenges motive us to develop a visual approach for assisting data miners on the exploration of hierarchical clusters.

Zhang *et al* [19] have harnessed a visual technique, called *Hypothesis Oriented Verification and Validation by Visualization* (HOV^3) [20], to explore partitioning cluster structures. This research builds on the work in [19] further to explore tree-structured clusters of data. Since partitioning clustering can be viewed as a special case of hierarchical clustering [2], we firstly project a dataset by HOV^3 onto a 2D space to have several well-separated data groups based on the enhanced separation feature of HOV^3[21]. Then we plot these groups further from top-down into lower-level subgroups recursively. In such a way, data miners can have the data groups in a hierarchy by the observation of their data distribution.

The rest of this paper is organized as follows. A review of current techniques on hierarchical clustering and cluster visualization are provided in Section 2. Section 3 briefly introduces the HOV^3 technique for completeness. Section 4 presents a detailed description of the top-down interactive approach to hierarchical cluster exploration by HOV^3. Then Section 5 demonstrates the effectiveness of our approach on several benchmark examples. Finally, section 6 summarizes the contributions of this paper.

2 Background and Related Work

2.1 Hierarchical Clustering

Based on different similarity measurements, such as single-link, average-link or complete-link, hierarchical clusters can have data points or representatives of lower-level clusters [17]. Hierarchical clustering is more general than partitioning clustering, because partitioning clustering can be viewed as a single-level special case of hierarchical clustering. However, once the tree-structured clusters are produced by automated hierarchical clustering algorithms, it is almost impossible to restructure the discovered dendrogram of clusters. This is because if data miners find any improper observation clustered by an automated clustering algorithm and attempt to reclassify it, they have to repeat the clustering process on the whole dataset. However, any change of the clustering hierarchy may cause it to collapse. Also, automated clustering algorithms cannot guarantee the reclassification of that observation to the right cluster for re-clustering the dataset. In addition, the higher computational complexity (between $O(N^2)$ to $O(N^3)$) of hierarchical clustering algorithms make them extremely hard to correct the previous misclassifications.

To deal with those problems of hierarchical clustering, several approaches have been proposed. For example, BIRCH [22] abstracts a given dataset as a Cluster Feature (CF)-tree, and then applies a hierarchical clustering algorithm with a multilevel compression technique to have the final clustering results. BIRCH is good at clustering in single scan due to its linear scalability feature. However, BIRCH is sensitive to the input order of given data objects. CURE [6] uses a set of representative points to describe the boundary of a cluster for further hierarchical clustering, but the representative points would be dramatically increased for depicting complicated cluster shapes in order to maintain the precision of clustering result. CHAMELEON [8] employs a multilevel graph partitioning algorithm on the k-nearest neighbour graph. But the high O(N^2) complexity of the algorithm hinders its application. LIMBO [1] first employs a summarized structure called Distributional Cluster Feature (DCF)-tree to condense large datasets, and then produces a range of k (the number of clusters) values for clustering. However, it is difficult for data miners to choose which k is the best value for LIMBO of a specific application. A recent survey of hierarchical clustering algorithms can be found in the literature [2, 17].

2.2 Hierarchical Cluster Visualization

Hierarchical data structure is common in data mining. Many visualization techniques have been proposed to reflect the hierarchy of high-dimensional data. However, most of them simply take information visualization as a layout problem for providing a more readable depiction of a dataset, rather than objectively reflecting the actual relationships in the dataset. As a result, those techniques are not very suitable to visualize hierarchical clusters of very large datasets. A recent survey of data visualization techniques can be found in the literature [4].

Several 3D techniques have been proposed to depict hierarchical clusters. For example, H-BLOB [10] visualizes clusters into nested blobs in a 3D hierarchy. Yang [18] proposed a 3D technique to project clusters in animations. However, the complex 3D graph formation limits their capability during interactive cluster exploration of a very large data by visualization. Bishop and Tipping proposed a latent variable model for hierarchical cluster visualization [3], however, their approach only performs well on the regular-shaped clusters. Kandoğan proposed the Star Coordinates technique to map high dimensional data to 2D space [7]. Star Coordinates has linear time computational complexity on the data mapping, and hence it is suitable to explore unknown data structures. But it is not easy to interpret the grouping results produced by Star Coordinates as its exploration is based on data miners' random adjustments.

The output of hierarchical clustering can be viewed as a dendrogram of clusters. Several dendrogram-based techniques have been proposed to visualize hierarchical clusters, such as, XmdvTool [16], HEC [13], Xgobi/Ggobi [11], TreeView [5], maxdView [9] and GrapClus [15]. However, dendrogram is limited to visualize a small number of observations [12].

In practice, instead of providing quantitative guidance during cluster exploration, most of the visual techniques employed in cluster analysis are typically used as a presentation mechanism to assist data miners having an intuitive comparison and better understanding of clustering results.

3 Preliminaries

The approach reported in this paper has been developed based on the projection feature of HOV^3 [20]. For the sake of completeness, we briefly describe HOV^3 below.

3.1 Data Projection of HOV^3

The idea of mapping high dimensional data to a 2D space by HOV^3 is intuitive, which extends the traditional orthogonal X-Y 2D coordinates technique to a higher dimensional space [20]. The mapping process divides a 2D plane into *n* equal sectors with *n* coordinate axes, where each axis represents a dimension and all axes share the initials at the centre of a circle on the 2D space.

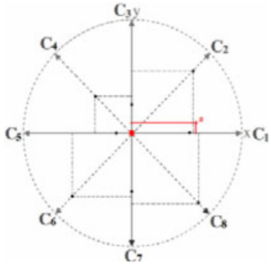


Fig. 1. Mapping an 8-dimensional datum to 2D Star coordinates space

First, data in each dimension are normalized over the interval of [0, 1]. Then the values of all axes are mapped to orthogonal X-Y coordinates which share the initial point on a 2D star coordinates space [7]. Thus, an *n*-dimensional data item is represented by a point in the X-Y 2D plane. Fig.1 illustrates an example of mapping an 8-dimensional datum *a* = (0.5, 0.75, 0.22, 0.33, 0.09, 0.7, 0.5, 0.9) to 2D X-Y Star coordinates space by HOV^3, where point "*a*" (marked between coordinate axes C_1 and C_2) is the mapped location.

To project high-dimensional data onto a 2D surface, we adopt the Polar Coordinates representation. Thus any high dimensional vector can be easily transformed to the orthogonal coordinates X and Y. In analytic geometry, the element-wise product of vector *A* and vector *B* (mathematically written as $A \circ B$) expresses the difference between them, and its geometrical meaning captures the data distribution plotted by vector *A* against vector *B* (and vice versa). Therefore the element-wise product between a dataset and a measure vector in HOV^3 can be geometrically viewed as a data distribution plotted by a matrix (a set of vectors) against the measure vector in the HOV^3 space.

For the clarification of our work, we adopt the Euler formula to depict the HOV^3 model. Let $z = x + i.y$ where *i* is the imaginary unit. According to the Euler formula, $e^{ix} = \cos x + i \sin x$. Let $z_0 = e^{2\pi i/n}$ such that $z_0^1, z_0^2, z_0^3, \ldots, z_0^{n-1}, z_0^n$ (with $z_0^n = 1$) divide the unit circle on the complex 2D plane into *n* equal sectors. Then given a non-zero measure vector *M* in R^n and a family of vectors P_j, the projection of P_j against *M* is mathematically written as:

$$P_j(z_0) = \sum_{k=1}^{n} [(d_{jk} - \min_k d_k)/(\max_k d_k - \min_k d_k) \cdot z_0^k \cdot m_k] \qquad (1)$$

where $\min_k d_k$ and $\max_k d_k$ represent the minimal and maximal values of the k^{th} coordinate respectively; and $(d_{jk} - \min_k d_k)/(\max_k d_k - \min_k d_k)$ is the normalized value of d_{jk}; m_k is the *k*th variable of measure *M*.

As presented in equation (1), the projection of HOV^3 can be viewed as a mapping from high-dimensional real space to 2D complex number space ($R^n \rightarrow C^2$) with measure *M*, where real part and imaginary part are mapped to X axis and Y axis respectively. Based on the description above, we can find that, if data points *a* and *b* are closely plotted by HOV^3 with measure vector *m*, data miners could interpret the result by analyzing *m*, as HOV^3 provides a quantitative mechanism on the data plot.

3.2 Features of HOV^3

We summarize the features of HOV^3 as follows.

Linear Time Complexity - Equation (1) is a standard form of linear transformation of *n* variables where m_k is the coefficient of the *k*th variable of P_j. The projection process of HOV^3 has only linear time complexity. Therefore, HOV^3 is effective for the interactive exploration on very large datasets by visualization.

Quantitative Exploration - The HOV^3 technique adopts the quantified user domain knowledge as a prediction to discover clusters [21]. Therefore HOV^3 can avoid the randomness and subjectivity introduced by data miners during the process of cluster exploration by visualization.

Enhanced Separation – Zhang *et al* [21] have shown that, if there are several data point groups that can be roughly separated by applying a measure vector *m* in HOV^3 to a dataset, then multiple applications of the projection in HOV^3 with the same measure vector *m* to the dataset would lead to the groups being more condensed, i.e., have a good separation of the groups. Based on this feature, data miners can apply a measure vector to a dataset several times to clearly identify data groups. This feature facilitates a clear geometrical separation of data points which may provide data miners an intuitive insight on the cluster observation.

4 Hierarchical Cluster Exploration

In this paper, we are only concerned with top-down hierarchical clustering. According to Shneiderman's visualization mantra, "*Overview, zoom and filter, details on demand*" [14], the idea of our approach is straightforward. Firstly, data miners apply HOV^3 to project a dataset with its statistical summaries to observe the groupings in the dataset. Then, the same process is performed iteratively to each group/subgroup for having lower level nested groups until a termination criterion is satisfied. Based on this idea, we propose an algorithm for hierarchical cluster exploration by HOV^3.

4.1 The Definitions

To clarify the description of our algorithm, we first present several definitions.

Definition 1 (a statistical measure of a dataset)**.** *Having a statistical measurement of a dataset* $\boldsymbol{D}$ *is an operation, denoted as* $S_t(\boldsymbol{D})\rightarrow\boldsymbol{m}$, *where* $\boldsymbol{m}$ *is a vector with the same dimensionality as* $\boldsymbol{D}$.

Definition 2 (HOV^3 projection)**.** *The element-wise product between a dataset* $\boldsymbol{D}$ *and a measure vector* $\boldsymbol{m}$ *in HOV*^{3} *is a projective operation, denoted as* $OP_H(\boldsymbol{D}, \boldsymbol{m})\rightarrow\boldsymbol{G}$, *where* $\boldsymbol{G}$ *is the projected data distribution of* $\boldsymbol{D}$ *in 2D HOV*^{3} *space.*

Data are only viewed by data miners when they are mapped to a specific space (2D or 3D normally). In fact, the data distribution projected by HOV^3 is a graph

G = (V(G), E(G)), where V(G) is the (nonempty) set of the vertices (data points) of G and E(G) is the set of edges of G.

In a given graph G, v represents a data point in the data distribution, thus the notation $v.a$ expresses a datum of v, and $| v.a |$ is the size of v.

To describe the hierarchical structure in HOV^3, we introduce the notions of a cluster container and a virtual edge in this study.

Definition 3 (Cluster container)**.** *A cluster container is a virtual data point denoted as* v_c, $v_c \in V \wedge |v_c.a| > 1$, *and* v_c *is invisible.*

A cluster container is used for representing a cluster in HOV^3, where the cluster container contains all the data points in the cluster that the cluster container represents. So the size of a cluster can be written as $|v_c.a|$. The geometrical features of a cluster container are expressed by the data points contained by the cluster container itself, such as the geometrical position and size.

Definition 4 (Virtual edge)**.** *A virtual edge is a directed link between a given data point (which can be a cluster container) to a cluster container, denoted as* $e(v, v_c)$, *and* $e(v, v_c) \in E(G)$.

A virtual edge is also invisible. It expresses the belonging relationship of a data point to a cluster container. To link a virtual edge between a data point v and a given cluster container v_c means that the data point v is a member of the cluster C which represents by the cluster container v_c.

Definition 5 (Virtual edge connection)**.** *This is an operation to create a virtual edge between a data point and a given cluster container* v_c, *denoted as* $op_e (v, v_c)$.

Definition 6 (Cluster in HOV^3)**.** *In HOV^3, a cluster C is a set of vertices of G projected by HOV^3 from a dataset D. C is represented by a cluster container* v_c *and* $(\forall v_i \in V(C)| \, e(v, v_c))$, *where* $C=\{ v_i \}$, $i=1,\ldots,k$.

In HOV^3, instead of using representative points to identify a cluster as done in some existing hierarchical clustering algorithms, we use a cluster container v_c to represent a cluster. The geometrical features of the cluster container can be expressed by its cluster members visually.

Also, instead of using class labels to mark membership in a cluster, in HOV^3, we use the virtual edges to identify the cluster members in the cluster, where all cluster members in the cluster are linked to their cluster container v_c. The use of virtual edges to identify cluster members provides a potential to combine HOV^3 with any hierarchical clustering algorithm directly, since it is not necessary to reclassify the existing data points with a class label in HOV^3.

Definition 7 (Cluster selection)**.** *Cluster selection is an interactive operation by data miners to collect a set of data points from a data distribution G, denoted as* $sub(G) \rightarrow G_c$, *and create virtual edges between the data points of* G_c *to a given cluster container* v_c, *written as* $((\forall v_i \in V(G_c)| \, op_e (v_i, v_c))$.

Cluster member selection in HOV^3 is the process of selecting a group of data points as a cluster by creating virtual edges from the data points to a given cluster container.

4.2 The Algorithm

Based on above definitions, we present the algorithm of top-down hierarchical cluster exploration by HOV^3 in Table 1. To demonstrate the effectiveness of our approach, we provide several experiments with detailed explanations of the algorithm in the next section.

Table 1. Hierarchical Cluster Exploration by HOV^3

```
Algorithm: Hierarchical Cluster Exploration by HOV$^3$
Input: $D$: a dataset; $M$: statistical measures of $D$;
Output: $G$: data distribution of $D$ or a subset of $D$;
1:   true → cluster exploration ;
2:   0 → c;   //initializing the counter of cluster id
3:       $D \rightarrow p$;
4:       $St(p) \rightarrow m_i$ ($m_i \in M$) ;// $m_i$, a statistical measure of p
5:       while (cluster exploration)
6:           $OP_H(p, m_i) \rightarrow G$;
7:           if (is $G$ well grouped?)// do clearly separated groups exist by observation?
8:               Group numbers $\rightarrow n_g$;
9:                   for( k=1 to $n_g$) {
10:                      $sub(G) \rightarrow G_c$;
11:                      $\forall v \in V(G_c) \mid op_e(v_i, v_{k+c})$;
12:                  }
13:                  $n_g + c \rightarrow c$;
14:              if (stop cluster exploration?)
15:              false →cluster exploration;
16:              break;
17:              endif
18:          endif
19:          if (is there a need for a new statistical measure of p?)
20:              $St(p) \rightarrow m_i$; //select the new statistical measurement to m$_i$
21:              break;
22:              else
23:              $m_i \rightarrow m$;
24:          endif
25:          $m \cdot m_i \rightarrow m_i$;//two times of m$_i$
26:      endwhile
```

5 Experimental Evaluation

In this section, we present an experimental evaluation of hierarchical cluster exploration by HOV^3. The results of the experiments are drawn by MATLAB. We directly introduce the statistical summaries as the measure vectors of a dataset in HOV^3. Hierarchical cluster exploration can be performed by HOV^3 by drilling down the nested groups iteratively. As shown in the steps 4 to 26 in table 1, when a data miner finds well grouped data points, he/she can choose one of those groups to do the iterative cluster exploration by HOV^3 as shown in steps 7 to 18 in table 1.

5.1 Experiment on the Shuttle Dataset

We first use the *shuttle* dataset, which is collected by NASA (available from http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/shuttle/). The dataset has 43,500 instances and 10 attributes. The original data distribution of *shuttle* projected by HOV^3 is shown in Fig. 2. Next, we apply the standard deviation *s_sd*=[0.124, 0.008, 0.070, 0.005, 0.035, 0.007, 0.086, 0.035, 0.041, 0.226] of *shuttle* twice to map *shuttle* to 2D space by HOV^3, as shown in Fig. 3, where there exist 7 well separated groups by observation. Then we circled these 7 groups as illustrated in Fig. 4.

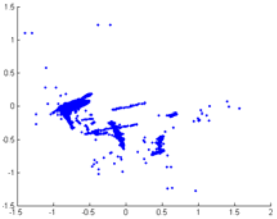


Fig. 2. The original data distribution of the *shuttle* dataset projected by HOV^3

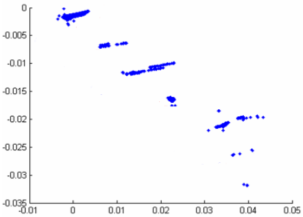


Fig. 3. The data distribution of *shuttle* projected by HOV^3 with twice of *s_sd* as the measurement

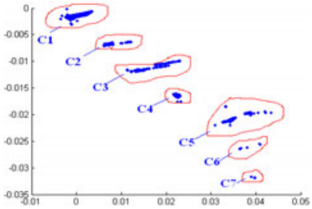


Fig. 4. The circled data points of data distribution in Fig. 3

By analyzing the data items of the groups, we find that the groups in Fig.4 are distinguished by the 10th attribute (*mark*) of *shuttle*. For example, group C1 has 34,108 items with class' label "1" (Rad Flow), C4 has 6,748 instances with label " 4" (High), etc. We are more interested in subgroups C1, C4 and C5 for exploring further nested groups because other groups are too small to detect internal clusters.

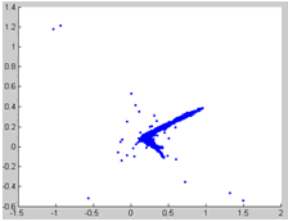


Fig. 5. The original data distribution of C1 dataset projected by HOV^3

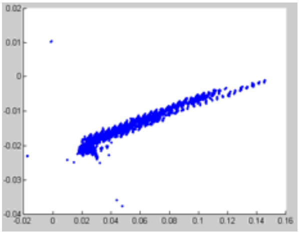


Fig. 6. The data distribution of C1 projected by HOV^3 with the standard deviation of C1 as the statistical prediction

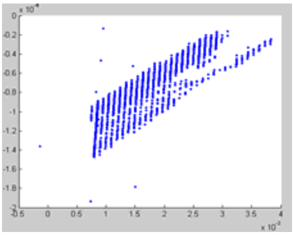


Fig. 7. The data distribution of C1 projected by HOV^3 with 3 times of *c1_std* of C1 as the statistical prediction

Next, we choose group C1 and remove its *mark* attribute. Thus, C1 now has 34,108 items and 9 attributes. We apply HOV^3 to C1 with its statistical predictions for nested cluster exploration. The original data distribution of C1 projected by HOV^3 is given in Fig. 5, where we cannot identify any group information by observation. Then we employ the standard deviation of C1, *c1_std* = [0.158, 0.007, 0.068, 0.005, 0.027, 0.006, 0.082, 0.030, 0.019] as a measurement to plot C1; its plotted data distribution is shown in Fig. 6, where we cannot observe any clearly separated groups. However, the wave shape of data distribution in Fig.6 compels us to explore C1 further. Then we apply three times and ten times of *c1_std* to C1. Their corresponding data distributions are shown in Fig. 7 and Fig. 8 respectively. It can be observed that there are 34 regular bar-shaped data points and several outliers.

By analyzing the standard deviation of C1 (*c1_std*), we find that the relative scale of the first element in vector *c_std* is much larger than the others. Thus, when we have the element-wise product between the vector *c1std* and itself, each element's value is attenuated, because they are less than 1.

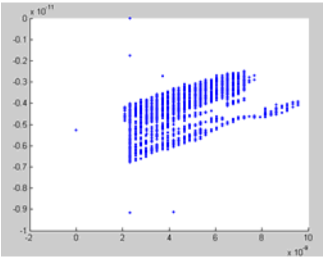


Fig. 8. The projected distribution of C1 projected by HOV^3 with 10 times of *c1_std*

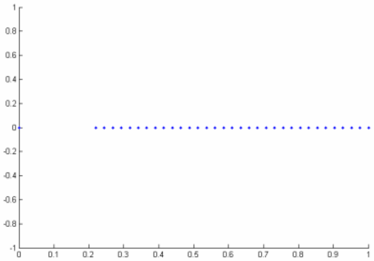


Fig. 9. The data distribution of the first column of C1 projected by HOV^3

With an increasing application of the element-wise product, each element's value in the output vector would be regressed to zero. However, the relative scale of the first value in the output vector to the others is increased. The output vector of ten times element-wise product between *c1_std* is [0.958, 0.000, 0.000, 0.000, 0.000, 0.000, 0.001, 0.000, 0.000] e-9. It is observed that, in this vector, the first element itself can almost have an effect on mapping C1 by HOV^3.

We therefore choose the first column of C1 and project it without any measurements. Its data distribution is shown in Fig. 9, where there are clearly 34 dots. This means that, the groups in Fig. 8 are mainly distinguished according to the category of the first column of C1. By further top-down exploration of the resulting groups, each bar-shaped data group in Fig. 8 can be further divided into smaller-sized subgroups.

5.2 Experiment on the Credit-Screening Dataset

The credit-screening (*crx*) dataset has 6 enumerated, 4 logical and 6 continuous attributes, and 690 instances. It is available from http://archive.ics.uci.edu/ml/machine-learning-databases/credit-screening/. To deal with non-digital data, the dataset is firstly normalized into [0, 1] interval for the next step of data analysis. The original data distribution of *crx* plotted by HOV^3 is presented in Fig. 10, where we cannot observe any groups of data. We simply employ the mean value of *crx*, named *crx_m* = [0.170, 0.433, 0.170, 0.087, 0.087, 0.433, 0.106, 0.078, 0.477, 0.572, 0.036, 0.458, 0.053, 0.088, 0.010, 0.555], as a statistical measurement to detect group information of *crx* by HOV^3. Its data distribution is shown in Fig.11, where no groups can be identified intuitively. Then we project the dataset further by HOV^3 with three, five and seven times of *crx_m* respectively;

their corresponding projected results are shown in Fig.12, Fig. 13, and Fig.14 respectively. Intuitively, we cannot identify any data groups in Fig. 12. There exist several fuzzy data groups in Fig. 13.

We find that there are 7 group data points in Fig. 14 by observation. The process of applying a measure vector multiple-times to a dataset is presented in the steps 6 to 26 in Table 1. We circle these obvious groups in Fig. 14 as 4 clusters as shown in Fig.15, where C1, C2, C3 and C4 have 98, 297, 209 and 86 instances respectively.

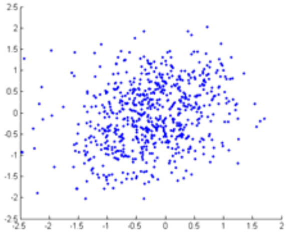


Fig. 10. The original data distribution of *crx* projected by HOV^3

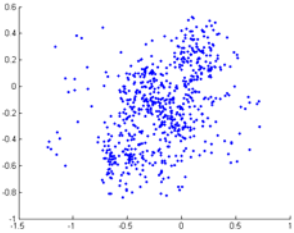


Fig. 11. The data distribution of *crx* projected by HOV^3 with *crx_m*

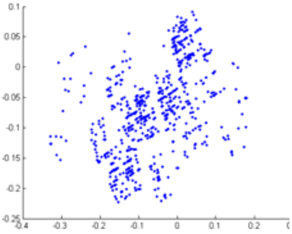


Fig.12. The data distribution of *crx* projected by HOV^3 with 3 times of *crx_m*

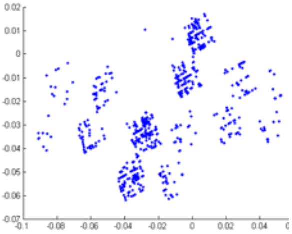


Fig. 13. The data distribution of *crx* projected by HOV^3 with 5 times of *crx_m*

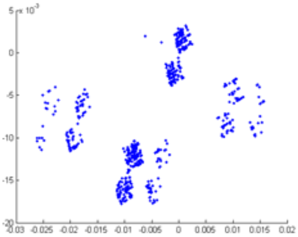


Fig. 14. The data distribution of *crx* projected by HOV^3 with 7 times of *crx_m*

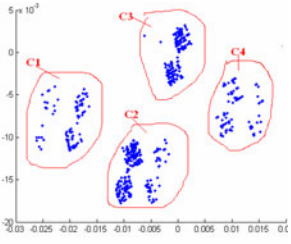


Fig. 15. The data distribution in Fig. 14 circled by data miners' observation

We choose cluster C2 for further cluster exploration by HOV^3. The original data distribution of C2 is shown in Fig. 16, where we cannot recognize any groups. Then we employ the standard deviation of C2, *C2_std*=[0.246, 0.287, 0.156, 0.247, 0.247, 0.265, 0.210, 0.153, 0.392, 0.000, 0.000, 0.497, 0.196, 0.131, 0.127, 0.000] as a statistical measurement to explore the inner clusters of C2 by HOV^3. Its data distribution is presented in Fig. 17, where we still cannot identify groups of C2. Based on the enhanced separation feature of HOV^3, we apply *C2_std* three times to C2, and its data distribution is shown in Fig. 18, where 4 groups can be clearly observed and we mark them as 4 sub-clusters (C21, C22, C23 and C24) of C2. C21, C22, C23 and C24 have 32, 24, 144 and 97 instances respectively. In such a way, hierarchical cluster exploration can be applied to the resulting sub-clusters further. We believe that, an expert on this domain could give a more meaningful explanation about this clustering result and any subsequent cluster exploration.

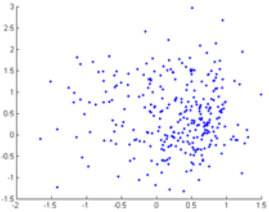


Fig. 16. The original data distribution of the cluster C1 in *credit-screening (crx) dataset* projected by HOV^3

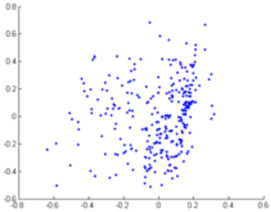


Fig. 17. The data distribution of the cluster C2 in *crx dataset* projected by HOV^3 with *C2_std*

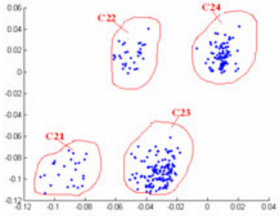


Fig. 18. The circled data distribution of the cluster C2 in *crxt* projected by HOV^3 with three times *C2_std*

We have also compared the clustering results produced by HOV^3 with several commonly used clustering algorithms in partitioning clustering elsewhere [21, 19], where HOV^3 provides better clustering results than those clustering algorithms based on the experiments we have done. This implies that, HOV^3 can be used for the local optimization of clustering in each level of hierarchical clustering.

6 Conclusions

This work has developed a top-down visual approach to assist data miners on the exploration of hierarchical clusters. Based on HOV^3, data miners can interactively project a high dimensional dataset by the statistical summaries of the dataset or its subsets. Data miners are then able to capture grouping information from the projected data distribution by their intuition based on the enhanced separation feature of HOV^3. This process can be performed to explore the nested clusters of each group / subgroup in a top-down manner until a termination criterion is reached. In addition, HOV^3 provides a combination mechanism of HOV^3 with existing hierarchical clustering algorithms with its virtual edge and cluster container to correct the previous misclassifications produced by a hierarchical clustering algorithm. As a consequence, based on the HOV^3 technique, data miners can effectively identify the nested clusters of very large data sets.

References

1. Andritsos, P., Tsaparas, P., Miller, R.J., Sevcik, K.C.: Limbo: scalable clustering of categorical data. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K., Ferrari, E. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 123–146. Springer, Heidelberg (2004)
2. Berkhin, P.: A Survey of Clustering Data Mining Techniques. In: Kogan, J., Nicholas, C., Teboulle, M. (eds.) Grouping Multidimensional Data, pp. 25–72. Springer, Heidelberg (2006)
3. Bishop, C.M., Tipping, M.E.: A Hierarchical Latent Variable Model for Data Visualization. IEEE Trans. on Pattern Analysis and Machine Intelligence 20(3), 281–299 (1998)

4. Chen, C.: A Brief History of Data Visualization. In: Hardle, W., Unwin, A. (eds.) Handbook of Computational Statistics: Data Visualization, vol. III, Springer, Heidelberg (2007)
5. Eisen, M.: Cluster and TreeView Manual (2007), `http://rana.lbl.gov/manuals/ClusterTreeView.pdf`
6. Guha, S., Rastogh, R., Shim, K.: CURE: An efficient clustering algorithm for large databases. In: Proceedings of ACM SIGMOD Conference 1998, pp. 73–84. ACM Press, New York (1998)
7. Kandogan, E.: Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In: Proceedings of ACM SIGKDD Conference 2001, pp. 107–116. ACM Press, New York (2001)
8. Karypis, G., Han, E.-H.S., Kumar, V.: Chameleon: hierarchical clustering using dynamic modeling. IEEE Computer 32(8), 68–75 (1999)
9. `http://www.bioinf.manchester.ac.uk/microarray/maxd/maxdView/overview.html`
10. Sprenger, T.C., Brunella, R., Gross, M.H.: H-BLOB: A Hierarchical Visual Clustering Method Using Implicit Surfaces. In: Proceedings of the Conference on Visualization 2000, pp. 61–68. IEEE Computer Society Press, Los Alamitos (2000)
11. Swayne, D.F., Cook, D., Buja, A.: XGobi: Interactive dynamic data visualization in the X Window System. Journal of Computational and Graphical Statistics 7, 113–130 (1998)
12. Schonlau, M.: Visualizing non-hierarchical and hierarchical cluster analyses with clustergrams. Journal of Computational Statistics 19, 95–111 (2004)
13. Seo, J., Shneiderman, B.: Interactively Exploring Hierarchical Clustering Results. IEEE Computer 35, 80–86 (2002)
14. Shneiderman, B.: Inventing discovery tools: Combining information visualization with data mining. Information Visualization 1, 5–12 (2002)
15. Todd, C.S., Toth, T.M., Robert, B.-F.: GraphClus, a MATLAB program for cluster analysis using graph theory. Journal of Computers and Geosciences 35, 1205–1213 (2009)
16. Ward, M.O.: XmdvTool: Integrating Multiple Methods for Visualizing Multivariate Data. In: Proceedings of IEEE Conference on Visualization 1994, pp. 326–333. IEEE Computer Society, Los Alamitos (1994)
17. Xu, R., Wunsch, D.C.: Clustering. John Wiley and Sons, Inc., Publication, Chichester (2009)
18. Yang, L.: Visual Exploration of Large Relational Data Sets through 3D Projections and Footprint Splatting. IEEE Trans. on Knowledge and Data Engineering 15, 1460–1471 (2003)
19. Zhang, K.-B., Huang, M.L., Orgun, M.A., Nguyen, Q.V.: A Visual Method for High-dimensional Data Cluster Exploration. In: Leung, C.S., Lee, M., Chan, J.H. (eds.) ICONIP 2009. LNCS, vol. 5863, pp. 699–709. Springer, Heidelberg (2009)
20. Zhang, K.-B., Orgun, M.A., Zhang, K.: HOV^3: An Approach to Visual Cluster Analysis. In: Li, X., Zaïane, O.R., Li, Z.-h. (eds.) ADMA 2006. LNCS (LNAI), vol. 4093, pp. 316–327. Springer, Heidelberg (2006)
21. Zhang, K.-B., Orgun, M.A., Zhang, K.: A Prediction-Based Visual Approach for Cluster Exploration and Cluster Validation by HOV^3. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 336–349. Springer, Heidelberg (2007)
22. Zhang, T., Ramakrishana, R., Livny, M.: An Efficient Data Clustering Method for Very Large Database. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 103–114. ACM Press, New York (1996)

Distributed Frequent Items Detection on Uncertain Data

Shuang Wang[1, 2], Guoren Wang[2], and Jitong Chen[1]

[1] Software College, Northeastern University, Shenyang 110819, China
[2] College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
{wangsh,wanggr}@mail.neu.edu.cn

Abstract. Frequent items detection is one of the valuable techniques in many applications, such as network monitor, network intrusion detection, worm virus detection, and so on. This technique has been well studied on deterministic databases. However, it is a new task on emerging uncertain database, especially in distributed environment. In this paper, a new definition of frequent items on uncertain data is defined. Based on the definition, a polynomial algorithm is proposed, which can efficiently answer the queries in central environment. Furthermore, this work designs the communication-efficient algorithms for retrieving the top-k items with the largest probability from distributed sites. The algorithms compute the upper bound of each round of the transmission, and filter the data as much as possible, which have no chance to influence the query result. Extensive experiments show that the algorithms can process the queries correctly and reduce communication cost efficiently with various data set.

Keywords: distributed query processing, frequent item, uncertain data, top-k query.

1 Introduction

An important problem in many data management systems is to identify frequent items. People are interested in tracking these items for a number of reasons. For example, in network traffic monitoring, frequent packets consuming most of the bandwidth should be efficiently reported for accounting purpose[1]; in other applications, frequent items are essential for answering the popular iceberg queries [2] and association rule mining [3]. Due to its importance in a wide range of applications, the topic of finding frequent items has been extensively explored on deterministic databases [4,5,6].

However, little is known about finding frequent items on uncertain data, which is an emerging area that has attracted a lot of attention recently. Uncertain data analysis has become an increasingly important issue due to the ubiquitous data uncertainty in many real world applications such as sensor data monitoring, location based services (LBS), RFID networks, and moving object search. Interestingly enough, many cases where uncertainty arises are distributed in nature, e.g., distributed sensor networks with imprecise measurements [7], multiple data sources for information integration based on fuzzy similarity scores [8,9].

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 509–520, 2010.

In the aforementioned application domains, it is often very expensive to communicate the data set entirely from each site to the centralized server for processing, due to the large amounts of data available nowadays and the network delay incurred, as well as the economic cost associated with such communication. In the paper, our work is to answer top-k frequent items in a communication-efficient manner on uncertain data from multiple distributed sites.

Our contributions. We study frequent items queries for distributed uncertain data. We design communication-efficient algorithms for retrieving the top-k items with the largest probability from distributed sites. In summary, our contributions are as follows:(1) We give a new frequent items definition on uncertain data.(2) We find the polynomial approach in probability computation and greatly improves the efficiency of single data detection.(3) We formalize the problem of distributed frequent items detection on uncertain data, and argue that the straightforward solution is communication-expensive.(4) We propose the distributed frequent items detection algorithms on uncertain data, which results in significantly less communication.(5) We present a comprehensive experimental study that confirms the effectiveness of our approach.

2 Related Work

Over the past several decades, many algorithms have been developed to find frequent items in deterministic database. Most of these algorithms mainly maintain a data structure that contains a synopsis of item frequency. The strategy of these algorithms can be summed up as three kinds: sampling [10][11], hash[5][6] and counting[6]. Sampling method calculates frequent items based on a sample of original data set. Hash method reduces the range of data set by performing hash mapping and selects the most concentrated items in the mapping as the approximate solution. Counting method uses limited counting units to maintain frequent items in the data set. Also, there are some works on distributed frequent items queries on deterministic database [12][13]. However, the distributed frequent items queries on uncertain data are still an untapped territory.

The most relevant works to this paper are [14][15]. [14] studies the problem of estimating various statistical aggregates on probabilistic streams, and in particular frequent item based on expectation is proposed. [15] formalizes the new concept of probabilistic frequent items with confidence, rather than relying on the expectation over all possible worlds, but that paper do not solve the problem of distributed environment. In this paper, we propose a new definition of probabilistic frequent items, which is slight variation of definition in [15], and design a communication-efficient algorithm of distributed frequent items detection. This is the first step towards the important and challenging problem of answering distributed frequent items queries on uncertain data.

3 Problem Formulation

The modeling of uncertain data has been studied for many years and the most prominent model is possible world model [16][17]. As shown in Table 1, the uncertain database D includes four tuples and each tuple contains data and the existential probability of the data. The tuples can be independent or dependant from one another (In this paper we only consider the former, a more simple model). For the uncertain database D in table 1, the number of instances is 2^4=16, shown in table 2. The probability of instance {t1, t2} is computed by assuming the existence of t1 and t2, and the absence of t3 and t4. Therefore, the P(W_8)=0.6*0.4*(1-0.8)*(1-0.3)=0.0336, where 0.6,0.4 are the existence probabilities of t1and t2, respectively, while(1-0.8), (1-0.3) is the absence probability of t3 and t4.

Table 1. Uncertain Database

ID	data	pr
t1	8	0.6
t2	8	0.4
t3	3	0.8
t4	8	0.3

Table 2. Possible World and Probability

PW	Pr	PW	Pr
W_0={}	0.0336	W_8=t1,t2	0.0336
W_1=t1	0.0504	W_9=t2,t4	0.0096
W_2=t2	0.0224	W_{10}=t1,t4	0.0216
W_3=t3	0.1344	W_{11}=t1,t2,t3	0.1344
$W_{4=}$=t4	0.0144	W_{12}=t1,t2,t4	0.0144
$W_{5=}$=t2,t3	0.0896	W_{13}=t2,t3,t4	0.0384
W_6=t1,t3	0.2016	W_{14}=t1,t3,t4	0.0864
$W_{7=}$=t3,t4	0.0576	W_{15}=t1,t2,t3,t4	0.0576

3.1 Frequent Items Definition on Uncertain Data

In this section, we give a new definition of frequent items on uncertain data, namely, the probabilistic top-k frequent items, or simply PK-FI. Our definition PK-FI is a slight variation of [15], called PHH. There are two differences between them.

First, in the PK-FI query, we do not set a probability threshold, but return k items with the highest probabilities being the frequent items. One undesirable problem with the PHH query is that the number of items returned may differ a lot over different databases even when using the same threshold value. The user must set the threshold carefully to make the result set contain k tuples. Second, PK-FI is the absolute frequent items. The item t is frequent item, which must satisfy its frequency no less the parameter min_sup. PHH is the relative frequent items, the frequency of t accounts for the total number of items is no less than the parameter s, $s \in (0,1)$. Formally, the PK-FI query is defined as follows.

Definition 1: *Probabilistic top-k frequent items (PK-FI)*
Let D denote an uncertain database, PW the possible world space for D. A PK-FI query returns a set of k items T = {t_1,…,t_k}, satisfying for $\forall t_i \in T, t_j \notin T$, $\sum_{\omega \in PW, m_{t_i} \geq \text{min\_sup}} P[\omega] \geq \sum_{\omega \in PW, m_{t_j} \geq \text{min\_sup}} P[\omega]$, m_{ti} denotes the frequency of item t_i.

The PK-FI query returns the k most probable items of being the top-k among all. For example, in the uncertain dataset of Table 1 and with k = 1, min_sup=2, the answer is {8}, as it is the highest probability of being among the top-k, with probability 0.396. In the last eight possible world instances, the frequency of 8 is no less than 2, the sum of these instance's probability is 0.396.

3.2 The Probability Computation on Frequent Items

For computing the probability of a given item t being a frequent item, i.e., the basic idea is listing the possible worlds of uncertain database and computing the probability based on the definition1. However, the generated possible worlds can be exponentially large. Therefore, the algorithm is exponential in both space and time, which makes it impractical for large databases. In this section, we present a new algorithm by constructing the polynomial that enable highly efficient processing frequent items queries over very large datasets.

Definition 2: *The polynomial $F_t(x)$*

$$F_t(x) = \prod_{i=1}^{n} a_i(x) \tag{1}$$

$$\begin{cases} a_i(x) = (1 - p_i) + p_i(x) & v_i = t \\ a_i(x) = 1 & v_i \neq t \end{cases}$$

Where n is the total number of tuples in database.For each distinct item t in uncertain database D, we construct the polynomial $F_t(x)$. If the value of ith tuple equals item t, then the ith factor $a_i(x)=(1-p_i)+p_ix$, else $a_i(x)=1$. Expanding the $F_t(x)$, $F_t(x)$ can be rewritten in the normalization form, as shown in equation 2. C is the total number of item t in uncertain database.

$$F_t(x) = \sum_{i=0}^{C} b_i x^i \tag{2}$$

In table 1, the total number of item 8 is 3, and the polynomial $F_8(x)$ $= (0.6x + 0.4) * (0.4x + 0.6) * 1 * (0.3x + 0.7) = 0.072x^3 + 0.324x^2 + 0.436x + 0.168$.

Theorem 1: The coefficient of x^j in $F_t(x)$ is the total probability of possible worlds for which item t appears j times.

For example, the probability of item 8 appears three times is $P(m_8 = 3) = 0.072$. In table 2, $P(m_8 = 3) = P(W_{12}) + P(W_{15}) = 0.0144+0.0576=0.072$.

The proof is by induction and is omitted due to limited space.

By constructing the polynomial, we can compute the probability of item t being the frequent item. The probability is

$$P(m_t \geq min\_sup) = \sum_{i=min\_sup}^{c} b_i \tag{3}$$

For example, in table 1, min_sup=2, $P(m_8 \geq min\_sup) = 0.072+0.324=0.396$. For each distinct item in uncertain database, we computed the probability based on definition 2, and get the top-k items with the highest probability value.

The polynomial approach can dramatically reduce the processing time of frequent item queries. The time complexity of polynomial approach is polynomial time, however that of the naïve possible world's method is exponential time. The time of constructing the polynomial is O(m^2), with items increasing, the running time grows rapidly. So we propose a pruning rule to reduce candidates of frequent items so as to reduce the search space and improve performance of frequent items queries.

Pruning Rule: If |t|<min_sup, t is not a frequent item.
|t| denotes frequency of item t in the original uncertain data set. By performing pruning rule, we delete a large number of candidates for frequent items thus reducing search space.

4 Distributed Frequent Items Detection on Uncertain Data

The previous section shows that our frequent items definition and the algorithm in central environment. Next, we present the algorithms for efficiently detecting distributed frequent items according to definition 1. We first introduce our distributed architecture, then present two algorithms for distributed computing.

In the paper, we consider a distributed environment with n remote sites and one central site, called coordinator site. Here, a remote site i is holding an uncertain dataset D_i. There is a two-way communication channel between the coordinator and each of the n sites, but there is no direct communication between any two remote sites. Obviously, we can always ask all remote sites to forward their databases to coordinator site and solve the problem at coordinator site locally. Then we compute the probability by method in section 3 and get the top-k items. This approach, however, is communication-expensive. In this case, the total communication cost is $|D| = \sum_{i=1}^{n} |D_i|$. This will be the baseline we compare against.

4.1 The Polynomial Approach

For each distinct item t at every remote site, we compute the local polynomial F$_{ti}$(x) based on definition 2, then the global polynomial for item t can be calculated accumulatively from all the sites. More precisely, we have the following.

Lemma 1: The global polynomial GF$_t$ (x) for item t is

$$GF_t(x) = \prod_{i=1}^{n} F_{ti}(x) \tag{4}$$

where F$_{ti}$(x) is computed using equation 1 if $t \in D_i$, otherwise, F$_{ti}$(x)=1, n is the number of remote sites. The proof is omitted due to space limit. Lemma 1 indicates that by forwarding all local polynomials of item t of each site, we can obtain its final global probability. This naturally leads to the following idea for computing the global top-k at the coordinator site.

The polynomial algorithm (A_Polynomial)
At the ith remote site, it compute the local polynomial $F_{ti}(x)$ for every distinct item t, transmits all the $F_{ti}(x)$ to the coordinator site. The coordinator site computes the global polynomial GF_t (x) for item t based on equation (4), then calculates the probability according to equation (3), and reports the final top-k list.

Algorithm 1: On the remote site i

```
Input: uncertain database $D_i$, K, min_sup, n
Initialize queue $PQ_i$
Compute the distinct item and it's frequency, i.e. <$t_{ij}$, $C_{tij}$ > , add it to $PQ_i$
Transmit $PQ_i$ to coordinator site
Receive the invalid items from coordinator site, and remove them from $PQ_i$
for each item $t_{ij}$ in $PQ_i$
    Compute the local polynomial $F_{tij}(x)$ according to equation (1), (2)
    Transmit the $F_{tij}(x)$ to the coordinator site
```

Subroutine: computing_GP (t_{ij}) // compute the global probability of t_{ij}

```
For i=1 to n do
    get the local polynomial of item $t_{ij}$
Compute the global polynomial GF($t_{ij}$) according to equation(4)
Compute the global probability gp($t_{ij}$) according to equation (3)
```

Algorithm 2: On the coordinator site

```
Input: K, min_sup, n
Output: top-k items and their probabilities of being frequent items
For i=1 to n do
    Receive the queue $PQ_i$
Compute the distinct item and it's total frequency, i.e. <$t_j$, $C_{tj}$ >, add it to PQ
Prune the invalid items based on pruning rule
Transmit the invalid items to remote sites
For each remaining item $t_{ij}$ in PQ
        global probability gp($t_{ij}$)=computing_GP($t_{ij}$)
return the top-k items and their global probabilities
```

The A_Polynomial algorithm filters the invalid items based on pruning rule, calculates the local polynomial of each item, and only transmits the local polynomials, which reduces the communication cost and computation on coordinator site.

4.2 Sorted Access on Local Probability

At remote site i, we first calculate the local probabilities (lp) to be frequent item for all distinct items according to equation(3). We sort the items based on their local probabilities in descending order, $lp(t_{i1}) \ge lp(t_{i2}) \ge ... \ge lp(t_{in_1})$ .The coordinator site S accesses items from the n sites in the descending order of their local probabilities. More precisely, S maintains a priority queue LQ of size n in which each site i has a representative local probability value and the item t_{ij} that corresponds to that local probability value, i.e., a triple <i, t_{ij}, lp(t_{ij})>. The triples in the priority queue are sorted by the local probability value in descending order. LQ is initialized by retrieving the first item and local probability from each site.

In each step, S obtains the first element from LQ, say <i, t_{ij}, lp(t_{ij})>. Then, S asks for item t_{ij+1} from site i as well as lp(t_{ij+1}), the local probability of the next item from site i. The triple <i, t_{ij+1}, lp(t_{ij+1})> will be inserted into the priority queue LQ. In order to compute the exact global probability of t_{ij} that S has just retrieved, S broadcasts t_{ij} to all sites and asks each site to report back the local polynomial of t_{ij} (based on equation (3)). By Lemma 1, S obtains the exact global probability of item t_{ij} . This completes a round.

Let the set of items seen by S be *Ds*. S dynamically maintains a priority queue GQ for items in *Ds* based on their global probabilities. In the λ-th round, let the k-th smallest probability from *Ds* be p_λ^+ . Let p_λ^- be the upper bound of global probabilities of unseen items (*D-Ds*) in coordinator site S. It is safe for S to terminate the search as soon as $p_\lambda^+ \ge p_\lambda^-$ at some round λ and output the top-k from the current Ds as the final result. We denote this algorithm as A_SLP.

The key issue of algorithm A_SLP is how we can derive an upper bound p_λ^- for the global probabilities of any unseen items in coordinator site. We introduce the Lemma 2 to find an upper bound for any unseen items at round λ.

$$P((m_{t1} + m_{t2} + ... + m_{tn}) \ge min\_sup) \le P(\bigcup_{i=1}^{n}(m_{ti} \ge \frac{min\_sup}{n}))$$

$$P(\bigcup_{i=1}^{n}(m_{ti} \ge \frac{min\_sup}{n})) = 1 - P(\bigcap_{i=1}^{n}(m_{ti} < \frac{min\_sup}{n})) = 1 - \prod_{i=1}^{n} P(m_{ti} < \frac{min\_sup}{n})$$

So

$$P((m_{t1} + m_{t2} + ... + m_{tn}) \ge min\_sup) \le 1 - \prod_{i=1}^{n} P(m_{ti} < \frac{min\_sup}{n}) \tag{5}$$

According to the inequality(5), in every remote site, for each distinct item t, we compute the probability $P(m_{ti} \ge \frac{min\_sup}{n})$, then sort the probabilities in descending order, τ_i denotes the probability value of the first element in the remote site i.

Lemma 2: $p_\lambda^- = 1 - \prod_{i=1}^{n}(1 - \tau_i)$

Proof: By inequality (5) and equation (3).

Based on the above illustration, we further present the A_SLP Algorithm.

Algorithm A_SLP

Algorithm 3:On the coordinator site S

```
Input: min_sup, n, K
Output: Top-k frequent items and their probabilities
// Initialize
1. Initialize the queue LQ(length=n) ,GQ(length=K)
2. For i=1 to n do
3.     get the first element < $t_{i1}$, $P_{t_{i1}}$ > of $Q_i$ and add the element to LQ
4.     get the first element $P'_{i1}$ of $Q_{i'}$
5. Sort the LQ in descending order by the probability
6. For each element $t_{i1}$ in LQ
7.     the global probability gp($t_{i1}$)=computing_GP($t_{i1}$)
8. Get the top-k items on the global probability, add the items to GQ in descending
   order, $p_\lambda^+$ =the kth item's global probability
9. Compute the upper bound $p_\lambda^-$ according to Lemma 2
10. For i=1 to n do
       get the next element of $Q_i$ and add the element to LQ, get the next element
of $Q_{i'}$
//Initialize end
11. while ( $p_\lambda^+ < p_\lambda^-$ ) do
12.        sort LQ in descending order
13.        get the first element $t'_{ij}$ of LQ
14.        get the next element of $Q_i$ and $Q_{i'}$ from the remote site i
15.        gp($t'_{ij}$)=computing_GP($t'_{ij}$)
16.        add $t'_{ij}$ to GQ , remain the top-k result in GQ
17.        compute the upper bound $p_\lambda^-$ according to Lemma 2
18. return GQ
```

Algorithm 4: On the remote site i

```
Input: Uncertain dataset $D_i$, min_sup, K, n
1. compute the distinct items and their number,<$t_{ij}$, $l_{t_{ij}}$ >
2. Initialize the queen $Q_i$ ,$Q_{i'}$
3. For each distinct item $t_{ij}$
4.     Compute the local polynomial $LP_{t_{ij}}$ ,the probabilities $P_{t_{ij}} = P(m_{t_{ij}} \geq min\_sup)$ ,
           $P'_{t_{ij}} = P(m_{tij} \geq \frac{min\_sup}{n})$ according to equation (2)and (3).
5.     Add element <$t_{ij}$, $P_{t_{ij}}$ > to $Q_i$, add $P'_{t_{ij}}$ to $Q_{i'}$
6. Sort the $Q_i$ and $Q_{i'}$ in descending order by the probability
```

5 Experiment

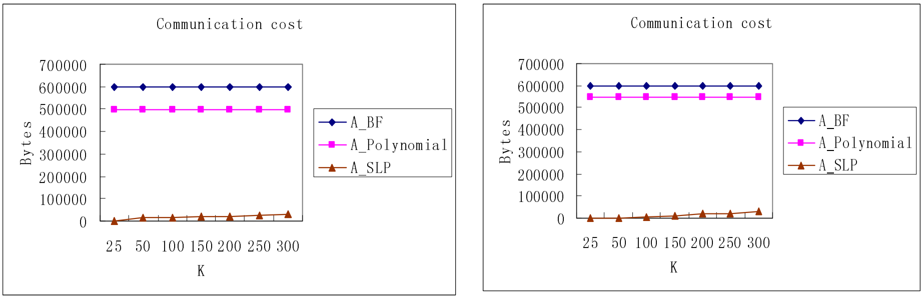
In this section, we begin to evaluate the performance of our approach through a series of experiments. We implemented all the algorithms proposed in this paper:

A_Polynomial, A_SLP and A_BF(the straightforward solution that sends all data to the coordinator site and process locally). Our platform is Pentium3.0 GHz CPU with 4Gbytes memory.

Datasets: We used matlab to generate two synthetic groups of data, where the item's value from the normal distribution N(0,1) and N(0,5). The confidence of each item is independent and uniformly distributed between 0.0 and 1.0.

In the simulation, the default values number of remote sites n=10, the size of uncertain database D=50000, each record from the uncertain database D is assigned to a remote site i chosen uniformly at random, K=50, min_sup=100. We measure the total communication cost in terms of bytes, running time in terms of milliseconds. For each item, the value of item is four bytes, and the probability is eight bytes.

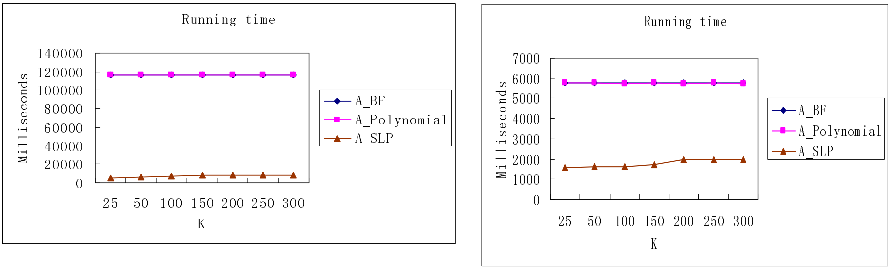
Results with different K. We first study the performance of all the algorithms for different k values from 25 to 300. Figure 1 shows the communication costs of the algorithms for the two data sets. Clearly, A_SLP saves the communication costs by several orders of magnitude compared with A_BF in all cases. A_Polynomial saves a little communication costs compared with A_BF, because some invalid items are pruned according to pruning rule, these invalid items do not transmit to the coordinator site. A_SLP algorithm has increasing communication costs as k gets larger, increases gradually as k gets larger, while A_BF and A_Polynomial remain unchangeable.



(a) N(0,1) (b) N(0,5)

Fig. 1. Communication costs when vary K

Figure 2 shows that the running times for A_BF are similar to that of the A_Polynomial, because the items to be processed are the same. A_BF does all of the work in coordinator site, while A_Polynomial calculates the local polynomial in remote sites, does other work in coordinator site, which reduces the computation overhead in coordinator site. As expected, the running times of A_SLP are very efficient. The running times for A_SLP increase with k increasing duo to more items to be processed, but increase gradually. The running times for N(0,5) is lower than that of N(0,1) dataset. Because the greater the variance, the more distinct items, the less times for constructing the polynomial.

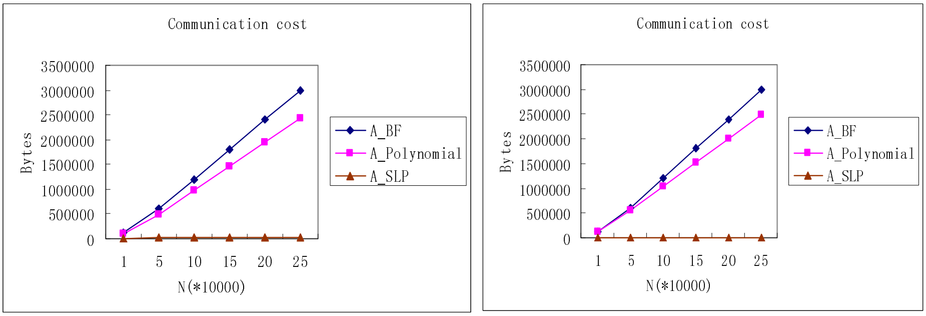


(a) N(0,1)

(b) N(0,5)

Fig. 2. Running times when vary K

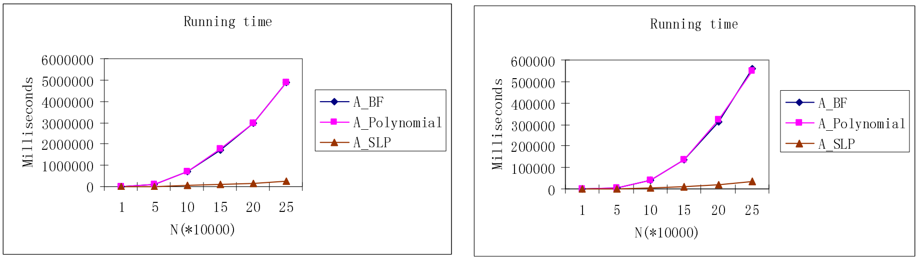
Results with different N. We next study the effects of N, the total number of records in the uncertain database D. Figure 3 shows that the communication cost of the A_BF and A_Polynomial approach linearly increases with N. A_SLP increases at a much slower rate. Figure 4 shows the running times of all algorithms. The running times for A_BF and A_Polynomial increases quadratic with N, and the algorithms become very slow for high N. The running times of A_SLP also increase with N increasing, but not significantly.



(a) N(0,1)

(b) N(0,5)

Fig. 3. Communication costs when vary N

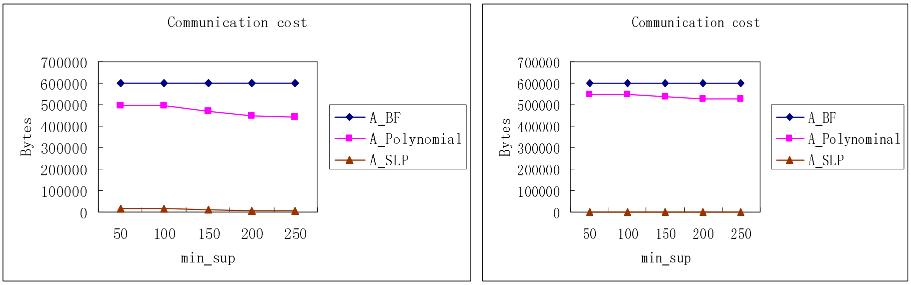


(a) N(0,1)

(b) N(0,5)

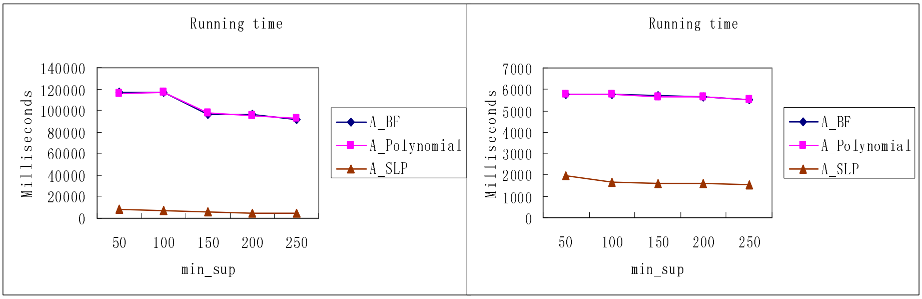
Fig. 4. Running times when vary N

Results with different min_sup. Finally, we study the effects of min_sup, the frequency threshold. Figure 5 shows that the communication costs for A_Polynomial and A_SLP decrease with min_sup increasing due to more items pruned according to pruning rule. Figure 6 shows that the running times for all algorithms decrease with min_sup increasing, it is also because more items are filtered out.



(a) N(0,1) (b) N(0,5)

Fig. 5. Communication costs when vary min_sup



(a) N(0,1) (b) N(0,5)

Fig. 6. Running times when vary min_sup

6 Conclusion and Future Work

This is the first work that studies frequent item queries for distributed uncertain data. We give a new definition of frequent item on uncertain data, propose the efficient polynomial method to detect the frequent item in central environment. We also present the distributed frequent item queries algorithm A_SLP. Our experimental results show that our algorithms can significantly save the communication cost and running time. In our future work, we plan to solve this problem on more complicated uncertain data model and on probabilistic data stream.

Acknowledgments. The work was supported by the National Natural Science Foundation of China under the grant No. 60933001.

References

1. Cristian, E., George, V.: New directions in traffic measurement and accounting. In: Proceedings of the 2001 ACM SIGCOMM International Conference on Data Communication, pp. 323–336. ACM Press, Pittsburgh (2002)
2. Jiawei, H., Jian, P., Guozhu, D., Ke, W.: Efficient computation of iceberg cubes with complex measures. In: Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, pp. 1–12. ACM Press, Santa Barbara (2001)
3. PRakesh, A., PRamakrishnan, S.: Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499. ACM Press, Santiago de Chile (1994)
4. Graham, C., Flip, K., Muthukrishnan, S., Divesh, S.: Diamond in the rough: finding hierarchical heavy hitters in multi-dimensional data. In: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, pp. 155–166. ACM Press, Paris (2004)
5. Graham, C., Muthukrishnan, S.: What's hot and what's not: tracking most frequent items dynamically. In: The Symposium on Principles of Database Systems, pp. 249–278. ACM Press, San Diego (2003)
6. Demaine, E.D., López-Ortiz, A., Munro, J.I.J.: Frequency estimation of internet packet streams with limited space. In: Möhring, R.H., Raman, R. (eds.) ESA 2002. LNCS, vol. 2461, pp. 348–360. Springer, Heidelberg (2002)
7. PAmol, D., PCarlos, G., Samuel, R.M., Joseph, M.H., PWei, H.: Model-driven data acquisition in sensor networks. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, pp. 588–599 (2004)
8. PNilesh, D., PDan, S.: Efficient query evaluation on probabilistic databases. The VLDB Journal 16, 523–544 (2007)
9. Christopher, R., Nilesh, D., Dan, S.: Efficient top-k query evaluation on probabilistic databases. In: Proceedings of the 23rd International Conference on Data Engineering, Istanbul, pp. 864–875 (2007)
10. Pjeffrey, S.V.: Random sampling with a reservoir. ACM Transactions on Mathematical Software 11, 37–57 (1985)
11. Gibbons, P., Matias, Y.: New sampling-based summary statistics for improving approximate query answers. In: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, pp. 331–342. ACM Press, Seattle (1998)
12. Ke, Y., Qin, Z.: Optimal Tracking of Distributed Heavy Hitters and Quantiles. In: Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 167–174. ACM Press, Providence (2009)
13. Qi, Z., PMitsunori, O., PHaixun, W., PJun, X.: Finding global icebergs over distributed data sets. In: Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 298–307. ACM Press, Chicago (2006)
14. Graham, C., PMinos, G.: Sketching probabilistic data streams. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, pp. 281–292. ACM Press, Beijing (2007)
15. Qin, Z., Fei, F.L., Ke, Y.: Finding Frequent Items in Probabilistic Data. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 819–832. ACM Press, Vancouver (2008)
16. Abiteboul, S., Kanellakis, P., Grahne, G.: On the representation and querying of sets of possible worlds. ACM SIGMOD Record 16, 34–48 (1987)
17. Green, T.J., Tannen, V.: Models for incomplete and probabilistic information. IEEE Date Engineering Bulletin 29, 17–24 (2006)

Mining Uncertain Sentences with Multiple Instance Learning

Feng Ji, Xipeng Qiu, and Xuanjing Huang

School of Computer Science and Technology,
Fudan University, Shanghai 201203, China
{fengji,xpqiu,xjhuang}@fudan.edu.cn

Abstract. Distinguishing uncertain information from factual ones in online texts is of essential importance in information extraction, because uncertain information would mislead systems to find useless even fault information. In this paper, we propose a method for detecting uncertain sentences with multiple instance learning (MIL). Based on the basic assumption, we derive two new constraints for estimating the weight vector by defining a probability margin, which is used in an online learning algorithm known as Passive-Aggressive algorithm. To demonstrate the effectiveness of our method, we experiment on the biomedical corpus. Compared with an intuitive method with conventional single instance learning (SIL), our method provide higher performance by raising the performance from 79.07% up to 82.55%, over 3% improvement.

Keywords: Uncertain sentence, Multiple instance learning, Passive-Aggressive algorithm.

1 Introduction

With the popularity of search engines, looking for useful information from online texts has obtained more and more attentions, especially in the communities of information extraction, computational biology and natural language process. In order to extract useful information, many research works are commonly implied such an assumption that the source articles are reliable and consistent with the facts. However, this assumption is weak. According to the statistics in English biomedical articles [9], nearly 18% sentences are containing uncertain information. These speculative information would be confused with the factual evidence and mislead the applications to extract unreliable information.

Uncertain information means the statements are not supported with facts. Linguistic devices such as hedges, some keywords indicating that authors cannot back up their opinions with facts, play important roles in detecting uncertain sentences. For examples of the sentence "*Our result suggests that the unknown amino acid encoded by stop codons does not exist.*", it is uncertain because word *suggests* conveys speculative meaning and help us make the right decision. However if we replace it with word *proves* while retaining the rests, the sentence will turn to be factual one. Usually, sentences containing at least one hedge are considered as uncertain, while sentences with no hedges are factual.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 521–528, 2010.

In this paper, we propose a novel method for identifying uncertain sentences by using multiple instance learning (MIL) [6]. Multiple instance learning (MIL) differs from conventional single instance learning (SIL). In MIL problem, a sample is represented as a bag composed of several single instances and labels of single instances are not available. In classic definition of MIL problem, a bag would be assigned as *positive* if at least one instance in the bag is *positive*, while assigned as *negative* if all instances are *negative*. This assumption obviously meets the problem of detecting uncertain sentences. We treat each word in a sentence as a single instance. Thus a sentence is considered to be a bag of word instance. Therefore, a sentence is classified as uncertain if at least one word is classified as hedge, otherwise is classified as factual. Based on the basic assumption, we derive two new constraints for estimating the model parameters when applying online Passive-Aggressive (PA) algorithm [2]. Our experiments on the biomedical corpus show that our method is more competitive than single instance learning.

The remainder of this paper is organized as follows. In Section 2, a brief review of related works on detecting uncertain sentences is presented. Then, we describe our new method with multiple instance learning (MIL) in Section 3.1, and parameter estimation within an online learning framework is presented in Section 3.2. Experiments and results are shown in the Section 4. Finally, the conclusion would be presented in Section 5.

2 Related Works

Although the concept of hedge information has been introduced in linguistic community for a long time, researches on automatic hedge detection emerged from machine learning or computational linguistic perspective in recent years. In this section, we give a brief review on the related works.

For speculative sentences detection, Medlock and Briscoe [8] reported their approach based on weakly supervised learning. In their method, a statistical model is initially derived from a seed corpus, and then iteratively modified by augmenting the training dataset with unlabeled samples according the posterior probability. They only employed bag-of-words features. Although they also tried to integrate more linguistic features, such as part-of-speech (POS), lemma and bigram, there is no significant improvements [7]. Morante and Daelemans [9] presented their research on identifying hedges by using IGTREE algorithm to train a 3-categories classifier. However, in the competitive shared task of CoNLL 2010 [3], many other models, such as sequence labeling models [10] or bag-of-words models [5], were exploited to improve the performance. Moreover, in the official corpus, hedges are annotated for each uncertain sentences. Although we use the same corpus, our method essentially does not require to annotate hedge words due to the basic MIL assumption, and in our experiments we have not used these word level annotation information as well.

3 Identifying Uncertain Sentences

In this section, we would present our method for identifying uncertain sentences in detail. First, we describe our method with multiple instance learning (MIL). By defining a probability margin, we obtain two new constraints respectively for uncertain sentences and factual ones. In the following section, an online learning algorithm, known as Passive-Aggressive (PA) algorithm, is employed to estimate the parameters of our method.

3.1 Detecting Uncertain Sentences with Multiple Instance Learning

Although traditional classification models [4,5] would be a not-bad choice for detecting uncertain sentences from natural raw texts, these models essentially do not describe the nature of detecting uncertain sentences. In uncertain sentences, at least one keyword, formally be known as hedge, can guide us to distinguish them from factual ones. This phenomenon is consistent with the basic assumption of multiple instance learning (MIL) [6], which states that a bag is *positive* if at least one instance is *positive*, and *negative* if all instances are *negative.*

In this setting, we treat an arbitrary sentence $\mathbf{x}$ as a bag consisting of word instances $x_i, i = 0, \ldots, n$. Each word instance x_i can be classified as *positive* denoting x_i is hedge, otherwise *negative.* Here notation y_i is the class label for word instance x_i, and bold notation $\mathbf{y}$ is the whole sentence class label. To predict the class label of the sentence, we need to first predict the class label of each word in the corresponding sentence.

According to the basic MIL assumption, we could conveniently deduce that all word instances in a sentence bag are *negative* if the sentence is factual. Unfortunately, if a sentence is uncertain, it is not clear to know which word instance is *positive*, since actual labels of word instances are not available. Thus the score of uncertain sentences cannot be easily calculated.

In order to employ the framework of PA algorithm, we first define a probability margin for MIL problem,

$$\gamma_p(\mathbf{x}, \mathbf{y}; \mathbf{w}) = Pr(\mathbf{x}, \mathbf{y}) - Pr(\mathbf{x}, \mathbf{y}^*) \ . \tag{1}$$

Basically we assume that all word instances in the same sentence bag are independent and identically distributed (I.I.D) and be in the form of maximum entropy model. We separately discuss the probability margin for two cases.

If sentence $\mathbf{x}$ is *positive*, the probability margin would be

$$\begin{aligned}
\gamma_p(\mathbf{x}, \mathbf{y}; \mathbf{w}) &= 1 - 2Pr(\mathbf{x}, \mathbf{y} = -1) \\
&= 1 - 2\prod_{i=0}^{n-1} Pr(x_i, y_i = -1) \\
&= 1 - 2\prod_{i=0}^{n-1} \frac{1}{Z(x_i)} exp(\mathbf{w}^T \phi(x_i, y_i = -1)) \\
&= 1 - 2\frac{exp(\sum_{i=0}^{n-1} \mathbf{w}^T \phi(x_i, y_i = -1)}{\prod_{i=0}^{n-1} Z(x_i)} \ .
\end{aligned} \tag{2}$$

where $Z(x_i)$ is the partition function over labels on each word instance x_i. We need to make the probability margin greater than ξ. However in Formula 2, it is difficult to calculate the denominator $\prod_{i=0}^{n-1} Z(x_i)$. By using Jensen inequality, we can derive its lower bound,

$$\begin{aligned}\prod_{i=0}^{n-1} Z(x_i) &= \prod_{i=0}^{n-1} \sum_{y} exp(\mathbf{w}^T \phi(x_i, y)) \\ &\geq \prod_{i=0}^{n-1} |y| exp(\sum_{y} \frac{1}{|y|} \mathbf{w}^T \phi(x_i, y)) \\ &= |y|^n exp(\sum_{i=0}^{n-1} \sum_{y} \frac{1}{|y|} \mathbf{w}^T \phi(x_i, y)) \ . \end{aligned} \tag{3}$$

Therefore, the first new constraint for *positive* sentence would be shown in the following formula,

$$\mathbf{w}^T (\sum_{i=0}^{n-1} \phi(x_i, y_i = -1) - \sum_{i=0}^{n-1} \sum_{y} \frac{1}{|y|} \phi(x_i, y)) \leq \log \frac{1-\xi}{2} - n \log |y| \ . \tag{4}$$

In the other hand, if sentence $\mathbf{x}$ is *negative*, for each word instance, the probability for *negative* label should be great than *positive* label. So the probability margin would be

$$\begin{aligned}\gamma_p(\mathbf{x}, \mathbf{y}; \mathbf{w}) &= \prod_{i=0}^{n-1} (Pr(x_i, y_i = -1) - Pr(x_i, y_i = +1)) \\ &\geq (Pr(x_k, y_k = -1) - Pr(x_k, y_k = +1))^n \ . \end{aligned} \tag{5}$$

where $k = \arg\min_i (Pr(x_i, y_i = -1) - Pr(x_i, y_i = +1))$, representing the most difficult classified word instance. By using Jensen inequality again, we can derive another lower bound,

$$\begin{aligned} Pr(x_k, y_k = +1) &= \frac{1}{Z(x_k)} exp(\mathbf{w}^T \phi(x_k, y_k = +1)) \\ &\leq \frac{exp(\mathbf{w}^T \phi(x_k, y_k = +1))}{|y| exp \sum_y \frac{1}{|y|} \mathbf{w}^T \phi(x_k, y)} \\ &\leq \frac{1 - \xi^{\frac{1}{n}}}{2} \ . \end{aligned} \tag{6}$$

Therefore, the other new constraint for *negative* sentence would be

$$\mathbf{w}^T (\phi(x_k, y_k = +1) - \sum_{y} \frac{1}{|y|} \phi(x_k, y)) \leq \log \frac{1 - \xi^{\frac{1}{n}}}{2} - \log |y| \ . \tag{7}$$

3.2 Parameter Estimation

To learn the weight vector $\mathbf{w}$, we employ Passive-Aggressive (PA) algorithm [2], which is an online error-driven learning algorithm. In the setting of online algorithm, we receive training samples in a sequential manner.

The weight vector $\mathbf{w}_0$ is initialized to 0. On round t, the new weight $\mathbf{w}_t$ can be analytically calculated by solving the following optimization problem,

$$\begin{aligned} \mathbf{w}_t = \arg\min_{\mathbf{w}} & \tfrac{1}{2}\|\mathbf{w} - \mathbf{w}_{t-1}\|^2 + C \cdot \xi \\ s.t. \qquad & \gamma_p(\mathbf{x}, \mathbf{y}; \mathbf{w}) \geq \xi \\ & \xi \geq 0 \; . \end{aligned} \tag{8}$$

where ξ is non-negative slack variable, C is a positive parameter which controls the influence of the slack term on the objective function, and y^* is the best prediction with current weight vector w_t.

Here in order to facilitate the presentation, we define some symbols,

$$\mathbf{M}_+ = \sum_{i=0}^{n-1} \phi(x_i, y_i = -1) - \sum_{i=0}^{n-1} \sum_{y} \frac{1}{|y|} \phi(x_i, y) \; . \tag{9}$$

$$\mathbf{M}_- = \phi(x_k, y_k = +1) - \sum_{y} \frac{1}{|y|} \phi(x_k, y) \; . \tag{10}$$

$$\xi_+ = \log \frac{1 - \xi}{2} - n \log |y| \; . \tag{11}$$

$$\xi_- = \log \frac{1 - \xi^{\frac{1}{n}}}{2} - \log |y| \; . \tag{12}$$

After applying the Lagrange multipliers, we can obtain the final weight update formulas (in which the subscript $*$ could be $+$ or $-$),

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \lambda_* \cdot \mathbf{M}_* \; . \tag{13}$$

$$\lambda_* = \min\{C, \frac{\xi_* - \mathbf{w}_{t-1}^T \cdot \mathbf{M}_*}{\|\mathbf{M}_*\|^2}\} \; . \tag{14}$$

To avoid the overfitting problem, we employ an average weighting strategy proposed in [1]. The detail implementations could be found in [5].

4 Experiments

4.1 Datasets, Baseline System and Features

To demonstrate the performance of our method, we experiment on the biomedical corpus [3]. Preliminary statistics in [9] shows that nearly about 18% sentences in training dataset and 15.8% in test dataset are speculative. Therefore, there are plenty of uncertain sentences in the real texts.

Our baseline system is similar to the work in [5], which is essentially a conventional binary classification model, except for the training algorithm. We substitute average perceptron with average PA algorithm. In the baseline system, each sentence is represented as a bag-of-words and a binary linear classifier is trained to predict a new coming sentence.

All sentences in the corpus are preprocessed with a maximum-entropy part-of-speech tagger[1], in which a rule-based tokenzier is used to separate punctuations or other symbols from regular words. At last, in order to make more meaningful of the comparisons, we also adopt the same predefined feature patterns in [5] to generate features. These features are used not only in the baseline system but also in our proposed method in this paper.

4.2 Experiment Results and Discussions

In all experiments, F-measure ($F_1 = \frac{2PR}{P+R}$) of the uncertain class is used as the chief evaluation metric. Parameter C, which controls update step, is set to be 0.01. We also try to experiment on different values of C, and found that a larger value of C implies a more aggressive update step and results to fast convergence, but it has little influence on the final results. All results are averaged by repeating 5 times experiments.

We first investigate the relationship between the performance and the number of iterations. ξ is set to be 1.0 for the baseline method, while 0.3 for our method. The results in biomedical domain are shown in Fig. 1.

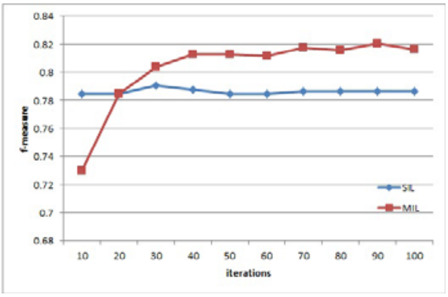


Fig. 1. relationship between the performance and the number of iterations

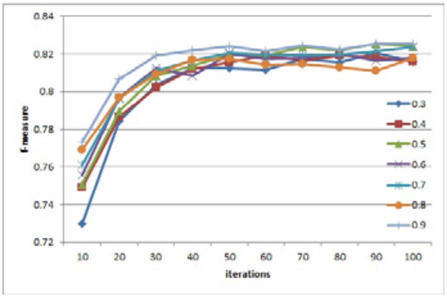


Fig. 2. relationship between ξ and the number of iterations

From Fig. 1, the baseline method is little influenced by the iterative numbers. After 10 iterations, the performance is stable and achieves best F_1 nearly 0.79. On the other hand our method is convergence after 40 iterations. However, our method outperforms the baseline only after 20 iterations, and achieves the maximum over 0.82 when the iterative number is 90. On the average, our method has increased about 2% than the baseline after convergence. Bottom 2 lines in Table 1 show the best results of these two methods. Although recall is slightly

[1] Available at http://nlp.stanford.edu/software/tagger.shtml

dropped down from 0.7532 to 0.7496, precision is obviously increased from 0.8322 to 0.9060.

In Table 1, we also list the best results with sequence labeling model [10] and our past work with average perceptron [5] in the competition of the shared task in CoNLL 2010. Their results are much better than ours, because hedge word instances are available in their model training phase but not available in our proposed method. Feature selection are also used to improve the model performance. However compared with our past work, we find that average PA algorithm can provide higher performance than average perceptron. It is also consistent with many previous works.

Table 1. Best results for different systems

	iterate	*Precision*	*Recall*	F_1
Tang [10]		0.85	0.877	0.864
Ji [5]		0.794	0.763	0.779
baseline	30	0.8322	0.7532	0.7907
our method	90	0.9060	0.7496	0.8204

Table 2. Best results with different ξ

ξ	*iterate*	*Precision*	*Recall*	F_1
0.3	90	0.9060	0.7496	0.8204
0.4	80	0.9077	0.7466	0.8193
0.5	90	0.9026	0.7600	0.8252
0.6	80	0.8964	0.7557	0.8201
0.7	100	0.8889	0.7673	0.8236
0.8	100	0.8848	0.7605	0.8180
0.9	100	0.8825	0.7754	0.8255

In the second experiment, we test the influence of ξ. ξ is set to be from 0.3 to 0.9 with interval 0.1. Similar to the first experiment, the maximum iterative number is set to be 100. Fig. 2 shows the experiment results. We can find that with the iterative numbers increased, the performance on test dataset is growing too. In the other hand, with the values of ξ goes up, the performance is slightly raised up, and the increment is nearly $1\% - 2\%$. This phenomenon can be easily explained because larger ξ implies that the ability of distinguishing *positive* samples from *negative* samples are stronger. We also list the best results for different values of ξ in Table 2. In spite of different ξ, the performances actually differ very little. The minimum F_1 is 0.8201, while the maximum is 0.8255. But compared with the baseline, when setting ξ to be greater than 0.5, precision is substantially increased as well as recall is also improved.

5 Conclusion

In this paper, we present a method for detecting uncertain sentences with multiple instance learning (MIL). Based on the basic assumption, we derive two new constraints by defining a probability margin, which is used in Passive-Aggressive algorithm for estimating the model parameters. Compared with an intuitive method with conventional single instance learning (SIL), our method provide higher performance by improving the performance from 79.07% to 82.55%, over 3% increment.

In the future, we will focus on the role of semantic information in this task, because uncertain information are semantic concept and more semantic information could be helpful. Applying our proposed method into a real system, such as web based question answering system, is another possible future work.

Acknowledgments. We would like to thank the anonymous reviewers for their helpful remarks. This work was (partially) funded by NSFC (No. 61003091), 973 Program (No. 2010CB327906), and 863 Program (No. 2009AA01A346).

References

1. Collins, M.: Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In: EMNLP, pp. 1–8. ACL (2002)
2. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. Journal of Machine Learning Research 7, 551–585 (2006)
3. Farkas, R., Vincze, V., Móra, G., Csirik, J., Szarvas, G.: The conll-2010 shared task: Learning to detect hedges and their scope in natural language text. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pp. 1–12. ACL, Uppsala (2010)
4. Georgescul, M.: A hedgehop over a max-margin framework using hedge cues. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pp. 26–31. ACL, Uppsala (2010)
5. Ji, F., Qiu, X., Huang, X.: Detecting hedge cues and their scopes with average perceptron. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pp. 32–39. ACL, Uppsala (2010)
6. Maron, O., Lozano-Prez, T.: A framework for multiple-instance learning. In: Advances in Neural Information Processing Systems, pp. 570–576. MIT Press, Cambridge (1998)
7. Medlock, B.: Exploring hedge identification in biomedical literature. Journal of Biomedical Informatics 41(4), 636–654 (2008)
8. Medlock, B., Briscoe, T.: Weakly supervised learning for hedge classification in scientific literature. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pp. 992–999. ACL, Prague (2007)
9. Morante, R., Daelemans, W.: Learning the scope of hedge cues in biomedical texts. In: Proceedings of the BioNLP 2009 Workshop, pp. 28–36. ACL, Boulder (2009)
10. Tang, B., Wang, X., Wang, X., Yuan, B., Fan, S.: A cascade method for detecting hedges and their scope in natural language text. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pp. 13–17. ACL, Uppsala (2010)

WeightLOFCC: A Heuristic Weight-Setting Strategy of LOF Applied to Outlier Detection in Time Series Data

Hongrui Xie, Yujiu Yang, and Wenhuang Liu

Information Science and Technology Division,
Graduate School at Shenzhen, Tsinghua University,
Shenzhen 518055, P.R. China
xhr04@mails.tsinghua.edu.cn,
{yang.yujiu,liuwh}@sz.tsinghua.edu.cn

Abstract. Finding outliers is more interesting than finding common patterns in many KDD applications. The local outlier factor method (LOF) is a popular approach to detect outliers, in which a degree of being an outlier will be assigned to each object. In this paper, we present a modification method called WeightLOFCC to better handle outliers in time series data. Differing from the traditional LOF algorithm, the proposed WeightLOFCC method utilizes the idea of semi-supervised learning and weight factor to model data, and makes use of the cross correlation to measure the similarity. We evaluated the proposed algorithm on a large variety of data sets, and the experiment results show that for most of the data sets, our solution for outlier detection can achieve the best performance compared with other classical techniques.

Keywords: WeightLOFCC, time series data, cross correlation, weight method.

1 Introduction

Most studies in KDD focus on finding common patterns. However, for many applications such as rare events detection and network intrusion detection, finding outliers can be more interesting than finding common cases.

A few studies have been conducted on outlier detection for time series data over these years, V. Chandola[1] has presented a comprehensive evaluation of some classical techniques. In this paper, we introduce a modification method on LOF[2] to find outliers in time series data and evaluate this method on the large variety of data sets provided by V. Chandola [1]. The results show that our solution can outperform the existing methods described in [1].

Specifically, our technical contributions in this paper are as following:

1. We explore a WeightLOFCC method to model time series data in a semi-supervised learning way and to measure the similarity using cross correlation.

2. Our experiments are performed on large variety of data sets, which have different characteristics. While most of the outlier detection techniques are shown to be effective for certain types of data and perform poorly on others. Our solution is robust with respect to the 16 different data sets.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 529–536, 2010.

The rest of the paper is organized as follows. In section 2, we discuss related work on outlier detection. In section 3, we introduce the WeightLOFCC and describe our solution in detail. In section 4, we present the experiments and results, and show the effectiveness of our solution. Section 5 concludes our work.

2 Related Work

A large number of outlier detection techniques have been proposed, such as statistical based[3], distance based[4], density based[2] and clustering methods[5]. In addition, a number of techniques have been proposed for related problems. For example, F.Angiulli and F.Fassetti[6] presented a novel definition of outlier in the context of inductive logic programming. C.Yushu and C.Yiming [7] proposed an Adaboost-IHMM method to combine IHMM and adaboost for outlier detection. S.Ando[8] presented an information theoretic analysis of outlier detection.

LOF is the representative method in the density based category. The main idea of LOF is to assign to each data object a degree of being outlier, called the local outlier factor. The local outlier factor of data object o is defined as[2]:

$$LOF_k(o) = \frac{\sum_{p \in N_k(o)} \frac{lrd_k(p)}{lrd_k(o)}}{|N_k(o)|},\ lrd_k(o) = 1 / \left(\frac{\sum_{p \in N_k(o)} reach-dist_k(o,p)}{|N_k(o)|} \right). \tag{1}$$

For the limitation of space, the general framework for LOF algorithm can be referred to the relevant documents [2, 9].

V. Chandola[1] has presented a comprehensive evaluation of the semi-supervised outlier detection techniques for time series data. These techniques can be classified into four categories, kernel based [10], window based [11], predictive techniques[12] and segmentation based [13]. Our solution utilizes the idea of semi-supervised learning as a variant of LOF.

3 Our Weight Strategy in LOF

3.1 LOFCC and SemiLOFCC

For describing the proposed WeightLOFCC method clearly, we will first discuss LOFCC and SemiLOFCC methods in this section.

In LOFCC, cross correlation is used to measure the similarity instead of Euclidean distance, it is better to handle time series data.

The definition of cross correlation[14] is given in equation (2) (continuous and discrete), where f^* denotes the complex conjugate of f .

$$(f * g)(t) \triangleq \int_{-\infty}^{\infty} f^*(\tau) g(t+\tau) d\tau,\ (f * g)[n] \triangleq \sum_{m=-\infty}^{\infty} f^*[m] g[n+m]\ . \tag{2}$$

Similar to the LOF method, we give the definition of $LOFCC_k(o)$ and $lacc_k(o)$. $LOFCC_k(o)$ depicts the average of the ratio of the local average cross correlation of o's k nearest neighbors and local average cross correlation of data object 'o'. $lacc_k(o)$ represents the similar meaning to the local reachability density in LOF algorithm. Since $scc_k(o,p)$ measures the similarity, there is no need to do the inverse.

The local outlier factor based on cross correlation is defined in equation (3) and the general framework for LOFCC algorithm is given as follows.

$$LOFCC_k(o) = \frac{\sum_{p \in Ns_k(o)} \frac{lacc_k(p)}{lacc_k(o)}}{|Ns_k(o)|}, \quad lacc_k(o) = \left(\frac{\sum_{p \in Ns_k(o)} scc_k(o,p)}{|Ns_k(o)|} \right). \tag{3}$$

```
Algorithm 1: The LOFCC method.
    Given: data set $D$
    Set the value of $k$ , which means the number of objects in neighborhood.
    Do:
    1. For each object $o \in D$ , compute $k\text{-}similarity(o)$ and $Ns_k(o)$ .
    2. For each object $o \in D$ and object $p \in Ns_k(o)$ , compute $scc_k(o,p)$ .
    3. For each object $o \in D$ , compute $lacc_k(o)$ .
    4. For each object $o \in D$ , compute $LOFCC_k(o)$ .
    Return LOFCCs for all data objects as a vector.
```

Where, *k-similarity(o)* means the cross correlation to the k-th nearest neighbor, $Ns_k(o)$ denotes k-similarity neighborhood, which is all points in a k-similarity sphere, $scc_k(o, p)$ represents cross correlation between object o and object p, $lacc_k(o)$ depicts the local average cross correlation based on k, $LOFCC_k(o)$ denotes local outlier factor based on cross correlation.

In the real world applications, it is usually not difficult to get normal data. The label could provide important information for outlier detection. Since LOFCC is an unsupervised method, it cannot make use of this information. We introduce the idea of semi-supervise learning and give the main framework of SemiLOFCC as follows.

```
Algorithm 2: The SemiLOFCC method.
    Given: data sets $D_L, D_U$ , represented the labeled data set and unlabeled data set
  respectively, and $D = D_L \cup D_U$ .
    Set the value of k, which means the number of objects in neighborhood.
    Do:
    1. For each object $o \in D$ , compute $k\text{-}similarity(o)$ and k-similarity
       neighborhood $Ns_k(o)$ with respect to the objects in data set $D_L$ .
    2. For each object $o \in D$ and object $p \in Ns_k(o)$ , compute $scc_k(o,p)$ .
    3. For each object $o \in D$ , compute $lacc_k(o)$ .
    4. For each object $o \in D_U$ , compute $SemiLOFcc_k(o)$ .
    Return SemiLOFCCs for all data objects as a vector.
```

3.2 WeightLOFCC

In order to further improve the performance and fully make use of the information provided in labeled and unlabeled data sets, we introduce a heuristic weight-setting strategy into LOFCC algorithm, which combine the LOFCC and SemiLOFCC together. We assign different weights to the labeled data and unlabeled data, and make use of LOFCC and SemiLOFCC to form our WeightLOFCC method.

The weighted local outlier factor based on cross correlation is defined in the equation (4), and the WeightLOFCC algorithm framework is summarized as follows.

$$WeightLOFCC_{k_L,k_U}(o)=\frac{w_1\sum_{p\in N_{L,k_L}(o)}\frac{lacc_{L,k_L}(p)}{lacc_{L,k_L}(o)}}{\left|N_{L,k_L}(o)\right|}+\frac{w_2\sum_{p\in N_{U,k_U}(o)}\frac{lacc_{U,k_U}(p)}{lacc_{U,k_U}(o)}}{\left|N_{U,k_U}(o)\right|},\quad where\quad w_1+w_2=1\,. \tag{4}$$

```
Algorithm 3: The WeightLOFCC method.
  Given: data sets $D_L, D_U$, and $D = D_L \cup D_U$ .
  Set the value of $k_L$, $k_U$, $w_1$, $w_2 = 1 - w_1$.
  Do:
  1. For each object $o \in D$ , compute $k\text{-}similarity_L(o)$ and k-similarity
     neighborhood $Ns_{L,k_L}(o)$ ; $k\text{-}similarity_U(o)$ and $Ns_{U,k_U}(o)$.
  2. For each object $o \in D$ and object $p \in Ns_{L,k_L}(o)$, compute $scc_{L,k_L}(o,p)$ ; for
     each object $o \in D$ and object $p' \in N_{U,k_U}(o)$, compute $scc_{U,k_U}(o,p')$ .
  3. For each object $o \in D$ , compute $lacc_{L,k_L}(o)$ and $lacc_{U,k_U}(o)$ .
  4. For each object $o \in D_U$ , compute $WeightLOFCC_{k_L,k_U}(o)$ .
  Return WeightLOFCCs for all data objects as a vector.
```

Where, D_L means the labeled data set, D_U denotes the unlabeled data set, k_L depicts the size of neighborhood in D_L, k_U means the size of neighborhood in D_U, w_1 denotes the weight assigned to D_L, w_2 represents the weight assigned to D_U, $k\text{-}similarity_L(o)$ depicts k-similarity with respect to the objects in D_L, $k\text{-}similarity_U(o)$ means k-similarity with respect to the objects in D_U, $Ns_{L,KL}(o)$ denotes k-similarity neighborhood in D_L, $Ns_{U,KU}(o)$ represents k-similarity neighborhood in D_U, $scc_{L,KL}(o, p)$ depicts cross correlation between object o and p in D_L, $scc_{U,KU}(o, p)$ means cross correlation between object o and p in D_U, $lacc_{L,KL}(o)$ represents the local average cross correlation in D_L, $lacc_{U,KU}(o)$ denotes the local average cross correlation in D_U.

4 Experiments and Results Analysis

4.1 Performance Evaluation

In this paper, we adopt the accuracy as the performance metric: We first rank the test time series in decreasing order based on the WeightedLOFCCs, then count the number of true anomalies in the top n test time series, where n is the number of true anomalies. Let there be t true anomalous time series in top n ranked sequences. The accuracy is computed as $Accuracy = t / n$.

4.2 Data Sets

We evaluated our solution on 16 different data sets[1] obtained from a broad spectral of application domains and the characteristics are summarized in Table 1[1].

Table 1. Details of different data sets used in the experiments[1]. where, L – Length of sequences, X – Training database, YN – Normal test time series, YA – Anomalous test time series, S – Synchronized (Yes - ✓, No - ×).

Name	L	|X|	|YN|	|YA|	S	Name	L	|X|	|YN|	|YA|	S
M1	1500	10	10	10	×	C1	2500	250	250	25	×
M2	1500	10	10	10	×	C2	2500	250	250	25	×
M3	1500	10	10	10	×	L1	2500	250	250	25	×
M4	1500	10	10	10	×	L2	2500	250	250	25	×
P1	672	11	33	8	✓	E1	250	500	500	50	×
V1	1000	4	4	8	✓	E2	250	500	500	50	×
S1	1614	10	10	10	×	Mi1	360	500	500	50	×
S2	1614	30	30	10	×	Mi2	360	500	500	50	×

These data sets belong to five different data collections: motor current data sets (M1 - M4), power usage data set (P1), NASA valve data set (V1), shape data sets (S1 and S2) and electrocardiogram data sets (the other eight data sets). For the detail explanation of normal and anomalous time series in the datasets, please refer to [1].

4.3 Results Analysis

Experiment Results across All the Methods. Six groups of experiments on LOFCC, SemiLOFCC, WeightLOFCC, and LOF, SemiLOF, WeightLOF were performed, the last two methods are variants of LOF in the same way as SemiLOFCC and WeightLOFCC. For the limitation of space, we didn't give the main framework of them. All the results are summarized in Table 2.

For all the methods, we experimented with different parameter values. For the parameter k, we tried different integer in [1, 30]; for the parameter w1, we tried different value from 0 to 1, with interval 0.1. The best performance was selected to fill in Table 2. In addition, we compute the mean, standard deviation and top counts for each method.

Comparing the series of cross correlation (LOFCC, SemiLOFCC, WeightLOFCC) and the series of Euclidean distance, we can see that for most data sets, the cross correlation measure is consistently better than the Euclidean distance, especially when the time series are not synchronized. That's because cross correlation can capture phase misalignments between time series while Euclidean measure cannot.

Comparing the unsupervised methods, semi-supervised methods and weight-setting methods, we can see that semi-supervise methods are better than unsupervised methods, and the weight-setting methods perform the best.

According to all the three metrics (abbr. Mn, St and Tc), WeightLOFCC not only achieves the best performance among all these LOF adaption methods, but also outperforms other classical techniques evaluated in[1]. KNNC with cross correlation as the distance measure and WINC are the best techniques according to Chandola's experiments[1]. So we compare our experiment results with the results of these two techniques. For the results of other techniques, please refer to [1].

Table 2. Experiment results across all techniques. Where, KC means KNNC with cross correlation, WN represents WINC, S_LOF means SemiLOF, and W_LOF denotes WeightLOF, Mn means the mean value, Sd represents standard deviation and Tc depicts top counts.

	KC	WN	LOF	LOFCC	S_LOF	S_LOFCC	W_LOF	W_LOFCC
M1	0.60	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
M2	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
M3	0.90	*1.00*	*1.00*	*1.00*	0.70	*1.00*	0.90	*1.00*
M4	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
P1	*0.88*	*0.88*	*0.88*	*0.88*	*0.88*	*0.88*	*0.88*	*0.88*
V1	0.75	0.75	*1.00*	0.75	*1.00*	0.75	*1.00*	0.75
S1	*1.00*	*1.00*	0.80	0.90	0.90	0.90	0.90	*1.00*
S2	0.80	0.50	0.50	0.30	0.80	*0.90*	0.80	*0.90*
C1	0.20	0.40	0.12	0.40	0.24	0.40	0.24	*0.44*
C2	0.40	0.32	0.16	0.44	0.20	0.44	0.28	*0.52*
L1	0.40	0.56	0.20	0.08	0.24	*0.60*	0.28	*0.60*
L2	*0.44*	0.28	0.12	0.16	0.20	0.36	0.20	0.36
E1	0.74	*0.78*	0.70	0.50	0.68	0.74	0.72	0.76
E2	*0.30*	0.16	0.20	0.22	0.26	0.24	0.26	0.24
Mi1	0.78	*0.90*	0.66	0.36	0.70	0.84	0.74	0.84
Mi2	*0.94*	*0.94*	0.72	0.24	0.76	*0.94*	0.76	*0.94*
Mn	0.70	0.72	0.63	0.58	0.668	0.75	0.69	0.76
Sd	0.27	0.30	0.36	0.34	0.32	0.26	0.32	0.26
Tc	7	9	6	5	5	8	5	11

Preliminary Analysis of the Parameter Sensitivity for WeightLOFCC. There are two parameters, k and w1, in WeightLOFCC method. We conducted some experiments to examine the parameter sensitivity.

Fig. 1 shows the w sensitivity and k sensitivity respectively. We can see that, the detection accuracies on different data sets change in different way with the increase of w1 and k. For example, for data set like Mi1 and Mi2, the accuracy increases with w1 increases while for data set like C1 and C2, the accuracy decreases with w1 increases. We also take Mi1 and L2 as examples to show the impact of k and w1 on accuracy. The x axis means the size of neighborhood, y axis indicates the accuracy and colored lines express the results by assigning different w1. We observe that k and w1 have totally different impact on these two data sets.

The performance of WeightLOFCC relies on the parameter selection. And in many cases, the optimal parameters are not the same for different datasets. Generally speaking, it seems complicated and difficult to choose the best parameter setting by trial and error.

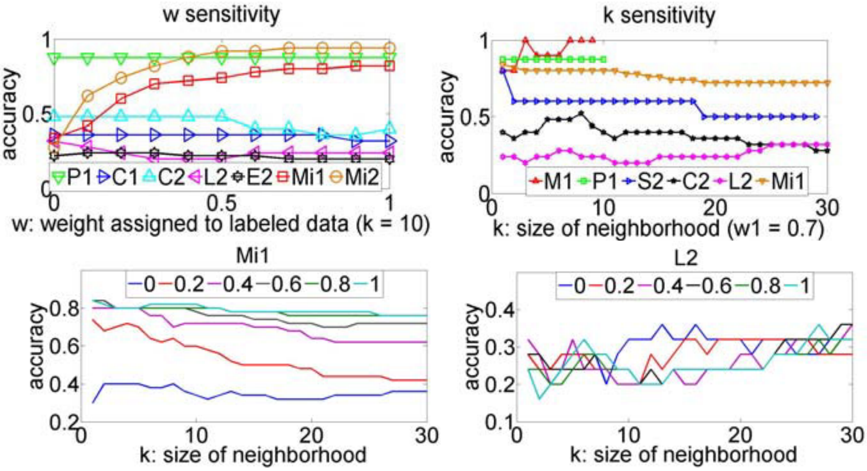


Fig. 1. The parameter sensitivity. Four pictures are presented here to describe the impact of parameter w and k on different datasets.

5 Conclusion and Future Work

In this paper, we give a variant of LOF to better handle outliers in time series data. The proposed WeightLOFCC method utilizes the cross correlation to measure the similarity, and introduces the idea of semi-supervised learning and weight factor to model data. We evaluated the proposed algorithm on a large variety of data sets. The experiment results show that our solution can achieve the best performance according to all the three metrics (mean, standard deviation and top counts on accuracy).

There are also some limitations in current work. The preliminary analysis of the parameter sensitivity indicates that the performance of WeightLOFCC relies on the parameter setting, and the optimal parameters are not identical for different datasets. We investigated many parameters, but exhaustive testing could be costly due to the large parameter space. Automatic parameter setting for a given data set is the ultimate aim. Our study provides a comparably good technique, appears to be a step in this direction. It is an important ongoing work to find a way to choose the best parameter setting without exhaustive testing.

Acknowledgments. This project is supported by Guangdong Provincial Nature Science Foundation (No. 9451805702004O4).

References

1. Chandola, V., Cheboli, D., Kumar, V.: Detecting Anomalies in a Timeseries Database. CS Technical Report 09-004, Computer Science Department, University of Minnesota (2009)
2. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: Identifying Density-Based Local Outliers. In: ACM SIGMOD conference, pp. 93–104. ACM, Dallas (2000)

3. Barnett, V., Lewis, T.: Outliers in Statistical Data. Wiley, New York (1994)
4. Knorr, E., Ng, R.: Finding Intensional Knowledge of Distance-Based Outliers. In: 25th VLDB Conference, pp. 211–222. Morgan Kaufmann, Edinburgh (1999)
5. Sheikholeslami, G., Chatterjee, S., Zhang, A.: Wavecluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases. In: 24th VLDB Conference, pp. 428–439. Morgan Kaufmann, New York (1998)
6. Angiulli, F., Fassetti, F.: Outlier Detection Using Inductive Logic Programming. In: 9th ICDM Conference, pp. 693–698. IEEE Computer Society, Miami (2009)
7. YuShu, C., YiMing, C.: Combining Incremental Hidden Markov Model and Adaboost Algorithm for Anomaly Intrusion Detection. In: ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics, pp. 3–9. ACM, Paris (2009)
8. Ando, S.: Clustering Needles in a Haystack: An Information Theoretic Analysis of Minority and Outlier Detection. In: 7th ICDM Conference, pp. 13–22. IEEE Computer Society, Omaha (2007)
9. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J.: A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. In: 3rd SIAM International Conference on Data Mining, pp. 25–36. SIAM, San Francisco (2003)
10. Wei, L., Keogh, E., Xi, X.: SAXually Explicit Images: Finding Unusual Shapes. In: 6th ICDM Conference, pp. 711–720. IEEE Computer Society, Hong Kong (2006)
11. Keogh, E., Lin, J., Fu, A.: HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence: Algorithms and Applications. In: 5th ICDM Conference, pp. 226–233. IEEE Computer Society, Houston (2005)
12. Ma, J., Perkins, S.: Online Novelty Detection on Temporal Sequences. In: 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 613–618. ACM, Washington (2003)
13. Chan, P.K., Mahoney, M.V.: Modeling Multiple Time Series for Anomaly Detection. In: 5th ICDM Conference, pp. 90–97. IEEE Computer Society, Houston (2005)
14. The definition of cross correlation in Wikipedia, `http://en.wikipedia.org/wiki/Cross-correlation`

TGP: Mining Top-K Frequent Closed Graph Pattern without Minimum Support[⋆]

Yuhua Li, Quan Lin, Ruixuan Li, and Dongsheng Duan

School of Computer Science and Technology
Huazhong University of Science and Technology, Wuhan 430074, P.R. China
linquan.hust@gmail.com,
{idcliyuhua,rxli}@hust.edu.cn,
duandongsheng@smail.hust.edu.cn

Abstract. In this paper, we propose a new mining task: mining top-k frequent closed graph patterns without minimum support. Most previous frequent graph pattern mining works require the specification of a minimum support threshold. However it is difficult for users to set a suitable value sometimes. We develop an efficient algorithm, called TGP, to mine patterns without minimum support. A new structure called Lexicographic Pattern Net is designed to store graph patterns, which makes the closed pattern verification more efficient and speeds up raising support threshold dynamically. In addition, Lexicographic Pattern Net can be stored in the file through serialization, so it doesn't need generate candidate patterns again in the next mining. It is found in the preliminary experiments that TGP can find top-k frequent closed graph patterns completely and accurately. Furthermore, TGP can be extended to mine other kinds of graphs or dynamic graph streams easily.

Keywords: data mining, top-k, frequent closed graph pattern.

1 Introduction

Frequent subgraph mining[1,2,3,4,7] has long been viewed as a challenging problem in data management. From the view of generation of candidate subgraph, there are two typical algorithms: Apriori and FP-growth[5]. Apriori and Fp-growth based mining framework need a min-support threshold to ensure the generation of candidate subgraph correct and necessary. Unfortunately, this framework, though simple, leads to the following two problems.

First, *min_support* is difficult to choose. If the *min_support* is too small, thousands of patterns will be mined out but a lot of which are meaningless. On the other hand, if the *min_support* is too big, large patterns will not be contained in mining result because frequent counts of lager patterns are often

[⋆] This work is supported by National Natural Science Foundation of China under Grant 70771043, 60873225, 60773191. National High Technology Research and Development Program of China under Grant 2007AA01Z403, Natural Science Foundation of Hubei Province under Grant 2009CDB298.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 537–548, 2010.

small. Choosing a suitable *min_support* needs users to try many times, which is a great burden. Second, the mining result contains a lot of redundant graph pattern. According to the Apriori property, all the subgraphs of a big frequent graph pattern are frequent and should be mined out, wasting a lot time but meaningless.

To solve the two problems above, the existing mining framework must be changed. The second problem has been noted and examined by researchers recently proposing to mine frequent closed graph patterns instead[6,8]. Mining closed patterns is more efficient and the mining result is compact without loss of information. Therefore, mining closed pattern will be a good choice for mining frequent patterns.

TSP [11]and TFP [12] have solved the $top - k$ frequent closed item and sequence pattern mining respectively, but mining $top - k$ frequent closed graph pattern is still a problem now because graph mining itself is more complex than item and sequence.

Our research focuses on the pattern of money laundering for detecting suspicious transactions in financial trade network. From domain knowledge we know that money laundering is a kind of group crime. We have a transaction database which contains many detected money laundering cases which can be modeled with trade graphs. Now we need to find $top - k$ frequent transaction patterns which can be regarded as money laundering patterns. There are similar problems in fraud detection and crime clue mining.

To solve these problems, we propose a new mining task: mining $top-k$ frequent closed graph patterns of size bigger than *min_size* without *min_support*. We propose the Lexicographic Pattern Net model based on Lexicographic Sequence Tree, which are used to store candidate graph pattern. Based on Lexicographic Pattern Net, a new efficient algorithm, called TGP, is developed to solve this mining task. Experiments show that TGP solves this new mining task very well. Applying TGP into our problem, we can find out most frequently used money laundering patterns quickly. Among these patterns, some match the domain knowledge very well, and others can be derived indirectly.

The rest of the paper is organized as follows. In Section 2, we define our new mining task and compare the difference between TGP and CloseGraph. In Section 3 we propose a new pattern storage structure Lexicographic Pattern Net and the mining process is described in Section 4. Experimental result is analyzed in Section 5. Extending TGP to mining other kinds of graphs is discussed in Section 6, and we conclude our study and discuss the future works in Section 7.

2 Problem Definition

In this section we propose the new mining task: $top - k$ frequent closed graph pattern. We denote the vertex set of a graph g by $V(g)$, the edge set by $E(g)$, then the graph can be denoted by $G(V(g), E(g))$. Since most of interesting graph patterns are connected graphs, the mentioned graph is simply connected with undirected graph with labeled vertex and labeled edge without multiple edges. Graphs with self-loop and directed graph are discussed in Section 6.

Definition 1. ***Top-k Frequent Closed Graph (TGP)*** *Subgraph g is a* ***frequent graph pattern*** *in a labeled graph dataset D, if its support in D is no less than $min\_support$. A grpah pattern g is a* ***closed graph pattern*** *if there exists no graph pattern g' such that(1)$g \sqsubset g'$ and (2)$support(g) = support(g')$. A closed graph pattern g is a* ***top-k closed graph pattern of minimum size*** *min_size if there exist no more than $(k-1)$ closed graph patters whose size is at least min_size and whose support is higher than that of g.*

Our new mining task adopts the advantages of CloseGraph. Comparing CloseGraph with traditional frequent subgraph, CloseGraph discards some patterns whose supports equal to their super patterns and such super patterns exist in the mining result. So the mining result of CloseGraph is more meaningful. Furthermore, CloseGraph adopts early termination to speed up pruning, so it's more efficient. TGP eliminates redundant subpatterns, and can control the amount of mining results and the size of pattern, which makes the mining result more meaningful. Furthermore, there is no need for user to specify a *min_support*.

3 Pattern Extension and Storage

To solve our new mining task, a new structure called DFS Code Net is proposed to store all the patterns from one graph and another structure called Lexicographic Pattern Net is proposed to store the merge result of DFS Code Nets for all graphs in graph dataset.

3.1 Graph Representation

Han, J. and Yan X. have used DFS Subscripting and Right-Most Extension [4] to generate DFS Codes for a graph. According to DFS Lexicographic Order, there must be a minimum DFS Code for the graph, which is used to denote the graph uniquely. This method has been proved to be an efficient coding technology especially in algorithms based on FP-growth. Our TGP algorithm is also based on FP-growth, so we also denote a graph by its minimum DFS Code.

3.2 DFS Code Net

DFS Code Tree has been used widely in graph mining algorithms based on FP-growth. It is a hierarchical tree structure for storing graph patterns. Every node of DFS Code Tree stores a DFS Code, which represents a graph pattern. The Pattern its child nodes represent is its supper pattern. But this structure is not perfect to mine closed pattern, because closed pattern verification need compare its frequent count with all of its direct subpatterns. We cannot get frequent count directly from DFS Code Tree structure because a pattern has several codes according to different extension orders as shown in Fig. 1(a).

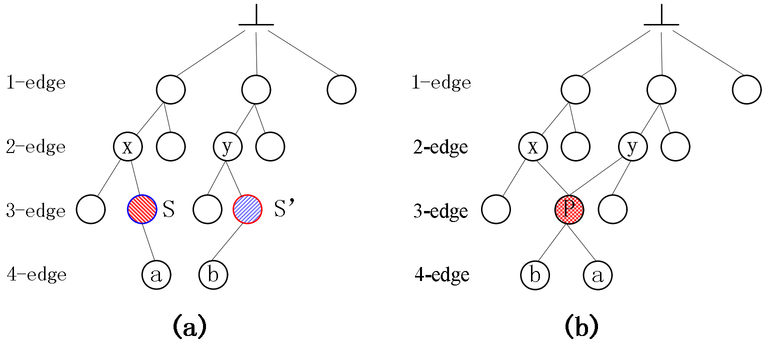


Fig. 1. (a) is a DFS Code Tree, nodes S and S' present for same graph pattern. (b) is a DFS Code Net, nodes S and S' in (a) are merged into node P. There exist no other nodes presenting the same graph pattern with P.

The DFS code of S and S' are different but represent same pattern P. If we want to get all the sub patterns of P, we have to find all P's DFS Code nodes and their parents such as node x and node y, but x and y may present for same graph pattern. So it is inefficient. We improve the DFS Code Tree to DFS Code Net, which is a hierarchical net structure, as Fig. 1(b)shows.

Algorithm 1. *Generate_DFS_Code_Net*(G)

```
input  : Graph $G$
output: DFS Code Net $D$
Initial DFS Code Net $D$ to a Null Net;
foreach Edge e of G do
    Create Node $N(C)$,insert$(N)$;
    for Node N in Net D do
        if N can be extended then
            Pattern $P$= $Extend(N)$,$C$=$DFS\_Code(P)$;
            if exist Node $N_{child}$ and $DFS\_Code(N_{child})$ represent $P$ then
                $DFS\_Code(N_{child}) = min(C, C')$,$N'child = N_{child}$;
            end
            else create Node NewN$(C)$,insert$(NewN)$, $N'child = NewN$;
        end
        else break;
    end
end
return $D$
```

DFS Code Net construction as algorithm 1 is a NP-Complete problems, it has the same time complexity with DFS Code Tree [4] with small additional cost. A node of DFS Code Net stores the minimum DFS Code of a graph pattern, and it is unique. If the graph size of g is n, then there is just one $n-th$ level node storing the minimum DFS Code of g, representing graph g itself.

3.3 Lexicographic Pattern Net

Lexicographic Pattern Net stores all patterns and their frequent counts for graph set, which is improved on the Lexicographic Sequence Tree. The structure of Lexicographic Pattern Net is the same as the DFS Code Net, but the node of Lexicographic Pattern Net stores the pattern's frequent count and graph ID which contains the pattern. Lexicographic Pattern Net is generated from DFS Code Nets by a kind of incremental method. Fig. 2 shows two graph g_1,g_2 and their DFS Code Nets N_1,N_2.

Algorithm 2 describes how to insert a DFS Code Net to a Lexicographic Pattern Net.

Algorithm 2. *Insert_DFS_Code_Net*(D,L)

```
input  : DFS Code Net $D$,Lexicographic Pattern Net $L$
output: Lexicographic Pattern Net $L$ after inserting

get root node $R$ of $D$;
if $R$ can be found in $D$ then
    foreach Node $N$ of subnet $R$ of $D$ do
        frequent++;
    end
    return $L$;
end
else
    insert $R$ to $L$ by lexicographic order;
    if $R$ has children then
        foreach subpattern $R'$ of $R$ do
            call $Insert\_DFS\_Code\_Net(R', L)$;
        end
    end
    else return $L$;
end
```

Algorithm 2 bases on an easy theory, that is if the root nodes of two DFS Code Nets are same, they represent for same graph pattern, so the two DFS Code Nets must be same. By this theory we can calculate the frequent count of nodes of Lexicographic Pattern Net quickly. In the inserting process if the root of DFS Code Net can be found in the Lexicographic Pattern Net, we can insert this net at one time just by raising the frequent count of nodes of corresponding subnet in Lexicographic Pattern Net. The Lexicographic Pattern Net N_{12} in Fig. 2 shows the Lexicographic Pattern Net generated by N_1 and N_2.

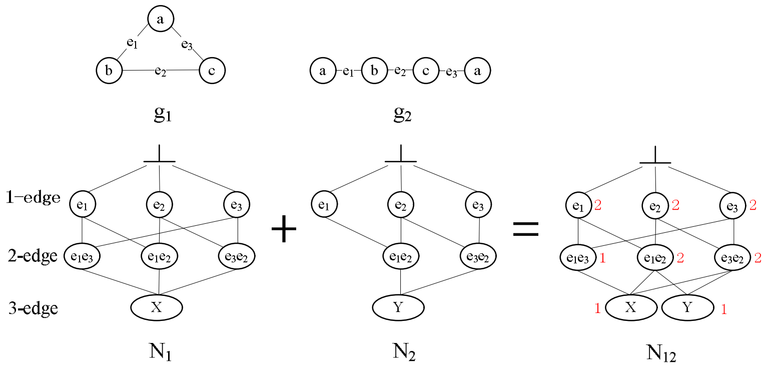


Fig. 2. DFS Code Nets N_1 and N_2 are generated from graphs g_1 and g_2 respectively. Nodes X and Y represent for the whole graphs g_1 and g_2. The Lexicographic Pattern Net N_{12} is merged from DFS Code Nets N_1 and N_2. The red numbers beside nodes are their frequent counts.

4 Mining Process

Section 3 introduces how to generate a Lexicographic Pattern Net from a graph set. In this section we explain step by step how to use Lexicographic Pattern Net to mine $top-k$ frequent closed graph pattern.

4.1 Closed Graph Pattern Verification

Our task is mining $top-k$ closed patterns, so we should guarantee that at least k closed patterns can be found. CloSpan [13] stores candidates for closed patterns during the mining process and in the last step it finds and removes the non-closed ones. This approach can not be used in mining $top-k$ closed patterns because it needs to know which patterns are closed and accumulates at least k closed patterns before it starts to raise the minimum support. Thus closed pattern verification must be done during the mining process. CloseGraph adopts Early Termination and Detecting Failure of Early Termination to find closed patterns directly during the mining process, but it still has the above problem. Lexicographic Pattern Net solves this problem very well. We can get all the direct superpatterns of any pattern, and verify that this pattern is closed quickly.

For example, if we want to verify whether the pattern e_2 in Fig.2 is closed or not, we get its direct superpatterns e_1e_2 and e_3e_2. Their frequent count is the same, so the pattern e_2 is not closed. Comparing frequent count e_3 with e_1e_3, they are different, so e_3 is closed. Lexicographic Pattern Net make closed pattern verification easily, that's why we cost a lot of time to construct it.

4.2 Applying Minimum Size Constraint

DFS Code Net and Lexicographic Pattern Net is a level structure, so the depth of node is the size of pattern. Thus it is easy to apply minimum size constraint by add a controlled condition in Algorithm 2. That is when the size of R is *min_size*, the subpatterns of R need not be inserted any more.

4.3 Find *top* − *k* Patterns

In order to mine most frequent closed patterns, an appropriate minimum support should be found to judge whether a pattern is a $top - k$ frequent pattern or not. We adopt a Support Threshold Raising method to find the minimum support. To raise the minimum support quickly and correctly, a simple data structure called *closed_pattern_order_array* is used to store current $top - k$ closed graph pattern ordering by their frequent count. The size of *closed_pattern_order_array* is equal or greater than k according to the definition of TGP. When new patterns insert into it, low frequent patterns are removed to make sure it always store top-k frequent closed patterns already known.

Algorithm 3. $top - k(L, min\_size, k)$

```
input  : Lexicographic Pattern Net $L$, minimum size $min\_size$, $top - k$
output: closed_pattern_order_array $A$ contains $top - k$ patterns

foreach Pattern $P$ of $L$ at $min\_size$ level do
    if $P$ is a closed Patten then insert $P$ to $A$;
    else
        Find Closed Patterns from $P$'s children;
        insert to $A$;
    end
end
remove unfrequent Patterns in $A$,keep size of $A$ to $k$;
$min\_frequent$=minimum $frequent$ of Patterns in $A$;
while $A$ changed do
    foreach Pattern $P$ of $A$ do
        Find Closed Patterns from $P'$s children that
        $frequent > min\_frequent$; insert to $A$;
        remove unfrequent Patterns in $A$,keep size of $A$ to $k$;
        $min\_frequent$=minimum $frequents$ of Patterns in $A$;
    end
end
return $A$;
```

The length of *closed_pattern_order_array* is bigger than or equal to k, and it may change during the mining process. Algorithm 3 shows how to generate the final *closed_pattern_order_array* by scanning the Lexicographic Pattern Net selectively.

Algorithm 3 need not scan every node of Lexicographic Pattern Net, and the Lexicographic Pattern Net makes the checking of closed pattern very easy. According to Apriori, the frequent count of sub pattern is small or equal to that of their parents, which reduces the search space. Experiments show that Algorithm 3 is efficient.

4.4 TGP Algorithm

Algorithm 4 describes our new mining framework. This new framework generates a Lexicographic Pattern Net which contains all sub patterns and the relationships between them. When Lexicographic Pattern Net is constructed, it can be reused for different mining task. For example, we can edit mining tasks line 6 in Algorithm 4. Experiments indicate that the runtime of line 6 is less than 1% of the whole runtime, so this new framework has a good performance in multitask.

Algorithm 4. $TGP(D, min\_size, k)$

```
  input  : graph set $D$,minimum size $min\_size$,$top - k$ value k
  output: closed_pattern_order_array $A$ contains $top - k$ patterns
1 Initial Lexicographic Pattern Net $L$;
2 foreach graph $g$ of $D$ do
3     DFS Code Net $N$= DFS_Code_Net($g$);
4     Insert_DFS_Code_Net($N$,$L$);
5 end
6 closed_pattern_order_array $A$=top-k($L$,$min\_size$,$k$);
7 return $A$;
```

5 Experimental Evaluations

A comprehensive performance study has been conducted in our experiments on money laundering case dataset, synthetic datasets and processed chemical compound datasets. TGP algorithm is implemented in java, compiled by Sun's stand compiler V1.5, and interpretively executed by Sun's stand java VM V1.5. All the experiments are done on a 2.4GHZ Intel Pentium-4 PC with 1GB main memory, running Windows XP.

Money Laundering Case Dataset. We collect 200 money laundering cases. Regarding the characteristics of money laundering, several important attributes are extracted such as money amount, account type, etc. After pretreatment, these cases are modeled as 200 graphs which contain 10 nodes and 15 edges averagely. For covering up money laundering, financial criminals usually adopt some tricks such as laundering large amounts through several transactions or through long path. By analyzing money laundering action, it's found that most of them consist of three or more transactions. So we set min_size=3, $top - k$=20. Comparing the mined-out patterns with research results of financiers, most of these patterns such as Fig. 3 (a), (b) and (c) show match very well, some can be derived indirectly and some doesn't match very well. These mismatched patterns as Fig. 3(d) shows are proved to be new money laundering patterns which have not been abstracted by financiers.

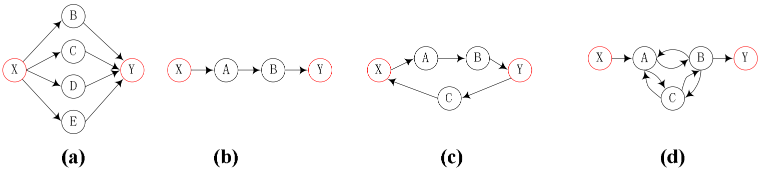


Fig. 3. (a), (b) and (c) are patterns matching Anti Money Laundering and(d) are patterns occurring frequently in real world

Synthetic Dataset. The synthetic datasets are generated using a graph procedure called GraphGen designed by us. The procedure has 3 parameters: G, N and E. Parameter G controls the amount of output graph and every graph contains N nodes and E edges averagely. Two contrast experiments have been conducted to analyze the performances of TGP on different graphs. The datasets of the first contrast experiment are generated by GraphGen with parameter G=300, N=10, E=10∼16 in Fig. 4(a) shows the experiment results.

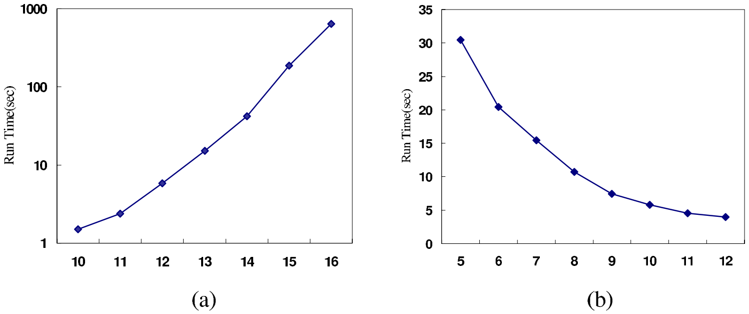


Fig. 4. (a) shows the TGP performance for different graph size (average size from 10 to 17). (b) shows performance for different graph density (graph size is fixed to 12, nodes count form 5 to 12).

The result shows that runtime grows exponentially as the graph size increases because it is a NP-Complete problem.

The datasets of the second contrast experiments are generated by GraphGen with parameter G=300, N=5∼12, E=12. This dataset contains graphs of different density. As Fig. 4(b) shows, when graphs are more density, it costs more times. For graphs of same size, if graph contains fewer nodes, it contains more sub graphs. For example, for a 5 nodes complete graph, it contains 25 sub graphs while for a 5 nodes tree, it contains 15 sub graphs at most. So it costs more time during pattern extension and combination.

Processed Chemical Compound Dataset. The chemical compound dataset can be retrieved through this URL.[1] The original dataset contains 340 compounds, 24 different atoms, 66 atom types, and 4 types of bonds. The dataset

[1] http://oldwww.comlabox.ac.uk/oucl/groups/machlearn/PTE.

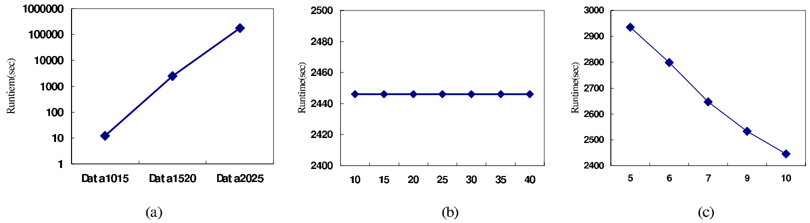


Fig. 5. (a) shows performance on three different data sets. (b) shows performance for different *top* − *k* values on Data1520. (c) shows performance for different *min_size* value on Data1520

is sparse, containing average 27 vertices per graph and 28 edges per graph. The largest one contains 214 edges and 214 vertices. Because TGP have no *min_support* to pruning, complete *DFS_Code_Net* must be generated. But generating complete *DFS_Code_Ne*t is a NP-Complete problem, it's impossible to generate *DFS_Code_Net* for big graph. Experiment shows that when graph has more than 25 edges, the runtime is longer than 4000 seconds. Furthermore, 3 synthetic datasets are generated: Data1015 (the graph size are from 10-15 at random), Data1520 (the graph size are from 15-20 at random) and Data2025 (the graph size are from 20-25 at random).

Fig. 5(a) shows the runtime for the 3 test dataset for mining top-20 frequent closed graph with the minimum size 10. This result shows that runtime grows linearly with the graph size.

Fig. 5(b) shows the runtime for different *top* − *k* values on Data1520. The runtime changes very little for different *top* − *k* values because mining patterns on the Lexicographic Pattern Net cost less than 1% time of the whole runtime.

Fig. 5(c) shows the runtime for different *min_size* values on Data1520. *Min_size* determines which levels of the DFS Code Net should be combined to Lexicographic Pattern Net. The bigger the *min_size* is, the smaller the subnet to be combined is. But combining costs litter time to DFS Code Net generation, the whole runtime doesn't change much.

6 Discussion

So far, we have proposed and investigated a general framework, TGP, for mining closed, labeled connected, undirected, frequent subgraph patterns. But in real world, graph is various. Here we show how the framework can be extended to mine other kinds of graphs.

1. **Mining directed graphs.** We have used Extinction Direction Code [14] to solve mining directed graph pattern. This method improves DFS Code by adding a direction code. If extension direction is the same with the edge direction, the direction code is 1, otherwise, it is 0.
2. **Mining unlabeled and partially labeled graphs.** For this kind of graph, we can transform it to labeled graph by adding specified label (this label does not appear in graph) to unlabeled nodes and edges.

3. **Mining non-simple graph.** Non-simple graphs may have self-loop and multiple edges. We can change extension order to solve it. In order to accommodate self-loops, the growing order can be changed to backward edges, self-loops and forward edges. Multiple edges can appear in these three kinds of edges. If we allow two neighbored edges in a DFS code to share the same vertices, actually the definition of DFS lexicographic order can accommodate multiple edges.
4. **Mining graph stream.** Dynamic graph stream [16] is a special graph set. Different from normal graph set, the graph count of graph stream is not fixed, so the mining framework based on Apriori cannot work on this kind of dataset. TGP algorithm can solve this problem very well because it is incremental. All graph pattern information in it is stored in the Lexicographic Pattern Net, and added into Lexicographic Pattern Net one by one. TGP is a stream framework. And for dynamic data, time tags can be add to Lexicographic Pattern Net nodes for mining frequent graph patterns in a time slice.

7 Conclusion

In this paper, we have studied the problem of mining $top - k$ frequent closed graph patterns with size no less than *min_size* and proposed an efficient mining algorithm TGP. This new algorithm adopts Lexicographic Pattern Net to store patterns and relationship between patterns, which makes close pattern verification easily during the mining process. So the algorithm can raise *min_support* correctly and ensures the complete and right results. Comparing TGP with CloseGraph, TGP doesn't need users to provide a proper *min_support* which is often difficult to set and it can find specified number of most frequent closed graph patterns. In addition, TGP can run with a parameter *min_size* to filter out small patterns. So the patterns mined out by TGP are practical and in many cases more preferable than the traditional minimum support threshold based graph pattern mining. Moreover TGP mining framework is incremental, so it is suitable for mining graph stream.

We have done experiment in financial dataset and received good results. In future, we will widen the range of TGP application, such as mining most frequent molecular formula in poisonous chemical substances dataset to find poisoning molecular structure. TGP is a NP-Complete problem and it cannot prune before extending patterns to *min_size* for ensuring its veracity and completeness. So for huge graph it doesn't work well. In future we will study how to sacrifice some accuracy and completeness to prune during generating DFS Code Net to apply TGP on huge graphs. Moreover, the extension of TGP for mining other complicated structured patterns, such as directed graph pattern, unlabeled and partially labeled graph pattern, is valuable for future research.

References

1. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: First IEEE International Conference on Data Mining (ICDM 2001), pp. 313–320. IEEE Computer Society, San Jose (2001)

2. Vanetik, N., Gudes, E., Shimony, S.E.: Computing Frequent Graph Patterns from Semistructured Data. In: Second IEEE International Conference on Data Mining (ICDM 2002), pp. 458–465. IEEE Computer Society, Maebashi City (2002)
3. Han, J., Wang, W., Prins, J.: Efficient Mining of Frequent Subgraph in the presence of Isomorphims. In: 3rd IEEE International Conference on Data Mining (ICDM 2003), pp. 549–552. IEEE Computer Society, Melbourne (2003)
4. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Second IEEE International Conference on Data Mining (ICDM 2002), pp. 721–723. IEEE Computer Society, Maebashi City (2002)
5. Christian, B.: An Implementation of the FP-growth Algorithm. In: Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations (OSDM 2005), pp. 1–5. ACM, Chicago (2005)
6. Han, J., Yan, X.: CloseGraph: Mining Closed Frequent Graph Patterns. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003), pp. 286–295. ACM, Washington (2003)
7. PTamas, H., PJan, R., Stefan, W.: Frequent subgraph mining in outerplanar graphs. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), pp. 1097–1111. ACM, Philadelphia (2006)
8. Yan, X., Zhou, J., Han, J.: Mining closed relational graphs with connectivity constraints. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD 2005), pp. 324–333. ACM, Chicago (2005)
9. Thomas, L.T., Valluri, S.R., Karlapalem, K.: MARGIN: Maximal Frequent Subgraph Mining. In: Proceedings of the Sixth International Conference on Data Mining (ICDM 2006), pp. 1097–1101. IEEE Computer Society, Hong Kong (2006)
10. Wang, N., Parthasarathy, S., Tan, K., Tung, A.K.H.: CSV: visualizing and mining cohesive subgraphs. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (KDD 2008), pp. 445–458. ACM, Vancouver (2008)
11. Wang, J., Han, J., Lu, Y., Tzvetkov, P.: TFP: An Efficient Algorithm for Mining Top-K Frequent Closed Itemsets. IEEE Trans. Knowl. Data Eng. 17(5), 652–664 (2005)
12. Tzvetkov, P., Yan, X., Han, J.: TSP: Mining Top-K Closed Sequential Patterns. In: 3rd IEEE International Conference on Data Mining (ICDM 2003), pp. 347–354. IEEE Computer Society, Maebashi (2003)
13. Yan, X., Han, J., Afshar, R.: CloSpan: Mining Closed Sequential Patterns in Large Databases. In: SIAM International Conference on Data Mining (SDM 2003), San Francisco, pp. 166–177 (2003)
14. Li, Y., Lin, Q., Zhong, G.: Duan. D.: A Directed Labeled Graph Frequent Pattern Mining Algorithm based on Minimum Code. In: The 3rd International Conference on Multimedia and Ubiquitous Engineering (MUE 2009), pp. 353–359. Conference Publishing Services, Qingdao (2009)
15. Maunz, A., Helma, C., Kramer, S.: Large-scale graph mining using backbone refinement classes. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009), pp. 617–626. ACM, Paris (2009)
16. Muthukrishnan, S.: Data streams: algorithms and applications. In: 14th ACM-SIAM Symposium on Discrete Algorithms (SODA 2003), pp.413–413. ACM, Baltimore (2003)

Research on Similarity Matching for Multiple Granularities Time-Series Data

Wenning Hao, Enlai Zhao, Hongjun Zhang,
Gang Chen, and Dawei Jin

Engineering Institue of Corps of Engineers, PLA University of Science
& Technology, 210007 Nanjing, China
{hwnbox,lgdxzel,jsnjzhj}@163.com, gchen1027@sina.com,
dave_nj@163.com

Abstract. Because of the appropriate algorithm of measuring multiple granularities time-series is few, this article advanced a similarity matching algorithm for multiple granularities time-series, which based on the ideal of time calibrator and hypothesis test. It firstly expounded the definition of multiple granularities time-series, and proposed a sample of distance; secondly, it put forward the similarity matching algorithm of multiple granularities time-series; finally, the experimental result proved that the algorithm can effectively reflect the time-series of multiple granularities.

Keywords: time-series, similarity matching, time granularity, distance measure.

1 Introduction

Time series data is ubiquitous in science, engineering and business. Time-series data are the sequences of real values sampled at a fixed time interval, and the database storing time-series database [1]. The similar sequence matching is a foundational issue all of the temporal data mining, moreover, many further analysis and mining are based on it. Similar sequence matching usually can be classified into two categories: whole sequence matching, WSM; subsequence matching, SM.

How to measure the difference between the two time-series data is an important issue of similar sequence matching. Generally, the difference is used to describe by distance measure or similarity measure. In order to measure the similarity of any two sequences, at present, there are many distance measure, such as L_p distance function[2][3], Dynamic Time Warping[4][5], Longest Common Subsequence[6], Edit Distance on Real Sequence, and Edit Distance with Real Penalty[7][8], etc.

Time granularity is a measuring unit of time-series data, and usually the temporal type is used as time granularity [9]. In similar sequence matching, time description and division is an important issue. There is second, minute, hour, day, mouth, year, etc. of time granularity in common use. In some practical applications, they often use multiple time granularities to describe the same problem, for instance, commonly used the *second* and the *day* as the time granularity to describe the trend of the stock price change in the stock statistics. However, the distance measure which mentioned above is useless to measure the two different granularities of time-series data.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 549–556, 2010.

In this paper, the problem of similarity matching of the two different granularities time-series data has been studied. First definitions are made, and a new distance measure is put forth in Section II; then the similarity matching algorithm is based on hypothesis test and time calibration is presented in Section III; and experimental results are achieved which verified the algorithm, and algorithm performance is analyzed in Section IV; finally, a summary and directions for further study are given in Section V.

2 Related Work

This article discusses the different granularities time-series data which are numeric, one variable and time interval.

Definition 1: Given the limited time granularities set TG= {$tg_1,tg_2,\ldots,tg_p$, p∈**N+**}, time granularity conversion matrix $T=\begin{pmatrix} t_{11} & \cdots & t_{1j} & \cdots & t_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ t_{i1} & \cdots & t_{ij} & \cdots & t_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ t_{p1} & \cdots & t_{pj} & \cdots & t_{pp} \end{pmatrix}$, the mapping of any of the two time granularity tg_i and tg_j are met:

$$T_{ij} \mapsto T_{ij}(tg_i,tg_j)=t_{ij} \tag{1}$$

In (1), $t_{ii}=1$, $t_{ij}=\dfrac{1}{t_{ji}}$, and $t_{ij}=t_{ik}*t_{kj}$. $i,k,j=1,2,...p$.

For example, $t_{\sec\ \min}=60$, $t_{\min\ hr}=60$, so $t_{\min\ \sec}=1/60$, $t_{\sec\ hr}=t_{\sec\ \min}*t_{\min\ hr}=3600$.

Definition 2: Given two different granularities of time-series data $\boldsymbol{S_a}$, $\boldsymbol{S_b}$:

$\boldsymbol{S_a}$= {$s_{a1},s_{a2},\ldots,s_{am}$}, time granularity is tg_a, the number of $\boldsymbol{S_a}$ is m;

$\boldsymbol{S_b}$= {$s_{b1},s_{b2},\ldots,s_{bn}$}, time granularity is tg_b, the number of $\boldsymbol{S_b}$ is n, assumption that m<n.

According to the time granularity conversion matrix T, the original time-series data $\boldsymbol{S_a}$, $\boldsymbol{S_b}$ can be transformed into:

$\begin{cases} S'_a=\{s'_{a1},s'_{a2},\ldots,s'_{an}\} \\ S_b=\{s_{b1},s_{b2},\ldots,s_{bn}\} \end{cases}$, distance sample ***Dis***= ($dis_1,dis_2,\ldots,dis_n$), in which $dis_i=\dfrac{s'_{ai}}{s_{bi}}$.

3 Similarity Matching Algorithm for Multiple Granularities Time-Series

Similarity is an extremely common phenomenon in nature. In philosophy, similarity is the dialectical unity of the sameness and variation about the objective things. In the similarity matching process, when most points of the corresponding time have a large extent of similarity, and only a few ones have less similarity, that the two time-series are likely similar. However, because the sampling interval of the two time-series data which are matching their similarity, i.e. the granularity of the two time-series is different, to measure the similarity of them by original data is impossible. Therefore, we must calibrate the different granularity time-series before similarity matching, that is to convert them into the same time granularity.

SMfMGT algorithm consists of two sub-algorithms: Preprocessing algorithm and Tester algorithm. Firstly the Preprocessing algorithm is used to convert different granularity time-series into a same time granularity, and remove the unnecessary fluctuation by *moving average*. Then the results, which processed by Preprocessing algorithm, are entered into the Tester algorithm to calculate the test statistics. Finally a conclusion is arrived at on the judgment by test statistics. The following two sections will explain the functions and implementation of these two sub-algorithms.

3.1 Preprocessing Algorithm

Because of the different time granularities of the two time-series $\boldsymbol{S_a}$ and $\boldsymbol{S_b}$, their items value can't be directly comparable and calculate the similarity. Hence, it is necessary to calibrate time granularity firstly before matching similarity of the two time-series. *Preprocessing algorithm* firstly convert the coarse granularity time-series into the fine granularity time-series by the method of interpolation according to the conversion relationship of different granularity of the two time-series. The flow char of *Preprocessing algorithm* is as follows:

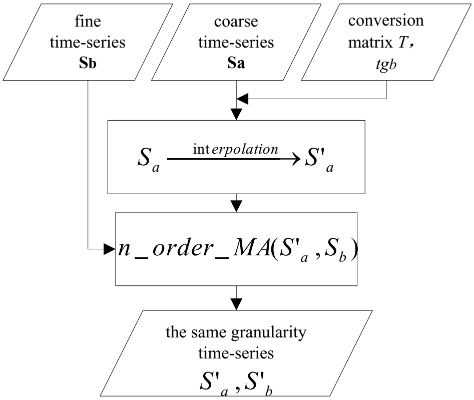

fine time-series Sb
coarse time-series Sa
conversion matrix T, tgb
$S_a \xrightarrow{interpolation} S'_a$
$n\_order\_MA(S'_a, S_b)$
the same granularity time-series
S'_a, S'_b

Fig. 1. Flow char of Preprocessing algorithm

3.2 Tester Algorithm

Assume two time-series data that have been processed by Preprocessing algorithm:

$S'_a = \{s'_{a1}, s'_{a2}, ..., s'_{an}\}$, time granularity is tg_b, the number of $\boldsymbol{S'_a}$ is n;

$S_b = \{s_{b1}, s_{b2}, ..., s_{bn}\}$, time granularity is tg_b, the number of $\boldsymbol{S_b}$ is n.

If two time-series data are similar, then at the same i time, s'$_{ai}$ and s$_{bi}$ should be the same or approximately same, i.e. the value of $dis_i = \frac{s'_{ai}}{s_{bi}}$ should approximately equal to 1. It is thus clear that for two similar time-series data ***S'***$_a$ and ***S***$_b$, most dis_i should fall close to 1, few ones depart from 1 great. On the basis of the statistical laws, the distance sample $Dis=(dis_1,dis_2,...,dis_n)$ should have the characteristics of normal distribution, i.e. $Dis \sim N(u_0,\sigma^2)$, in which u_0=1. The follow steps of Tester algorithm is to determine whether ***S'***$_a$ and ***S***$_b$ is similar or not by hypothesis test. Consider each element of *Dis* from a same sample of the same general, and the sample size is *k*. Advance the following assumptions:

H_0 u$=u_0=1$(Two time-series data is similar)

H_1 u$\neq u_0=1$(Two time-series data is not similar)

At a significance level α, when the test statistic $T=\frac{|\overline{dis}-u_0|}{S*/\sqrt{n}}$ is bigger than $t_{1-\frac{\alpha}{2}}(n-1)$, then refused to H_0, i.e. the two time-series is not similar. In test statistic T, $\overline{dis}=\frac{\sum_{i=1}^{n} dis_i}{n}=\frac{\sum_{i=1}^{n}\frac{s'_{ai}}{s_{bi}}}{n}$, $S*=\sqrt{\frac{\sum_{i=1}^{n}(dis_i-\overline{dis})^2}{n-1}}$, α is the significance level, which is the occurrence probability of a small probability event. In this paper, we take α=0.05. When *n* is bigger than 45, $t(n)$ will approximatively obey standard normal distribution. The flow chart of Tester algorithm is shown below:

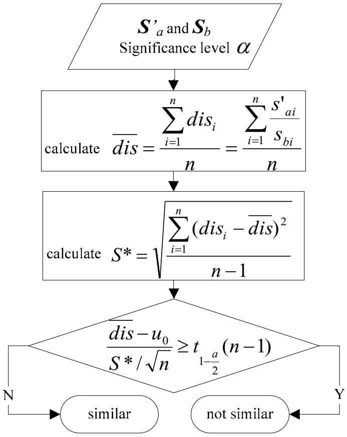

S'a and Sb
Significance level α
calculate $\overline{dis}=\frac{\sum_{i=1}^{n} dis_i}{n}=\frac{\sum_{i=1}^{n}\frac{s'_{ai}}{s_{bi}}}{n}$
calculate $S*=\sqrt{\frac{\sum_{i=1}^{n}(dis_i-\overline{dis})^2}{n-1}}$
$\frac{\overline{dis}-u_0}{S*/\sqrt{n}}\geq t_{1-\frac{a}{2}}(n-1)$
N
Y
similar
not similar

Fig. 2. Flow char of Tester algorithm.

4 Experimental Results and Analysis

4.1 Experimental Results

Experimental data come from http://www.bundesbank.de/[10], which is the time-series WU0053: Gross sales of domestic debt securities at nominal value/Total under stock market. The data are taken from 1990 to 2008. After processing, the time-series S_a, whose time granularity is half year, have 38 items in total; the time-series S_b, whose time granularity is quarter, have 76 items in total. The broken-line figure (the unit of vertical axis value is million) of the different granularity time-series S_a and S_b is follow:

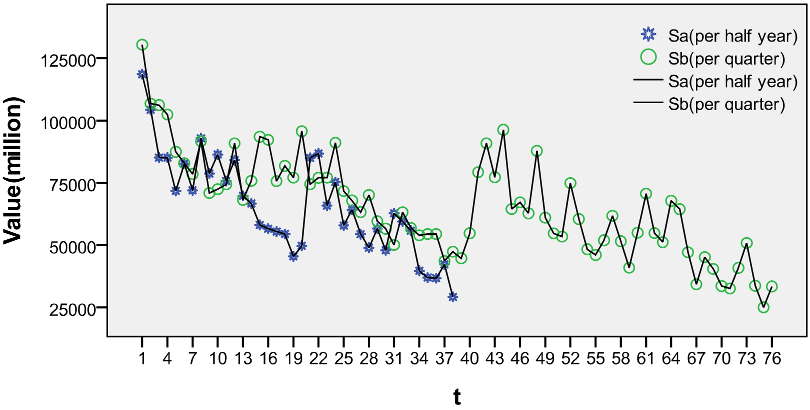


Fig. 3. Different time granularities of time-series data. the abscissa *t* value of time-series S_a range from [1,38], and the granularity of S_a is *half year*; the abscissa *t* value of time-series S_b range from [1,76], and the granularity of S_b is *quarter*. It is obvious that it is difficult to measure the two different granularities time-series by the traditional distance measure.

In Preprocessing algorithm, time-series S_a make use of mean interpolation to insert corresponding items, and the items of new time-series S'_a are corresponding with items of time-series S_b; then use the second-order *moving average* to smooth the fluctuation of the two time-series. The results of Preprocessing algorithm are shown in Fig.4 as follows:

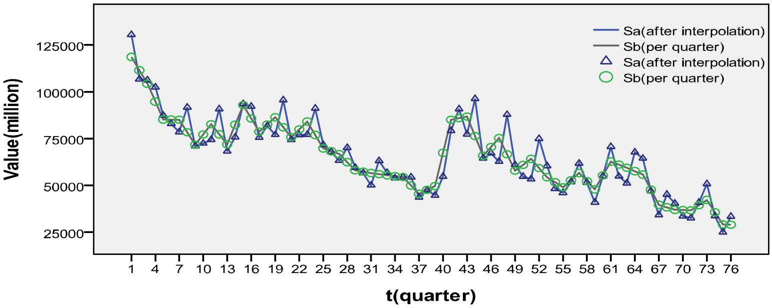


Fig. 4. Time-series Sa processed by Preprocessing algorithm

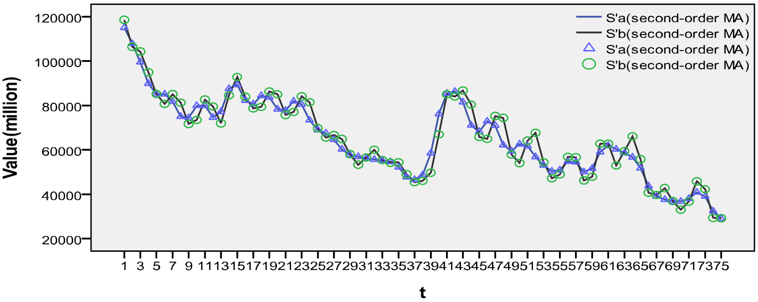


Fig. 4. (*continued*)

The results that have been disposed by Preprocessing algorithm are put into Tester algorithm to test whether the two time-series are similar or not. In this paper, it sets α =0.05, i.e. $t_{1-\frac{\alpha}{2}}(k-1)=t_{0.975}(73)\approx u_{0.975}=1.96$, test statistic $T=\frac{|\overline{dis}-u_0|}{S^*/\sqrt{k}}=\frac{|0.99799-1|}{0.072701/\sqrt{75}}$ = 0.23841. Because of $\mathrm{T}<t_{1-\frac{a}{2}}(k-1)$, the two time-series $\boldsymbol{S}_a$ and $\boldsymbol{S}_b$ are similar. As the experimental data are selected from the same debt securities, in essential, the two time-series are sampled from a same thing at different time granularity. Hence, it is not difficult to explain the similarity of two time-series sampled form the same thing. The frequency histogram of $\frac{s'_{ai}}{s_{bi}}$ as follows:

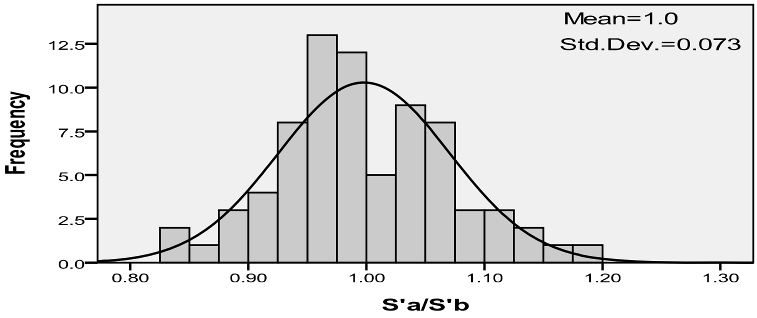


Fig. 5. Histogram of $\frac{s'_{ai}}{s_{bi}}$. the values of $\frac{s'_{ai}}{s_{bi}}$ are between 0.8 and 1.2, the black line is the Gaussian distribution whose mean is 1 and standard deviations is 0.073. It is obvious from Fig.5 that the majority values of $\frac{s'_{ai}}{s_{bi}}$ fell about 1, that is, most of $\frac{s'_{ai}}{s_{bi}}$ are the same; only few values are far different.

4.2 Algorithm Analysis

SMfMGT algorithm solves the problem of the similarity measure of different granularities time-series data. The algorithm is simpleness of distance measure; has multi-selectivity of time-calibrate methods; has good robustness, and high efficiency of calculation. Preprocessing algorithm, which is the sub-algorithm of the SMfMGT algorithm, has many methods to choose. Users can choose one of the interpolation methods, interpolation methods include polynomial interpolation, hermite interpolation, piecewise interpolation, spline interpolation, etc; the algorithm is robust, it is impossible that the similarity of the two time-series $\boldsymbol{S'_a}$ and $\boldsymbol{S'_b}$ is low because that the value of S'_{ai} and S'_{bi} are quite different in the same i time; SMfMGT algorithm is simple in distance measure sample, and easy to calculate, just order one scan data set, what's more important, the test statistic $T = \frac{|\overline{dis} - u_0|}{S^* / \sqrt{k}}$ can be incremental calculated, thus it improves efficiency of the algorithm.

5 Conclusions

In this paper, it proposed a novel solution to solve the similarity matching of different granularity time-series. Firstly, it calibrated time, i.e. unified time granularity; then calculated the distance sample of the two time-series; finally judged whether the two time-series is similar by calculated the statistics. Experimental results show that the algorithm is simple, robust, high performance, and can support incremental calculation. To expand the SMfMGT algorithm into the one, which could handle the time-series subsequence matching, is the author's the next direction.

References

1. Agrawal, R., Faloutsos, C., Swami, A.: Efficient Similarity in Sequence Databases. In: Proceeding International Conference on Foundations of Data Organization and Algorithms, Chicago, Illinois, pp. 69–84 (1993)
2. Agrawal, R., Lin, K.I., Sawhney, H.S., Shim, K.: Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In: 21st International Conference on Very Large Databases, Swizerland, pp. 490–501 (1995)
3. Keogh, E.: Similarity Search in Massive Time Series Databases. PhD dissertation, University of California, Irvine, Department of Computer Science (2001)
4. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and Information Systems 7, 358–386 (2008)
5. Berndt, D.J., Clifford, J.: Finding patterns in time series: a dynamic programming approach. In: Procddeings of the Advances in Knowledge Discovery and Data Mining, Menlo Park, pp. 229–248 (1996)
6. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. In: Proceedings of the 18th International Conference on Data Engineering, SanJose, pp. 673–684 (2002)

7. Chen, L., Ozsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: Proceedings of the ACM SIGMOD Conference, Maryland, pp. 491–502 (2005)
8. Chen, L., Ng, R.T.: On the marriage of Lp-norms and edit distance. In: Proceedings of the 30th International Conference on Very Large Data Bases, Toronto, pp. 792–804 (2004)
9. Meng, Z.Q.: Stduy of Temporal Type and Time Granularity in the Temporal Data Mining. Xiangtan University Science Journal 3, 1–4 (2000)
10. Gross sales of domestic debt securities at nominal value, `http://www.bundesbank.de/statistik/statistik_zeitreihen.en.php?lang=en&open=&func=row&tr=WU0053`

A Novel Algorithm for Hierarchical Community Structure Detection in Complex Networks

Chuan Shi, Jian Zhang, Liangliang Shi, Yanan Cai, and Bin Wu

Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia
Beijing University of Posts and Telecommunications, Beijing, China, 100876
{shichuan,strubby,stona126,caiyanan,wubin}@bupt.edu.cn

Abstract. Networks have in recent years emerged as an invaluable tool for describing and quantifying complex systems in many branches of science. Recent studies suggest that network often exhibit hierarchical organization, where vertices divide into groups that further subdivided into groups of groups, and so forth over multiple scales. In this paper, we introduce a novel algorithm that searches for the hierarchical structure. The method iteratively combines the similar communities with the elaborate design of community similarity and combination threshold. The experiments on artificial and real networks show that the method is able to obtain reasonable hierarchical structure solutions.

Keywords: hierarchical structure, community detection, similarity matrix, possibility matrix.

1 Introduction

Community detection in complex networks has attracted a lot of attentions in recent years. The communities are groups of nodes that are densely interconnected but only sparely connected with the rest of the network [1] [2]. Since a decisive advance was made by Newman and Girvan [3] in 2002, the community detection algorithms are experiencing a surge. The contemporary community detection algorithms can be roughly classified into two categories: optimization based methods and heuristic methods. These algorithms can effectively reveal the community structures; however, they cannot solve the networks with hierarchical structure.

Traditionally, hierarchical structure is represented by a tree or dendrogram in which closely related pairs of vertices have lowest common ancestors that are lower in the tree than those of more distantly related pairs. This structure can be modeled mathematically using a probabilistic approach. Although these methods could create hierarchical structure, these hierarchical structures cannot be efficiently identified.

More recently, several schemes have been designed to identify hierarchical organization of complex systems. Marta Sales-Pardon et al. [4] proposed a universal method that comprises two major steps: (1) estimating the "proximity" in the hierarchy between all pairs of nodes; (2) uncovering the overall hierarchical organization

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 557–564, 2010.

of nodes with an unsupervised algorithm. Ravasz et al. [5] studied the hierarchical structure of metabolic networks, but in their analysis the authors put emphasis on detecting “global signatures” of hierarchical network architecture. More direct methods to investigate the hierarchical organization of the nodes in a network have also been recently proposed [6-8]. Although useful in some contexts, these methods do not clearly identify hierarchical levels and thus fail to identify the different levels in the hierarchy as well as the number of modules and their composition at each level.

In this paper, we propose a simple but effective algorithm to detect hierarchical structure, which is called Iteration Hierarchical Community Detection algorithm (IHCD). IHCD first considers each node as a community, and then repeatedly combines “similar” communities into larger one. With the elaborated design of the community similarity and the combination condition, IHCD can automatically reveal the valuable hierarchical structure. The experiments give excellent results.

2 Preliminary

A network N can be modeled as a graph $G=(V,E)$ where V is a set of objects, called nodes or vertices, and E is a set of links, called edges, that connect two elements of V. Network also can be equivalently represented as a connectivity matrix A. In the case of unweighted and undirected networks, the adjacent matrix can be used to illustrate the network.

$$A[i,j]=\begin{cases}0 & V(i,j)\in G\\ 1 & V(i,j)\notin G\end{cases}. \tag{1}$$

The set of node i’s adjacency nodes can be noted as N_i, and the number of elements in N_i is $|N_i|$. A definite community partition is to find a partition of nodes, i.e., $P=\{\{C_1,C_2,\cdots\}|C_i\subset V, C_i\bigcap C_j=\varnothing\}$. The number of elements (i.e., vertices) in C_i is $|C_i|$. An ideal community (also call cluster or module) in a network is a group of vertices (i.e. a sub-graph) having a high density of edges within them, and a lower density of edges between groups. A hierarchical structure in a network means that there are several partitions $P_1,P_2,\cdots$. For any two partitions $P_i=\{C_{i1},C_{i2},\cdots\}$, $P_j=\{C_{j1},C_{j2},\cdots\}(i<j)$ if any one community in P_j is constituted by several communities in P_i (i.e., $C_{jk}=C_{ik1}\bigcup C_{ik2}\bigcup\cdots$), we can say that there are hierarchical structure in the graph, and $P_1<P_2<\cdots$. That is, the resulting network will have a larger density of connections between nodes grouped in the same sub-module at the first level, a smaller density of connections between groups of nodes grouped in the same module at the second level, and an even. Thus, the network has by construction a hierarchical organization.

3 Iteration Hierarchical Community Detection Algorithm

3.1 Similarity Matrix

We first calculate the similarity of pairs of nodes, which constructs a similarity matrix of the graph. There are many methods to measure the node similarity. After extensive experiments, we select a fundamental approach measures “topological overlap”, which is defined as the ratio between the number of common neighbors of two nodes and the number of nodes in the combination of their neighbor set. This measure identifies affinity between nodes with a dense pattern of local connections.

$$SM[i,j] = \begin{cases} 1 & (i = j) \\ \dfrac{|N_i \cap N_j|}{|N_i \cup N_j|} & (i \neq j) \end{cases} . \tag{2}$$

Based on node affinity between every pair nodes, the similarity matrix *SM* creates, the rows and columns of the matrix correspond to the indices of nodes. Note that the degree *Ki* of a generic node i can be calculated as $K_i = \sum_j A[i,j]$. The value $|N_i \cap N_j|$ and $|N_i \cup N_j|$ can be separately the result of “and”, “or” operation between the two vectors constructed by row i and j.

3.2 Possibility Matrix

During the combination of communities, we need to measure the similarity of communities. Here we introduce a Possibility Matrix to describe the similarity between communities. After comparing many potential methods, we define the Possibility Matrix as follows:

$$PM[i,j] = \frac{|C_i| * |C_j|}{\sum_{m=1}^{|C_i|} \sum_{n=1}^{|C_j|} \frac{1}{SM[m,n]}} . \tag{3}$$

Note that the denominator $SM[m,n]$ may be 0; in our experiment we selected a very little value 0.001 in replace of 0 values.

3.3 The Method

IHCD consists of two major steps: (i) calculating the similarity matrix of the graph and (ii) iteratively combining similar communities. Algorithm 1 is the framework of IHCD. For each iteration, we can get a hierarchical structure.

The key operation in IHCD is the community combination. After extensive experiments and comparisons, a simple self-adapting method is proved to be effective. For the possibility matrix PM_t of community partition P_t, the combination threshold $\alpha = (MAX(PM[i,j]) - MIN(PM[i,j]))/2$; if the similarity of community i and j $PM[i,j] > \alpha$, the method combines community i and j.

Algorithm 1. Main framework of IHCD

```
1.  procedure
2.     calculate similarity matrix $SM$ based on the adjacent matrix $A$
3.     initialize graph partition $P_0$ with each node is a community
4.     set t = 0
5.     while( | $P_t$ | > 1 ) do
6.        calculate possibility matrix $PM_t$ based on $P_t$ and $SM$
7.        combine communities to construct partition $P_{t+1}$
8.        t=t+1
9.     end while
10. end procedure
```

Algorithm 2 gives the explicit description of combining communities, which is the operation in line 7 of Algorithm 1.

Algorithm 2. Combine Communities

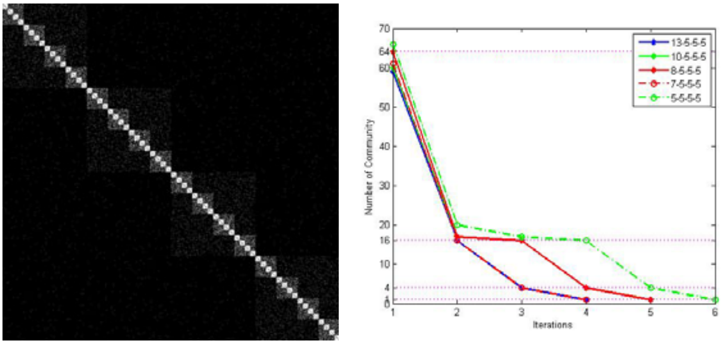
```
procedure
   set $\alpha = (MAX(PM[i, j]) - MIN(PM[i, j])) / 2$
   for each community i set isCombined[i] = false
   set m = 0
   initialize graph partition $P_0$
   for each $C_i$ in $P_t$ do
      if (isCombined[i] = false)  do
         add $C_i$ to $C_m$ in $P_{t+1}$
         isCombined[i] = true
         for $C_j$ in $P_t$  (j>i) do
            if( isCombined[j] = false  and  $PM[i, j] > \alpha$ ) do
               add $C_j$ to $C_m$ in $P_{t+1}$
               isCombined[j] = true
            end if
         end for
         m++
      end if
   end for
end procedure
```

The process stops when all nodes are in one community. It's obvious we can get a hierarchical structure of the network on each iteration. The hierarchical structure is also the combination process of communities. In order to reveal the meaningful hierarchical structure, we should elaborately design the combination condition. The design of α value in each procedure can be in various ways. The simplest way may be that α ranges from 1 to 0 with 0.1 intervals, which divides the network from the finest granularity to the coarsest granularity. Here we set α as the middle value of the largest and smallest similarity. The experiments show that this combination condition fastens the convergence with more meaningful hierarchical structure. Other combination condition can be explored, which is our future work.

4 Experiments

4.1 Artificial Network

In this experiment, we test the performance of IHCD with a benchmark which is an extension of the classical benchmark proposed by Girvan and Newman. There are 1024 nodes, arranged in 64 groups of 16 nodes each. Each node in the network has K_1 links with the most internal community (formed by 16 nodes), K_2 links with the second external community (formed by 64 nodes), K_3 links with the most external community(formed by 256 nodes). In addition, each node has a number K_4 of links with the rest of the network. As shown in Fig.1 (A), the network has three hierarchical levels: the coarse level consisting of the 4 large groups, and the middle level with 16 subgroups and the fine level has 64 small groups. K_1, K_2, K_3 *and* K_4 are used to control the notable degree of communities. Larger K_1 means the most internal cluster is more community-like. Similarly, larger K_2 means the second hierarchical is more obvious. Here we set K_2=K_3=K_4=5 and K_1 ranges from 5 to 13. The smaller K_1 means the most internal communities are more



(A) (B)

Fig. 1. (A) A network with hierarchical structure. Each of the four large clusters is made out of 256 nodes and has an internal subdivision in four clusters with 64 nodes. Every cluster of 64 nodes has been divided into 4 smaller groups containing 16 nodes each. In this way, three hierarchical levels emerge. (B) Results of the artificial network. The legend corresponds to value K1-K2-K3-K4.

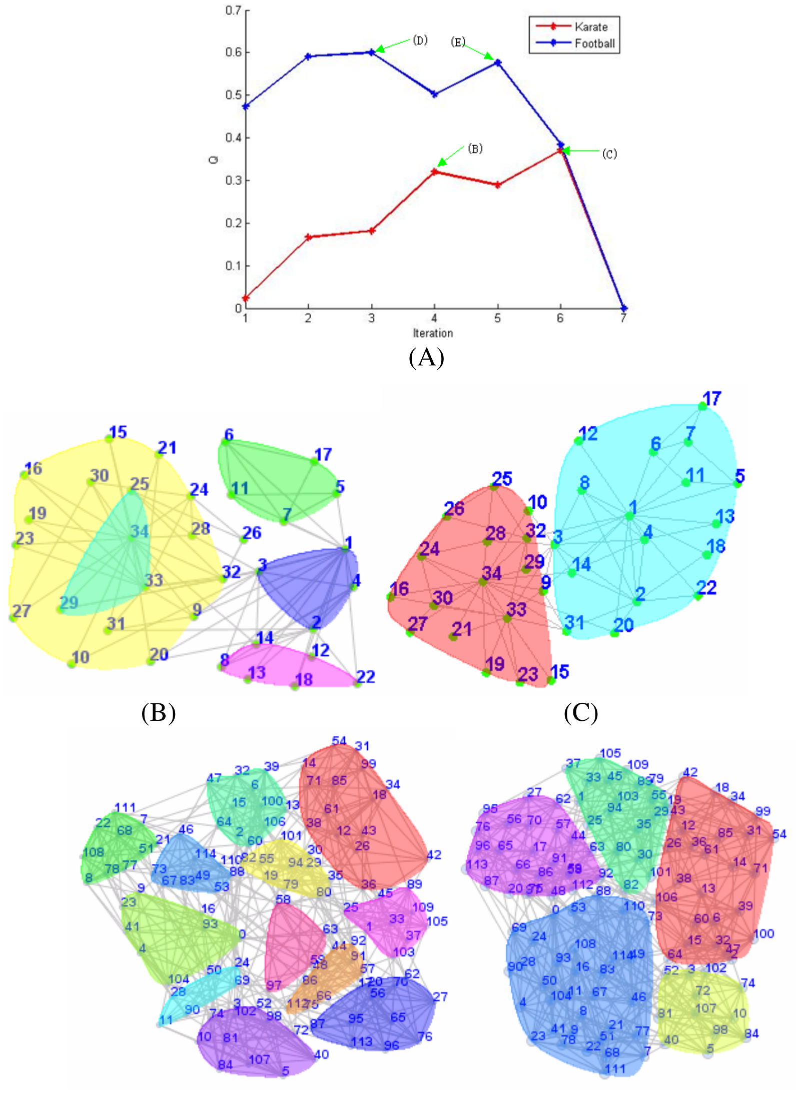
“fuzzy”, which indicates it is harder for the algorithm to detect the hierarchical structure.

The experimental results are shown in Fig.1 (B). It is obvious that, for most cases, IHCD accurately reveals the three hierarchies: the most internal level (i.e., 64 communities), the second level (16 communities), and the third level (4 communities). We also find that the speed of convergence turns down as K_1 decreases. Note that, with the decrease of K_1, the connectivity of the most inner level becomes looser. Thus, it is more difficult for IHCD to find the hierarchy. Especially, IHCD only costs 4 iterations to explicitly reveal the three hierarchy when K_1=13. For K_1=5, it takes IHCD 6 iterations to reveal the hierarchy. The experiments show that IHCD can accurately reveal hierarchical structure with a small number of iterations.

4.2 Real Network

This section further validates IHCD with two real networks: Zachary’s karate club [9], and the network of American college football teams [9]. The social network of karate club members studied by the sociologist Wayne Zachary has become a benchmark for all methods of community detection. The network consists of 34 nodes, which separated into two distinct groups due to a contrast between one of the instructors and the administrator of the club. The question is whether one is able to detect the social fission of the network. The second example is the network of American college football teams. Here, there are 115 nodes, representing the teams, and two nodes are connected if their teams play against each other. The teams are divided into 12 conferences. Games between teams in the same conference are more frequent than game between teams of different conferences, so one has a natural cover where the communities correspond to the conferences.

IHCD iterates 7 times for Karate and Football networks, when the algorithm converges. We use a simple method to further select meaningful hierarchical structures from these hierarchies. We calculate the Q value of each hierarchy, and select the hierarchy with local optimal Q value. Fig. 2 (A) shows the relation between hierarchy and its Q value on Karate and Football networks. Two hierarchical structures with local optimal are selected for these two networks, respectively. For Karate network, the real partition is revealed at the 6 iteration (see Fig.2 (C)), and the network can be further divided into 5 communities with a fine granularity (see Fig.2 (B)) which has been found by many researchers. For the Football network, IHCD correctly discovers the 12 conferences at the 3 iteration (see Fig.2 (D)). Moreover, these conferences can further be combined into 5 regions at 6 iteration as shown in Fig.2 (E). For these real networks, IHCD not only reveals the real community structure, but also discovers the hierarchical structure that complies with the real situation.



(A)

(B)

(C)

(D)

(E)

Fig. 2. (A) The result of the relation between iterations and the Q value of corresponding community structure. (B)(C) The two hierarchical structure of the Karate network corresponding to two local optimal of Q value. (D)(E) The two hierarchical structure of the Football network corresponding to two local optimal of Q value.

5 Conclusions

We present in this paper a novel way to detect hierarchical structure in complex networks. The algorithm includes two phases. The first phase calculates the similarity matrix of the graph and initializes each node as a community. The second phase iteratively combines similar communities. After extensive experiments, we design the simple but effective method to calculate the community similarity and determine the combination condition of communities. The experiments on artificial network show that the algorithm can accurately reveal built-in hierarchical structure with few iterations. The experiments on two real networks demonstrate that the algorithm reveals the real community structure as well as other meaningful hierarchical structure.

Some work are desire to be done for improvement in the near future. The algorithm needs to be validated by large real networks. Moreover, new combination threshold and similarity calculation can be designed.

Acknowledgments. This work is supported by the National Science Foundation of China (No. 60905025, 90924029, 61074128). It is also supported the National High-tech R&D Program of China (No.2009AA04Z136) and the National Key Technology R&D Program of China (No.2006BAH03B05).

References

1. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.-U.: Complex networks: Structure and dynamics. Physics Report 424(4-5), 175–308 (2006)
2. Guimera, R., Amaral, L.A.N.: Functional cartography of complex metabolic networks. Nature 433, 895–900 (2005)
3. Pizzuti, C.: Community detection in social networks with genetic algorithms. In: The 10th Annual Conference on Genetic and Evolutionary Computation, USA, pp. 1137–1138 (2008)
4. Sales-Pardo, M., Guimera, R., Andre, A.M., Amaral, L.A.N.: Extracting the hierarchical organization of complex systems. PNAS 104(39), 15224–15229 (2007)
5. Ravasz, E., Somera, A.L., Mongru, D.A., Oltvai, Z.N., Barabasi, A.-L.: Hierarchical Organization of Modularity in Metabolic Network. Science 30, 297(5588), 1551–1555 (2002)
6. Guimera, R., Danon, L., Diaz-Guilera, A., Giralt, F., Arenas, A.: Self-similar community structure in a network of human iteractions. Phys. Rev. E 68(065103) (2003)
7. Clauset, A., Moore, C., Newman, M.E.J.: Structural Inference of Hierarchies in Networks. In: Airoldi, E.M., Blei, D.M., Fienberg, S.E., Goldenberg, A., Xing, E.P., Zheng, A.X. (eds.) ICML 2006. LNCS, vol. 4503, pp. 1–13. Springer, Heidelberg (2007)
8. Pons, P.: Post-Processing Hierarchical Community Structures: Quality improvements and Multi-scale View. Computer Science, Data Structure and Algorithms, cs/0608050 (2006)
9. Newman, M.E.J.: Detecting community structure in networks. Eur. Phys. J. B 38, 321–330 (2004)

Investigating Sequential Patterns of DNS Usage and Its Applications*

Jun Wu, Xin Wang, Xiaodong Lee, and Baoping Yan

China Network Information Center,
Computer Network Information Center, Chinese Academy of Sciences, Beijing, China
Graduate University of Chinese Academy of Sciences, Beijing, China
{wujun,wangxin,lee}@cnnic.cn,
ybp@cnic.cn

Abstract. DNS is an important infrastructure of the Internet whose smooth operation and performance are vital to both the Internet users and applications. In this paper, we investigate the DNS usage patterns by applying the mixture of Markov chains (*MMC*) to DNS usage data. Results on cluster analysis of real DNS query data demonstrate that the method is capable of revealing insightful patterns of sequential activities of DNS users. Furthermore, typical DNS user groups are investigated. Finally, two particular application scenarios of user clustering using this method are proposed, which might help improve the DNS performance by adding plug-ins into the DNS server software.

Keywords: DNS, mixture of Markov chains, maximum likelihood estimation, clustering.

1 Introduction

The Domain Name System (DNS) is one of the most important Internet infrastructures which consists of a distributed database and a corresponding software system implementing the mapping between domain names and IP addresses. Virtually every Internet application relies on looking up the DNS data. Therefore DNS is regarded as the bridge between the users and Internet applications and the performance of DNS is vital to both the Internet applications and users.

At present, the work of DNS research communities is mainly focused on the measurement and the protocol. Analysis of root server logs by Danzig *et al.* [1] was the first published large-scale study of DNS. They found a large number of implementation errors that caused unnecessarily wasted bandwidth. Nemeth *et al.* [2] analyzed a few hours of traces collected at the *F*-root name server and found an astounding number of bogus queries. Several projects are devoted to continuous monitoring of the DNS root servers' performance [3], [4]. Although a lot of research work has been carried out, there is little work devoted to mining the DNS query patterns.

* This research is supported by the China Next Generation Internet Project: Industrialization of Next Generation Trusted Domain Name Service System (Grant number: CNGI-09-03-04).

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 565–576, 2010.

DNS usage data contains useful information about the users accessing Internet applications via domain name look-up activities. DNS servers, especially those run by large ISPs and domain registrars, are characterized by a large number of users who access the diversified applications or web sites. Therefore, the DNS user behavior is highly heterogeneous.

In this paper we study the heterogeneity of DNS user behavior using a mixture approach. An extended study on how to make use of the heterogeneity and corresponding locality to improve the performance of the DNS servers is also provided.

The paper proceeds as follows. Section 2 provides a detailed account of the underlying mixture model and the associated EM clustering algorithm. In section 3, we describe the data set and illustrate the result of clustering approach can perform an exploratory data analysis that provides direct insight into the heterogeneity and dynamic nature of this type of DNS query data. The method for improving the performance of DNS usage by utilizing the user clustering is introduced in section 4. Section 5 concludes the paper with a summary and some possible extensions of this work.

2 Modeling DNS Query Data Using MMC

Based on the *i.i.d.* assumption, the data from different users are generated independently given the mixture model with *K* components each of which is a *Markov chain* [5]. MMC is selected out of the following considerations:

1. A DNS user issues queries consecutively, which form a time sequence.
2. Based on consideration of computational complexity, only the *first-order Markov* model is used. In other words, we assume that the next query issued is only affected by the current one.
3. Because of the heterogeneity of a large number of DNS users' query patterns, the MMC is selected.

2.1 Model Formulation

Let $\mathbb{D} = \{ \boldsymbol{Y}^1,\ldots,\boldsymbol{Y}^S \}$ be a set of sequences. Each sequence $\boldsymbol{Y}^s = \{y_1^s,\ldots,y_{L_s}^s\}$ ($1 \le s \le S$) consisting of L_s observed states describes the user's query behavior through the DNS Recursive name Server (*RS*) with each element in a sequence corresponding to *M* possible categories of the domain names that the DNS user could have requested. $C_s \in \{1,\ldots,K\}$ corresponds to the unknown cluster assignment for users. $\boldsymbol{\Theta} = \{\boldsymbol{\alpha},\boldsymbol{\pi},\boldsymbol{\beta}\}$ denotes the parameters of the model and can be written as detail:

1. $\boldsymbol{\alpha}$ is a vector of *K* mixture weights:

$$\boldsymbol{\alpha} = \{ \alpha_1,\ldots,\alpha_K \}, \sum_k \alpha_k = 1$$

2. $\boldsymbol{\pi}$ is a set of *K* initial state probability vectors of *S* - dimensions:

$$\boldsymbol{\pi} = \{ \pi_1,\pi_2,\ldots,\pi_K \}$$

3. $\boldsymbol{\beta}$ is a set of K transition matrices:

$$\boldsymbol{\beta} = \{ \beta_1, \beta_2, \ldots, \beta_K \}$$

where the per-component transition probability matrices β_k ($1 \le k \le K$) are matrices of $M \times M$:

$$\beta_k = [\beta_{k,j,l}]_{M \times M}, \beta_{k,j,l} = P(y_t = l | y_{t-1} = j, k, \boldsymbol{\Theta})$$

A mixture model for the sequence $\boldsymbol{Y}^s$ with K components has the general form:

$$\begin{aligned} P(\boldsymbol{Y}^s | \boldsymbol{\Theta}) &= \sum_{k=1}^{K} P(C_s = k | \boldsymbol{\Theta}) P(\boldsymbol{Y}^s | k, \boldsymbol{\Theta}) \\ &= \sum_{k=1}^{K} P(C_s = k | \boldsymbol{\Theta}) P(y_1^s | k, \boldsymbol{\Theta}) \prod_{t=2}^{L_s} P(y_t^s | y_{t-1}^s, k, \boldsymbol{\Theta}) \\ &= \sum_{k=1}^{K} \alpha_k \pi_{k, y_1^s} \prod_{t=2}^{L_s} \beta_{k, y_{t-1}^s, y_t^s} \, . \end{aligned} \tag{1}$$

So, the *likelihood* of observing a full data set $\mathbb{D}$ can be defined under the *i.i.d* assumption as:

$$P(\mathbb{D} | \boldsymbol{\Theta}) = \prod_{s=1}^{S} P(\boldsymbol{Y}^s | \boldsymbol{\Theta}) = \prod_{s=1}^{S} \sum_{k=1}^{K} \alpha_k \pi_{k, y_1^s} \prod_{t=2}^{L_s} \beta_{k, y_{t-1}^s, y_t^s} \, . \tag{2}$$

2.2 Model Estimation by EM

At first we assume that the number of components is known given data set $\mathbb{D}$. *Maximum Likelihood Estimation* (*MLE*) [6] is used by us to identify the parameters for $\boldsymbol{\Theta}$:

$$\begin{aligned} \boldsymbol{\Theta}_{ML} &= \arg\max_{\boldsymbol{\Theta}} \log P(\mathbb{D} | \boldsymbol{\Theta}) \\ &= \arg\max_{\boldsymbol{\Theta}} \sum_{s=1}^{S} \log\left(\sum_{k=1}^{K} \alpha_k \pi_{k, y_1^s} \prod_{t=2}^{L_s} \beta_{k, y_{t-1}^s, y_t^s} \right) . \end{aligned} \tag{3}$$

For maximizing the likelihood function, the simplest choices are to employ a gradient search method [7] or an iterative EM method. Here, the latter is chosen. The Expectation Maximization (*EM*) algorithm [8] is a parameter estimation method which falls into the general framework of MLE, and is applied in cases where part of the data can be considered to be incomplete, or “hidden”. It is essentially an iterative optimization algorithm which, at least under certain conditions, will converge to parameter values at a local maximum of the likelihood function. EM algorithm alternates two steps: (1) an expectation (E) step where posterior probabilities are computed based on the current estimates of the parameters, (2) a maximization (M) step, re-estimate the parameters in order to maximize the expectation of the complete data likelihood.

The EM algorithm begins with some initial values of $\boldsymbol{\alpha}$, $\boldsymbol{\pi}$ and $\boldsymbol{\beta}$. In the expectation step we compute the posterior probabilities by *Bayes' rule* [9]:

$$\gamma_{ks}(\boldsymbol{\Theta}) = P(C_s = k \mid \boldsymbol{Y}^s, \boldsymbol{\Theta}) = \frac{\alpha_k P(\boldsymbol{Y}^s \mid k, \boldsymbol{\Theta})}{\sum_{k=1}^{K} \alpha_{k'} P(\boldsymbol{Y}^s \mid k', \boldsymbol{\Theta})}$$

$$\propto \alpha_k P(\boldsymbol{Y}^s \mid k, \boldsymbol{\Theta}) . \tag{4}$$

where "$\propto$" stands for "proportional to". Once these probabilities are computed, we can either assign the user to the cluster with highest probability—a *hard* assignment—or assign the user fractionally to the set of clusters according to this distribution—a *soft* assignment. Both types of assignment are commonly used in practice. In the current implementation, hard assignment is used. In the maximization step, we aim at maximizing the expectation of the complete data likelihood $\mathbb{E}(L^C)$:

$$\mathbb{E}(L^C) = \sum_{s=1}^{S} \sum_{k=1}^{K} \gamma_{ks}(\boldsymbol{\Theta}_{old}) \log[\alpha_k P(\boldsymbol{Y}^s \mid k, \boldsymbol{\Theta})] . \tag{5}$$

Through the use of *Lagrange Multipliers* [10], we get the following update rules for each set of re-estimated parameters:

1. Mixture weights:

$$\alpha_k \propto \sum_{s=1}^{S} \gamma_{ks} . \tag{6}$$

2. Initial state probabilities:

$$\pi_{k,j} \propto \sum_{s=1}^{S} \gamma_{ks} \delta(y_1^s, j) . \tag{7}$$

where $\delta(y_1^s, j)$ is an indicator function that is equal to 1 if the arguments are equal and 0 otherwise.

3. Transition probabilities:

$$\beta_{k,j,l} \propto \sum_{s=1}^{S} \gamma_{ks} N_{j,l}(\boldsymbol{Y}^s) . \tag{8}$$

where $N_{j,l}(\boldsymbol{Y}^s)$ is the frequency of transitions from state j to state l in the *s-th* sequence.

Iterating the above expectation and maximization steps monotonically increases the likelihood of the observed data until a local optimal is reached.

2.3 Model Selection

For choosing an optimum value for *K*, the number of model components, the *Akaike Information Criterion* (*AIC*) [11] is used, which is a measure of the goodness of fit of an estimated statistical model. Given a data set $\mathbb{D}$ and several candidate models, the one having the lowest AIC is selected. The AIC score of a model $\mathbb{M}$ with parameter $\boldsymbol{\Theta}$ is:

$$AIC(\hat{\boldsymbol{\Theta}}) = -2\log L(\hat{\boldsymbol{\Theta}} \mid \quad) + 2N_f(K)\ . \tag{9}$$

$L(\hat{\boldsymbol{\Theta}} \mid \mathbb{D})$ is the likelihood of the parameters in model $\mathbb{M}$ evaluated by MLE, $N_f(K)$ is the number of free parameters of $\mathbb{M}$, or the degrees of freedom of model $\mathbb{M}$ that need to be estimated.

3 Experiments on DNS Usage Data

3.1 Data Set

The query data was recorded by a RS operated by a large ISP in China during a 24-hours period of June, 18, 2009. During this day, there are totally 12,433,097 queries sent to this RS by 9,242 DNS users for 568,484 domain names, 15.5% of which are targeted at names suffixed with ".CN". After preprocessing, we get 8,135 sequences with an average of 236.8 names per sequence. Each user is described by a sequence of domain names he/she queried during the day, which are further mapped into categories. The category information is obtained from *.CN Domain Name Registration Information Database*. The 19 categories we obtain are *not available* (*na* for short), *government* (*go*), *consulting* (*co*), *social service* (*ss*), *taxation* (*ta*), *sports entertainment* (*se*), *transportation* (*tr*), *news and advertising* (*ne*), *business* (*bu*), *science education* (*sc*), *real estate* (*re*), *postal* (*po*), *information technology* (*it*), *manufacturing* (*ma*), *public utility* (*pu*), *construction* (*cs*), *generalized agriculture* (*ga*), *mining* (*mi*), and *other* (*ot*)}. It is worth noting that "na" means there is no category information associated with the domain names. Table 1 shows a fragment of the sequences.

Table 1. A fragment of user query sequences

Users	Sequences
1	real estate, real estate, taxation, public utility, business
2	science education, consulting, IT, science education
4	IT, IT, IT, IT, NA, IT, business, consulting
5	science education, science education, science education, other
6	business, NA, business, IT, sports entertainment, other
7	social service, public utility, generalized agriculture
8	sports entertainment, sports entertainment, news advertising

3.2 DNS User Clusters

By applying MMC and AIC for model estimation and selection respectively, 8,135 users who sent the DNS queries for "CN" names are clustered into 33 clusters, as shown in Fig. 1, where each cluster has been represented by a "vector of state transition probability" which is a flat form of the corresponding transition probability matrix. It can be seen that each cluster is obviously different from others because the query behaviors of the users in them focus on different transitions from one name category to another.

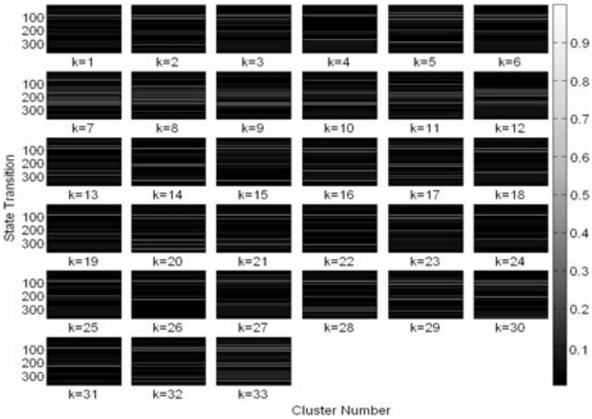


Fig. 1. The coordinate values on *Y*-axis of each subgraph denote all the possible state transitions. Because there are totally 19 domain name categories, the number of elements of the transition probability matrix is 361 (19^2) that is equal to the dimension of the vector of state transition probability. If $a_{i,j}$ and b_n are respectively the elements of the matrix and the vector, they are equal when $n = (i\text{-}1) \times M + j$. In each subgraph the line whose index is *n* denotes there exists the *n-th* transition between two domain name categories. The brighter the color of the line, the greater the transition probability.

3.3 Finding Distinct Clusters

In order to find the most distinct clusters among the *K* clusters, *Kullback-Leibler* (*KL*) divergence is utilized. KL divergence (also *information gain*) is a special case of a broader class of divergences called *f-divergences* [12] and it measures the expected number of extra bits required to code examples from *Q* when using a code based on *R* [13]. For probability distributions *Q* and *R* of a discrete random variable *x*, the KL divergence is calculated as:

$$D_{KL}(Q\|R) = \sum_{x} Q(x)\log\frac{Q(x)}{R(x)}. \tag{10}$$

KL divergence is often used to compute the divergence between two probability distributions, which is less applicable for our purpose and further processing is needed. Firstly, the KL divergences of any two clusters are computed. The result matrix Z is shown in Fig. 2 in which the color bar on the right indicates the range of the values. It is also denoted by the colors that cluster 33, 17 and 7 look more distinct from other clusters. Using the matrix Z , we define a new metric—*averaged KL distinction* (*AKLD*)—to describe the distinction of a cluster from others:

$$AKLD_k \overset{def}{=} \frac{1}{2K} \sum_{j=1, j\neq k}^{K} (Z_{k,j} + Z_{j,k})(1 \leq k \leq K)\,. \tag{11}$$

where k is the index of cluster and the value of $AKLD_k$ denotes the distinction level of the *k-th* cluster. After computing the AKLD values for all 33 clusters, Fig. 3 is given in “stairs” style. We can find from Fig. 3 that the top 3 distinct clusters are cluster 33, 17 and 7, which comes to the same conclusion of Fig. 2.

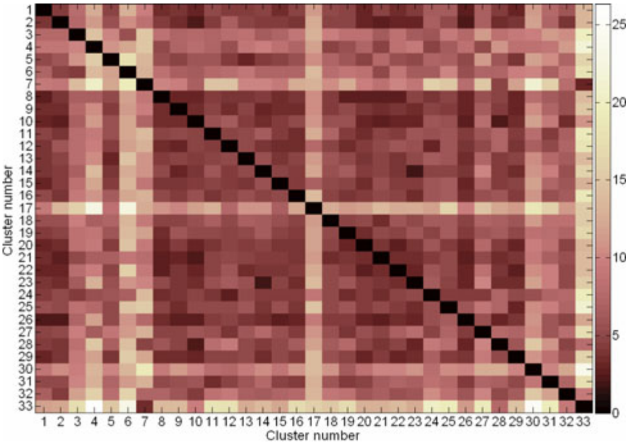


Fig. 2. KL divergence values between any two clusters

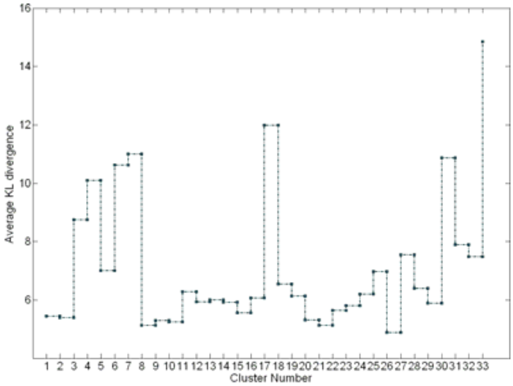


Fig. 3. Values of AKLD for all clusters

3.4 Investigation of Typical Clusters

Due to space limitation, only the top 2 distinct clusters are analyzed. The transition probability matrices of cluster 33 and 17 are shown in Fig. 4, where we can find that the users in these two clusters share the following two points:

1. They are all inclined to query the domain names of the same category continually, which is revealed by the diagonals of the two matrices.
2. The *9-th* columns of the two matrices show that they all tend to query the domain names of "business" category after querying any other domain names.

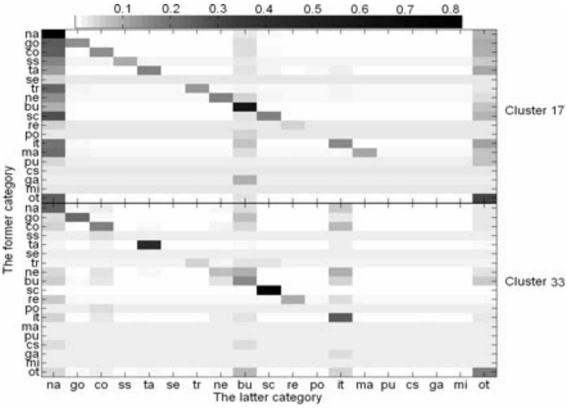


Fig. 4. Transition probability matrices of cluster 17 and cluster 33

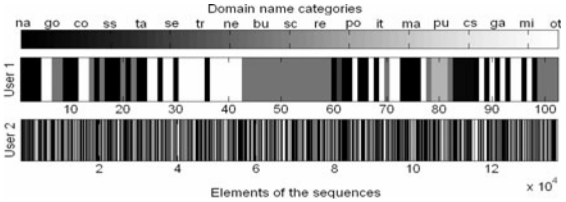


Fig. 5. Query sequences of the two users in cluster 17. The two bars marked by user ID represent the two users' query sequences of which the elements corresponding to the categories is represented by the different colors in grey scale like what the colorbar shows.

Beside the similarities, the differences between the two clusters shown by this figure are:

1. The users in cluster 17 are more likely to query the domain names of "not available" category and "other" category. Although interests of the users cannot be accurately described by these two kinds of domain names, this query pattern can indirectly reflect that users in cluster 17 have a wider range of interests.
2. The users in cluster 33 have more interests in "IT" and they are prone to query the "taxation" names and "science education" names successively.

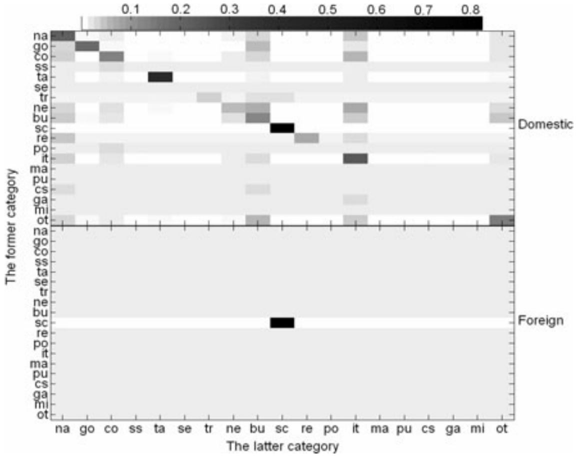


Fig. 6. Transition probability matrices of the domestic users and foreign users in cluster 33

There are only two users in cluster 17, whose query sequences are shown in Fig. 5. We can see that although the lengths of the two users' sequences differ significantly from each other, they are both concerned with the domain names of "not available", "business" and "other" categories. In fact, the sequence of the first user can be regarded as a "substring" of the second user's query sequence. This may be why these two users are grouped into the same cluster.

The significant difference between cluster 33 and other clusters is that the users in cluster 33 come from more than 30 countries. We separate the sequences according to the geographic location of the users into "domestic China" and "foreign" sub-clusters. Fig. 6 shows the transition probability matrices of these two sub-clusters. An interesting pattern revealed by Fig. 6 is that the foreign users normally traversed the domain names of "science education" category with the probability of 100%.

4 Application of User-Clustering in DNS Usage

4.1 Optimizing the TTL of Records in DNS Cache

DNS RS recursively sends queries to the Authoritative name Server (AS) as an agent of the DNS end users [14]. If it gets a record (mapping of domain name and IP address, in our context) as the answer, this record is cached and associated with a lifetime, that is, *TTL* (time to live). For later queries matching records in the cache, the server will response directly with the answer in the cache instead of querying the AS again. On the other hand, if a TTL expires, the corresponding record is cleared.

Deciding on the TTL is essentially a trade-off between performance and consistency. A small TTL ensures that data is consistent across the DNS, however, increases the average resolution time. A large TTL reduces the average resolution time but the drawback is that information in RSs will be inconsistent longer if changes are made to the data on ASs.

Normally the TTL is configured manually by the administrator of ASs. In many cases, DNS recursive service operators increase the TTLs for shortening the response time for better resolution performance. However, this "simple" action ignores the refreshing frequency at the authoritative end and may ramp up the risk of data inconsistency.

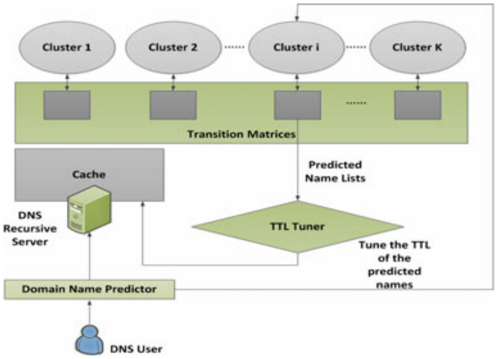


Fig. 7. DNS cache optimization via user-clustering

The scenario of applying user-clustering to DNS cache optimization is shown in Fig. 7. It needs to be noted that the user query sequences here are all represented by the time series of the domain names themselves. As shown in Fig. 7, domain name predictor captures the DNS query datagram of a user and extracts the IP address according to which the predictor classifies this user into one cluster (Cluster *i*} in the figure). These clusters are generated by learning from large amounts of query data as described in section 2, what is different is the transition probability here is computed from one domain name to another, not the category. According to the transition probability matrix of Cluster *i* and the domain name currently queried by this user, a predicted name list can be obtained. Takes as input the predicted name list, TTL tuner looks up the name in the cache one by one. If the name exists in the cache and its TTL has expired, the TTL will be updated as follow:

$$TTL_{new} = TTL_{old} \times (1 + prob)\,. \tag{12}$$

where TTL_{old} is the original TTL, *prob* is the corresponding probability in the predicted name list. The modified lifetime TTL_{new} is applied and the domain name will not be cleared.

This method effectively reduces the frequency of query from RS to AS, that is, reduces the round trip time during the whole DNS resolution process by taking into consideration the local feature of query behavior and obeying the data consistency of DNS.

4.2 Load Scheduling for DNS Authoritative Server

In a similar way we can cluster the query sequences of RSs and propose a new *Source Hashing Scheduling* [15] method in Fig. 8. Every AS is associated with one or several

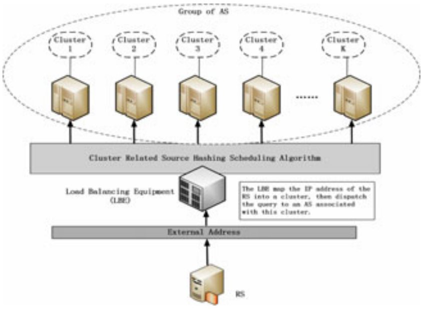

Group of AS
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster K
......
Cluster Related Source Hashing Scheduling Algorithm
Load Balancing Equipment (LBE)
The LBE map the IP address of the RS into a cluster, then dispatch the query to an AS associated with this cluster.
External Address
RS

Fig. 8. Cluster related load scheduling

clusters that include RSs' IP addresses and the domain names of their concern. Through this association, every AS is only responsible for one or several RSs, which shrinks the search range. On the other hand, AS can predict the coming queries according to the transition probability matrix of its corresponding cluster and prefetch can be done. It is worth noting that the clustering use historical query data set, which is changing as time goes on. Incremental training should be performed to rebuild the correlation between clusters and servers.

5 Conclusion and Future Work

In this paper, *MMC* was applied to DNS query data for investigating the DNS usage patterns. Results on real DNS query data demonstrated that the approach was capable of grouping the patterns of sequential activities of DNS users and revealing the heterogeneity of DNS user behavior. At last, we provided an extended study on how to make use of the heterogeneity to improve the performance of the DNS from two aspects.

There are several extensions to our approach that could be investigated. One is to model the duration of each domain name query using duration models. Another way is to diversify the granularity of *M*. According to the hierarchical structure of domain name space, the elements of a query sequence can also be the type of top level domain, second level domain, etc. In such extension, Markov models can be used to characterize both the transitions among categories and the transitions among top level domains within a given category. Alternatively, we can use a hidden-Markov mixture model to learn categories and category transitions simultaneously.

References

1. Danzig, P., Obraczka, K., Kumar, A.: An Analysis of Wide-Area Name Server Traffic: A Study of the Internet Domain Name System. In: Proc. ACM SIGCOMM, New York (1992)
2. Nemeth, E., Claffy, K., Brownlee, N.: DNS Measurements at a Root Server. In: Proc. IEEE GLOBECOMM, San Antonio, Texas (2001)

3. Cho, K., et al.: A Study on the Performance of the Root Name Servers, `http://mawi.wide.ad.jp/mawi/dnsprobe`
4. Thomas, R.: DNS Data Page, `http://www.cymru.com/DNS`
5. Shridharbhai, T.K., Trivedi, K.S.: Probability and Statistics with Reliability, Queuing, and Computer Science Applications. John Wiley & Sons, New York (2002)
6. Hald, A.: On the History of Maximum Likelihood in Relation to Inverse Probability and Least Squares. Statistical Science 14(2), 214–222 (1999)
7. Arnold, D., Salomon, R.: Evolutionary Gradient Search Revisited. IEEE Trans. on Evolutionary Computation 11(4), 480–495 (2007)
8. Hastie, T., et al.: The Elements of Statistical Learning. Springer, Heidelberg (2001)
9. Jaynes, G., Edwin, T.: Probability Theory: the Logic of Science. Cambridge University Press, London (2003)
10. Vapnyarskii, I.B.: Lagrange Multipliers. In: Hazewinkel, Michiel (eds.) Encyclopaedia of Mathematics. Springer, Heidelberg (2001)
11. Burnham, K.P., Anderson, D.R.: Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach, 2nd edn. Springer, New York (2002)
12. Liese, F., Vajda, I.: On Divergences and Informations in Statistics and Information Theory. IEEE Trans. on Information Theory 52(10), 4394–4412 (2006)
13. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
14. Albitz, P., Liu, C.: DNS and BIND, 4th edn. O'Reilly & Associates, Sebastopol (2001)
15. Syme, M., Goldie, P.: Optimizing Network Performance with Content Switching: Server, Firewall, and Cache Load Balancing. Prentice Hall, London (2003)

Key Issues and Theoretical Framework on Moving Objects Data Mining

Rong Xie[1] and Xin Luo[2]

[1] International School of Software, Wuhan University,
430079 Wuhan, Hubei
xierong@whu.edu.cn
[2] State Key Lab of Digital Manufacturing Equipment and Technology,
Huazhong University of Science and Technology,
430074Wuhan, Hubei
mexinluo@hust.edu.cn

Abstract. Considering technical difficulties and bottlenecks in moving objects data mining, such as massive movement data, high dimensional data, topological complexity, and knowledge semantic representation etc., this paper focuses on the study of theory and methods of moving objects data mining. First, it presents two key scientific issues of the research topic, i.e. integration and modeling of heterogeneous data, and information aggregation and interpretation. Second, a theoretical framework of moving object data mining is proposed based on different perspectives of "*space-time data→space-time concept→space-time pattern*". Two aspects of the framework are then discussed in details, including (1) moving objects data modeling and semantic expression; (2) mining methods and algorithms of association rules based on concept lattice. Finally, future works are discussed.

Keywords: Moving objects, Spatio-temporal data model, Spatio-temporal data mining, Ontology, Concept lattice, Natural language.

1 Introduction

Under development and applications of advanced high-precision positioning technology, it is becoming possible to track and acquire location of moving objects in real time, which brings about new problems in the aspects of explosive growth of complexity and quantity of moving objects data. Study of spatial information system is now changing from data management of static objects to data modeling and distributed management of moving objects. Increasing expansion of these complex objects data goes beyond the capacity of people's interpretation. The existing technologies (databases, artificial intelligence and statistics etc.) cannot independently maximize the use of data. Thus, new theories, methods and technologies are urgently required to analyze and predict movement characteristics of moving objects, particularly knowledge discovery. The emerging data mining is considered one of the most promising interdisciplinary developments in the information technology [1].

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 577–584, 2010.

Traditional moving objects database technologies, as well as space-time analytical methods, have been successfully solved some applications of quantitative query and analysis, like "*existence of moving objects within a certain space during a certain period of time*". However, it cannot fully uncover some characteristics and patterns of moving objects which hide under the complicated surface, which can be more valuable support for scientific decision-makings. Taking an example of traffic situated in some big cities, we would expect to process everyday phenomena of detecting and predicting traffic jams, and discovering relations between these jams. It is becoming challenging for us to discover space-time characteristics, relationships, unknown patterns, decision-making knowledge from massive database of moving objects.

Moving objects data mining is currently meeting two key scientific issues. Because of multiple sources, heterogeneous, dynamic, complicated moving objects data, it is required to analyze moving objects characteristics, studying data model and semantics representation, including location, topology and semantic description. On the other hand, it is required to study spatio-temporal granularity, clustering, association and evolution, discovering and interpreting behavior patterns of moving objects.

Researchers have started their research in moving objects data mining and knowledge discovery. Some results were made in the directions of data model, data mining methods and algorithms [2-5]. However, it is still at an early stage that theoretical foundations and approaches are not decided yet [6]. For a long time, it is basically to regard moving object data mining as an application of data mining so that general mining approaches are directly moved to moving object data mining, e.g. focusing on mining algorithms efficiency on spatio-temporal data. As ignoring different characteristics between moving objects and general objects, such moving objects data processing is very weak, leading to incomplete, even wrong knowledge sometimes. Although there are some similarities between these two kinds of mining, their discipline basis and thinking way are very different. Considering technical difficulties and bottlenecks in moving objects data mining, such as massive movement data, high dimension, topological complexity, and knowledge semantic representation etc., this paper focuses on the study of theory and methods of moving objects data mining.

The rest of the paper is organized as follows. Section 2 presents framework of moving objects data mining. Section 3 introduces moving objects data modeling and semantic representation. Section 4 introduces methods of moving objects data mining based on lattice theory. Section 5 concludes the paper and future works are discussed.

2 Overall Framework of Moving Objects Data Mining

Moving objects change their positions over time in the real world, having their own characteristics, that is spatial, temporal, multidimensional, massive, complicated. So, it is more complicated in data analysis and knowledge discovery than static objects or discretely changing objects. From the perspectives of human cognition, when discovering knowledge from moving objects database, essence of data mining is the viewpoint transformation based on different perspectives "*space-time data→space-time concept→space-time pattern*". Cognitive levels are represented by spatio-temporal granularities and scales. Based on different granularities and scales, abstract levels are established to help knowledge extraction in gradual steps. A framework is proposed as shown in Fig. 1 to model and analyze moving object data. Theoretical framework and technological methods are studied, focusing on moving objects data modeling and semantic representation, moving object data mining methods and algorithms based on lattice theory, spatio-temporal knowledge representation.

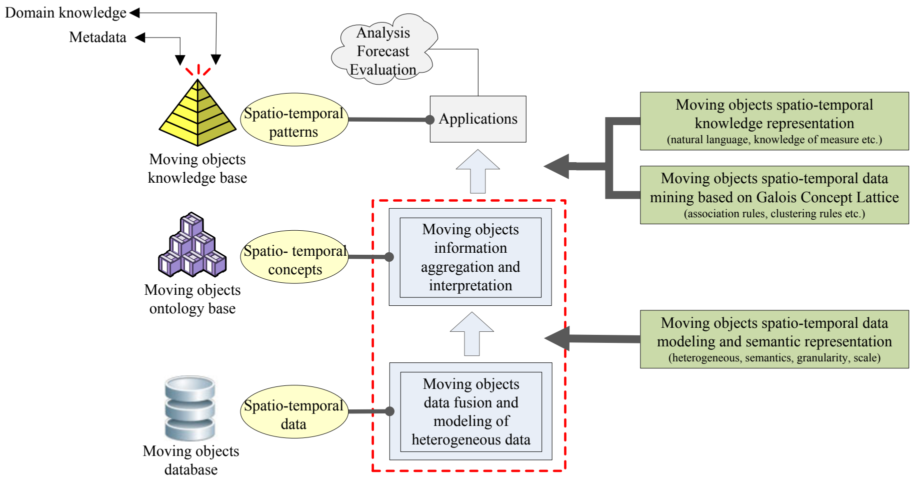

Domain knowledge
Metadata
Analysis
Forecast
Evaluation
Spatio-temporal patterns
Applications
Moving objects knowledge base
Moving objects spatio-temporal knowledge representation
(natural language, knowledge of measure etc.)
Moving objects spatio-temporal data mining based on Galois Concept Lattice
(association rules, clustering rules etc.)
Spatio- temporal concepts
Moving objects information aggregation and interpretation
Moving objects ontology base
Moving objects spatio-temporal data modeling and semantic representation
(heterogeneous, semantics, granularity, scale)
Spatio-temporal data
Moving objects data fusion and modeling of heterogeneous data
Moving objects database

Fig. 1. Overall framework of moving objects data mining

3 Spatio-temporal Data Modeling and Semantic Representation

Moving objects data model is the basis of moving objects data mining and their applications, which defines spatio-temporal objects, topology, processes and maintenance rules of moving objects database. A complete data model should have the capabilities of data query, information analysis and knowledge discovery. Main tasks are: (1) Organize multi-source, heterogeneous data of moving objects, eliminate inconsistencies and uncertainties of source data, extract concepts from data, and represent ontology and semantics of entities, attributes and instances. (2) Based on spatio-temporal granularities and spatio-temporal scales, define semantic relationships of spatio-temporal ontology, establish conceptual levels with different granularities and scales. (3) Describe spatio-temporal ontology and semantics by ontology description language; interpret and use these spatio-temporal ontology and semantics.

Moving objects, i.e. location and shape changes with time continuously, mainly constitute three characteristics of space, time, attributes. Spatial characteristics are related to the spatial position and geometric shapes of moving objects. Time characteristics are described by time instant or time period. Attributes describe non spatio-temporal features of moving objects. Moving objects ontology model, as shown in Fig. 2, has the following components, including metadata, spatial model, temporal model, and spatio-temporal model and ontology language.

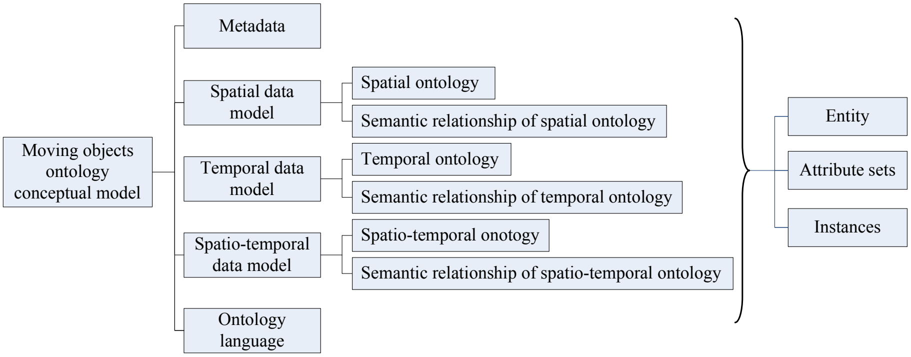

Metadata
Spatial data model
Spatial ontology
Semantic relationship of spatial ontology
Moving objects ontology conceptual model
Temporal data model
Temporal ontology
Semantic relationship of temporal ontology
Spatio-temporal data model
Spatio-temporal onotogy
Semantic relationship of spatio-temporal ontology
Ontology language
Entity
Attribute sets
Instances

Fig. 2. Framework of moving objects spatio-temporal ontology model

3.1 Spatio-temporal Ontology

Definition 3.1 (spatial ontology)**:** Spatial ontology models geometric data in spatial database, with basic triple structure *SO*=(*SE*, *SA*, *SI*), where *SE* is spatial entity, *SA* is attribute sets of *SE*, *SI* is instances of *SE*. Basic conceptual entities of spatial objects identified in spatial schema [7] are *GM_Primitive*, consisting in *GM_Point*, *GM_Curve*, *GM_Surface* and *GM_Solid*.

Definition 3.2 (temporal ontology)**:** Temporal ontology is a basic triple structure *TO*=(*TE*, *TA*, *TI*), where *TE* is temporal entity, *TA* is attribute sets of *TE*, *TI* is instances of *TE*. Temporal entities describes valid time dimension in temporal schema [8] which includes *TM_Instant* and *TM_Period*.

Definition 3.3 (spatio-temporal ontology)**:** Spatio-temporal ontology is a basic triple structure *STO*=(*STE*, *STA*, *STI*), where *STE* is spatio-temporal entity, *STA* is attribute sets of *STE*, *STI* is instances of *STE*. Moving objects are referred to as continuous spatiotemporal objects, changing their position, size/shape over time, so its concept is viewed as mapping from time objects *t* into space objects *s* with semantics $\tau(s)$:

$$\tau(s) = f : t \rightarrow s \ . \tag{1}$$

According to (1), 8 abstract entities of moving objects are identified, such as *ST_PointInstant*, *ST_CurveInstant*, *ST_SurfaceInstant*, *ST_SolidInstant*, *ST_PointPeriod*, *ST_CurvePeriod*, *ST_SurfacePeriod*, *ST_SolidPeriod*. Fig. 3 presents moving objects spatio-temporal ontology model.

3.2 Semantic Relationship

Topological relationship is recognized to be valuable information about integration among moving objects in the real world.

Definition 3.4 (spatial relationship)**:** 8 spatial topological relationships [9] are valid such as {*Meet*, *Disjoint*, *Overlap*, *Contains*, *Inside*, *Equal*, *Covers*, *CoveredBy*}.

Definition 3.5 (temporal relationship)**:** 13 binary operators [10] define mutually exclusive relationships between time intervals {*Equals*, *Before*, *After*, *Meets*, *MetBy*, *During*, *Contains*, *Starts*, *StartedBy*, *Finishes*, *FinishedBy*, *Overlaps*, *Overlapped-by*}.

Definition 3.6 (spatio-temporal relationship)**:** Spatiotemporal topological relationship between two moving objects can be defined as pairs of spatial relationship and temporal relationship in accordance with their evolvement over time.

$$f: \Theta st \to \Theta s \times \Theta t = \{(s, t), s\in \Theta s, t\in \Theta t\} \ . \tag{2}$$

As temporal relationship is concerned, in case of *Before*, *After*, *Meets* and *MetBy*, temporal overlap among moving objects never occurs. Therefore, in the procedure of evolution, only definitions of topology make sense upon the overlapped time period.

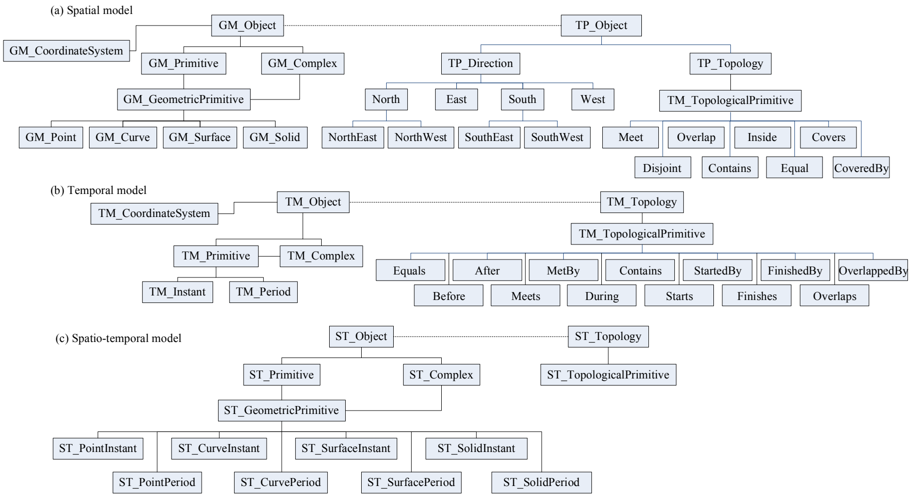


Fig. 3. Moving objects spatio-temporal ontology model

4 Moving Objects Data Mining Based on Lattice Theory

Data mining and knowledge discovery can be understood as concept formation from database, and representation of content and extension of concepts as well as abstract relationships among concepts. Concept lattice can be a good method for such mining process. Main tasks are as follows. (1) Design a unified data structure to represent nodes of lattice, construct concept lattice, and define generalization relationship and specialization relationship between concepts. (2) Using Pruning strategy to reduce numbers of nodes to achieve simplification of lattices. (3) Extract association rules from concept lattice. (4) Assessment of association patterns.

4.1 Concept Lattice Construction

Definition 4.1 (concept lattice)**:** Given a formal context of triple *T*=(*O*, *D*, *R*), where *O* is set of objects, *D* is set of descriptor (attribute), *R* is a binary relation between *O* and D, there exists one partially ordered set, which can produce a lattice structure. Lattice L induced from T is called a concept lattice. Each node in L is a pair, called concept, denoted by (Y, X), where $Y \in P(O)$, called concept extension; $X \in P(D)$, called concept content. Each pair about R should be complete, that is, only the largest expansion of the order even can appear in the lattice structure.

A partial order can be established between nodes of lattice. Given $H_1=(Y_1, X_1)$ and $H_2=(Y_2, X_2)$, then $H_1<H_2 \Leftrightarrow X_1 \subseteq X_2$, H_1 is parent node of H_2. In fact, there is a dual relationship between X_1 and X_2, that is $X_1 \subseteq X_2 \Leftrightarrow Y_2 \subseteq Y_1$, thus $H_1<H_2 \Leftrightarrow Y_2 \subseteq Y_1$. Concept lattice is essentially linked the two lattices. Hasse diagram of lattice is generated according to the partial order: If $H_1<H_2$, and there is no element H_3 making $H_1<H_3<H_2$, then there exists an edge from H_1 to H_2. Hasse diagram reveals generalization and specialization among connotation and extension of concepts.

Algorithm of construction of concept lattice is described as below.

(1) Initialize lattice L as an empty.
(2) Take an object o from O.
(3) For each $H_1=(Y_1, X_1)$ in concept lattice L, if $X_1 \subseteq f(o)$, then put o into Y_1.
(4) If $X_1 \cap f(o) \neq \varnothing$, $X_1 \cap f(o) \neq X_1$ and there does not exist a parent node (Y_2, X_2) of (Y_1, X_1) to meet $X_2 \supseteq X_1 \cap f(o)$, then create a new node.
(5) Put new node into L, and adjust the connections between nodes.
(6) Repeatedly (2)~(5) until all objects are processed.

Output concept lattice L. In the worst case, nodes of concept lattice will increase at exponential growth when context increases. Hence, in the case of large data sets, we need to prune grid nodes of concept lattice to control the growth of concept lattice.

4.2 Association Rules Generation

Suppose $I=\{i_1, i_2 \ldots i_n\}$ is a set of items. D is a set of database transaction. Each transaction T is a collection of items, making $T \subseteq I$. Each transaction has a unique ID, denoted by TID. An association rule is the implication of $A \Rightarrow B$, where $A \subset I$, $B \subset I$, and $A \cap B = \varnothing$.

Definition 4.2 (support)**:** Ratio of number of transactions including A and B and all transactions, denoted by $sup\ (A \Rightarrow B)$, i.e. $sup\ (A \Rightarrow B)=(A \cup B)$.

Definition 4.3 (confidence)**:** Ratio of number of transactions including A and B and number of transactions including A, denoted by $conf\,(A \Rightarrow B)$, i.e. $conf\,(A \Rightarrow B)=P(B|A)$.

Definition 4.4 (strong association rule)**:** Mining association rules whose support and confidence are greater than minimum support (*min_supp*) and minimum confidence (*min_conf*).

Definition 4.5 (frequent itemsets)**:** If frequency of itemsets is greater than or equal to the product of *min_supp* and total affairs in D, then called frequent item sets.

Theorem 4.1. All non-empty set in the frequent itemsets is also frequent.

For concept lattice of concepts $C=(A, B)$, support of B is $sup(B)=|A|/M$, where M is number of all objects. If $sup(B)>min\_sup$, then C becomes a frequent concept and B becomes a frequent item.

Theorem 4.2. If concept $C_1=(A_1, B_1)$, $C_2=(A_2, B_2)$ meet that C_2 is the super concept of C_1, then a rule is available: $B_2 \Rightarrow B_1 - B_2$, where $sup=|A_1|/M$ and conf$=|A_1|/|A_2|$. And conf$(B_1 - B_2 \Rightarrow B_2)$ is 100%.

Theorem 4.3. If concept $C_1=(A_1, B_1)$, $C_2=(A_2, B_2)$ do not meet that C_2 is the super concept of C_1 but exist non-empty extension of the largest public sub-concept $C=(A, B)$, then a rule is available between B_1 and B_2: $B_1 \Rightarrow B_2$, $B_2 \Rightarrow B_1$ with confidence are $|A|/|A_1|$ and $|A|/|A_2|$.

After mining association rules, it is also necessary interpret and understand the meanings of the rules. According to domain knowledge and knowledge types discovered, transformation model from qualitative description and quantitative representation is needed to develop, as well as interpretation method.

4.3 An Illustrative Example

Table 1 shows a formal context three moving objects appear within a certain region (A~E), where $O=\{1, 2, 3\}$, $D=\{Object\_Id, Location, Time\}$. Fig. 4 gives result of Hasse diagram of concept lattice and association rules corresponding to Table 1.

Table 1. An example of context

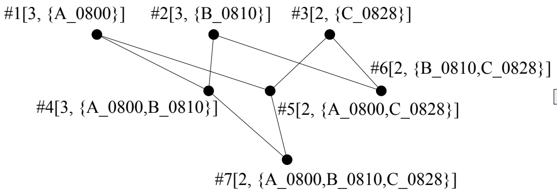
Object_ID	Location	Time
1	A	08:00
1	B	08:10
1	B	08:20
1	C	08:28
2	A	08:00
2	B	08:10
2	D	08:18
2	E	08:35
3	A	08:00
3	B	08:10
3	B	08:18
3	C	08:28

Spatio-Temporal Baskets

ST_Object	ST_PointInstant
1	A_0800, B_0810, B_0820, C_0828
2	A_0800, B_0810, D_0818, E_0835
3	A_0800, B_0810, B_0818, C_0828

Frequent Itemset Mining (>=2)

{A_0800}	3
{B_0810}	3
{C_0828}	2
{A_0800,B_0810}	3
{A_0800,C_0828}	2
{B_0810,C_0828}	2
{A_0800,B_0810,C_0828}	2



Associate Rule Mining (=1.0)

Association rule	Confidence
{A_0800}⇒{B_0810}	1.0
{C_0828}⇒{ A_0800,B_0810}	1.0
{A_0800,C_0828}⇒{B_0810}	1.0
{B_0810,C_0828}⇒{A_0800}	1.0

Fig. 4. Concept lattice diagram and association rules corresponding to Table 1

5 Conclusions and Future Work

Technical difficulties and bottlenecks mentioned in the paper always occur in moving objects data mining, which are decided by moving objects characteristics. Therefore, it is necessary to consider all aspects for mining during the life cycle of moving objects, such as data collection, storage, indexing, query and analysis, putting them in a unified framework. On the other hand, when discovering knowledge from moving objects database, data mining mechanism is based on the different perspectives of "*space-time data→space-time concept→space-time pattern*". Firstly, extract spatio-temporal concepts from moving objects data to establish conceptual model of moving objects. And then summarize spatio-temporal characteristics from concept spaces. Finally conclude spatio-temporal knowledge from characteristic space gradually.

An ideal theoretical framework should be able to model moving objects, which is the basis of moving objects data mining. According different types of data mining tasks in applications (such as clustering, association etc.), discover knowledge in moving objects database using mining algorithms. Theoretical framework of data mining, data models, data mining algorithms, and knowledge interpretation and representation in nature language are still big challenging tasks in the future research.

Acknowledgments. This work has been supported by the Natural Science Foundation of Hubei Province (Grant No. 2009CDB227), the Scientific Research Foundation for the Returned Overseas Chinese Scholars of State Education Ministry, and National 863 Funds (Grant No. 2009AA12Z324).

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
2. Giannotti, F., Pedreschi, D. (eds.): Mobility. Data Mining and Privacy–Geographic Knowledge Discovery. Springer, Berlin (2007)
3. Han, J., Cheng, H., Xin, D., Yan, X.: Frequent Pattern Mining: Current Status and Future Directions. Data Mining and Knowledge Discovery 15(1), 55–86 (2007)
4. Giannotti, F., Nanni, M., Pedreschi, D., Pinelli, F.: Trajectory Pattern Mining. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 330–339. ACM Press, New York (2007)
5. Chen, J., Meng, X., Guo, Y., Grumbach, S., Wang, H.: Modeling and Predicting Future Trajectories of Moving objects in a Constrained Network. In: Proceedings of the 7th International Conference on Mobile Data Management (MDM 2006), pp. 156–163. IEEE Computer Society Press, Washington (2006)
6. Mannila, H.: Theoretical Frameworks for Data Mining. SIGKDD Explorations 1(2), 30–32 (2000)
7. IS0 19107: ISO/TC 211 Geographical Information – Spatial Schema, p. 179 (2000)
8. IS0 19108: ISO/TC 211 Geographical Information – Temporal Schema, p. 56 (2000)
9. Egenhofer, M.J., Franzosa, R.D.: Point-set Topological Spatial Relations. International Journal for Geographical Information Systems 5(2), 161–194 (1991)
10. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Communications of the ACM 26(11), 832–843 (1983)

An Improved KNN Based Outlier Detection Algorithm for Large Datasets

Qian Wang and Min Zheng

School of Computer Science, Chongqing University, Chongqing, China
wangqian@cqu.edu.cn, brisis@qq.com

Abstract. Outlier detection is becoming a hot issue in the field of data mining since outliers often contain useful information. In this paper, we propose an improved KNN based outlier detection algorithm which is fulfilled through two stage clustering. Clustering one is to partition the dataset into several clusters and then calculate the Kth nearest neighbor in each cluster which can effectively avoid passing the entire dataset for each calculation. Clustering two is to partition the clusters obtained by clustering one and then prune the partitions as soon as it is determined that it cannot contain outliers which results in substantial savings in computation. Experimental results on both synthetic and real life datasets demonstrate that our algorithm is efficient in large datasets.

Keywords: Data mining, Knn, Outlier Detection.

1 Introduction

An outlier in a set of data is an observation or a point that is considerably dissimilar or inconsistent with the remainder of the data [1]. The earlier viewpoint that outliers must be quickly identified and removed so that they will not interfere with the data analysis is too narrow because this “nuisance” may contain useful information and has a number of applications in customer segmentation, weather prediction, medical analysis, telecom and credit card fraud and financial applications.

Recently, many studies have been conducted on outlier detection which can be classified into 5 categories: distribution based, distance based, depth based, density based and clustering based [2]. The problem of outlier detection has been extensively studied in the statistic community and distribution based approaches are typically based on this, see [3] for more details about statistical techniques. The main problem of distribution-based approaches is that in a number of situations, the user might simply not have enough knowledge about the underlying data distribution. In order to solve this problem, distance based approach is proposed by Knorr and Ng [4] in which knowledge of data distribution is not reqiured. Sridhar Ramaswamy and Rajeev Rastogi propose a novel formulation based on the distance of a point from its Kth nearest neighbor and declare the top n points in this ranking to be outliers [1]. Depth-based approaches take the consideration that outliers are more likely to be objects with smaller depth which are based on computational geometry and computing different layers of k-d convex hulls [5][6].Density-based approach proposed by

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 585–592, 2010.

Breunig defines LOF for an object which enjoys many desirable properties [7]. Clustering-based approaches detect outliers as by-products [8].

In this paper, we propose an outlier detection algorithm which can be divided into two clustering steps. Step 1, the simplest clustering algorithm is adopted to obtain clusters so that we can calculate the Kth nearest neighbor for each object in each cluster instead of in the whole dataset. Step 2, K-MEDOIDS is applied to re-cluster the clusters obtained by step 1 into disjoint subsets, and then prunes the subsets cannot contain outliers to reduce the unnecessary computing. In the end, experiments are conducted to evaluate the performance of our algorithm.

The rest of the paper is arranged as follows. Section 2 elaborates two outlier detection algorithms that motivate our algorithm. Section 3 discusses our two stage clustering-based algorithm. Section 4 analyzes the performance of the algorithm by experiments. Section 5 concludes the paper.

2 Two Outlier Detection Algorithms

In this section, we will elaborate two outlier detection algorithms that motivate our algorithm which are partition-based and clustering-based.

2.1 Partition-Based Algorithms for Outlier Detection

The key idea underlying the partition-based algorithm is to first partition the data space, and then prune partitions as soon as it can be determined that they cannot contain outliers. Since n will typically be very small, this additional preprocessing step performed at the granularity of partitions rather than points eliminates a significant number of points as outlier candidates. Consequently, Kth nearest neighbor computations need to be performed for very few points, thus speeding up the computation of outliers. Furthermore, since the number of partitions in the preprocessing step is usually much smaller compared to the number of points, and the preprocessing is performed at the granularity of partitions rather than points, the overhead of preprocessing is low.

Generate Partitions: In the first step, a clustering algorithm BIRCH [13] is used to cluster the data and treat each cluster as a separate partition.

Compute Bounds on Dk (the Kth nearest neighbor of p) for Points in Each Partition: For each partition P, compute itslower and upper bounds (stored in P.lower and P.upper, respectively) on Dk for points in the partition. Thus, for every point p∈ P, Dk(p) ≥P.lower and Dk(p) ≤P.upper.

Identify Candidate Partitions Containing Outliers: In this step, the candidate partitions are identified, that is, the partitions containing points which are candidates for outliers. Here minDkDist is computed, the lower bound on Dk for the n outliers. Then, if P.upper for a partition P is less than minDkDist, none of the points in P can possibly be outliers. Thus, only partitions P for which P.upper ≥ minDkDist are candidate partitions.

Compute Outliers from Points in Candidate Partitions: In the final step, the outliers are computed from among the points in the candidate partitions. For each candidate partition P, let P.neighbours denote the neighboring partitions of P, which are all the partitions within distance P.upper from P. Points belonging to neighboring partitions of P are the only points that need to be examined when computing Dk for each point in P.

2.2 KNN Based Outlier Detection Algorithm in Large Datasets

The idea of this algorithm is to firstly obtain clusters on the dataset so that the similar objects are more likely within the same cluster. Then calculate the Kth nearest neighbour in each cluster and find outliers utilizing the prune methods. Since the neighbours of objects are almost searched within the cluster rather than on the entire dataset, this algorithm is efficient for outlier detection in large dataset. In this algorithm, Squeezer Algorithm is used to identify groups of similar behaviour in the datasets during the clustering phase and R-tree is utilized to calculate Dk.

3 Two Stage Clustering-Based Algorithm

Two stage clustering-based algorithm proposed in this section is motivated by the algorithms presented in the previous section. The first step, we cluster the entire dataset into several clusters by the simplest clustering algorithm. The purpose of this step is to calculate the Kth nearest neighbor in each cluster. The second step, we then partition each cluster into small partitions and then delete the partitions when they are determined cannot contain outliers. In this step, K-MEDIODS is adopted.

3.1 Clustering One

In this phase, the simplest clustering algorithm is applied to cluster the entire dataset. We have stated before that the purpose of this clustering is to calculate the Kth nearest neighbor in each cluster so the clusters generated here do not need to be small, on the contrary, they should be respectively large since we have to ensure the Kth nearest neighbor of p and p itself are in the same cluster. So we do not need to choose some complex clustering algorithms.

The simplest clustering algorithm is presented as follow:

```
Input: dataset R with n objects and the similarity measure (distance) threshold D.
Output: the clusters C1,C2,C3…
i=2, j=1;
Step 1: Take an object Pi from R and make it the center of cluster 1 Cj;
Step 2: If i≤ n, take another object pi from R, calculate the distance D1i,D2i…Dji between C1,C2…Cj and pi(we take the distance between the center of Cj to pi as Dji), and to step 3, Otherwise output the clustering results;
Step 3: Find the minimum of the distances calculated above, If it is no bigger than D, assign pi to Cj where Dji is the minimum, i++, and go to step 2, otherwise, go to step 4;
Step 4: j=j+1, make pi the center of cluster j Cj. Go to Step 2.
```

3.2 Clustering Two

After clustering one, we will treat each cluster as a separate partition, or we could call it sub-dataset. We have clustered the entire dataset into several sub-datasets so the sub-dataset is no longer a large dataset. It is respectively small so K-MEDIODS—very effective for small data sets—is applied. K-MEDIODS is an algorithm which improves K-MEANS and able to deal with any type of data, K-MEDIODS can also eliminate the sensitivity of abnormal data. In this paper, we choose PAM which is the first version of K-MEDIODS and the simplest. See [14] for more about PAM.

We do not cluster each sub-dataset obtained by clustering one, but only the ones with more than certain number of objects.

3.3 Computing Bounds for Partitions

For the purpose of identifying the candidate partitions, we need to first compute the bounds P.lower and P.upper, which have the following property: for every point p∈ P, Dk(p) ≥P.lower and Dk(p) ≤P.upper. Since Procedure computeLowerUpper for computing P.lower and P.upper for partition P is too long, we only state some key procedure below .

Computation of Lower and Upper Bounds for Partitions

```
begin
   nodeList := { Root} ;P.lower := P.upper :=∞
   lowerHeap := upperHeap := Φ
   while nodeList is not empty do {
     delete the first element, Node, from nodeList
     if (Node is a leaf) {for each partition Q in Node {
      if (MINDIST(P,Q)< P.lower) {lowerHeap.insert(Q)
        while lower Heap.numPoints()-
        lowerHeap.top().numPoints ≥ k do
        lowerHeap.deleteTop()
      if (lowerHeap.numPoints() ≥ k)
             P.lower := MINDIST(P, lowerHeap.top())}
      if (MAXDIST(P,Q)< P.upper) {upperHeap.insert(Q)
       while upperHeap.numPoints()
          upperHeap.top().numPoints()≥ k do
            upperHeap.deleteTop()
          if (upperHeap.numPoints()≥ k)
            P.upper := MAXDIST(P, upperHeap.top())
              if (P.upper≤minDkDist) return} } }
   else ......
  end.
```

3.4 Computing Candidate Partitions

This is a crucial step in our algorithm in which we identify the candidate partitions that can potentially contain outliers, and prune the remaining partitions. The idea is to use the bounds computed in the previous section to first estimate minDkDist, which is a lower bound on Dk for an outlier. Then a partition P is a candidate only if P.upper ≥ minDkDist. The lower bound minDkDist can be computed using the P.lower values for the partitions as follows. Let P1…,Pl be the partitions with the maximum values for P.lower and containing at least n points. Then minDkDist= min {Pi.lower:1≤i≤l} is a lower bound on Dk for an outlier.

Computation of Candidate Partitions

```
   Procedure computeCandidatePartitions(PSet, k, n)
   begin
    for each partition P in PSet doinsertIntoIndex(Tree, P)
     partHeap := Ö;minDkDist :=0
    for each partition P in PSet do {
     computeLowerUpper(Tree.Root, P, k, minDkDist)
     if (P.lower >minDkDist) {partHeap.insert(P)
     while partHeap.numPoints()-partHeap.top().numPoints()
     ≥n do partHeap.deleteTop()
      if (partHeap.numPoints()≥ n)
      minDkDist := partHeap.top().lower} } candSet := Ö
    for each partition P in PSet do
    if (P.upper≥minDkDist) {candSet := candSet ∪ {P}
     P.neighbors := {Q: Q∈PSet and MINDIST(P,Q)≤P.upper}
     } return candSet
  end.
```

3.5 Computing Outliers from Candidate Partitions

In the final step, we compute the top n outliers from the candidate partitions in candSet. If points in all the candidate partitions and their neighbors fit in memory, then we can simply load all the points into a main memory spatial index. The index-based algorithm [4] can then be used to compute the n outliers by probing the index to compute Dk values only for points belonging to the candidate partitions. Since both the size of the index as well as the number of candidate points will in general be small compared to the total number of points in the data set, this can be expected to be much faster than executing the index-based algorithm on the entire data set of points.

Index-Based Algorithm for Computing Outliers

```
Procedure computeOutliersIndex(k,n)
begin
 for each point p in input data set do
```

```
  insertIntoIndex(Tree, p); outHeap := Ö; minDkDist := 0
 for each point p in input data set do  {
  getKthNeighborDist(Tree.Root, p, k, minDkDist)
    if (p.DkDist>minDkDist) {outHeap.insert(p)
    if (outHeap.numPoints()>n) outHeap.deleteTop()
    if (outHeap.numPoints() = n)
    minDkDist := outHeap.top().DkDist } }
return outHeap
End
```

4 Experimental Results

4.1 Performance Results on Real Life Dataset

To assess the effectiveness and accuracy of our algorithm, we performed several experiments on Intel(R) Core(TM)2 Duo CPU 2.10GHz with 2GB RAM running windows 7. We empirically compared the performance of our algorithm with partition-based algorithm and another KNN based Algorithm proposed in [1] and [12] with the experiments conducted on both synthetic datasets and real life dataset obtained from the UCI machine learning repository.

The real life dataset used here is the adult data set extracted from US Census Bureau's Income data set with labels indicating the individual's yearly income level (6 continuous and 8 categorical attributes). And the following data records are marked as the top 3 outliers:

A 39 year old male from Asia with a doctorate, working in a university as the professor makes less than 50,000 dollars per year.

A 42 year old self-employed male from Iran, with a bachelor degree, working in an Executive position makes less than 50,000 dollars per year.

A 45 year old Alaska Native female with an Asian Pacific Islander origin out of work and makes more than 50,000 dollars per year.

The rest outliers found tend to come from people from foreign countries which are not well represented in this data set. We also conducted this experiment by implementation of the other two algorithms stated above; the executive time of our two stage clustering-based algorithm is much faster.

4.2 Performance Results on Synthetic Data

For our experiments, we use a data generator to produce synthetic datasets in which all data are generated with equal probability. The execution times for the partition based algorithm and our algorithm as N is varied from 100000 to 500000 are shown using a log scale in Figure 1.

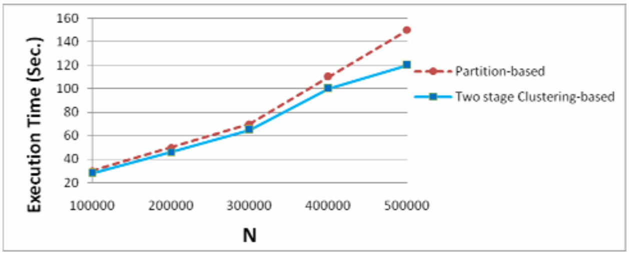


Fig. 1. Performance Results for N

As the figure illustrates, the executive times for all the three algorithms increases about linearly with input size but our algorithm takes less time than the other two and the slope is less than the other two which means with the size increasing, the executive time of our algorithm increases slower. This is all because the dataset is big and we cluster it twice which results in tremendous savings in both I/O and computation, and enables the two stage clustering-based to outperform the other two algorithms.

Now consider the scalability of two stage clustering-based algorithm. We conduct the experiment with varying N from 10000 to 50000 and dimensionality from 10 to 30. As showed in figure 2, with the increasing of size and dimensionality of dataset, our algorithm can achieve approximately linear scalability.

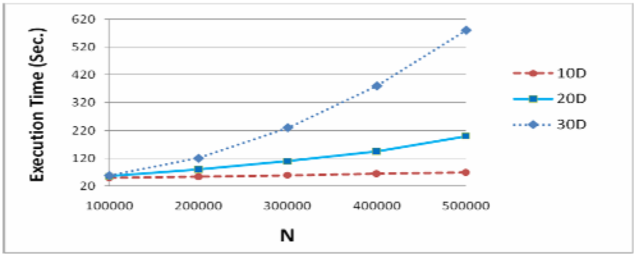


Fig. 2. Performance Results on Scalability

5 Conclusions

In this paper, we proposed an efficient outlier detection algorithm based on clustering the dataset twice. Clustering one implements the simplest clustering algorithm to partition the dataset into several clusters and then calculate the Kth nearest neighbor in each cluster. Clustering two implements PAM (Partitioning Around Medoids) to partition the clusters obtained by clustering one and then prune the partitions as soon as it is determined that it cannot contain outliers. Experimental results on both synthetic and real life datasets show that our algorithm is more efficient than the traditional algorithms in large datasets and also has good scalability.

References

1. Ramaswamy, S., Rastogi, R., Kyuseok, S.: Efficient algorithms for mining outliers from large data sets. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 93–104. ACM Press, New York (2000)
2. Birant, D., Kut, A.: Spatio-temporal outlier detection in large databases. In: Proceedings of Conf. Information Technology Interfaces, pp. 179–184 (2003)
3. Barnett, V., Lewis, T.: Outliers in Statistical Data. John Wiley and Sons, New York (1994)
4. Knorr, E., Ng, R.: Algorithms for mining distancebased outliers in large datasets. In: Proceedings of the 24th Conference on VLDB, New York, pp. 392–403 (1998)
5. Johnson, T., Kwok, I., Ng, R.: Fast Computation of 2-Dimensional Depth Contours. In: Proceedings of 4th. Int. Conf. on KDD, New York, pp. 224–228 (1998)
6. Ruts, I., Rousseeuw, P.: Computing Depth Contours of Bivariate Point Clouds. Journal of Computational Statistics and Data Analysis (23), 153–168 (1996)
7. Breunig, M.M., Kriegel, H.P., Ng, R.T.: LOF: Identifying density based local outliers. In: Proceedings of ACM Conference, pp. 93–104 (2000)
8. Jain, A., Murty, M., Flynn, P.: Data Clustering: A Review. ACM Computing Surveys 31(3), 264–323 (1999)
9. Ng, R.T., Han, J.: Efficient and Effective Clustering Methods for Spatial Data Mining. In: Proceedings of 20th Int. Conf. on Very Large Data Bases, Santiago, Chile, pp. 144–155 (1994)
10. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: Clustering for Mining in Large Spatial Databases. KI-Journal (Artificial Intelligence), Special Issue on Data Mining 12(1), 18–24 (1998)
11. Guha, S., Rastogi, R., Shim, K.: CURE: An Efficient Clustering Algorithms for Large Databases. In: Proceedings of ACM SIGMOD Int. Conf. on Management of Data, Seattle, WA, pp. 73–84 (1998)
12. Yang, P., Huang, B.: An efficient outlier mining algorithm for large dataset. In: Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering, vol. 1, pp. 199–202 (2008)
13. Zhang, T., Ramakrishnan, R., Birch, M.L.: An efficient data clustering method for very large databases. In: Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada, pp. 103–114 (June 1996)
14. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, Chichester (1990)

Some Developments of Determinacy Analysis

Rein Kuusik and Grete Lind

Department of Informatics, Tallinn University of Technology
Raja 15, Tallinn 12618, Estonia
kuusik@cc.ttu.ee, grete@staff.ttu.ee

Abstract. This paper deals with development of Determincy Analysis (DA), a method for data mining. There are two approaches to DA that give a different result consisting of non-overlapping (exclusive each other) rules. The first method finds exactly one system consisting of rules that have equal length. The second, step by step approach enables to find very many different rule systems where the rules have different length. In the first case the rules contain a lot of redundant attributes, in the second case there are too many different (formally complete) systems of rules what makes the selection hard. A better result can be obtained by finding overlapping rules. This paper presents DA approach and technique that enables to find overlapping rules with different length and algorithm realizing it. Such approach for DA has not been created earlier.

Keywords: Determinacy Analysis, Data Mining, Overlapping rules.

1 Introduction

In this paper we deal with a data mining method called Determinacy Analysis. Determinacy Analysis (DA) is a system of methods for the analysis of rules that was created at the end of 1970s. Its approach combines mathematical statistics and logic. DA's methodology and the underlying mathematics are developed by a Russian scientist Sergei Chesnokov [1], [2].

The primary goal of DA is data mining specifying the objects of class Y to try to answer the question "who/what are they?" i.e. to describe them the best way using specified attributes and to output several sets of rules from which each one covers Y 100%. The user makes the decision which set or sets are the best for describing. Diffrerent orders of attributes and different sets of attributes are tried until the most suitable description in the user's opinion is found.

In the middle of 90s a next version of DA was developed by Chesnokov and it is known in Russia and the former republics of USSR as a software package DA-system[1] (Determinacy Analysis System or shortly DAS).

Known methods of DA can extract one or more rule systems where the rules do not overlap. The two main approaches to DA are: 1) DAS, 2) step by step method.

The initial data table and a feature Y (as a certain class) are given. The goal is to describe Y (possibly) completely by the non-overlapping (possibly) accurate rules.

[1] Software package DA-system (Russian version only) by "Context Media" (http://www.context.ru), is described also in [3].

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 593–602, 2010.

Different sets of rules may be found and different ways to find them can be used.

DAS extracts the rules with equal length (= the number of factors in the rule), all of them contain the same attributes – this is a simple way to get a set of non-overlapping rules.

In step by step approach described in [3] the extracted rules can have different lengths while they do not overlap. Attributes (factors) are added into the rules one by one. If a rule is accurate it will not be expanded by adding the next factor. At the same time the completenesses of found accurate rules are summed up. Reaching 100% the coverage is found. The user decides about the order in which the attributes are included into the rules, from the beginning until the situation when all objects of the class are covered. The order of factors (attributes) is essential, different orders lead to the different results.

Both approaches have problems with the amount of rules and the lenght of rules. We deal with these problems, analyze them and describe a new approach to DA which can solve these problems. It can extract a rule system with a lower number of rules and with shorter rules.

2 Determinacy Analysis

DA-technology assists in obtaining regularities, explanations and description rules. DA has been used in sociology [4], linguistics [5], medicine [6], and other areas (for the complete list of references see [7] or [8]).

The overview of determinacy analysis is based on [1], [2], [8], [9].

2.1 Determination and Its Characteristics

The main idea behind DA is that a rule can be found based on the frequencies of joint occurrence or non-occurrence of events. Such rule is called a determinacy or determination, and the mathematical theory of such rules is called determinacy analysis [8].

If it is observable that an occurrence of X is always followed by an occurrence of Y, this means that there exists a rule "If X then Y", or X→Y. Such correlation between X and Y is called a *determination* (from X to Y). Here X is a *determinative (determining)* and Y is a *determinable*.

The determinative (X) consists of one or more factors. A factor is an attribute with its certain value. Each attribute can give as many different factors as many different values it has. The factors coming from the same attribute are not contained in the same X.

Each rule has two characteristics: accuracy and completeness[2].

The accuracy of the determination X→Y shows to what extent X determines Y. It is defined as a proportion of occurrences of Y among the occurrences of X: $A(X \rightarrow Y) = n(X\ Y) / n(X)$, where $A(X \rightarrow Y)$ is the accuracy of determination, n(X) is the number of objects having feature X and n(X Y) is the number of objects having both features X and Y.

[2] In the beginning (in [1], for example) "accuracy" (Russian "точность") was called "intensity" and "completeness" ("полнота") was called "capacity" ("емкость").

The completeness of the determination X→Y shows which part of the objects having Y can be explained by the determination X→Y. It is a percentage of occurrences of X among the occurrences of Y: C(X→Y) = n(X Y) / n(Y), where C(X→Y) is the completeness of determination, n(Y) is the number of objects having feature Y and n(X Y) is the number of objects having both features X and Y.

Both accuracy and completeness can have values from 0 to 1. Value 1 shows maximal accuracy or completeness, 0 means that rule is not accurate or complete at all. Value between 0 and 1 shows quasideterminism.

If all objects having feature X have also feature Y then the determination is (maximally) accurate. In case of an accurate determination A(X→Y) = 1 (100%). Most of the rules are not accurate. In case of an inaccurate rule A(X→Y) < 1.

In order to make a determination more (or less) accurate the complementary factors are added into the left part of the rule. Adding the factor Z into the rule X→Y we get a rule XZ→Y.

The contribution of factor Z *to accuracy* of rule XZ→Y is measured by the increase of accuracy A(Z) caused by addition of factor Z into the rule X→Y: A(Z) = A(XZ→Y) – A(X→Y). The contribution to accuracy falls into interval from -1 to 1.

If A(Z)>0 then Z is a positive factor. Addition of a positive factor makes the rule more accurate, sometimes the resultant rule is (maximally) accurate. If A(Z)<0 then Z is a negative factor. Addition of a negative factor decreases the rule's accuracy, sometimes until zero. If A(Z)=0 then Z is a zero (or inessential) factor. Addition of a zero factor does not change the rule's accuracy. *An accurate rule* contains no negative factors, all factors are positive or zero factors [9].

If C(X→Y)=1 (100%) then the rule X→Y is (maximally) complete. It means that Y is always explained by X. In case of an incomplete rule C(X→Y)<1, it means that X does not explain all occurrences of Y.

The contribution of factor Z *to completeness* of rule XZ→Y is measured by the increase of completeness C(Z) by addition of factor Z into the rule X→Y: C(Z) = C(XZ→Y) – C(X→Y). The contribution of whatever factor to completeness is negative or zero [9].

The system of rules is a set of rules in the form of $S_q = \{x_i \to y \mid i=1,2,...,q\}$, where q is the number of rules. Every system is characterised by average accuracy, summarised completeness and summarised capacity (the number of objects covered by the rules).

The system of rules $S_q = \{x_i \to y \mid i=1,2,...,q\}$ is *additive* when x_i-s pairwise do not overlap. The completeness and capacity of the additive system are just summed up completenesses and capacities of the rules. The accuracy of the additive system is not additive (i.e. equal to the sum of rules' accuracies), it is found as a weighted average.

A system is called *complete* if its completeness is 1. A s*ystem* is called *accurate* if its accuracy is 1. A system is accurate when all of its rules are accurate.

The rank of a rule is a dimension of its left side. A rule in the form of $z_1z_2...z_r \to y$ is called a rule of rank r ($r \geq 1$). A system of rules in the form of $z_1z_2...z_r \to y$ is called a system of rules of rank r by variables z_1, z_2, ..., z_r relative to the feature y. Every system of rules of rank $r \geq 1$ by fixed set of r variables is additive [9].

The theory of DA covers also the non-additive systems of (overlapping) rules, but to our knowledge no algorithms for finding such systems have been published.

2.2 Example

The following example illustrates step by step approach. The data from [10] will be used in example (see Table 1). This data table (object-attribute system) contains 8 objects described by 4 attributes. The last attribute shows the object's belonging to a certain class (feature Y). Attributes Height, Eyes and Class have two possible values each, attribute Hair has three alternative values.

We will describe persons (objects) belonging to class "+". This class consists of five objects (n(Y)=5).

Attributes will be added into the rules in the following (freely chosen) order: Hair, then Eyes and then Height.

Table 1. Quinlan's data

Height	Hair	Eyes	Class
tall	dark	blue	+
short	dark	blue	+
tall	blond	blue	–
tall	red	blue	–
tall	blond	brown	+
short	blond	blue	–
short	blond	brown	+
tall	dark	brown	+

Rules containing attribute Hair only are given in Table 2.

Table 2. The rules consisting of attribute Hair

Hair	n(X)	n(XY)	A	C	ΣC
dark	**3**	**3**	**1**	**3/5**	**3/5**
red	1	0	0	0	
blond	4	2	1/2	2/5	

The rule Hair.dark→Class.+ is accurate (A=1) and needs no additional factors. This rule is included into the result. It covers 60% of the objects of the class (C=3/5). The completenesses C of the accurate rules are summed up (ΣC), with hope to reach the 100% coverage (by accurate rules).

The rule with Hair.red has zero accuracy (i.e. does not exist) in given class and will not be expanded.

The rule with Hair.blond has accuracy between 0 and 1, thus we expand it by adding the next attribute (Eyes) into it (see Table 3).

The second of found rules (with Eyes.brown) has accuracy 1 and will be included into the result. Its completeness is 40% (C=2/5). Now the summed completeness is 100%, thus the class "+" is (fully) covered by the (accurate) rules.

Table 3. The rules consisting of attributes Hair and Eyes

Hair	Eyes	n(X)	n(XY)	A	C	ΣC
blond	blue	2	0	0	0	
blond	**brown**	**2**	**2**	**1**	**2/5**	**1**

At the same time we can see that the other branch (Hair.blond and Eyes.blue) has zero accuracy and there is no reason to expand it.

As the found rules cover the class completely there is no need to use the next attribute (Height), so the class is described without using it.

Class "+" is covered by two rules: 1) Hair.dark → Class.+ (C = 60%); 2) Hair.blond&Eyes.brown → Class.+ (C = 40%).

This is one possible description for class "+" beginning from the attribute Hair. In the "language" of DA it means that they are people with dark hair or with blond hair and brown eyes. The user has to know what to do with this knowledge. If "+" means "beauties", for example, then if the user is satisfied with such description then (s)he accepts it. If this knowledge is not satisfying then the user can change the order of attributes or add some new attributes.

3 Problems

First we present some examples of different systems of rules using the data from [10] (see Table 1). The purpose is to determine Class "+" by additive rules containing attributes Eyes and Hair.

1. The additive system consisting of accurate rules of different rank (number of factors) got using step by step approach can be (*S1*): 1) Hair.dark → Class.+, C = 60% (3 objects); 2) Hair.blond & Eyes.brown → Class.+, C = 40% (2 objects) or (*S2*): 1) Eyes.brown → Class.+, C = 60% (3 objects); 2) Eyes.blue & Hair.dark → Class.+, C = 40% (2 objects).

   S1 is found when attributes are included in the order: first Hair, then Eyes. In case of S2 the order is: first Eyes, then Hair. In both cases the system is accurate and complete and the rules do not cover the same objects. As we can see the different rule systems describe the objects practically differently giving a different knowledge about the subset of objects. The user decides how many rule systems is enough. If the description given by the rule systems is not good enough then some other set of attributes can be chosen or just some attributes can be exchanged.

2. The additive system with a fixed set of factors (rank r=2) got using DAS (*S3*): 1) Eyes.blue & Hair.dark → Class.+, C = 40% (2 objects); 2) Eyes.brown & Hair.dark → Class.+, C = 20% (1 object); 3) Eyes.brown & Hair.blond → Class.+, C = 40% (2 objects).

   The system is (maximally) accurate (the overall accuracy is 100%) and (maximally) complete (the sum of completenesses is 100%). Maximal completeness shows that all objects (belonging to the class) are covered and maximal accuracy says that there is no need to add any more factors into the analysis.

Looking at these examples we can see that an additive system containing rules with different lengths (both S1 and S2) has lesser number of rules than an additive system with a fixed rank and set of attributes (S3). S3 contains the longer rule from S1 and the longer rule from S2 and also a combination of shorter rules from S1 and S2.

It is desired that the rules were relatively short – then it is easier to interpret them. Also the number of rules has not to be very large – for the same reason.

The task to find an additive system of rules can be understood in several ways:

1. Originating from DAS, all the rules contain all given attributes and have equal length.
2. Originating from the step by step approach, it is possible to find a system of rules of different length. Different order of inclusion of attributes can give different systems of rules.

In the first case, on the assumption of the essence of the DAS approach, there is exactly one solution. If each rule has to contain all (given) attributes, then we cannot take away any factors from any rules and consequently we cannot change such system of rules. Determining a different set of attributes (for description) we get a different system which is also the only possible solution for that set of attributes (in case of the same requirement – to contain all attributes). In this case the purpose is to find the most suitable set of attributes i.e. exclude redundant attributes (like Height in our example) and include all necessary attributes to avoid contradictions – a situation where the set of attributes is not sufficient to distinguish between different classes. A suitable set can be found by testing different sets of attributes.

In the second case – non-overlapping rules (i.e. additive system) of different lenght (rank) – the constitution of the system depends on the order in which the factors are included into the rules. It might be hard to find a compact system of such rules. It would be good to allow non-fixed order of including factors.

The recent approach to DA emanates from logic: if one rule system does not satisfy then the next one will be considered etc. Such treatment comes from the fact that the rules of one rule system are derived from the uniform order of attributes and thus they are mutually related. It means that we cannot change the order of attributes in the emergent rule system. This is an important restriction, according to it the best description in a sense can form from single rules of different rule systems. One solution here is to generate all rule systems and according to some criteria find from them a cover i.e. a rule set (with completeness 100%) consisting of the shortest rules or a rule set with the least number of rules. This turns out to be very labor-consuming because all the possible sets and orders (permutations) of attributes should be found. Next we will show that simpler solutions exist. A better result can be obtained by finding overlapping rules i.e. non-additive system of rules.

4 New Approach

Here we describe a new approach to DA and a simple way for finding overlapping rules, the algorithm realizing it and a new very simple technique for rule detection. The system of rules it finds is non-additive (i.e. objects may be covered by more than one rule), accurate (i.e. consists of accurate rules only) and complete (i.e. covers all objects of determinable class) if there are no contradictions in the data (the algorithm can find them). The findable set of rules is possibly small – the potential rules are not included into the result if they cover only such objects that are covered already.

The selection criteria for choosing the next factor are based on frequencies, the maximal frequency in Fy_t, for example. In case of equal maximal frequencies in Fy the one having bigger frequency in Fx is preferred.

```
Algorithm
Determine tables X and Y
S0. t:=0; $U_t$:=∅
    If all the objects in Y are covered then Goto End
S1. Find frequencies in tables $X_t$ and $Y_t$: $Fx_t$, $Fy_t$
S2. For each factor A such that $Fy_t$(A)=$Fx_t$(A) and all
    objects containing A are not covered by rule(s)
       output rule {$U_i$}&A, i=0,…,t
    If at least one new rule was found Goto S0
S3. Choose a new (free) factor $U_t$
    If there are no factors to add then
       {$U_i$}, i=0,…,t is a contradiction; Goto S0
    t:=t+1; extract subtable of objects containing $U_t$;
       Goto S1
End. System of rules is found
```

4.1 Example 1

In the examples we use the same Quinlan's data [10] as in section 2.2 (see Table 1), but this time a numerical representation of data is used. The coding used for attributes and their values is shown in Table 4. The initial data table is given in Table 5. Let X is X(8,3), X_{ij} = 1,...,3 and Y=4.1 {Y_i: 1,2,5,7,8} (i.e. class "+"). The frequencies of attributes' values for X and Y (Fx and Fy accordingly) are given in Table 6.

Table 4. The coding used for Quinlan data

Attribute	**Height**	**Hair**	**Eyes**	**Class**
Code	1	2	3	4
1	short	dark	blue	–
2	tall	red	brown	+
3		blond		

Table 5. The initial data table

i\j	1	2	3	4
1	2	1	1	1
2	1	1	1	1
3	2	3	1	2
4	2	2	1	2
5	2	3	2	1
6	1	3	1	2
7	1	3	2	1
8	2	1	2	1

Table 6. The frequencies for X and Y

Fx	Kj \ j	1	2	3	Fy	Kj \ j	1	2	3
	1	3	3	5		1	2	3	2
	2	5	1	3		2	3	0	3
	3	0	4	0		3	0	2	0

For the attribute 2 value 1 (shortly 2.1) the frequencies in Fx and Fy are equal which means that all objects having 2.1 belong to Y. Hence we get a rule 2.1=3 (Hair.dark→Class.+). The frequency after "=" shows that the rule covers three objects (namely objects 1, 2 and 8) – this is additional information. Also for 3.2 the frequencies are equal and we get another rule 3.2=3 (Eyes.brown→Class.+) that also covers three objects (5, 7 and 8).

By those two rules all the objects belonging to Y are covered. The found rules are overlapping, both cover object 8.

In order to demonstrate other steps of the algorithm another example is presented.

4.2 Example 2

The same data as in the previous example is used (see Table 5). This time Y=4.2 {Y_i: 3,4,6} (class "–"). The frequency tables Fx and Fy are given in Table 7.

Table 7. The frequencies for X and Y

Fx	Kj \ j	1	2	3	Fy	Kj \ j	1	2	3
	1	3	3	5		1	1	0	3
	2	5	1	3		2	2	1	0
	3	0	4	0		3	0	2	0

For 2.2 the frequencies in Fx and Fy are equal. The rule 2.2=1 (Hair.red→Class.–) covers object 4.

The "free" frequencies i.e. the frequencies over non-covered objects (in Y) after extracting the first rule are shown in Table 8.

Table 8. Free frequencies for Y after extracting first rule

Fy	Kj \ j	1	2	3
	1	1	0	2
	2	1	0	0
	3	0	2	0

From here the factor starting a new rule is chosen by maximal frequency in Fy. As there are two (equal) maximal frequencies, the choice is made by bigger frequency in Fx where the factor 3.1 has frequency 5 and 2.3 has frequency 4. Thus 3.1 is selected.

Next the objects having 3.1 are extracted (see Table 9) and the frequencies for X and Y over extracted data are found (see Table 10). The frequencies of covered objects (Fc) are given in Table 11.

Table 9. Extract by 3.1

i \ j	1	2	3	4
1	2	1	1	1
2	1	1	1	1
3	2	3	1	2
4	2	2	1	2
6	1	3	1	2

Table 10. The frequencies of extracted data

Fx	Kj \ j	1	2	3	Fy	Kj \ j	1	2	3
	1	2	2	5		1	1	0	3
	2	3	1	0		2	2	1	0
	3	0	2	0		3	0	2	0

Table 11. The frequencies of covered objects after extracting the first rule

Fc	Kj \ j	1	2	3
	1	0	0	1
	2	1	1	0
	3	0	0	0

In Table 10 we can see two potential rules, (3.1) with 2.2 and (3.1) with 2.3. Looking into Fc (Table 11) we see that 2.2 has equal frequency here indicating that the object(s) containing factor 2.2 is(are) already covered by the found rules. Therefore 2.2 does not produce a new rule (rule 3.1&2.2=1 would be redundant covering the same object as rule 2.2=1). The factor 2.3 is suitable for ending a rule, its frequency in Fc is less than 2. So, the second rule is 3.1&2.3=2 (Eyes.blue&Hair.blond→Class.–), it covers objects 3 and 6.

Table 12. The frequencies of covered objects after extracting the second rule

Fc	Kj \ j	1	2	3
	1	1	0	3
	2	2	1	0
	3	0	2	0

The two rules we have found cover all the objects belonging to Y. We detect it turning back to the initial level and comparing initial Fy with Fc (after extracting the last rule – see Table 12): all the frequencies in both tables are equal.

5 Conclusion

This paper gives an overview of Determinacy Analysis, introducing the theory and different approaches. The theory by Chesnokov treats both additive and non-additive systems of rules. Solutions for finding additive systems of rules are known earlier.

Non-additive systems allowing overlapping rules could be more compact, but to our knowledge no method or application of DA for non-additive systems is published. In this paper we propose a method and the corresponding algorithm for finding a non-additive system of rules. Our algorithm is based on frequency tables which makes it easy to detect a potential rule and whether it covers any uncovered object. One rule system is found in which the number of rules is not bigger (and usually is smaller) than in case of DAS or step by step method and the rules are shorter also. In addition, the algorithm is able to detect a contradiction – the situation where identically described objects belong to different classes.

Realizing the algoritm it is possible to apply several different principles for selecting the next factor, one possibility is to let the user make the decision. Selection criteria is the direction for the future work.

References

1. Chesnokov, S.V.: Determination-Analysis of Social-Economic Data in Dialogical Regime. Preprint. All-Union Institute for Systems Research, Moscow (1980) (in Russian)
2. Chesnokov, S.V.: Determinacy Analysis of Social-Economic Data. Nauka, Moscow (1982) (in Russian)
3. Lind, G., Kuusik, R.: Some Ideas for Determinacy Analysis Realisation. In: Proceedings of the 11th IASTED International Conference on Artificial Intelligence and Soft Computing, pp. 185–190. ACTA Press (2007)
4. Chesnokov, S.V.: Determinacy Analysis of Social-Economic Data. Sociological Studies 3, 179–189 (1980) (in Russian)
5. Luelsdorff, P.A., Chesnokov, S.V.: Determinacy Form as the Essence of Language. Prague Linguistic Circle Papers 2, 205–234 (1996)
6. Chesnokov, S.V.: Determinacy Analysis and the Search for Diagnostic Criteria in Medicine (the Case of Comprehensive Ultrasonography). Ultrasonic Diagnostics 4, 42–47 (1996) (in Russian)
7. Main scientific works of S.V. Chesnokov (in Russian), `http://www.context.ru/publications`
8. DA-system 4.0, version 4.0 for Windows 95, Windows 98 and Windows NT. Questions and Answers. DA-system and Technology of Data Analysis. Context (1999) (in Russian)
9. Chesnokov, S.V.: Determinacy Analysis of Socio-Economic Data. Illustrative Materials to Lectures. Lecture 2: Rules. Lecture 3: Systems of Rules. Lomonosov Moscow State University, Faculty of Economics, Moscow (2002) (unpublished, in Russian)
10. Quinlan, J.R.: Learning Efficient Classification Procedures and Their Application to Chess and Games. In: Carbonell, J.G., Michalski, R.S., Mitchell, T.M. (eds.) Machine Learning. An Artificial Intelligence Approach, pp. 463–482. Springer, Heidelberg (1984)

A New Computational Framework for Gene Expression Clustering

Shahreen Kasim[1,3], Safaai Deris[2], and Razib M. Othman[3]

[1] Department of Information System, Faculty of Information Technology and Multimedia,
Universiti Tun Hussein Onn Malaysia, 86400 Batu Pahat, Malaysia
[2] Laboratory of Computational Intelligence and Biotechnology, Universiti Teknologi
Malaysia, 81310 UTM Skudai, Malaysia
[3] Artificial Intelligence and Bioinformatics Research Group, Faculty of Computer Science
and Information Systems
Universiti Teknologi Malaysia, 81310 UTM Skudai, Malaysia
shahreen@uthm.edu.my, safaai@utm.my, razib@utm.my

Abstract. Clustering of gene expression is a useful exploratory technique for gene expression dataset as it groups similar objects together and identify potentially meaningful relationships between the objects. However, there are several issues arise for instance data intensive and redundancy in the cluster. Therefore, the new computational framework is needed in order to handle these issues. The results showed that the proposed computational framework achieved better results compared with other methods.

Keywords: gene expression, gene function prediction, biological knowledge, and functional annotations.

1 Introduction

Gene expression is the process that takes information from every structure contained in genes of Deoxyribonucleic Acid (DNA) and turns that information into proteins. A set of genes is co-expressed frequently and this implies that the genes share a biological function and are under common regulatory control [1]. Genes are also identified as co-expressed with different groups of genes, each governed by a distinct regulatory mechanism, in response to the varying demands of the cell. The yeast *S. cerevisiae* is used in this study by reason of the availability of nutrients and the conditions of growth vary constantly. Due to the connection between gene expression regulation and gene product function, computational analysis of gene expression data is used extensively to identify groups of similarly expressed genes. However, most of the commonly used algorithms are unable to identify genes whose expression is similar to multiple and distinct gene groups. This raises issues and brings challenges to the analysis of expression data. The first challenge needed to be tackle in this paper is to handle the changes of the messenger of Ribonucleic acid (mRNA) during experiments conducted on the gene expression dataset. The results from the experiments show that positive numbers represent an increase in expression, while negative numbers represent decreases in

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 603–610, 2010.

expression. This situation produces extensive data which leads to the computational challenge. The increase in intensive data also depends on the variety types of experiments conducted by the researcher in which the volume of the dataset might reach up to approximately 50MB. Another challenge is the nature of gene. Biologically, one gene can belong to multiple functions and multiple traits [2,3]. This introduces the challenge of data redundancy. These challenges are due to the ambiguity of gene expression datasets which comes from the *ab initio* process, thus producing inaccurate results. The existing algorithms, for example the *k*-means algorithm [4], has by necessity weakness when genes can belong to only one cluster in which they contradict the nature of gene biologically. Also, the drawbacks of Self-Organizing Map (SOM) algorithm [5] is that it is not always effective since genes with the main interesting patterns may be merged into only one or two clusters and so cannot be identified. Then, we look into the drawbacks of hierarchical algorithm, which are high in computational intricacy, lack robustness, are vague with respect to termination criteria, and fail to function accurately with a large number of genes as the datasets grow in complexity. Meanwhile the Principal Component Analysis (PCA) algorithm [6] is useful only when features are highly correlated or redundant and the biclustering algorithm generates highly redundant biclusters. Although these clustering algorithms are capable of clustering gene expression datasets, the ambiguity of the datasets is not handled efficiently.

Therefore, based on its ability to handle data ambiguity in the gene expression clustering, fuzzy *c*-means (FCM) algorithm is chosen. This is because of its capability to facilitate the identification of overlapping groups of objects by allowing the objects to belong to more than one group [7]. This algorithm was developed by Dunn, 1973 [8] while improved by Bezdek, 1981 [4] and is very similar to the *k*-means algorithm [4]. The essential difference between FCM clustering and standard *k*-means clustering is in the partitioning of genes into each group. Rather than the hard partitioning of standard *k*-means clustering, where genes belong to only a single cluster, FCM clustering considers each gene to be a member of every cluster, with a variable degree of memberships. Each gene has a total membership of 1.0, which is apportioned to clusters on the basis of the similarity between the gene's expression pattern and that of each cluster centroid. Genes whose expression patterns are very similar to a given centroid will be assigned a high membership in that cluster, whereas genes that bear little similarity to the centroid will be assigned a low membership. Importantly, genes can be assigned significant memberships to more than one cluster, thus revealing genes whose expressions are similar to multiple, distinct groups of genes. In the next section, we show the investigations by FCM that produce non-exclusive clusters while handling the fuzziness of the datasets in order to solve their intensity and redundancy.

2 A New Computational Framework

The proposed computational framework consists of five phases; the requirement analysis and preparation of the data (Phase 1), the basic FCM for gene expression clustering (Phase 2), an improved fuzzy *c*-means with biological knowledge integration (Phase 3), an extended fuzzy *c*-means with multi stage filtering (Phase 4), and the analysis of the results (Phase 5). In Phase 1, data sources (Gene Ontology (GO) datasets, functional annotation databases, and testing datasets) were pre-processed and analyzed while the FCM algorithm is developed in Phase 2. Next in Phase 3, an improved fuzzy *c*-means algorithm is developed that is able to deal with insufficient knowledge by incorporating GO datasets and multiple functional annotation databases in the fuzzy *c*-means algorithm. Then in Phase 4, the results achieved in the Phase 3 are enhanced which can handle the inaccuracies by conducting the filtering stages and applying the enhanced *apriori* algorithm. Lastly, the whole framework is analyzed by computational evaluation and biological validation. In this paper, we present the Phase 2 algorithm in order to justify the choice of algorithm which supports the whole new computational framework.

2.1 The FCM: Fuzzy *C*-Means Clustering Algorithm

2.1.1 Number of Cluster Initialization

This algorithm is initiated by partitioning the genes into l group of clusters. The value of l needs to be pre-determined before the algorithm is performed. In this algorithm, the number of l is determined randomly and needs to be defined by the user. In contrast with hard clustering, FCM appears to be less sensitive to over-fitting due to the genes that are not forced to belong to only a single cluster. Therefore, in order to achieve a fair comparison with the algorithm that will be investigated in Phase 3 and Phase 4, we set 76 as the number of clusters.

2.1.2 Fuzzy Membership Initialization

Once the l has been determined, the algorithm begins its fuzzy membership initialization. Let $X = \{x_i : i = 1, \dots, g_n\}$ where x_i is a vector expression value for gene g and $C = [c_j]$. The objective function is minimized in FCM [4] as given below:

$$J\left(X, C\right) = \min \sum_{j=1}^{l} \sum_{i=1}^{n} \left(u_{ij}\right)^m d(x_i, c_j) \ , \tag{1}$$

where $d(x_i, c_j)$ is a Euclidean distance function from expression value of gene x_i to centroid c_j as follows:

$$d(x_i, c_j) = \| x_i - c_j \| = \sqrt{\sum_{j=1}^{l} (x_{ij} - c_{ij})^2} \ , \tag{2}$$

Accordingly, l is the number of clusters while n is the number of genes in the dataset and m is the fuzzy parameter. Concurrently u_{ij} is the membership value of gene x_i in cluster cl in which it is subject to $u_{ij} \in [0,1]$ and $\sum_{cl=1}^{cl} u_{ij} = 1, \forall i$ constraints, which means that the sum of the membership values of the objects to all of the fuzzy clusters must be one. This algorithm takes the values of membership values, u_{ij} which lie between 0 and 1. For a given gene, an index close to 1 indicates a strong association to the cluster. Inversely, indexes close to 0 indicate the absence of a strong association to the corresponding cluster.

2.1.3 Centroid Calculation

Next, the membership vector values u_{ij} and cluster centroids c_j can be obtained after minimization of the Equation (1) and (2), as stated by Bezdek, 1981 [4]. In order to computes the centroid of the clusters this definition is performed:

$$c_j = \frac{\sum_{i=1}^{n} \left(u_{ij}^{m}\right) x_i}{\sum_{i=1}^{n} \left(u_{ij}^{m}\right)} . \tag{3}$$

2.1.4 Fuzzy Membership Update

At this stage, the membership values, u_{ij} computes the membership of object i to cluster cl. The membership update function is defined as follows:

$$u_{ij} = \frac{1}{\sum_{j=1}^{cl} \left(d(c_j, x_i) / (c_j, x_j)\right)^{\frac{1}{m-1}}} , \tag{4}$$

This step continues and repeats itself from stage 2.1.3 until the stopping criterion is met.

3 Results and Discussion

3.1 The Multiple Functions of the Gene

As mentioned above, the FCM algorithm is able to solve the redundancy and data intensive issues. These bring an essential difference between FCM clustering and standard *k*-means clustering, although FCM has similarity with *k*-means algorithm. Rather than the hard partitioning of standard *k*-means clustering, where genes belong to only one single cluster, FCM clustering considers each gene to be a member of every cluster, with a variable degree of membership. Each gene has a total membership of 1.0, which is apportioned to clusters on the basis of the similarity between the gene's expression pattern and that of each cluster centroid. Genes whose expression patterns are very similar to a given centroid are assigned a high membership in that cluster, whereas genes that bear little similarity to the centroid have a low membership. Significant, genes can be assigned membership to more than one cluster, thus revealing genes whose expression is similar to multiple, distinct groups of genes. In Table 1, we see that there are many genes having a significant membership in more than one of the fuzzy clusters. When the genes are assigned to all clusters with an empirically defined membership cutoff of 0.06, 35% of the assigned genes are able to be placed in more than one group (see column number three). At a slightly lower cutoff of 0.04, 64% of all of the assigned genes were able to be placed into multiple clusters. This procedure has also been conducted on Gasch [9] dataset, as shown in Table 2. It can be seen that the smaller the membership cutoff value, the more genes would be assigned to the clusters.

Table 1. Fuzzy assignment of genes to clusters from Eisen dataset

Membership Cutoff	Number of Assigned Genes	Number of Assigned Genes to > 1 cluster
0.10	1,341 (22%)	230 (17%)
0.08	1,843 (30%)	334 (18%)
0.06	2,631 (43%)	913 (35%)
0.04	4,233 (69%)	2,719 (64%)
0.02	4,785 (72%)	3,460 (72%)

Table 2. Fuzzy assignment of genes to clusters from Gasch dataset

Membership Cutoff	Number of Assigned Genes	Number of Assigned Genes to > 1 cluster
0.10	1,365 (22%)	227 (17%)
0.08	1,859 (30%)	353 (19%)
0.06	2,782 (42%)	935 (34%)
0.04	4,323 (69%)	2,725 (63%)
0.02	5,276 (86%)	3,214 (72%)

3.2 Comparison with Other Methods

We have compared the performance of FCM with several algorithms in gene-based (*k*-means, SOM, hierarchical), sample-based (PCA), and subspace method (biclustering) algorithms using Eisen [1] and Gasch [9] datasets. Using the implementation of CLUSTER software [10], we configured SOM, *k*-means, hierarchical, and PCA algorithms by using the default setting. Meanwhile, the implementations used in our experiments for the biclustering algorithm were obtained from Biclustering Analysis Toolbox (BicAT) [11]. The default setting for the maximum number of iterations for *k*-means, SOM, and hierarchical are 100, 100,000, and 20 respectively. To achieve a fair comparison, we set the number of clusters cl at 76 for all of those algorithms, which is the same number of clusters generated by the algorithm in Phase 3 and Phase 4. We also used the default setting 0.06 for the membership cutoff value for biclustering, which is the same value reported in Barkow *et al.,* 2006 [11]. Further, in Table 3 and Table 4, we show the comparison of the FCM with *k*-means and the biclustering algorithm for the *z*-score and the processing time in 10, 30, 50, 76, and 100 clusters for both datasets. We have not compared them with SOM, hierarchical, and PCA algorithms due to the inapplicability of the number of clusters setting in the software. Based in this table, we can say that our chosen number of clusters is appropriate and acceptable. Therefore, a comparison of the *z*-scores with other algorithms conforms the suitability of the chosen algorithm to the gene annotation data in which the gene annotation data will be used in the proposed computational framework, while requiring a reasonable amount of running time.

Table 3. The comparison of *z*-score with several methods in different clusters for Eisen dataset. The values in brackets '[]' indicate processing time (seconds).

No. of Clusters / Algorithm			**z-score**		
	10	**30**	**50**	**76**	**100**
FCM	[62] 62.22	[65] 63.90	[90] 67.97	[165] 68.17	[168] 68.25
k-means	[251] 41.8	[795] 34.8	[1,150] 32.1	[1,540] 30.5	[1,841] 23.3
Biclustering	[10,200] 51.22	[10,243] 52.97	[103,356] 52.76	[119,761] 52.53	[120,255] 52.61

Table 4. The comparison of *z*-score with several methods in different clusters for Gasch dataset. The values in brackets '[]' indicate processing time (seconds).

No. of Clusters / Algorithm			***z*-score**		
	10	**30**	**50**	**76**	**100**
FCM	[298] 80.27	[401] 80.19	[430] 81.97	[449] 83.45	[455] 83.65
***k*-means**	[768] 93.7	[2,400] 70.2	[2,498] 61.3	[3,969] 56.2	[4,839] 49.7
Biclustering	[110,200] 52.11	[120,943] 52.17	[120,796] 52.76	[131,135] 55.34	[143,255] 56.44

4 Summary

DNA microarray has been widely used in biological studies. When analyzing gene expression data, a clustering method is often used. Although the goal of clustering is to subdivide a set of genes in such a way that functionally similar genes fall into the same cluster, the actual results merely reflect the similar abundance of gene products (mRNA). However, the challenge of changes in expression values in the gene expression datasets will cause data intensiveness, while the gene's multiple functions will bring out the issues of data redundancy. In this paper, we have investigated FCM clustering algorithm in order to handle data ambiguity in the gene expression datasets. This investigation has proved the choice of the algorithm throughout the whole of the computational framework. The algorithm consists of the initialization of fuzzy centroids with genes with the use of a Euclidean correlation distance in its objective function. The *z*-score evaluation measurement employed here are useful in quantifying the results of the chosen algorithm in its grouping of genes with similar biological functions. Our initial evaluation of the algorithm on a yeast datasets revealed promising results: the algorithm produced more functionally significant clusters, and assigned more genes to functional groups. It can be observed that the algorithm is capable of solving the intensiveness of the data based on to the changes in datasets. Furthermore, the algorithm is also able to pay determine the data redundancy due to the multiple functions of gene. Therefore, the results in this phase show that genes have been assigned to their clusters with better results when compared to other algorithms. However, issues remain, as insufficient knowledge for the solution of the data quality remains. This issue can be resolved by incorporating GO and multiple functional annotation databases into the FCM algorithm (Phase 3 and Phase 4).

References

1. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster Analysis and Display of Genome-wide Expression Patterns. In: Proceedings of the National Academy of Sciences of the United States of America, pp. 14863–14868. The National Academy of Sciences, Washington (1998)
2. Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., Dzeroski, S.: Predicting Gene Function Using Hierarchical Multi-Label Decision Tree Ensembles. Bioinformatics 11(2) (2010)
3. Lobo, I.: Pleiotropy: One Gene Can Affect Multiple Traits. Nature Education 1(1) (2008)
4. Bezdek, J.C.: Pattern Recognition With Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
5. Tamayo, P., Solni, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., Golub, T.R.: Interpreting Patterns Of Gene Expression With Self-Organizing Maps: Methods And Application To Hematopoietic Differentiation. In: Proceedings of The National Academy of Sciences of The United States of America, pp. 2907–2912. The National Academy of Sciences, Washington (1999)
6. Alter, O., Brown, P.O., Bostein, D.: Singular Value Decomposition For Genome-Wide Expression Data Processing And Modeling. In: Proceedings Of The National Academy Of Sciences Of The United States Of America, pp. 10101–10106. The National Academy Of Sciences, Washington (2000)
7. Bezdek, J.C.: Fuzzy Mathematic. In: Pattern Classification. Cornell University, Ithaca (1973)
8. Dunn, J.C.: A Fuzzy Relative Of The ISODATA Process And Its Use In Detecting Compact Well-Separated Clusters. Journal of Cybernetics 3, 32–57 (1973)

9. Gasch, A.P., Spellman, P.T., Kao, C.M., Carmel-Harel, O., Eisen, M.B., Storz, G., Botstein, D., Brown, P.O.: Genomic Expression Programs In The Response Of Yeast Cells To Environmental Changes. Molecular Biology of the Cell 11(12), 4241–4257 (2000)
10. CLUSTER Software, `http://rana.lbl.gov/EisenSoftware.htm`
11. Barkow, S., Bleuler, S., Prelic, A., Zimmermann, P., Zitzler, E.: Bicat: A Biclustering Analysis Toolbox. Bioinformatics 22(10), 1282–1283 (2006)
12. Gibbons, F., Roth, F.: Judging The Quality Of Gene Expression-Based Clustering Methods Using Gene Annotation. Genome Research 12, 1574–1581 (2002)

Forecasting Short-Term Trends of Stock Markets Based on Fuzzy Frequent Pattern Tree

Defu Zhang, Bo Wu, Xian Hua, and Yangbin Yang

Department of Computer Science, Xiamen University,
361005 Xiamen, China
dfzhang@xmu.edu.cn, destinyfog@gmail.com

Abstract. Stock forecasting is a non-linear financial time series forecasting problem. Stock index contains tremendous noise and is affected by numerous factors. Fuzzy time series takes advantage of such problems. In this paper, a novel model based on the fuzzy frequent pattern tree (FFPT) is proposed to forecast short-term trends of stock markets. Fuzzy frequent pattern tree is a combination of fuzzy set theory and frequent pattern tree. Frequent pattern tree is a highly compressed data structure store the information of association rules to be mined. In this paper, an FFPT is built using fuzzy stock time series. Then we forecast short-term trends by a new method called FFPT-Search. And stock data from several famous stock markets is picked up to evaluate the effectiveness of our model. Computational results indicate it works well.

Keywords: stock forecasting, fuzzy frequent pattern tree, fuzzy time series, association rules, data mining.

1 Introduction

Investors have trying to find a way to make more profit from the stock market. Accurate short-term trends forecasting can give them great help. In this paper, FFPT model which is a combination of fuzzy time series and frequent pattern tree is proposed to forecast the stock market.

Fuzzy time series was first introduced by Song and Chissom [1]. It was developed to handle problems involving linguistic terms. Researches [2,3] indicate that it handles problems with noise better than other models. In this paper, stock price rates of change are converted into fuzzy stock time series.

Association rules learning is a popular method which is first proposed by Agrawal [4] for discovering interest relations in large databases. And Han [5] proposed an extended prefix-tree named frequent pattern tree (FP-tree) to store the database in a compressed form. In this paper, we build an FFPT by fuzzy stock time series under the constraint of minimum support. The FFPT contains all useful frequent patterns. At last, short-term trends of stock are forecasted by finding the match with highest confidence in the FFPT.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 611–617, 2010.

Remaining parts of this paper are organized as follows. Basic concepts on fuzzy time series, association rules and frequent pattern tree are presented in Section 2. And preprocess of stock data is presented in Section 3. Section 4 describes details of proposed model. Experimental results and related analysis are discussed in Section 5. Section 6 concludes the paper.

2 Related Concepts

2.1 Fuzzy Set Theory

Fuzzy set theory was originally developed to handle ambiguity or uncertainty problems involving linguistic terms [6]. Fuzzy time series is an extensive application of fuzzy set theory on time series. It was first proposed by Song and Chissom [1,7] to forecast university enrollments. And fuzzy rules are useful rules generated from fuzzy datasets through different methods [8,9]. And some related definitions are given as follows:

Definition 1. Fuzzy set. A fuzzy set A is a group of linguistic terms, which are characterized by a real membership function $f_A(t)$ in the interval [0, 1], representing the grade of membership of t in A .

Definition 2. Fuzzy rule. A fuzzy rule is the relationship between $F(t)$ and $F(t-1)$ which is denoted by the form $F(t-1) \rightarrow F(t)$.

Definition 3. High order fuzzy rule. If $F(t)$ is only caused by $F(t-1)$, it's called one order fuzzy rule. If $F(t)$ is caused by $F(t-1)$, $F(t-2)$, … , $F(t-m)$ (m > 1), it's called m-order fuzzy rule denoted by the form of { $F(t-1)$, $F(t-2)$, …, $F(t-m)$ }$\rightarrow F(t)$.

2.2 Mining Algorithms for Association Rules

It is useful to extract knowledge from large databases and represent it in a comprehensible form. Several mining approaches have been proposed to find association rules from transaction data. For example, Agrawal [4] proposed the Apriori algorithm which generates and tests candidate frequent itemsets level by level in 1993. It has to store and do a considerable amount of work on dealing with a large number of candidate frequent itemsets. In [5], Han et al. introduced the FP-growth method. In this method, a data structure called FP-tree is applied to store frequency information in a compressed form of the original database. It only scans the original database twice and no candidate generation is required. And some related definitions are given as follows:

Definition 4. Attribute. An item i_j is an attribute. An element in fuzzy stock time series is an attribute.

Definition 5. Itemset. Let $I = \{ i_1, i_2, \ldots, i_n \}$ be a set of n attributes called an itemset. Sequence of fuzzy stock time series is an itemset.

Definition 6. Support. Support of an itemset I is the proportion of transactions in the database which contains I .

Definition 7. Frequent itemset. An itemset which satisfy a user-specified minimum support is called a frequent itemset.

Definition 8. Association rule. An association rule is an implication of the form $X \Rightarrow Y$, where X and Y are itemsets. Support of the rule $X \Rightarrow Y$ is equal to support of $X \; U \; Y$.

Definition 9. Confidence. Confidence of an association rule is conf($X \Rightarrow Y$) = support ($X \; U \; Y$) / support(X).

3 Preprocess of Stock Index

Prices of stock index are real values. Most papers convert stock index into fuzzy time series directly. But in order to forecast short-term trends more accurately, price rate of change should be chosen as it better reflects the fluctuations. Price rate of change is a calculation of current price divided by price change some days before. A stock has in an uptrend if price rate of change is positive and downtrend if negative. For stock prices $s_k \; (k \in [1..n])$, price rate of change p_k can be calculated by formula (1).

$$p_k = (s_{k+1} - s_k) \, / \, s_k \ . \tag{1}$$

And then price rates of change are converted into four fuzzy attributes $\{ D_1, D_2, U_1, U_2 \}$ formula (2).

$$f_k = \begin{cases} U2 & \text{if } p_k \text{ larger than half of rising price rates of change} \\ U1 & \text{if } p_k > 0 \text{ and does not satisfy U2's condition} \\ D2 & \text{if } p_k \text{ smaller than half of falling price rates of change} \\ D1 & \text{if } p_k < 0 \text{ and does not satisfy } D2\text{'s condition} \end{cases} \tag{2}$$

From formula (2), fuzzy attribute has its own linguistic meaning. U_1 means the stock has been in a small uptrend while U_2 means a big uptrend. And for D_1 and D_2 , vice versa. An example is applied for illustration by Table 1. Stock prices in Table 1 come from Shanghai stock market.

Table 1. Convert stock prices into fuzzy attributes

Stock prices	Price rates of change	Fuzzy attributes
1390.461	0.00387	U_1
1395.848	0.00353	U_1
1400.769	-0.01128	D_2
1384.964	-0.01211	D_2
1368.196	0.00019	U_1
1368.45	-0.01019	D_1
1354.511	0.00420	U_2
1360.2	-0.00146	D_1
1358.216	-0.01308	D_2
1340.45	0.00096	U_1
1341.738	-0.01217	D_2
1325.413	0.01450	U_2
1344.626	-0.00360	D_1
1339.784	-0.00400	D_1
1334.422	-0.01035	D_2
1320.614	-0.00091	D_1
1319.407	0.01717	U_2
1342.062	-0.01535	D_2
1321.462	0.00489	U_2
1327.927		

4 The Proposed Fuzzy Frequent Pattern Tree (FFPT) Model

4.1 Building a Frequent Pattern Tree with Training Fuzzy Attributes

In the FFPT, each node has three fields: item-name, count, node-link. Data structure of the node is shown in Fig.1.

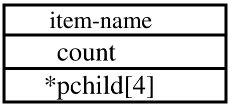


Fig. 1. Data structure of the node

Element count refers to the frequency of appearances for sequence of item-names from tree's root to current node in fuzzy stock time series. And element pchild[4] are pointers which link to its children. FFPT is built with frequent fuzzy stock sequences by algorithm 1 under the constraint of minimum support.

Algorithm 1. BuildFFPT

```
 BuildFFPT(Tree T)
   for each fuzzy frequent sequence $f[i..j]$
          seqcount←number of $f[i..j]$ occurrences in fuzzy
 stock time series;
          match $f[i..j]$ with T top down
          increase count of current node by seqcount;
```

To illustrate Algorithm 1, an FFPT is built by the data of Table 1 under the constraint of minimum support to be 10%. Frequent sequences generated from Table 1 are $\{\{[U_1]{:}4\}$, $\{[U_2]{:}4\}$, $\{[U_2, D_1]{:}2\}$, $\{[D_1]{:}5\}$, $\{[D_1, D_2]{:}2\}$, $\{[D_1, U_2]{:}2\}$, $\{[D_2]{:}6\}$, $\{[D_2, U_1]{:}2\}$, $\{[D_2, U_2]{:}2\}\}$. The number after the colon is count of occurrences of sequence in fuzzy stock time series. An example of FFPT built with data in Table 1 is given as Fig.2.

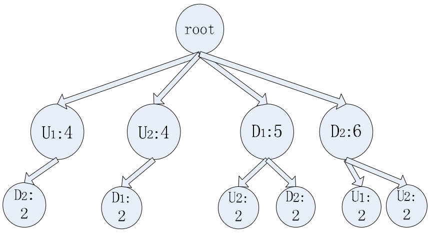


Fig. 2. Tree built with data of Table 1

From Fig.2, count of occurrences of a sequence in fuzzy stock time series can be acquired by matching item-names of nodes from tree's root top down. For example, sequence $[D_2]$ appears 6 times in fuzzy stock time series. So in T's second level, count of the node with item-name D_2 is 6. And sequence $[U_1, D_2]$ appears 2 times in fuzzy stock time series. So in T's third level, count of the node whose item-name is D_2 and parent's item-name is U_1 is 2.

4.2 Forecasting Short-Term Trends by FFPT-Search Method

Let notation *Height* means height of the frequent pattern tree. Trend of jth day is forecasted by FFPT-Search method. It searches the tree to find the match with past fuzzy stock time series seq[0..j-1] with highest confidence top down.

Algorithm 2. FFPT-Search

```
FFPT-Search(int j, fuzzy stock time series seq)
  maxconf←0;
  trend←NULL;
  for i← 1 to $Height$ - 1 do
```

```
      if T matches sequence seq[j-i..j-1] top down
      /*Assume that it stays at node n after matching
        and $D_1$ , $D_2$ ,  $U_1$ , $U_2$ are the children of node n */
              downcount ← $D_1.count + D_2.count$ ;
              upcount ← $U_1.count + U_2.count$ ;
              conf←max(upcount,downcount) / $n.count$ *100%;
              if conf > maxconf then
                     maxconf ←conf;
              if upcount > downcount and maxconf
                  satisfies minimum confidence then
                     trend←up;
              else if maxconf satisfies minimum
                        confidence then
                     trend←down;
 return trend;
```

5 Experimental Results

5.1 Performance of Proposed Model

Historical stock prices including Standard & Poor's, NASDAQ, New York Composite, Shanghai Index and Shenzhen Index are picked up to check our model. Stock data in this paper can be downloaded from http://finance.yahoo.com/ and http://finance.cn.yahoo.com.

For Standard & Poor's , NASDAQ and New York , 1000 days' stock prices from Dec. 9th, 2004 to Nov.26th, 2008 are selected as training data and 300 days' stock prices from Nov.28th, 2008 to Feb.8th, 2010 are selected as test data. For Shanghai Stock Index and Shenzhen Stock Index, 1000 days' stock prices from Aug.4th, 2004 to Sep.12th, 2008 are selected as training data and 300 days' stock prices from Sep.16th, 2008 to Dec.9th, 2009 are selected as test data. Experimental results are shown in Table 2 and Table 3.

Table 2. Experimental results for different stock indexes

Stock name	Hit rate
SP 500	0.609
Nasdaq	0.667
NY	0.606
SH China	0.571
SZ China	0.565

5.2 Compare with Other Models

From Table 2, the average hit rate of proposed model is 60.36%. And comparison of our model with most famous other models is shown in Table 3. Hit rate data of those models comes from Atsalakis's paper [10].

Table 3. Comparing with other models

Author	Model	Average hit rate
Lin et al.(2002)	NF	58.03
Fernandez et al. (2000)	ANN	58.00
Harvey et al. (2000)	NN	59.00
Perez-Cruz et al. (2003)	MLP	57.00
Doesken et al. (2005)	TS-FIS	56.00

6 Conclusions and Future Work

In this paper, a novel model based on FFPT and FFPT-Search is proposed to forecast short-term trends of stock markets. The model combines fuzzy set and association rules mining technique. And experimental results show that it perform well in forecasting several world famous stock markets.

Our future work is to put the model into practice.

References

1. Song, Q., Chissom, B.S.: Forecasting enrollments with fuzzy time series. In: Part I, Fuzzy Sets and Systems, pp. 1–9 (1993)
2. Yungho, L., Chien Pang, L., Jou, Y.Z.: A distance-based fuzzy time series model for exchange rates forecasting. In: Expert Systems with Applications, pp. 8107–8114 (2009)
3. Hsing-Hui, C., Tai-Liang, C., Ching-Hsue, C., Chen-Chi, H.: Fuzzy dual-factor time-series for stock index forecasting. In: Expert Systems with Applications, pp. 165–171 (2009)
4. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: An Introduction to Signal Detection and Estimation, pp. 207–216 (1993)
5. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate Generation. In: Proceeding of ACM SIGMOD International Conference Management of Data, pp. 1–12 (2000)
6. Zadeh, L.A.: Fuzzy Sets. Information and control, 338–353 (1965)
7. Nai-Yi, W., Shyi-Ming, C.: Temperature prediction and TAIFEX forecasting based on automatic clustering techniques and two-factors high-order fuzzy time series. In: Expert Systems with Applications, pp, pp. 2143–2154 (2009)
8. Chen, S.M., Lee, S.H.: A new method for generating fuzzy rules from numerical data for handling classification problems. In: Applied Artificial Intelligence, pp. 645–664 (2001)
9. Chen, S.M., Lee, S.H.: Neuro-Fuzzy and Soft-Computing: A Computational Approach to Learning and Machine Intelligence. In: Applied Artificial Intelligence, pp. 645–664 (2001)
10. Atsalakis, G.S.: Forecasting stock market short-term trends using a neuro-fuzzy based methodology. Expert Systems with Applications 36, 10696–10707 (2009)

Inspired Rule-Based User Identification

Peng Yang and Yan Zheng

School of Computer Science,
Beijing University of Posts and Telecommunications, 100876 Beijing
samerry@163.com, yanzheng@bupt.edu.cn

Abstract. Web log mining is an important branch of Web usage mining. It is a kind of specific application of data mining. It can find Web user mode of behavior through processing server-side log files, and further improve the structure of the website or provide users with personalized service. Data preprocessing is an important step of Web log mining. In general, data preprocessing is divided into four steps: data cleaning, user identification, session identification, path supplement, transaction identification. This paper proposes a user identification method which is based on inspired rules. Experiment results prove the effectiveness of this method.

Keywords: Web Log Mining, Preprocessing, User Identification.

1 Introduction

With rapid development of Internet technology, more and more network resource exists. Therefore, there is an urgent need for a new technology which can quickly and accurately discover useful knowledge to be used for people from these resources. Web mining is an effective way. Web mining is the data mining techniques to Web data, in order to find interesting patterns and implicit information. As an important area of Web mining, Web log mining is a process of obtaining useful information from the large number of users to access records which were recorded on the server, and such information can be used to improve the site's structure or to provide users with personalized service.

Data mining on the data is handled by the strict requirements. Build the appropriate data set is an important step in the process of data mining, so the data preprocessing before data mining is more important. According to statistics, two thirds of the data mining analysts believe that a complete pre-processing takes up the whole data mining about 60% of the time.

Web mining is generally divided into three categories: Web Structure Mining, Web Content Mining, Web Usage Mining. As to the Web Log Mining, the user identification is a more crucial aspect. User identification is a step to identify the corresponding user from the log records of each visit, it directly affects the accuracy of the next steps in data preprocessing.

L. Cao, J. Zhong, and Y. Feng (Eds.): ADMA 2010, Part I, LNCS 6440, pp. 618–624, 2010.

2 Web Log Mining Data Preprocessing Process

Web log mining is generally divided into the following steps: data cleaning, user identification, session identification, path supplement, transaction identification. The steps are showed as Fig. 1:

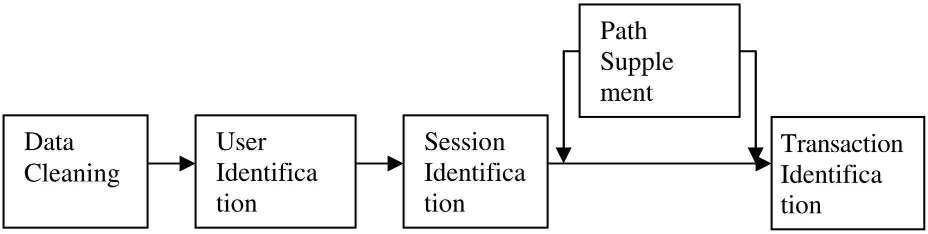


Fig. 1. Data pre-processing steps

Data Cleaning is the primary task of data preprocessing, which aims to remove the Web log of a record which does not reflect the user behavior. In general, it should be remove the records which is unnecessary to our mining purpose and it also should be delete the unnecessary domains.

Next, the user must be identified, the user is to access the site through a browser individuals. User identification is a process of identifying each user from the records which are obtained from the cleaning process. As the local cache proxy server and firewall exists, it makes the task of accurately identify each user more difficult.

To an identified user, all of his access sequence may go beyond a very long time, and it indicates that the user may visited the site more than once in this time period. The session identification aims at identifying all of the user's access sequence into a number of individual user access sequences. A simple way is to set the time threshold. In general, when a user visits two difference adjacent pages between more than 30 minutes, it will open a new session.

The purpose of the Path Supplement is to determine the reference page of current access page. If the user's access history have more than one page which contains a reference to the current page link, it will recognizes the page which was accessed more close in time to the current page accessed time as reference page, and the related information will be added to the user's session file.

The Transaction Identification is to identify a user session on semantic groups, after the Transaction Identification on behalf of the user access information, the sequence is more meaningful.

3 A User Identification Method Based on Inspired Rules

3.1 Inspired Rule-Based User Identification

Many methods can identify the user, such as Cookies-based and using embedded user id, client-side software agent based and registered for use, etc. But the reality is that the users usually close Cookies for security reasons or are unwilling to give real privacy information when they make a registration. Generally, it is only to analyze log file information can identify different users.

There are many main information will be used to identify users, such as User IP, Agent, etc, but if we only use one aspect, it can not give use a accurate result, so we use all this information as constraints.

In addition, due to the instability of the network itself, many browsers have the ability of automatically re-connected features when connection dropped, so there are many redundant log information in the access records, we should delete them.

Considering the reasons above, we propose a user identify method based on inspired rules. We use four constraints (time information, IP address, agent information and reference page) to identify a user. It is agreed that the different IP indicates different user, and different agent information also shows different user. In the same IP address, and agent information is also exactly the same circumstances, it uses site topology to identify the user: If the current access page can not be reached from the all exists access sequence, it identifies the user as a new user. When the current visiting page's reference page is empty, it uses time information and accessing page information to identify users, when a new page is accessed, it identifies the user as a new user. When the same page is accessed, and the difference between the last access time less than 10 seconds, it identifies as off-line auto reconnection, and removes the record.

The method is described as follows:

```
input：WebLog(Web log file)
output：UList(users set)
for(R belongs WebLog){
 if （R.ip != all IPs in UList）
    new  R.user；
    add R.user to UList;
 else if(R.agent!=UList all user.R.agent &&
          user.R.ip==R.ip)
     new  R.user；
      add R.user to UList;
    else if （R.refer == ″-″）
    if(R.uri!=UList all user.R.uri &&user.R.ip==R.ip
      &&user.R.agent==R.agent)
      new  R.user；
      add R.user to UList;
    else if （R.time-user.R.time>10seconds）
      new  R.user；
      add R.user to UList;
```

```
        else
            delete
  }
```

Method symbol Description

R is a record of the log file; R.user is the user of R record; R.ip is the client ip of R.record; R.time is the time of user access the cite; R.uri is the resource that user accessed; R.refer is the reference page; R.agent is the client browser and operating system information.

The method identifies users by the multiple constraints, so computational efficiency is not high, but it can accurately identify new users, considering non-real time application, the algorithm is feasible.

3.2 Experiments

The following is the column of the log file which used in the user identification method, as following Table 1:

Table 1. Part of the log file

Date	Time	IP	Agent	URI	Reference
2010-04-21	00:00:25	110.75.166.39	A	/cs_web/research/kyhj.html	-
2010-04-21	00:03:07	219.220.30.239	B	/cs_web/	-
2010-04-21	00:05:44	211.68.71.60	C	/cs_web/	-
2010-04-21	00:06:09	211.68.71.60	D	/cs_web/recruit/zhaosheng.html	-
2010-04-21	00:06:09	211.68.71.60	D	/cs_web/recruit/lxwm.html	http://scs.bupt.edu.cn/cs_web/recruit/zhaosheng.html
2010-04-21	00:06:09	211.68.71.60	E	/cs_web/	-
2010-04-21	00:06:10	211.68.71.60	D	/cs_web/index.aspx	http://scs.bupt.edu.cn/cs_web/recruit/zhaosheng.html
2010-04-21	00:11:31	211.68.71.60	F	/cs_web/	http://www.bupt.edu.cn/pages1/out/index.asp
2010-04-21	00:11:35	211.68.71.60	F	/cs_web/	-
2010-04-21	00:11:40	211.68.71.60	F	/cs_web/	-
2010-04-21	00:11:41	211.68.71.60	F	/cs_web/	-
2010-04-21	00:11:47	118.229.135.247	G	/cs_web/class_web/2007-1/index.php	-
2010-04-21	00:12:07	211.68.71.60	A	/cs_web/center/school_regulation.html	-

Table 1. (*continued*)

2010-04-21	00:12:10	211.68.71.60	A	/cs_web/center/school_regulation.html	-
2010-04-21	00:14:41	211.68.71.60	H	/cs_web/	http://www.bupt.edu.cn/pages1/out/index.asp
2010-04-21	00:14:41	211.68.71.60	H	/cs_web/	-
2010-04-21	00:17:38	211.68.71.60	I	/cs_web/	http://www.bupt.edu.cn/pages1/out/index.asp
2010-04-21	00:17:42	211.68.71.60	J	/cs_web/	-
2010-04-21	00:18:01	211.68.71.60	K	/cs_web/introduce/xxaq center.html	http://scs.bupt.edu.cn/cs_web/
2010-04-21	00:18:35	59.64.136.59	J	/cs_web/	-

Identification method is applied to the user log data in the Table 1, The first 3 lines are identified as new user because of different IP; Although the IP of line 4 and the IP of line 3 is the same, different agent information indicates that they are different user; The line 4,5,7 indicate the same user; Line 6 of table 2 recognized as a new user because of different browsers: Line 8 is also recognized as a new user because of the different browsers; the 9,10,11 line is considered the line 8 represents the user's record for it's browser automatically re-link behavior, so delete them; the results obtained is in the Table 2:

Table 2. Table 1 Recognition results by using new method

UserId	Time	IP	Agent	URI	Reference
1	00:00:25	110.75.166.39	A	/cs_web/research/kyhj.html	-
2	00:03:07	219.220.30.239	B	/cs_web/	-
3	00:05:44	211.68.71.60	C	/cs_web/	-
4	00:06:09	211.68.71.60	D	/cs_web/recruit/zhaosheng.html	-
	00:06:09			/cs_web/recruit/lxwm.html	http://scs.bupt.edu.cn/cs_web/recruit/zhaosheng.html
	00:06:10			/cs_web/index.aspx	http://scs.bupt.edu.cn/cs_web/recruit/zhaosheng.html -
5	00:06:09	211.68.71.60	E	/cs_web/	-
6	00:11:31	211.68.71.60	F	/cs_web/	http://www.bupt.edu.cn/pages1/out/index.asp

Table 2. (*continued*)

7	00:11:47	118.229.135.247	G	/cs_web/class_web/2007-1/index.php	-
8	00:12:07	211.68.71.60	A	/cs_web/center/school_regulation.html	-
9	00:14:41	211.68.71.60	H	/cs_web/	http://www.bupt.edu.cn/pages1/out/index.asp
10	00:17:38	211.68.71.60	I	/cs_web/	http://www.bupt.edu.cn/pages1/out/index.asp
11	00:17:42	211.68.71.60	J	/cs_web/	-
12	00:18:01	211.68.71.60	K	/cs_web/introduce/xxaqcenter.html	http://scs.bupt.edu.cn/cs_web/
13	00:18:35	59.64.136.59	J	/cs_web/	-

3.3 Comparsion

There are some user identification methods also based on inspire rules, but they only use IP information as constraints. If we use it to process data in the Table 1, we can get a result showed in Table 3.

Table 3. Table 2 Recognition results by one old method

UserID	IP
1	110.75.166.39
2	219.220.30.239
3	211.68.71.60
4	118.229.135.247
5	59.64.136.59

From the results showed in the Table 3, we can get a conclusion that the above old method is simple and its efficiency is satisfied,but its accuracy is low. The method only considers the IP information and it agreed that if the log records have the same IP information, they belong one user. It lacks of considering local server technology. We consider the method of multiple constraints, while low efficiency, but accuracy increased significantly.

The method which depends on the cooperation of the user is the best way of user identification. Considering the personal privacy issues, this kind of method is difficult to implement, but our method is based on the log file information, and it is not involved user privacy, so it have higher feasibility.

4 Conclusion

With the rapid development of Internet technology, web resources become more and more plentiful, and people's information demand is increased day by day, Web mining as an important branch of data mining will attract more and more attention. From the perspective of the site, Web log mining can provide good advice for Web site development. From the user's point of view, Web log mining can provide better personalized service. Web log mining has great commercial value and broad application space, so the methods on user identification of Web log mining data preprocessing is worth studying.

Acknowledgments. This research is supported by the Fundamental Research Funds for the Central Universities.

References

1. Luotonen, A.: The common log file format (1995)
2. Qingqing, C.: Research on Data Preprocessing process in the Web Log Mining. In: The 1st International Conformation Science and Engineering, pp. 941–945 (2009)
3. Jiling, L., Jun, F.: Web log mining data preprocessing based on user access Tree. Computer Science 36, 154–156 (2009) (in Chinese)
4. Jiawei, H., Xiaofeng, M., Jing, W., Shengen, L.: Research on Web Mining. Journal of Computer Research & Development 38, 405–414 (2001)
5. Xiaoyu, G.: Web log mining techniques and algorithms (in Chinese). Harbin Engineering University, Master Thesis (2009)
6. Yubin, Z.: The study of Web log mining association algorithm. National University of Defense Technology, Master’s Thesis (2006) (in Chinese)
7. Rong, W.: User identification algorithm of Web log mining (in Chinese). Microcomputer Applications 23, 61–62 (2007)
8. Xuan, L., Zhenquan, Z.: User Identification Algorithm in Preprocessing of Web Usage Mining (in Chinese). Computer Engineering and Applications 7, 173–176 (2002)

Author Index