AusDM05

Proceedings
4th Australasian Data Mining Conference

5 - 6th December, 2005, Sydney, Australia

Edited by
Simeon J. Simoff, Graham J. Williams, John Galloway
and Inna Kolyshkina

Collocated with
the 18th Australian Joint Conference on
Artificial Intelligence AI2005
and
the 2nd Australian Conference on Artificial
Life ACAL05

University of Technology Sydney
2005

Proceedings of the 4th Australasian Data Mining Conference – AusDM05, 5 – 6th, December, 2005, Sydney, Australia, collocated with the 18th Australian Joint Conference on Artificial Intelligence AI2005 and the 2nd Australian Conference on Artificial Life ACAL05.

S. J. Simoff, G. J. Williams, J. Galloway and I. Kolyshkina (eds).

Conference Web Site:
http://www.togaware.com/ausdm05/

Published by the University of Technology Sydney

ISBN 1-86365-716-9

Supported by:

togaware — Web address: www.togaware.com

The e-Markets Research Group — Web address: www.e-markets.org.au

University of Technology Sydney, UTS: Information Technology — Web address: www.uts.edu.au
Web address: www.it.uts.edu.au

Institute of Analytics Professionals of Australia Limited — Web address: www.iapa.org.au

NetMap analytics — Web address: www.netmapanalytics.com

ARC Research Network on Data Mining and Knowledge Discovery — Web address: www.dmkd.flinders.edu.au

Foreword

The Australasian Data Mining Conference series **AusDM**, initiated in 2002, is the annual flagship venue where data mining and analytics professionals - scholars and practitioners, can present the state-of-art in the field. Together with the Institute of Analytics Professionals of Australia **AusDM** has a unique profile in nurturing this joint community. The first and second edition of the conference (held in 2002 and 2003 in Canberra, Australia) facilitated the links between different research groups in Australia and some industry practitioners. The event the event has been supported by:

- Togaware, again hosting the website and the conference management system, coordinating the review process and other essential expertise;
- the University of Technology, Sydney, providing the venue, registration facilities and various other support at the Faculty of Information Technology;
- the Institute of Analytic Professionals of Australia (IAPA) and NetMap Analytics Pty Limited, facilitating the contacts with the industry;
- the e-Markets Research Group, providing essential expertise for the event;
- the ARC Research Network on Data Mining and Knowledge Discovery, providing financial support.

The conference program committee reviewed 42 submissions, out of which 16 submissions have been selected for publication and presentation. **AusDM** follows a rigid blind peer-review process and ranking-based paper selection process. All papers were extensively reviewed by at least three referees drawn from the program committee. We would like to note that the cut-off threshold has been very high (4.1 on a 5 point scale), which indicates that the quality of submissions is very high. We would like to thank all those who submitted their work to the conference. We will be extending the conference format to be able to accommodate more papers.

Today data mining and analytics technology has gone far beyond crunching databases of credit card usage or retail transaction records. This technology is a core part of the so-called “embedded intelligence” in science, business, health care, drug design, security and other areas of human endeavour. Unstructured text and richer multimedia data are becoming a major input to the data mining algorithms. Consistent and reliable methodologies are becoming critical to the success of data mining and analytics in industry. Accepted submissions have been grouped in four sessions reflecting these trends. Each session is preceded by invited industry presentation.

Special thanks go to the program committee members and external reviewers. The final quality of selected papers depends on their efforts. The **AusDM** review cycle runs on a very tight schedule and we would like to thank all reviewers for their commitment and professionalism.

Last but not least, we would like to thank the organisers of AI 2005 and ACAL 2005 for assisting in hosting **AusDM**.

Simeon, J. Simoff, Graham J. Williams
John Galloway and Inna Kolyshkina

November 2005

Conference Chairs

Program Committee

AusDM05 Conference Program, 5th – 6th December 2005, Sydney, Australia

Monday, 5 December, 2005

9:00 - 9:05 **Opening and Welcome**

09:05 - 10:05 **INDUSTRY KEYNOTE "Text Mining"**
Inna Kolyshkina, PricewaterhouseCoopers, Sydney

10:05 - 10:30 Coffee break

10:30 - 12:00 **Session I: Text Mining**

- 10:30 - 11:00 INCORPORATE DOMAIN KNOWLEDGE INTO SUPPORT VECTOR MACHINE TO CLASSIFY PRICE IMPACTS OF UNEXPECTED NEWS
  Ting Yu, Tony Jan, John Debenham and Simeon J. Simoff
- 11:00 - 11:30 TEXT MINING - A DISCRETE DYNAMICAL SYSTEM APPROACH USING THE RESONANCE MODEL
  Wenyuan Li, Kok-Leong Ong and Wee-Keong Ng
- 11:30 - 12:00 CRITICAL VECTOR LEARNING FOR TEXT CATEGORISATION
  Lei Zhang, **Debbie Zhang** and Simeon J. Simoff

12:00 - 12:30 **Panel: Data Mining State-of-the-Art**

12:30 - 13:30 Lunch

13:30 - 14:30 **INDUSTRY KEYNOTE "Network Data Mining"**
John Galloway, NetMap Analytics, Sydney

14:30 - 15:00 Coffee break

15:00 - 17:00 **Session II: Data Linking, Enrichment and Data Streams**

- 15:00 - 15:30 ASSESSING DEDUPLICATION AND DATA LINKAGE QUALITY: WHAT TO MEASURE?
  Peter Christen and Karl Goiser
- 15:30 - 16:00 AUTOMATED PROBABILISTIC ADDRESS STANDARDISATION AND VERIFICATION
  Peter Christen and Daniel Belacic
- 16:00 - 16:30 DIFFERENTIAL CATEGORICAL DATA STREAM CLUSTERING
  Weijun Huang, Edward Omiecinski, Leo Mark and Weiquan Zhao
- 16:30 - 17:00 S-MONITORS: LOW-COST CHANGE DETECTION IN DATA STREAMS
  Weijun Huang, Edward Omiecinski, Leo Mark and Weiquan Zhao

Tuesday, 6 December, 2005

09:00 - 10:00 INDUSTRY KEYNOTE "The Analytics Profession: Lessons and Challenges"
Eugene Dubossarsky, Ernst & Young, Sydney

10:00 - 10:30 Coffee break

10:30 - 12:30 Session III: Methodological issues

- 10:30 - 11:00 DOMAIN-DRIVEN IN-DEPTH PATTERN DISCOVERY: A PRACTICAL METHODOLOGY
  Longbing Cao, Rick Schurmann, Chengqi Zhang
- 11:00 - 11:30 MODELING MICROARRAY DATASETS FOR EFFICIENT FEATURE SELECTION
  Chia Huey Ooi, Madhu Chetty, Shyh Wei Teng
- 11:30 - 12:00 PREDICTING INTRINSICALLY UNSTRUCTURED PROTEINS BASED ON AMINO ACID COMPOSITION
  Pengfei Han, Xiuzhen Zhang, Raymond S. Norton, and Zhiping Feng
- 12:00 - 12:30 A COMPARATIVE STUDY OF SEMI-NAIVE BAYES METHODS IN CLASSIFICATION LEARNING
  Fei Zheng and Geoffrey I. Webb

12:30 - 13:30 Lunch

13:30 - 14:30 INDUSTRY KEYNOTE "Analytics in The Australian Taxation Office"
Warwick Graco, Australian Taxation Office, Canberra

14:30 - 15:00 Coffee break

15:00 - 17:00 Session IV: Methodology and Applications

- 15:00 - 15:30 A STATISTICALLY SOUND ALTERNATIVE APPROACH TO MINING CONTRAST SETS
  Robert J. Hilderman and Terry Peckham
- 15:30 - 16:00 CLASSIFICATION OF MUSIC BASED ON MUSICAL INSTRUMENT TIMBRE
  Peter Somerville and Alexandra L. Uitdenbogerd
- 16:00 - 16:30 A COMPARISON OF SUPPORT VECTOR MACHINES AND SELF-ORGANIZING MAPS FOR E-MAIL CATEGORIZATION
  Helmut Berger and Dieter Merkl
- 16:30 - 17:00 WEIGHTED EVIDENCE ACCUMULATION CLUSTERING
  F. Jorge Duarte, Ana L. N. Fred, André Lourenço and M. Fátima C. Rodrigues

Table of Contents

Incorporate Domain Knowledge into Support Vector Machine to Classify Price Impacts of Unexpected News

Ting Yu, Tony Jan, John Debenham and Simeon Simoff

Institute for Information and Communication Technologies
Faculty of Information Technology, University of Technology, Sydney,
PO Box 123, Broadway, NSW 2007, Australia
{yuting,jant,debenham,simeon}@it.uts.edu.au

Abstract. We present a novel approach for providing approximate answers to classifying news events into simple three categories. The approach is based on the authors' previous research: incorporating domain knowledge into machine learning [1], and initially explore the results of its implementation for this particular field. In this paper, the process of constructing training datasets is emphasized, and domain knowledge is utilized to pre-process the dataset. The piecewise linear fitting etc. is used to label the outputs of the training datasets, which is fed into a classifier built by support vector machine, in order to learn the interrelationship between news events and volatility of the given stock price.

1 Introduction and Background

In macroeconomic theories, the Rational Expectations Hypothesis (REH) assumes that all traders are rational and take as their subjective expectation of future variables the objective prediction by economic theory. In contrast, Keynes already questioned a completely rational valuation of assets, arguing those investors' sentiment and mass psychology play a significant role in financial markets. New classical economists have views these as being *irrational*, and therefore inconsistent with the REH. In an *efficient* market, 'irrational' speculators would simply lose money and therefore fail to survive evolutionary competition.
Hence, financial markets are viewed as evolutionary systems between different, competing trading strategies [2]. In this uncertain world, nobody really knows what exactly the fundamental value is; good news about economic fundamental reinforced by some evolutionary forces may lead to deviations from the fundamental values and overvaluation.
Hommes C.H. [2] specifies the Adaptive Belief System (ABS), which assumes that traders are *boundedly rational*, and implied a decomposition of return into two terms: one martingale difference sequence part according to the conventional EMH theory, and an extra speculative term added by the evolutionary theory. The phenomenon of volatility clustering occurs due to the interaction of heterogeneous traders. In periods of low volatility fundamentalists dominate the market. High volatility may be trig-

gered by news about fundamental values and may be amplified by technical trading. Once a (temporary) *bubble* has started, evolutionary forces may reinforce deviations from the benchmark fundamental values.
As a non-linear stochastic system, ABS: $X_{t+1} = F(X_t; n_{1t}, \ldots, n_{Ht}; \lambda; \delta_t; \varepsilon_t)$
Where F is a nonlinear mapping, the noise term ε_t is the model approximation error representing the fact that a model can only be an approximation of the real world. In economic and financial models one almost has to deal with intrinsic uncertainty represented here by the noise term δ_t. For example one typically deals with investors' uncertainty about economic fundamental values. In the ABS there will be uncertainty about future dividends.
Maheu and McCurdy [3] specified a GARCH-Jump model for return series. They label the innovation to returns, which is directly measurable from price data, as the news impact from latent news innovations. The latent news process is postulated to have two separate components, *normal* and *unusual* news events. These news innovations are identified through their impact on return volatility. The unobservable normal news innovations are assumed to be captured by the return innovation component, $\varepsilon_{1,t}$. This component of the news process causes smoothly evolving changes in the conditional variance of returns. The second component of the latent news process causes infrequent large moves in returns, $\varepsilon_{2,t}$. The impacts of these unusual news events are labelled *jumps*. Given an information set at time t-1, which consists of the history of returns $\Phi_{t-1} = \{r_{t-1}, \ldots, r_t\}$, the two stochastic innovations, $\varepsilon_{1,t}$ and $\varepsilon_{2,t}$ drive returns: $r_t = \mu + \varepsilon_{1,t} + \varepsilon_{2,t}$, $\varepsilon_{1,t}$ is a mean-zero innovation ($E[\varepsilon_{1,t} \mid \Phi_{t-1}] = 0$) with a normal stochastic forcing process, $\varepsilon_{1,t} = \sigma_t z_t, z_t \sim NID(0,1)$ and $\varepsilon_{2,t}$ is a jump innovation.
Both of the previous models provide general frameworks to incorporate the impacts from news articles, but with respect to thousands of news articles from all kinds of sources, these methods do not provide an approach to figure out the significant news of the given stocks. Therefore, these methods cannot make significant improvement in practice.
Numerous publications describe machine-learning researches that try to predict short-term movement of stock prices. However very limited researches have been done to deal with unstructured data due to the difficulty of the combination of numerical data and textual data in this specific field. Marc-Andre Mittermayer developed a prototype NewsCATS [4], which provides a rather completed framework. Being different from this, the prototype developed in this paper, gives an automatic pre-processing approach to build training datasets and keyword sets. Within the NewsCATS, experts do these works manually, and this is very time consuming and lack of flexibility to dynamic environments of stock markets. A similar work has been done by B. Wuthrich and V. Cho et al [5]. The following part of this paper emphasizes the pre-processing approach and the combination of the rule-based clustering and nonparametric classifications.

2. Methodologies and System Design

Being different from common interrelationships among multiple sequences of observations, heterogeneous data e.g. price (or return) series and event sequences are considered in this paper. Normally, the price (or return) series is numerical data, and the later is textual data. At the previous GARCH-Jump model, the component $\varepsilon_{2,t}$ incorporates the impacts from events into price series. But it is manual and time consuming to measure the value of $\varepsilon_{2,t}$ and the model does not provide a clear approach.

Moreover, with respect to thousands of news from overall the world, it is almost impossible for one individual to pick up the significant news and make a rational estimation immediately after they happen. At the following parts, this paper will propose an approach that uses machine learning to classify influent news

The prototype of this classifier is a combination of rule-based clustering, keywords extraction and non-parametric classification e.g. support vector machine (SVM). To initiate this prototype, some training data from the archive of press release and a closing price series from the closing price data archive are fed into the news pre-processing engine, and the engine tries to “align” news items to the price (or return series). After the alignment, training news items are labelled as three types of news using a rule-based clustering. Further the training news items are fed into a keywords extraction engine within the news pre-processing engine [6], in order to extract keywords to construct an archive of keywords, which will be used to convert the news items into term-frequency data understood by the classification engine (support vector machine).

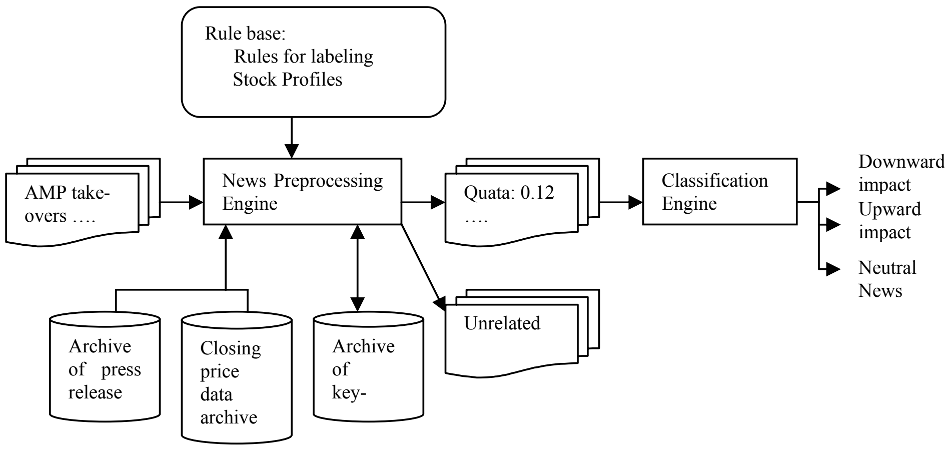


Fig 2.1. Structure of the classifier

After the training process is completed, the inflow of news will be converted as a term-frequency format and fed into the classification engine to predict its impact to the current stock price.

On the other hand, before news items are fed into the classifier, a rule-based filter, "stock profile", screens out the unrelated articles. Given a stock, a set of its casual links is named as its "Stock Profile", which represents a set of characteristics of that stock. For example, AMP is an Australia-based financial company. If a regional natural disease happens in Australia, its impact to AMP is much stronger than its impact to News Corp, which is multi-national news provider. The stock price of AMP is more sensitive to this kind of news than the stock price of News Corp is.

2.1. Temporal Knowledge Discovery

John Roddick et al [7] described that time-stamped data can be scalar values, such as stock prices, or events, such as telecommunication signals. Time-stamped scalar values of an ordinal domain form curves, so-called "time series", and reveal trends. They listed several types of temporal knowledge discovery: Apriori-like Discovery of Association Rules, Template-Based Mining for Sequences, and Classification of Temporal Data. In the case of trend discovery, a rationale is related to prediction: if one time series shows the same trend as another but with a known time delay, observing the trend of the latter allows assessments about the future behaviour of the former.
In order to more deeply explore the interrelationship between sequences of temporal data, the mining technique must be beyond the simple similarity measurement, and the further causal links between sequences is more interesting to be discovered.
In financial research, the stock price (or return) is normally treated as a time series, in order to explore the autocorrelation between the current and previous observations. On the other hand, events, e.g. news arrival, may be treated as a sequence of observations, and it will be very significant to explore correlation between these two sequences of observations.

2.2. A Rule Base Representing Domain Knowledge

How to link two different sequences of observations? A tradition way is employing financial researchers, who use their expertise and read through all of news articles to distinguish. Obviously it is a very time consuming task and not react timely to the dynamic environment. To avoid these problems, this prototype utilizes some existing financial knowledge, especially some time series analysis to price (or return), to label news articles. Here financial knowledge is named as domain knowledge: knowledge about the underlying process, 1) Functional form: either parametric (e.g. addictive or multiplicative), or semi-parametric, or nonparametric; and 2) identify economic cycles, unusual events, and causal forces.
Numerous financial researches have demonstrated that high volatilities often correlate with dramatic price discovery processes, which are often caused by unexpected news arrival, so-called "jump" in the GARCH-Jump model or "shock". On the other hand, as the previous ABS suggested, high volatility may be triggered by news about fundamental values and may be amplified by technical trading, and the ABS model also implied a decomposition of return into two terms: one martingale difference sequence part according to the conventional EMH theory, and an extra speculative term added

by the evolutionary theory. Some other financial researches also suggest that volatility may be caused by two groups of disturbance: traders' behaviours, e.g. trading process, inside the market, and impacts from some events outside the markets, e.g. unexpected breaking news. Borrowing some concepts from the electronic signal processing, "Inertial modelling" is the inherent model structure of the process even without events, and "Transient problem" is the changes of flux after new event happens. Transient problem may cause a shock at series of price (or return), or may change the inherent structure of the stock permanently, e.g. interrelationship between financial factors.

How to represent the domain knowledge into machine learning system? Some researches have been done by Ting Yu et al [1]. The rule base represents domain knowledge, e.g. causal information. Here, in case of unexpected news announcement, the causal link between the news and short-range trend is represented by knowledge about the subject area.

2.2.1. Associating events with patterns in volatility of stock price

A large amount of financial researches have indicated that important information releases are already followed by dramatic price adjustment processes, e.g. extremely increase of trading volume and volatility. This phenomena normally lasts one or two days [8].

In this paper, a filter will treat the observation beyond 3 standard derivations as abnormal volatilities, and the news released at these days with abnormal volatilities will be labelled as shocking news.

Different from the often-used return, e.g. $R_t = \frac{P_t - P_{t-1}}{P_{t-1}}$, the net-of-market return is the difference between absolute return and index return: $NR_t = R_t - IndexR_t$. This indicates the magnitude of information released and excludes the impact from the whole stock market.

Piecewise Linear Fitting:

In order to measure the impact from unexpected news event, the first step is to get rid of the inertial part of the series of return. At the price series, the piecewise linear regression is used to fit into the real price series and detect the change of trend. Here, piecewise linear fitting screens out the disturbance caused by traders' behaviours, which normally are around 70% total disturbances.

Linear regression falls into the category of so-called parametric regression, which assumes that the nature of the relationships (but not the specific parameters) between the dependent and independent variables is known *a priori* (e.g., is linear). By contrast, nonparametric regression does not make any such assumption as to how the dependent variables are related to the predictors. Instead it allows the regression function to be "driven" directly from data [9].

Three major approaches to segment time series [10]: sliding windows, top-down and bottom-up. Here the bottom-up segmentation algorithm is used to fit a piecewise linear function into the price series, and the algorithm is developed by Eamonn Keogh el at [11]. The piecewise segmented model *M* is given by [12]:

$$\begin{aligned} Y &= f_1(t, w_1) + e_1(t), (1 < t < \theta_1) \\ &= f_2(t, w_2) + e_2(t), (\theta_1 < t < \theta_2) \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &= f_k(t, w_k) + e_k(t), (\theta_{k-1} < t < \theta_k) \end{aligned}$$

An $f_i(t, w_i)$ is the function that is fit in segment *i*. In case of the trend estimation, this function is a linear one between price and date. The θ_i's are change points between successive segments, and $e_i(t)$'s are error terms.

In the piecewise fitting of a series of stock price, the connecting points of piecewise release points of the significant change of trends. In the statistics literature this has been called the *change point detection* problem [12].

After detecting the change points, the next stage is to select an appropriate set of news stories. Victor Lavrenko el at named this stage as "Aligning the trends with news stories" [13].

In this paper, these two rules, extreme volatilities detection and change point detection, are employed to label training news items, and at the same time, some rules are employed to screen out the unrelated news. This rule base contains some domain knowledge, which has been discussed at the previous part, and bridges the gap between different types of information.

Collopy and Armstrong have done some similar researches. The objective of their rule base [14] are: to provide more accurate forecasts, and to provide a systematic summary of knowledge. The performance of rule-based forecasting depends not only on the rule base, but also on the conditions of the series. Here conditions mean a set of features that describes a series.

An important feature of time series is *a change in the basic trend* of a series. A piecewise regression line is fitted on the series to detect the level discontinuity and changes of basic trend.

The pseudo-code for an example of the algorithms:

```
rule_base();
Piecewise (data);
While not finish the time series
     If {condition 1, condition 2}  then
         a_set_of_news=scan_news(time);
          Episode_array[i]= a_set_of_news;
     End if
     Return Episode_array;
End loop
/**Rule base**/
rule_base() {
     Condition 1:
       Day ∈ {upward, neutral, downward};
     Condition 2: shock == true;
}
```

The combination of two rules are quite straightforward: unanticipated negative news = within downward trend + large volatility, unanticipated positive news = within upward trend + large volatility.

2.3. Text Classification:

The goal of text classification is the automatic assignment of documents, e.g. company announcements, to simple three categories. In this experiment, the commonly used Term Frequency-Inverse Document Frequency (TF-IDF) is utilized to calculate the frequency of predefined key words in order to represent documents as a set of term-vectors. The set of key words is constructed by comparing general business articles come from the website from the Australian Financial Reviews, with companied announcements collected and pre-processed by Prof Robert Dale [15]. The detailed algorithms are developed by eMarket group. Keywords are not restricted to single words, but can be phrases. Therefore, the first step is to identify phrases in the target corpus. The phrases are extracted based on the assumption that two constituent words form a collocation if they co-occur a lot [6].

2.3.1. Extracting Document Representations

Documents are represented as a set of fields where each field is a term-vector. Fields could include the title of the document, the date of the document and the frequency of selected key words.
In a corpus of documents, certain terms will occur in the most of the documents, while others will occur in just a few documents. The inverse document frequency (*IDF*) is a factor that enhances the terms that appear in fewer documents, while downgrading the terms occurring in many documents. The resulting effect is that the document-specific features get highlighted, while the collection-wide features are diminished in importance. TF-IDF assigns the term *i* in document *k* a weight computed as:

$$TF_{ik} * IDF(t_i) = \frac{f_k(t_i)}{\sqrt{\sum_{t_i \in D_k} f_k^2(t_i)}} * \log(\frac{n}{DF(t_i)})$$

Here DF (Document frequency of the term (t_i)) – the number of documents in the corpus that the term appears; n – the number of documents in the corpus; TF$_{ik}$ – the occurrence of term i at the document k [16]. As a result, each document is represented as a set of vectors $F^{dk} =< term_i, weight >$.

2.3.2. Train the Classifier

Without the clear knowledge about the ways how the news influence a stock price, nonparametric methods seems to be the better choice than the parametric methods that base on prior assumptions, e.g. Logistic Regression. Here, the frequencies of selected key words are used as the input of Support Vector Machine (SVM). Under a supervised learning, the train sets consist of $< F^{dk}$, {upward impact, neutral impact, downward impact}>, which are constructed by the methods discussed at the previous part of this paper. Some of similar researches have been found at papers published by Ting Yu et al [1] and James Tin-Yan Kwok et al [17].

3 Experiments

Here the price series and return series of AMP are used to carry out some experiments. The first figures (Fig. 3.1) are the closing price and net return series of AMP from 15/06/1998 to 16/03/2005. On the other hand, more than 2000 company announcements are collected as a series of news items, which covers the same period as the closing prices series.

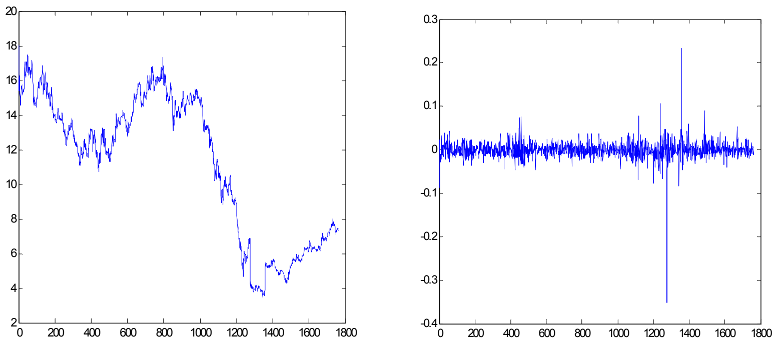


Fig. 3.1, Closing price and net return series of AMP

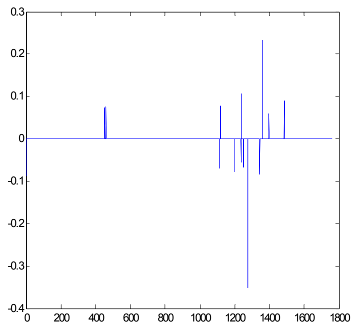


Fig. 3.2a. Shocks (large volatilities)

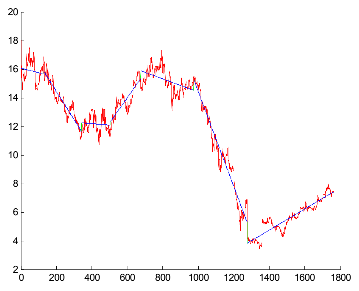


Fig. 3.2b. Trend and changing points

The second figures indicate shocks (large volatilities) (Fig. 3.2a), and the trend changing points detected (Fig. 3.2b) by the piecewise linear fitting. After pre-processing, the training dataset consists of 464 upwards news items, 833 downward news items and 997 neutral news items. The keywords extraction algorithm constructs a keyword set consisting of 36 single or double terms, e.g. vote share, demerg, court, qanta, annexure, pacif, execut share, memorandum, cole etc. these keywords are stemmed following the Porter Stemming Algorithm, written Martin Porter [18]. The dataset is split into two parts: training and test data. The result of classification, e.g. upwards or downwards, is compared with the real trends of the stock price. Under LibSVM 2.8 [19], the accuracy of classification is 65.73%, which is significant higher than 46%, the average accuracy of Wuthrich's experiments [5].

4 Conclusions and Further Work

This paper provides a brief framework to classify the coming news into three categories: upward, neural or downward. One of the major purposes of this research is to provide financial participants and researchers an automatic and powerful tool to screen out influential news (information shocks) among thousand of news around this world everyday. Another main purpose is to discuss an AI based approach to quantify the impact from news events to stock price movements.
The current prototype has demonstrated promising results of this approach, although the result of experiments is long distance from the practical satisfaction. On the further researches, the mechanism of impacts will be discussed more deeply to get better domain knowledge to improve the performance of machine learning. More experiments will be carried to compare the results between different types of stocks and between different stock markets.
In the further work, three major issues must be concerned, which are suggested by Nikolaus Hautsch [20]: 1) Inside information: if inside information has already been disclosed at the market, the price discovery process will be different. 2) Anticipated vs. unanticipated information: if traders' belief has absorbed the information, so-called anticipated information, the impact must be expressed as a conditional probability with the brief as a prior condition. 3) Interactive effects between information: at the current experiment all news at one point are labelled as a set of upward impacts or other, but the real situation is much more complex. Even at one upward point, it is common that there is some news with downward impacts. It will be very challenging to distinguish the subset of minor news and measure the interrelationship between news.

Acknowledgment

The authors would like to thank Dr. Debbie Zhang and Paul Bogg for their invaluable comments and discussion, and thank Prof Robert Dale for his company announcements as XML formats.

References

1. Yu, T., T. Jan, J. Debenham, and S. Simoff. *Incorporating Prior Domain Knowledge in Machine Learning: A Review*. in *AISTA 2004: International Conference on Advances in Intelligence Systems - Theory and Applications in cooperation with IEEE Computer Society*. 2004. Luxembourg.
2. Hommes, C.H., *Financial Markets as Nonlinear Adaptive Evolutionary Systems*, in *Tinbergen Institute Discussion Paper*. 2000, University of Amsterdam.
3. Maheu, J.M. and T.H. McCurdy, *News Arrival, Jump Dynamics and Volatility Components for Individual Stock Returns.* Journal of Finance, 2004. **59**(2): p. 755.
4. Mittermayer, M.-A. *Forecasting Intraday Stock Price Trends with Text Mining Techniques*. in *The 37th Hawaii International Conference on System Sciences*. 2004.
5. Wuthrich, B., V. Cho, S. Leung, D. Permunetilleke, K. Sankaran, J. Zhang, and W. Lam, *Daily Stock Market Forecast from Textual Web Data*, in *IT Magazine*. 1998. p. 46-47.
6. Zhang, D., S.J. Simoff, and J. Debenham. *Exchange Rate Modelling using News Articles and Economic Data*. in *The 18th Australian Joint Conference on Artificial Intelligence*. 2005. Sydney Australia.
7. Roddick, J.F. and M. Spiliopoulou, *A survey of temporal knowledge discovery paradigms and methods.* Knowledge and Data Engineering, IEEE Transactions on, 2002. **14**(4): p. 750-767.
8. Lee, C.C., M.J. Ready, and P.J. Seguin, *Volume, volatility, and New York Stock Exchange trading halts.* Journal of Finance, 1994. **49**(1): p. 183-214.
9. StatSoft, *Electronic Statistics Textbook*. 2005.
10. Keogh, E., S. Chu, D. Hart, and M. Pazzani, *Segmenting Time Series: A Survey and Novel Approach*, in *Data Mining in Time Series Databases*. 2003, World Scientific Publishing Company.
11. Keogh, E., S. Chu, D. Hart, and M. Pazzani. *An Online Algorithm for Segmenting Time Series.* in *In Proceedings of IEEE International Conference on Data Mining*. 2001.
12. Guralnik, V. and J. Srivastava. *Event Detection From Time Series Data*. in *KDD-99*. 1999. San Diego, CA USA.
13. Lavrenko, V., M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, and J. Allan, *Language Models for Financial News Recommendation*. 2000, Department of Computer Science, University of Massachusetts: Amhers, MA.
14. Collopy, F. and J.S. Armstrong, *Rule-Based Forecasting: Development and Validation of an Expert Systems Approach to Combining Time Series Extrapolations.* Journal of Management Science, 1992. **38**(10): p. 1394-1414.
15. Dale, R., R. Calvo, and M. Tilbrook. *Key Element Summarisation: Extracting Information from Company Announcements*. in *Proceedings of the 17th Australian Joint Conference on Artificial Intelligence*. 2004. Cairns, Queensland, Australia.
16. Losada, D.E. and A. Barreiro, *Embedding term similarity and inverse document frequency into a logical model of information retrieval.* Journal of the American Society for Information Science and Technology, 2003. **54**(4): p. 285 - 301.
17. Kwok, J.T.-Y. *Automated text categorization using support vector machine*. in *Proceedings of {ICONIP}'98, 5th International Conference on Neural Information Processing*. 1998. Kitakyushu, Japan.
18. Porter, M., *An algorithm for suffix stripping.* Program, 1980. **14**(3): p. 130-137.
19. Chang, C.-C. and C.-J. Lin, *LIBSVM: a Library for Support Vecter Machine*. 2004, Department of Computer Sicence and Information Engineering, National Taiwan University.

20. Hautsch, N. and D. Hess, *Bayesian Learning in Financial Markets - Testing for the Relevance of Information Precision in Price Discovery.* Journal of Financial and Quantitative Analysis, 2005.

Text Mining – A Discrete Dynamical System Approach Using the Resonance Model

Wenyuan Li[1], Kok-Leong Ong[2], and Wee-Keong Ng[1]

[1] Nanyang Technological University, Centre for Advanced Information Systems
Nanyang Avenue, N4-B3C-14, Singapore 639798
liwy@pmail.ntu.edu.sg, awkng@ntu.edu.sg

[2] School of Information Technology, Deakin University
Waurn Ponds, Victoria 3217, Australia
leong@deakin.edu.au

Keywords: text mining, biclique, discrete dynamical system, text clustering, resonance phenomenon.

Abstract. Text mining plays an important role in text analysis and information retrieval. However, existing text mining tools rarely address the high dimensionality and sparsity of text data appropriately, making the development of relevant and effective analytics difficult. In this paper, we propose a novel pattern called *heavy bicliques*, which unveil the inter-relationships of documents and their terms according to different density levels. Once discovered, many text analytics can be built upon this pattern to effectively accomplish different tasks. In addition, we also present a discrete dynamical system called the *resonance model* to find these heavy bicliques quickly. The preliminary results of our experiments proved to be promising.

1 Introduction

With advancements in storage and communication technologies, and the popularity of the Internet, there is an increasing number of online documents containing information of potential value. Text mining has been touted by some as the technology to unlock and uncover the knowledge contained in these documents. Research on text data has been on-going for many years, borrowing techniques from related disciplines (e.g., information retrieval and extraction, and natural language processing) including entity extraction, N-grams statistics, sentence bound, etc. This has led to a wide number of applications in business intelligence (e.g., market analysis, customer relationship management, human resources, technology watch, etc.) [1, 2], and in inferencing biomedicine literature [3–6] to name a few.

In text mining, there are several problems being studied. Typical problems include information extraction, document organization, and finding predominant themes in a given collection [7]. Underpinning these problems are techniques such as summarization, clustering, and classification, where efficient tools exist, such as CLUTO [3] [8] and SVM [4] [9]. Regardless of the text feature extraction method, or the linguistic technique used, these tools fail to meet the needs of the analyst due to high dimensionality and sparsity of the text data. For example, text clustering based on traditional formulations (e.g., optimization of a metric) is insufficient for a text collection with complex and reticula topics. Likewise, a simple flat partition (or even a hierarchical partition; see [10]) of the text collection is often insufficient to characterize the complex relationships between the documents and its topics.

To overcome the above, we propose the concept of `Heavy Biclique` (denoted simply as `HB`) to characterize the inter-relationships between documents and terms according to their densities levels. Although similar to recent biclusters, which identify coherence, our patterns determine the density of a submatrix, i.e., the number of non-zeros. Thus, our proposal can also be viewed as a variant of heavy subgraphs and yet, are more descriptive and flexible than traditional cluster definitions. Many text mining tasks can be built upon this pattern. One application of `HB` is to find those candidate terms with sufficient density for summarization. Compared against existing methods, our algorithm that discovers the `HB`s are more efficient at dealing with high dimensionality, sparsity, and size. This efficiency is achieved by the use of a discrete dynamical system (DDS) to obtain `HB`, which simulates the resonance phenomenon in the physical world. Since it can converge quickly to a give a solution, the empirical results proved to be promising.

The outline of this paper is as follows. We give a formal definition of our problem and propose the novel pattern call `Heavy Biclique` in the next section. Section 3 presents the discrete dynamical system to obtain `HB`, while Section 4 discusses the initial results. Section 5 discusses the related work before we conclude in Section 6 with future directions and works.

2 Problem Formulation

Let $\mathcal{O}$ be a set of objects, where $o \in \mathcal{O}$ is defined by a set of attributes $\mathcal{A}$. Further, let w_{ij} be the magnitude (absolute value) of o_i over $a_j \in \mathcal{A}$[5]. Then we can represent the relationship of all objects and their attributes in a matrix $W = (w_{ij})_{|\mathcal{O}|\times|\mathcal{A}|}$ for the weighted bipartite graph $G = (\mathcal{O}, \mathcal{A}, E, W)$, where E is the set of edges, and $|\mathcal{O}'|$ is the number of elements in the set $\mathcal{O}$, similarly $|\mathcal{A}|$. Thus, the relationship between the dataset W and the bipartite graph G is established to give the definition of a `Heavy Biclique`.

Definition 1. *Given a weighted bipartite graph G, a σ-`Heavy Biclique` (or simply σ-`HB`) is a subgraph $G' = (\mathcal{O}', \mathcal{A}', E', W')$ and $W' = (w_{ij})_{|\mathcal{O}'|\times|\mathcal{A}'|}$ of G*

[3] `http://www-users.cs.umn.edu/~karypis/cluto/`

[4] `http://svmlight.joachims.org/`

[5] By default, all magnitude (absolute value, or the modulus) of o_i are non-negative. If not, they can be scaled to non-negative numbers.

	A_1	A_2	A_3	A_4	A_5
O_1	5	20	15	8	16
O_2	16	8	5	19	2
O_3	18	6	7	17	3
O_4	3	12	20	5	20

	A_2	A_5	A_3	A_4	A_1
O_1	**20**	**16**	**15**	8	5
O_4	**12**	**20**	**20**	5	3
O_2	8	2	5	**19**	**16**
O_3	6	3	7	**17**	**18**

Heavy Biclique

	A_3	A_2	A_4	A_5	A_1
O_1	**15**	**20**	8	**16**	5
O_4	**20**	**12**	5	**20**	3
O_3	7	6	**17**	3	**18**
O_2	5	8	**19**	2	**16**

Fig. 1. The matrix with 4 objects and 5 attributes: (a) original matrix; (b) reordered by non-linear model; (c) reordered by linear model.

satisfying $|W'| \geqslant \sigma$*, where* $|W'| = \frac{1}{|\mathcal{O}'||\mathcal{A}'|} \sum_{\substack{i \in \mathcal{O}' \\ j \in \mathcal{A}'}} w_{ij}$*. Here,* σ *is the density threshold.*

Suppose we have a matrix, as shown in Figure 1(a), with 4 objects and 5 attributes containing entries scaled from 1 to 20. After reordering this matrix, we may find its largest heavy biclique in the top-left corner as shown in Figure 1(b) (if we set $\sigma = 16$). This biclique is $\{O_1, O_4\} \times \{A_2, A_3, A_5\}$. If we assume objects as documents, attributes as terms, and each entry as the frequency of a term occurring in a document, we immediately find that a biclique describes a topic in a subset of documents and terms. Of course, real-world collections are not as straightforward as Figure 1(b). Nevertheless, we may use this understanding to develop better algorithms to find subtle structures of collections.

A similar problem is the **Maximum Edge Biclique Problem (MBP)**: given a bipartite graph $G = (V_1 \cup V_2, E)$ and a positive integer K, does G contain a biclique with at least K edges? Although this bipartite graph G is unweighted, the problem is NP-complete [11]. Recall from Definition 1, letting $K = \sigma|\mathcal{O}'||\mathcal{A}'|$ makes G unweighted. Then, the problem of finding σ-`Heavy Biclique` by setting $\sigma = 1$ reduces to the **MBP** problem, i.e., our problem of finding largest σ-`HB` is very hard as well. Hence, it is therefore important to have a method to efficiently find `HB`s in a document-term matrix. This will also lay the foundation for future works in developing efficient algorithms based on `HB`s.

3 The Resonance Model – A Discrete Dynamical System

Given the difficulty of finding σ-`HB`, we seek alternative methods to discover the heavy bicliques. Since our objective is to find the bicliques with high density $|W'|$, then some approximation to the heaviest bicliques (that is computationally efficient) should suffice. To obtain the approximation of heaviest biclique for a dataset, we used a novel model inspired by the physics of resonance. This resonance model, which is a kind of discrete dynamical system [12], is very efficient even on very large and high-dimensional datasets.

To understand the rationale behind its efficiency, we can discuss a simple analogy. Suppose we are interested in finding the characteristics of students who are fans of thriller movies. One way is to poll each student. Clearly, this is time-consuming. A better solution is to gather a sample but we risk acquiring a wrong sample that leads to a wrong finding. A smarter approach is to announce the free screening of a blockbuster thriller. In all likelihood, the fans of thrillers will turn up for the screening. Despite the possibility of 'false positives', this sample is easily and quickly obtained with minimum effort.

The scientific model that corresponds to the above is the principle of resonance. In other words, we can simulate a resonance experiment by injecting a response function to elicit objects of interest to the analyst. In our analogy, this response function is the blockbuster thriller that fans automatically react to by going to the screening. In sections that follow, we present the model and discuss its properties and support practicality of the model by discussing how it improves analysis using some real-world applications.

3.1 Model Definition

To simulate a resonance phenomenon, we require a forcing object $\tilde{o}$, such that when an appropriate response function $\mathtt{r}$ is applied, $\tilde{o}$ will resonate to elicit those objects $\{o_i, \ldots\} \subset \mathcal{O}$ in G, whose 'natural frequency' is similar to $\tilde{o}$. This 'natural frequency' represents the characteristics of both $\tilde{o}$ and the objects $\{o_i, \ldots\}$ who resonated with $\tilde{o}$ when $\mathtt{r}$ was applied. For the weighted bipartite graph $G = (\mathcal{O}, \mathcal{A}, E, W)$ and $W = (w_{ij})_{|\mathcal{O}|\times|\mathcal{A}|}$, this 'natural frequency' of $o_i \in \mathcal{O}$ is $\mathbf{o_i} = (w_{i1}, w_{i2}, \ldots, w_{i|\mathcal{A}|})$. Likewise, the 'natural frequency' of the forcing object $\tilde{o}$ is defined as $\mathbf{\tilde{o}_i} = (\tilde{w_1}, \tilde{w_2}, \ldots, \tilde{w}_{|\mathcal{A}|})$.

Put simply, if two objects of the same 'natural frequency' will resonate and therefore, should have a similar distribution of frequencies, i.e., those entries with high values and the same attributes shall be easily identified. The evaluation of resonance strength between objects o_i and o_j is given by the response function $\mathtt{r}(\mathbf{o_i}, \mathbf{o_j}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. We defined this function abstractly to support different measures of resonance strength. For example, one existing measure to compare two frequency distributions is the well-known *rearrangement inequality theorem*, where $\mathtt{I}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} x_i y_i$ is maximized when the two positive sequences $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ are ordered in the same way (i.e. $x_1 \geqslant x_2 \geqslant \cdots \geqslant x_n$ and $y_1 \geqslant y_2 \geqslant \cdots \geqslant y_n$) and is minimized when they are ordered in the opposite way (i.e. $x_1 \geqslant x_2 \geqslant \cdots \geqslant x_n$ and $y_1 \leqslant y_2 \leqslant \cdots \leqslant y_n$).

Notice if two vectors maximizing $\mathbf{I}(\mathbf{x}, \mathbf{y})$ are put together to form $M = [\mathbf{x}; \mathbf{y}]$ (in MATLAB format), we obtain the entry value tendency of these two vectors. More importantly, all σ-HB are immediately obtained from this 'contour' of the matrix with no need to search every σ-HB! This is why the model is efficient – it only needs to consider the resonance strength among objects once the appropriate response function is selected. For example, the response function $\mathtt{I}$ is a suitable candidate to characterize the similarity of frequency distributions of two objects. Likewise, $\mathtt{E}(\mathbf{x}, \mathbf{y}) = \exp(\sum_{i=1}^{n} x_i y_i)$ is also an effective response function.

To find the heaviest biclique, the forcing object $\tilde{o}$ evaluates the resonance strength of every objects o_i against itself to locate a 'best fit' based on the 'contour' of the whole matrix. By running this iteratively, those objects that resonated with $\tilde{o}$ are discovered and placed together to form the heaviest biclique within the 2-dimensional matrix W. This iterative learning process between $\tilde{o}$ and G is outlined below.

Initialization Set up $\tilde{o}$ with a uniform distribution: $\mathbf{\tilde{o}} = (1, 1, \ldots, 1)$; normalize it as $\mathbf{\tilde{o}} = \mathtt{norm}(\mathbf{\tilde{o}})$[6]; then let $k = 0$; and record this as $\mathbf{\tilde{o}}^{(0)} = \mathbf{\tilde{o}}$.

Apply Response Function For each object $o_i \in \mathcal{O}$, compute the resonance strength $\mathtt{r}(\mathbf{\tilde{o}}, \mathbf{o_i})$; store the results in a vector $\mathbf{r} = \big(\mathtt{r}(\mathbf{\tilde{o}}, \mathbf{o_1}), \mathtt{r}(\mathbf{\tilde{o}}, \mathbf{o_2}), \ldots, \mathtt{r}(\mathbf{\tilde{o}}, \mathbf{o}_{|\mathcal{O}|})\big)$; and then normalize it, i.e., $\mathbf{r} = \mathtt{norm}(\mathbf{r})$.

Adjust Forcing Object Using $\mathbf{r}$ from the previous step, adjust the frequency distribution of $\tilde{o}$ for all $o_i \in \mathcal{O}$. To do this, we define the adjustment function $\mathtt{c}(\mathbf{r}, \mathbf{a_j}) : \mathbb{R}^{|\mathcal{O}|} \times \mathbb{R}^{|\mathcal{O}|} \to \mathbb{R}$, where the weights of the j-th attribute is given in $\mathbf{a_j} = (w_{1j}, w_{2j}, \ldots, w_{|\mathcal{O}|j})$. For each attribute a_j, $\tilde{w_j} = \mathtt{c}(\mathbf{r}, \mathbf{a_j})$ integrates the weights from $\mathbf{a_j}$ into $\tilde{o}$ by evaluating the resonance strength recorded in $\mathbf{r}$. Again, $\mathtt{c}$ is abstract, and can be materialized using the inner product $\mathtt{c}(\mathbf{r}, \mathbf{a_j}) = \mathbf{r} \bullet \mathbf{a_j} = \sum_i w_{ij} \cdot \mathtt{r}(\mathbf{\tilde{o}}, \mathbf{o_i})$. Finally, we compute $\mathbf{\tilde{o}} = \mathtt{norm}(\mathbf{\tilde{o}})$ and record it as $\mathbf{\tilde{o}}^{(k+1)} = \mathbf{\tilde{o}}$.

Test Convergence Compare $\mathbf{\tilde{o}}^{(k+1)}$ against $\mathbf{\tilde{o}}^{(k)}$. If the result converges, go to the next step; else apply $\mathtt{r}$ on $\mathcal{O}$ again (i.e., forcing resonance), and then adjust $\tilde{o}$.

Reordering Matrix Sort the objects $o_i \in \mathcal{O}$ by the coordinates of $\mathbf{r}$ in descending order; and sort the attributes $a_i \in \mathcal{A}$ by the coordinates of $\mathbf{\tilde{o}}$ in descending order.

We denote the resonance model as $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{r}, \mathtt{c})$, where the instances of functions $\mathtt{r}$ and $\mathtt{c}$ can be either $\mathtt{I}$ or $\mathtt{E}$. Interestingly, the instance $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{I}, \mathtt{I})$ is actually the HITS algorithm [13], where W is the adjacency matrix of a directed graph. However, this instance is actually different from HITS in 3 ways: (i) the objective of our model is to obtain an approximate heaviest biclique of the dataset (through the resonance simulation), while HITS is designed for Web IR and looks at only the top k authoritative Web pages (a reinforcement learning process); (ii) the implementation is different by the virtue that our model is able to use a non-linear instance, i.e., $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{E}, \mathtt{E})$, to discover heavy bicliques while HITS is strictly linear; and (iii) we study a different set of properties and functions from HITS, i.e., *heaviest biclique.*

3.2 Properties of the Model

We shall discuss some important properties of our model in this section. In particular, we show that the model gives a good approximation to the heaviest biclique, and that its iterative process converges quickly.

[6] $\mathtt{norm}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|_2$, where $\|\mathbf{x}\|_2 = (\sum_{i=1}^{n} x_i^2)^{1/2}$ is 2-norm of vector $\mathbf{x} = (x_1, \ldots, x_n)$.

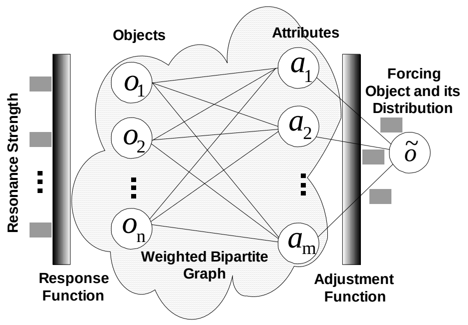


Fig. 2. Architecture of the resonance model

Convergence Since the resonance model is iterative, it is essential that it converges quickly to be efficient. Essentially, the model can be seen as a type of *discrete dynamical system* [12]. Where the functions in the system is linear, then it is a *linear dynamical system.* For linear dynamical systems, it corresponds to eigenvalue and eigenvector computation [12–14]. Hence, its convergence can be proved by eigen-decomposition for R where the response and adjustment functions are linear. In the non-linear case (i.e., $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{E}, \mathtt{E})$), its convergence is proven below.

Theorem 1. $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{r}, \mathtt{c})$*, where* $\mathtt{r}$*,* $\mathtt{c}$ *are* $\mathtt{I}$ *or* $\mathtt{E}$*, converges in limited iterations.*

Proof. When $\mathtt{r}$ and $\mathtt{c}$ are $\mathtt{I}$, we get $\tilde{\mathbf{o}}^{(k)} = \mathtt{norm}\big(\tilde{\mathbf{o}}^{(0)}(W^TW)^k\big)$ by linear algebra [14]. If A is symmetric and $\mathbf{x}$ is a row vector that is not orthogonal to the first eigenvector corresponding to the first largest eigenvalue of A, then $\mathtt{norm}(\mathbf{x}A^k)$ converges to the first eigenvector as k increases. Thus, $\tilde{\mathbf{o}}^{(k)}$ converges to the first eigenvector of W^TW. As the exponential function has Maclaurin series $\exp(x) = \sum_{n=0}^{\infty} x^n/n!$, the convergence of the non-linear model with $\mathtt{E}$ functions can be decomposed to the convergence of the model, when $\mathtt{r}$ and $\mathtt{c}$ are simple polynomial functions x^n.

So far, either implementations converge quickly if a reasonable precision threshold ϵ is set. In practice, this is acceptable because we are only interested in the convergence of orders of coordinates in $\tilde{\mathbf{o}}^k$ and $\mathbf{r}^k$, i.e., we are not interested in how closely $\tilde{\mathbf{o}}^k$ and $\mathbf{r}^k$ approximate the converged $\tilde{\mathbf{o}}^*$ and $\mathbf{r}^*$. Furthermore, $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{E}, \mathtt{E})$ converges faster than $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{I}, \mathtt{I})$. Therefore, each iteration of learning is bounded by $\mathbf{O}(|\mathcal{O}| \times t_{\mathtt{r}} + |\mathcal{A}| \times t_{\mathtt{c}})$), where $t_{\mathtt{r}}$ and $t_{\mathtt{c}}$ is the runtime of the response function $\mathtt{r}$, and the adjustment function $\mathtt{c}$ respectively. With k iterations, the final complexity is $\mathbf{O}(k \times (|\mathcal{O}| \times t_{\mathtt{r}} + |\mathcal{A}| \times t_{\mathtt{c}}))$. Since the complexity of $\mathtt{r}$ is $\mathbf{O}(|\mathcal{O}|)$ and $\mathtt{c}$ is $\mathbf{O}(|\mathcal{A}|)$, we have $\mathbf{O}(k \times |\mathcal{O}| \times |\mathcal{A}|)$. In our experiments (in Section 4), our model converges within 50 iterations even on the non-linear configurations giving a time complexity of $\mathbf{O}(|\mathcal{O}| \times |\mathcal{A}|)$. In all cases, the complexity is sufficiently low to efficiency handle large datasets.

Average Inter-resonance Strength $\frac{1}{\binom{k}{2}} \sum_{\substack{i \neq j \in \mathcal{O}' \\ |\mathcal{O}'|=k}} \mathtt{r}(\mathbf{o}_i, \mathbf{o}_j)$ among Objects
Theorem 2 is in fact an optimization process to find the best k objects, whose average inter-resonance strength is the largest among any subset of k objects.

Lemma 1. *Given a row vector $\mathbf{u} = (u_1, u_2, \ldots, u_n)$, where $u_1 \geqslant u_2 \geqslant \ldots \geqslant u_n \geqslant 0$, we generate a matrix $U = \lambda \mathbf{u}^T \mathbf{u}$, where $\lambda > 0$ is a scale factor. We then define the k-sub-matrix of U as $U_k = U(1:k, 1:k)$ (in MATLAB format). Then, U has the following 'staircase' property*

$$|U_1| \geqslant |U_2| \ldots \geqslant |U_k| \ldots \geqslant |U_n| = |U| \tag{1}$$

where $|U|$ of a symmetric matrix $U = (u_{ij})_{n \times n}$ is given as $|U| = \frac{1}{\binom{n}{2}} \sum_{1 \leqslant i \neq j \leqslant n} u_{ij}$.

Proof. By induction, when $n = 2$ (base case), we prove $|U_2| \geqslant |U_3|$. Since $|U_2| = u_1 u_2$, $|U_3| = \frac{1}{3}\big(u_1 u_2 + u_1 u_3 + u_2 u_3\big)$ and $u_1 \geqslant u_2 \geqslant u_3 \geqslant 0$, we have $|U_2| \geqslant |U_3|$. When $n = k$, we prove $|U_k| \geqslant |U_{k+1}|$. We first define

$$x_{k+1} = \frac{1}{2k+1}\Big(u_{k+1}^2 + 2 \sum_{1 \leqslant i \leqslant k} u_i u_{k+1}\Big)$$

which after a straightforward calculation, we have the following

$$|U_{k+1}| \geqslant x_{k+1} \tag{2}$$

$$|U_{k+1}| - |U_k| = \frac{2k+1}{(k+1)^2}\Big(x_{k+1} - |U_k|\Big) \tag{3}$$

and finally from Equations (2) and (3), we have

$$|U_k| \geqslant |U_{k+1}|$$

Lemma 2. *Given a resonance space $R_{|\mathcal{O}| \times |\mathcal{O}|} = WW^T$ of $\mathcal{O}$, its first eigenvalue λ, and the eigenvector $\mathbf{u} = (u_1, u_2, \ldots, u_n) \in \mathbb{R}^{1 \times n}$, we have*

$$\begin{aligned} &\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^{1 \times n} \\ &\|R - \lambda \mathbf{u}^T \mathbf{u}\|_F \leqslant \|R - \mathbf{x}^T \mathbf{y}\|_F \end{aligned} \tag{4}$$

where $\| \bullet \|_F$ denotes the Frobenius norm of a matrix.

Proof. We denote the the first singular value of A as s (its largest absolute value), and the corresponding left and right singular vectors as $\mathbf{p}$ and $\mathbf{q}$, respectively. By the Eckart-Young theorem, any given matrix $B_{n \times n}$ that satisfies the rank is 1. Therefore, we have

$$\|A - s\mathbf{p}\mathbf{q}^T\|_F \leqslant \|A - B\|_F$$

and by the symmetric property of A, it can be proved that $s = \lambda$ and $\mathbf{p} = \mathbf{q} = \mathbf{u}$. Rewriting the inequality will give us

$$\|A - \lambda \mathbf{u}\mathbf{u}^T\|_F \leqslant \|A - B\|_F \tag{5}$$

where for any two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n\times 1}$, the rank of $\mathbf{x}\mathbf{y}^T$ is 1. Therefore, substituting $\mathbf{x}\mathbf{y}^T$ for B in the inequality (5) gives us Equation 4.

Theorem 2. *Given the reordered matrix W' by the resonance model, the average inter-resonance strength $\frac{1}{\binom{k}{2}} \sum_{1\leqslant i\neq j\leqslant k} \mathtt{r}(\mathbf{o}_i, \mathbf{o}_j)$ of the first k objects, w.r.t. the resonance strength with $\tilde{o}$, is largest for any subset with k objects.*

Proof. For linear models, i.e., $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{I}, \mathtt{I})$, $\mathbf{r} = \big(\mathtt{r}(\tilde{\mathbf{o}}, \mathbf{o_1}), \mathtt{r}(\tilde{\mathbf{o}}, \mathbf{o_2}), \ldots, \mathtt{r}(\tilde{\mathbf{o}}, \mathbf{o}_{|\mathcal{O}|})\big)$ converges to the first eigenvector $\mathbf{u}$ of WW^T, i.e. $\mathbf{r} = \mathbf{u}$ as shown in Theorem 1. And since the functions are linear, we can rewrite them as $WW^T = \big(\mathtt{r}(\mathbf{o}_i, \mathbf{o}_j)\big)_{|\mathcal{O}|\times|\mathcal{O}|}$. Further, since W and R are already reordered in descending order of their resonance strength $\mathbf{u}$, we have the following (together with Lemma 1 and Lemma 2)

$$|R_1| \geqslant |R_2| \ldots \geqslant |R_k| \ldots \geqslant |R_n| = |R| \tag{6}$$

and because $|R_k| = \frac{1}{\binom{k}{2}} \sum_{1\leqslant i\neq j\leqslant k} \mathtt{r}(\mathbf{o}_i, \mathbf{o}_j)$ is the average inter-resonance strength of the first k objects, we have Theorem 2.

Approximation to Heaviest Biclique In the non-linear configuration of our model, i.e., $\mathrm{R}(\mathcal{O}, \mathcal{A}, W, \mathtt{E}, \mathtt{E})$, we have another interesting property that is not available in the linear model: the approximation to the *heaviest biclique*. Our empirical observations in Section 4 further confirmed this property of the non-linear model in finding the heaviest σ-HB. Given the efficiency of our model, it is therefore possible to find heavy biclique by running the model on different parts of the matrix with different σ. We exploited this property to find heavy bicliques, i.e., the algorithm that we shall discuss in the next subsection.

3.3 Algorithm of Approximating the Complete 1-HB

Recall from Theorem 2, the first k objects have the highest average inter-resonance strength. Therefore, we can expect a higher probability of finding the heaviest biclique among these objects. This has also been observed in various experiments earlier [15], and we note that the exponential functions in the non-linear models are better at eliciting the heavy biclique from the top k objects (compare Figure 1(b) and 1(c)). We will illustrate this with another example using the MovieLens [15] dataset. The matrix is shown in Figure 3(a). Here, we see that the non-zeros are scattered without any distinct clusters or concentration. After reordering using both models, we see that the non-linear model in Figure 3(c) better shows the heavy biclique than that of the linear model in Figure 3(b).

While the non-linear model is capable of collecting entries with high values to the top-left corner of the reordered matrix, a strategy is required to extend

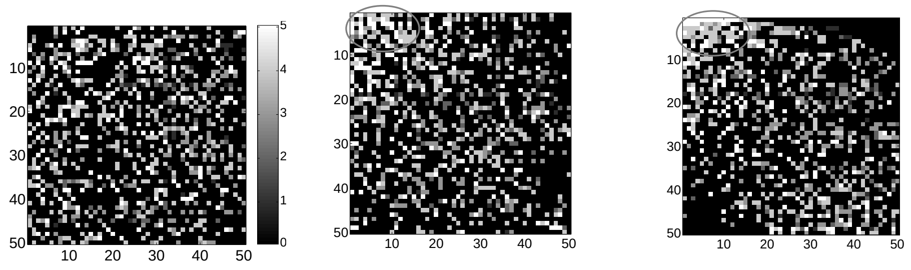


Fig. 3. Gray scale images of original and reordered matrix with 50 rows and 50 columns by different resonance models: (a) original matrix; (b) reordered by linear model; (c) reordered by non-linear model. In (b) and (c), the top-left corner circled by gray ellipse is the initial heavy biclique found by the models.

the 1-`HB` biclique found to the other parts of the matrix. The function `Find_B` is to find a 1-`HB` biclique by extending a row of the reordered matrix to a biclique using the heuristic in Line 5 of `Find_B`. The loop from Line 4 to 9 in `Find_1HB` is needed to get the bicliques computed from each row. The largest 1-`HB` biclique is then obtained by comparing the size $|B.L||B.R|$ among the bicliques found. The complexity of `Find_B` is $\mathbf{O}(|\mathcal{O}||\mathcal{A}|)$. Hence, the complexity of `Find_1HB` is $\mathbf{O}((k_1 + k_2)|\mathcal{O}||\mathcal{A}|)$, where k_1 is the convergence loop number of the non-linear model, and k_2 is the loop number in the **FOR** statement of `Find_1HB`. If computing on all rows, k_2 is $|\mathcal{O}|$. However, because most large bicliques are concentrated on the left-top corner, the loop for `Find_1HB` is insignificant, i.e., we could set k_2 to a small value to consider only the first few rows to reduce the runtime complexity of `Find_1HB` to $\mathbf{O}(|\mathcal{O}||\mathcal{A}|)$.

4 Preliminary Experimental Results

Our result is preliminary, but promising. In our experiment, we used the `re0` text collection[7], that has been widely used in [10, 16]. This text collection contains 1504 documents, 2886 stemmed terms and 13 predefined classes ("*housing*", "*money*", "*trade*", "*reserves*", "*cpi*", "*interest*", "*gnp*", "*retail*", "*ipi*", "*jobs*", "*lei*", "*bop*", "*wpi*"). Although `re0` has 13 predefined classes, most of the clusters are small with some having less than 20 documents while a few classes ("*money*", "*trade*" and "*interest*") made up 76.2% of documents in `re0`, i.e., the remaining 10 classes contain 23.8% of the documents. Therefore, traditional clustering algorithms may not be applicable in finding effective clusters. Moreover, due to the diverse and unbalanced distribution of classes, traditional clustering algorithms may not be helpful for users to effectively understand the relationships and details among documents. This is made more challenging when the 10 classes are highly related. Therefore, we applied our initial method based

[7] `http://www-users.cs.umn.edu/~karypis/cluto/files/datasets.tar.gz`

Algorithm 1 $B = \texttt{Find\_1HB}(G)$, Find the complete 1-HB in G

```
Input   : $G = (\mathcal{O}, \mathcal{A}, E, W)$ and $\sigma$
Output  : 1-HB, $B = (L, R, E', W')$, where $L \subseteq \mathcal{O}$ and $R \subseteq \mathcal{A}$
 1: convert $W = (w_{ij})$ to the binary matrix $W_b = (b_{ij})$, by setting $b_{ij}$ as 1 if $w_{ij} > 0$ and 0
    otherwise
 2: get reordered binary matrix $W_b^*$ by doing R($\mathcal{O}, \mathcal{A}, W_b$, E, E)
 3: $maxsize = 0$ and $B = \emptyset$
 4: for $i = 1$ to $k_2$ do {comment: $i$ is index of row, $k_2$ can be set with a small fixed value by
    users.}
 5:    $B = \texttt{Find\_B}(W_b^*, i)$
 6:    if $(|B.L||B.R| > maxsize)$ then
 7:       record $B$
 8:    end if
 9: end for
10: if $(B \neq \emptyset)$ then
11:    get $B.W'$ from $W$ by $B.L$ and $B.R$
12: end if
```

```
$B = \texttt{Find\_B}(W_b^*, start\_row)$
 1: set $B.L$ empty and addset$(B.L, start\_row)$
 2: $B.R$=binvec2set$(\mathbf{b}^*_{start\_row})$ and $maxsize = |B.R|$
 3: for $i = (start\_row + 1)$ to $|\mathcal{O}|$ do
 4:    $R = B.R/$binvec2set$(\mathbf{b}_i^*)$
 5:    if $((|B.L| + 1)|R| > maxsize)$ then
 6:       $B.R = R$ and addset$(B.L, i)$
 7:       $maxsize = |B.L||B.R|$
 8:    end if
 9: end for
10: $B = \texttt{Extend\_B}(W_b^*, B)$
```

```
$B' = \texttt{Extend\_B}(W_b^*, B)$
 1: $start\_row = \texttt{min}(B.L)$
 2: for $i = 1$ to $(start\_row - 1)$ do
 3:    $R = $ binvec2set$(\mathbf{b}_i^*)$
 4:    if $(B.R \subseteq R)$ then
 5:       addset$(B.L, i)$
 6:    end if
 7: end for
 8: $B' = B$
```

Note on set functions:

`binvec2set` returns elements with indices of non-zero coordinates in the binary vector.
`addset` adds a value to a set.
`min` returns the minimum value among all elements of a set.
`A/B` returns a set whose elements are in A, but not in B.

on the resonance model, Algorithm 1, to find something interesting in `re0`, that may not be discovered by traditional clustering algorithms.

We used the binary matrix representation of `re0`, i.e. the weights of all terms occurring in documents are set to 1. In the experiment, we implemented and used Algorithm 1 to find 1-HB. That is to say, we find the complete large bicliques in the unweighted bipartite graph. Here, we present some interesting results in the following.

Result 1: we found a biclique with 287 documents, where every document contains several stemmed terms: *pct*, *bank*, *rate*, *market*, *trade*, *billion*, *monei*, *billion*, *expect* and so on. This means these documents are highly related each other in terms of money, banking and trade. However, these documents are from 10 classes except "*housing*", "*lei*", "*wpi*". So this result indicates how these documents are related in key words and domains, although they come from different classes. Traditional clustering algorithms can not find such subtle details among documents.

Result 2: We also found several bicliques with small numbers of documents, where they share a large number of terms. That is to say, documents in a biclique may be duplicated in whole of in part. For example, a biclique with three documents has 233 terms. This means these three documents do duplicate each other.

Result 3: Some denser sub-cluster in a single class were found by our algorithm. For example, a biclique whose all documents belong to "*money*" was found. It is composed of 81 documents with the key terms: *market*, *monei*, *england*, *assist*, *shortag*, *forecast*, *bill* and *stg* (the abbreviation of sterling). From this biclique, we may find that documents in this sub-cluster contain more information about assistance and shortage in money and market areas.

In this initial experiment, three types of denser sub-clusters were found as shown above. They represent dense sub-cluster across different classes, in single classes and duplicated documents. Further experiments can be done in more text collections.

5 Related Work

Biclique problems have been addressed in different fields. There are traditional approximation algorithms rooted in mathematical programming relaxation [17]. Despite their polynomial runtime, their best result is 2-approximations, i.e., the subgraph discovered may not be a biclique but must contain the exact maximum edge biclique that is double in size. The other class of algorithms is to exhaustively enumerate all maximum bicliques [18] and then do a post-processing on all the maximum bicliques to obtain the desired results. Although efficient algorithms have been proposed and applied to computational biology [19], the runtime cost is too high. The third class of algorithms are developed based on some given conditions. For example, the bipartite graph $G{=}(\mathcal{O}, \mathcal{A}, E, W)$ must be of d-bounded degree, i.e., $|\mathcal{O}| < d$ or $|\mathcal{A}| < d$ [20] to give a complexity of $\mathbf{O}(n2^d)$ where $n{=}\max(|\mathcal{O}|, |\mathcal{A}|)$. While this gives the exact solution, the given conditions often do not satisfy the needs of real-world datasets and the runtime cost can be high for large d.

We can also view our work as a form of clustering. Often, clustering in high-dimensional space is problematic [21]. Therefore, subspace clustering and biclustering were proposed to discover the clusters embedded in the subspaces of the high-dimensional space. Subspace clustering, e.g., CLIQUE, PROCLUS, ORCLUS, fascicles, etc., are extensions of conventional clustering algorithms that seek to find clusters by measuring the similarity in a subset of dimensions [22]. Biclustering was first introduced in gene expression analysis [23], and then applied in data mining and bioinformatics [24]. Biclusters are measured based on submatrices and therefore, is equivalent to the maximum edge biclique problem [24]. Under this context, a σ-B is similar to a bicluster. However, these algorithms are inefficient, especially in the data with very high dimensionality and massive size. Therefore, they are only suitable to datasets with tens of hundreds of dimensions and medium size, such as gene expression data, and they are not applicable to text data with thousands and tens of thousands of dimensions and massive size.

Since our simulation of the resonance phenomenon involves an iterative learning process, where the forcing object would update its weight distribution, our work can also be classified as a type of dynamical system, i.e., the study of how one state develops into another over some course of time [12]. Actually, the application and design of discrete dynamical system has been widely used in neural networks. Typical applications include the well-known Hopfield network [25] and bidirectional associative memory network [26] for combinatorial optimization and pattern meomories. In the recent years, this field has contributed to many important and effective techniques in information retrieval, e.g., HITS [13], PageRank [27] and others [28]. In dynamical systems, the theory on its linear counterpart is closely related to the eigenvectors of matrices as used in HITS and PageRank; while the non-linear aspect is what forms the depth of dynamical systems theory. From its success in information retrieval, we were motivated to apply this field of theory to solve combinatorial problems in data analysis. To the best of our knowledge, our application of dynamical systems for analysis of massive and skewed datasets is completely novel.

6 Conclusions

In this paper, we proposed a novel pattern call `heavy bicliques` to be discovered in text data. We show that finding these heavy bicliques proved to be difficult and computationally expensive. As such, the resonance model – which is a discrete dynamical system simulating the resonance phenomenon in the physical world – is used to approximate the heavy bicliques. While this result is approximated, our initial experiments confirmed the effectiveness in producing heavy bicliques quickly and accurately for analytics purposes.

Of course, the initial results present a number of future works possible. In addition to further and more thorough experiments, we are also interested in developing algorithms that uses the heaviest bicliques to mine text data according to the different requirements of the users as illustrated in Algorithm 1. We are also interested in testing our work on very large data sets leading to the development of scalable algorithms for finding heavy bicliques.

References

1. Halliman, C.: Business Intelligence Using Smart Techniques : Environmental Scanning Using Text Mining and Competitor Analysis Using Scenarios and Manual Simulation. Information Uncover (2001)
2. Sullivan, D.: Document Warehousing and Text Mining: Techniques for Improving Business Operations, Marketing, and Sales. John Wiley & Sons (2001)
3. Afantenos, D.S., V. Karkaletsis, P.S.: Summarization from medical documents: A survey. Artificial Intelligence in Medicine **33** (2005) 157–177
4. Hoffmann, R., Krallinger, M., Andres, E., Tamames, J., Blaschke, C., Valencia, A.: Text mining for metabolic pathways, signaling cascades, and protein networks. Signal Transduction Knowledge Environment (2005) 21

5. Krallinger, M., Erhardt, R.A., Valencia, A.: Text-mining approaches in molecular biology and biomedicine. Drug Discovery Today **10** (2005) 439–445
6. Ono, T., Hishigaki, H., Tanigami, A., Takagi, T.: Automated extraction of information on protein-protein interactions from the biological literature. Bioinformatics **17** (2001) 155C161
7. Tkach, D.: Text mining technology turning information into knowledge: A white paper from IBM. Technical report, IBM Software Solutions (1998)
8. Karypis, G.: Cluto: A clustering toolkit. Technical Report #02-017, Univ. of Minnesota (2002)
9. Joachims, T.: Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms. PhD thesis, Kluwer (2002)
10. Zhao, Y., Karypis, G.: Hierarchical clustering algorithms for document datasets. Technical Report 03-027, Univ. of Minnesota (2003)
11. Peeters, R.: The maximum edge biclique problem is NP-complete. Disc. App. Math. **131** (2003) 651–654
12. Sandefur, J.T.: Discrete Dynamical Systems. Oxford: Clarendon Press (1990)
13. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. J. ACM **46** (1999) 604–632
14. Golub, G., Loan, C.V.: Matrix Computations. The Johns Hopkins University Press (1996)
15. Li, W., Ong, K.L., Ng, W.K.: Visual terrain analysis of high-dimensional datasets. Technical Report TRC04/06, School of IT, Deakin University (2005)
16. Zhao, Y., Karypis, G.: Criterion functions for document clustering: Experiments and analysis. Technical Report 01-40, Univ. of Minnesota (2001)
17. Hochbaum, D.S.: Approximating clique and biclique problems. J. Algorithms **29** (1998) 174–200
18. Alexe, G., Alexe, S., Crama, Y., Foldes, S., Hammer, P.L., Simeone, B.: Consensus algorithms for the generation of all maximum bicliques. Technical Report 2002-52, DIMACS (2002)
19. Sanderson, M.J., Driskell, A.C., Ree, R.H., Eulenstein, O., Langley, S.: Obtaining maximal concatenated phylogenetic data sets from large sequence databases. Molecular Biology and Evolution **20** (2003) 1036–1042
20. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics **18** (2002) 136–144
21. Beyer, K.: When is nearest neighbor meaningful? In: Proc. ICDT. (1999)
22. Parsons, L., Haque, E., Liu, H.: Subspace clustering for high dimensional data: a review. ACM SIGKDD Explorations Newsletter **6** (2004) 90–105
23. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Proc. 8th Int. Conf. on Intelligent System for Molecular Biology. (2000)
24. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. IEEE Transactions on computational biology and bioinformatics **1** (2004) 24–45
25. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proc. National Academy of Sciences **79** (1982) 2554–2558
26. Kosko, B.: Bidirectional associative memories. IEEE Transaction on Systems, Man and Cybernetics **SMC** (1988) 49–60
27. Lawrence, P., Sergey, B., Rajeev, M., Terry, W.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Tech. Project (1999)
28. Tsaparas, P.: Using non-linear dynamical systems for web searching and ranking. In: Proc. PODS. (2004)

Critical Vector Learning for Text Categorisation

Lei Zhang, Debbie Zhang, and Simeon J. Simoff

Faculty of Information Technology, University of Technology, Sydney
PO Box 123 Broadway NSW 2007 Australia
{leizhang, debbiez, simeon}@it.uts.edu.au

Abstract. This paper proposes a new text categorisation method based on the critical vector learning algorithm. By implementing a Bayesian treatment of a generalised linear model of identical function form to the support vector machine, the proposed approach requires significantly fewer support vectors. This leads to much reduced computational complexity of the prediction process, which is critical in online applications.

Key words: Support Vector Machine, Relevance Vector Machine, Critical Vector Learning, Text Classification

1 Introduction

Text categorisation is the classification of natural text or hypertext documents into a fixed number of predefined categories based on their content. Many machine learning approaches have been used in the text classification problem [1]. One of the leading approaches is the support vector machine (SVM) [2], which has demonstrated successfully in many applications. SVM is based on generalisation theory of statistical inference. SVM classification algorithms, proposed to solve two-class problems, are based on finding a separation between hyper planes. In the application of SVM in text categorisation [3–6], it fixes the representation of text document, extracts features from the set of text documents needed to be classified, then selects subset of features, transforms the set of documents to a series of binary classification sets, and final makes kernel from document features. SVM has good performance on large data sets and scales well. It is linear efficient and scalable to large document sets. Using the Reuters News Data Sets, Rennie and Rifkin [7] compared the SVM with Naive Bayes algorithm based on two data sets: 19,997 news related documents in 20 categories and 9649 industry sector data documents in 105 categories. Another researcher Joachims [8] compared the performance of several algorithms with SVM by using 12,902 documents from the Reuters 21578 document set and 20,000 medical abstracts from the Ohsumed corpus. Both Rennie and Joachims has shown that SVM performed better.

Tipping [9] introduced the relevance vector machine (RVM) methods which can be viewed from a Bayesian learning framework of kernel machine and produces an identical functional form to the SVM. Tipping compared the RVM with SVM and demonstrated that the RVM has a comparable generalisation performance to the SVM and requires dramatically fewer kernel functions or model terms than the SVM. As Tipping stated, SVM suffer from its limitation of probabilistic prediction and Mercer's condition that it must be the continuous symmetric kernel of a positive integral operator. While RVM adopt a fully probabilistic framework and sparsity is achieved because the posterior distributions of many of the weights are sharply peaked around zero. The relevance vector comes from those training vectors associated with the remaining non-zero weights. However, a draw back of the RVM algorithm is a significant increase in computational complexity, compared with the SVM. Orthogonal least square (OLS) was first developed for the nonlinear data modelling, recently Chen [10–12] derived the locally regularised OLS (LROLS) algorithm to construct sparse kernel models, which has shown to possess computational advantages compared with RVM. The LROLS only selects the significant terms, while RVM starts with the full model set. Moreover, LROLS only use a subset matrix of the full matrix that has been used by RVM. The subset matrix is diagonal and well-conditioned with small eigen-value spread. Further to Chen's research, Gao [13] has derived a critical vector learning (CVL) algorithm and improved the LROLS algorithm for the regression model, which has shown to possess more computational advantages. In this paper, the critical vector classification learning algorithm is applied to the text categorisation problem. Comparison results of SVM and CVL using the Reuters News Data Sets are presented and discussed.

The rest of this paper is organised as follows: In section 2, the basic idea of SVM is reviewed and explains its limitation compared with RVM. The algorithm of RVM with critical vector classification is presented in section 3. The detail implementation of applying critical learning algorithm in text categorisation is described in section 4. In section 5, the experiments are carried out using the Reuters data set, followed by the conclusions in section 6.

2 The Support Vector Machine

SVM is a learning system that uses a hypothesis space of linear functions in a high dimensional feature space. Joachims [8] explained the reason that SVM works well for text categorisation. Let's consider the binary classification problems about text document categorisation with SVM. Linear support vector machine trained on separable data. Let f be a function of $f : X \subseteq R^n \to R$, where X is the term frequency representation of documents. The input $x \in X$ is assigned to the positive class, if $f(x) \geq 0$; otherwise to negative class. When consider the $f(x)$ is a linear function, it can be rewritten as

$$f(x) = \langle w \cdot x \rangle + b = \sum_{i=1}^{n} w_i x_i + b \qquad (1)$$

where w is the weight vector. The basic idea of the support vector machine is to find the largest margin to do the classification in the hyper-plane, which means

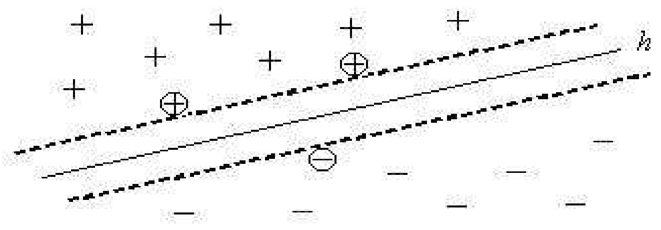


Fig. 1. Support vector machines find the hyper-plane h, which separates the positive and negative training examples with maximum margin. The examples closest to the hyper-plane in Figure 1 are called Support Vectors (marked with circles).

to minimise $\|w\|^2$, subject to

$$(x_i \cdot w) + b \geq +1 - \xi_i, \text{ for } y_i = +1, \tag{2}$$

$$(x_i \cdot w) + b \leq -1 + \xi_i, \text{ for } y_i = -1. \tag{3}$$

where the ξ_i is the slack variable. The optimal classification function is given by

$$g(x) = \operatorname{sgn}\{\langle w \cdot x \rangle + b\} \tag{4}$$

An appropriate inner product kernel $K(x_i, x_j)$ will be selected to realise the linear classification for non-linear problem. Then the equation (1) can be written as:

$$y(\mathbf{x}; \mathbf{w}) = \sum_{i=1}^{N} w_i K(\mathbf{x}, \mathbf{x}_i) + w_0 \tag{5}$$

Support vector machine has demonstrated successfully in many applications. However SVM suffers four major disadvantages: unnecessary use of basis functions; predictions are not probabilistic; entails a cross-validation procedure and the kernel function must satisfy Mercer's condition.

3 Critical Vector Learning

Tipping introduced the relevance vector machine (RVM), which does not suffer from the limitations mentioned in section 2. RVM can be viewed from a Bayesian learning framework of kernel machine and produces an identical functional form to the SVM. RVM generates predictive distributions which is a limitation of the SVM. And also RVM requires substantially fewer kernel functions.

Consider the scalar-valued target functions and giving the input-target pairs $\{\mathbf{x}_n, t_n\}_{n=1}^{N}$. The noise is assumed to be zero-mean Gaussian distribution with a variance of σ^2 . The likelihood of the complete data set can be written as

$$p\left(\mathbf{t}|\mathbf{w}, \sigma^2\right) = \left(2\pi\sigma^2\right)^{-N/2} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{t} - \mathbf{\Phi}\mathbf{w}\|^2\right\} \tag{6}$$

where $\mathbf{t} = (t_1...t_N)^T$, $\mathbf{w} = (w_1...w_N)^T$, and $\mathbf{\Phi} = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2)..., \phi(\mathbf{x}_N)]^T$, wherein $\phi(\mathbf{x}_n) = [1, K(\mathbf{x}_n, \mathbf{x}_1), K(\mathbf{x}_n, \mathbf{x}_2), ..., K(\mathbf{x}_n, \mathbf{x}_N)]^T$. To make a simple function for the Gaussian prior distribution over w , 6 can be written as:

$$p(\mathbf{w}|\alpha) = \prod_{i=0}^{N} \mathcal{N}\left(w_i|0, \alpha_i^{-1}\right) \tag{7}$$

where α is a vector of $N+1$ hyper parameters.

Relevance vector learning can be looked as the search for the hyper parameter posterior mode, i.e. the maximisation of $p\left(\alpha, \sigma^2|\mathbf{t}\right) \propto p\left(\mathbf{t}|\alpha, \sigma^2\right) p(\alpha) p\left(\sigma^2\right)$ with respect to α and $\beta(\beta \equiv \sigma^2)$. RVM involves the maximisation of the product of the marginal likelihood and priors over α and σ^2. And MacKay [14] has given

$$\alpha_i^{new} = \frac{\gamma_i}{2\mu_i^2},\ \beta^{new} = \frac{\|\mathbf{t} - \mathbf{\Phi}\mu\|^2}{N - \sum_i \gamma_i} \tag{8}$$

where μ_i is the $i-th$ posterior mean weight and N in the denominator refers to the number of data examples and not the number of basis functions. $\gamma_i \in [0,1]$ can be interpreted as a measure of how well-determined its corresponding parameter w_i is by the data.

A drawback of the RVM is a significant increase in computational complexity. Based on kernel methods and least squares algorithm, a locally regularised orthogonal least squares (LROLS) algorithm has been derived by Chen [10] to construct sparse kernel model.

$$\begin{gathered} y(k) = f\left(y(k-1), ..., y(k-n_y), u(k-1), ..., u(k-n_u)\right) + e(k) \\ y(k) = f(x(k)) + e(k) \end{gathered} \tag{9}$$

where, $x(k) = \left[y(k-1), ..., y(k-n_y), u(k-1), ..., u(k-n_u)\right]^T$ denotes the system "input" vector, f is the unknown system mapping. Considering a general discrete-time nonlinear system represented by a nonlinear model, $u(k)$ and $y(k)$ are the system input and output variables, respectively, n_y and n_u are positive integers representing the lags in $y(k)$ and $u(k)$, respectively, $e(k)$ is the system white noise.

The system identification involves in construct a function (model) to approximate the unknown mapping f based on an N-sample observation dataset $D = \{x(k), y(k)\}_{k=1}^{N}$, i.e., the system input-output observation data $\{u(k), y(k)\}$. The most popular class of such approximating functions is the kernel regression model of the form:

$$y(k) = \widehat{y}(k) + e(k) = \sum_{i=1}^{N} \omega_i \phi_i(k) + e(k),\ 1 \le k \le N \tag{10}$$

where $\widehat{y}(k)$ denotes the "approximated" model output, ω_i's are the model weights, and $\phi_i(k) = k(x(i), x(k))$ are the classifiers generated from a given kernel function $k(x,y)$ [15].

Focus on the single kernel function and by definitions in [13], the model can be viewed as the following matrix form:

$$y = \Phi\omega + e \tag{11}$$

The goal is to find the best linear combination of the columns of Φ (i.e. the best value for ω) to explain y according to some criterion. The normal criterion is to minimise the sum of squared errors,

$$E = e^T e \tag{12}$$

where the solution ω is called the least squares solution to the above model. Detail implementation is given in [16].

An equivalent regularisation formula can be adopted in the critical vector algorithm with PRESS statistic for the regularised objective [13]. The regularised critical vector algorithm with PRESS statistic is based on the following regularised error criterion

$$E(\omega, \alpha, \beta) = \beta e^T e + \sum_{i=1}^{n_M} \alpha_i \omega_i^2 = \beta e^T e + \omega^T H \omega \tag{13}$$

where n_M is the number of involved critical vectors, β is the noise parameter and $H = diag\{\alpha_1, ..., \alpha_{n_M}\}$ consisting of the hyper parameters used for regularising weights. The key issue in regularised regression formulation is to automatically optimise the regularisation parameter. The Bayesian evidence technique [14] can readily be used for this objective. Estimating hyper parameters is implemented in a loop procedure based on the calculation for α and β [17].

Define

$$A = \beta\Phi^T\Phi + H \tag{14}$$

and

$$\gamma_i = 1 - \alpha_i \left(A^{-1}\right)_{ii}, \ \gamma = \sum_{i=1}^{n_M} \gamma_i \tag{15}$$

Then the update formulas for hyper parameters α_i and β can be given by

$$\alpha_i^{new} = \frac{\gamma_i}{2\omega_i^2}, \ \beta^{new} = \frac{N-\gamma}{2e^T e} \tag{16}$$

The iterative hyper parameter and model selection procedure can be summarised:

Initialisation Set initial value for α_i and β for $i = 1, 2, ..., N$, for example, using estimated noise variance for the inverse of β and a small value 0.0001 for all α_i.

Step 1 Given the current α_i and β, use the procedure with PRESS statistic to select a subset model with critical vectors.

Step 2 Update α_i and β using equation 16. If α_i and β remains sufficiently unchanged in two successive iterations or a pre-set maximum iteration number is reached, then stop the algorithm; otherwise go to step 1.

4 Applying Critical Vector Learning in Text Categorisation

The document collection with n documents is represented by a term frequency document matrix

$$C = \begin{bmatrix} d_1 \\ \vdots \\ d_j \\ \vdots \\ d_n \end{bmatrix} \in \Re^{m \times n} \tag{17}$$

where document vector $d_j \in \Re^{m \times 1}$ represents the term frequency of m key terms in each document. The target variable

$$y = [y_1, \cdots, y_j, \cdots, y_n]^T \tag{18}$$

where y_j denotes the corresponding output of d_j, which represents the category that d_j belongs to.

The procedures of the training process was implemented as follows:

1. Calculate the keyword frequency of each document to construct the term frequency document matrix.
2. Construct the kernel matrix. Its (i, j)-th element is $K(d_i, d_j)$. Denote $\mathbf{x}_i$ as the i-th row of the kernel matrix Φ.
3. Select the k best $\mathbf{x}_i$ by repeating the following steps k times:
   (a) For every $\mathbf{x}_i$, use the least square algorithm to estimate the $\omega_\mathbf{i}$ in equation 11.
   (b) Select the $\mathbf{x}_i$ with the smallest error.
   (c) Remove the i-th row of the kernel matrix (corresponding to the selected $\mathbf{x}_i$ f) to form a new matrix.
   (d) Remove the corresponding i-th element in the target variable vector y and form a new target variable as:

$$y = [y_1 - \mathbf{x}_i\omega_i, \cdots, y_{i-1} - \mathbf{x}_i\omega_i, y_{i+1} - \mathbf{x}_i\omega_i, \cdots, y_n - \mathbf{x}_i\omega_i]^T$$

4. Construct the training kernel model, $K\_training(\mathbf{x}_k) = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k)$.

The prediction (or test) is conducted using the constructed training kernel.

5 Experimental Results

Experimental studies have been carried out to compare the performance of CVL and SVM. In this study, a java library for SVM (LIBSVM) was utilised while CVL was implemented using Scilab.

The Reuters News Data Sets, which has been frequently used as benchmarks for classification algorithms, has been used in this paper for the experiments. The

Reuters 21578 collection is a set of 21,578 short (average 200 words in length) news items, largely financially related, that have been pre-classified manually into 118 categories.

The experiments were conducted using 100 and 200 documents from three news group: C15 (performance group), C22 (new products/services group) and C21 (products/services group). The first set of experiments used C15 and C22 data, while the second set of experiments used C21 and C22. The second set of data is more difficult to classify than the first set since data sets C21 and C22 are closely related. This is confirmed by the experimental results, as shown in table 1 and table 2.

Table 1. Results of SVM and CVL classifiers on C15 and C22 data

No. of Documents	No. of Keywords	nSv (SVM)	Accuracy (SVM)	nSv (CVL)	Accuracy (CVL)
100	50	83	92.3%	13	91.02%
	100	83	92.3%	13	91.02%
200	50	122	92.4%	14	93.6%
	100	122	92.4%	14	93.6%

Table 2. Results of SVM and CVL classifiers on C21 and C22 data

No. of Documents	No. of Keywords	nSv (SVM)	Accuracy (SVM)	nSv (CVL)	Accuracy (CVL)
100	50	86	85.89%	14	84.61%
	100	86	85.89%	14	84.61%
200	50	153	84.81%	14	89.24%
	100	153	84.81%	14	89.24%

The result of the experiment shows that critical vector learning algorithm achieves the comparable accuracy with SVM. The advantage of using critical vector learning algorithm is that it requires dramatically fewer support vectors to construct the training model. This means it has less computation complexity and requires less computation time in conducting the prediction after the model is being built.

SVM performs slightly better when the number of document increase, while the CVL remain almost the same. However the number of support vectors required by SVM grows linearly with the size of the training set, while CVL various slightly.

The result of the experiment also shows that both SVM and CVL are not sensitive to the number of keywords, which the accuracy and the number of support vectors remain the same with different keyword attributes.

While SVM and CVL are implemented in different languages, comparison of computational time cannot be conducted at this stage. The next step is to implement CVL using JAVA which allows meaningful comparison of execution times.

6 Conclusions

The critical learning algorithm based on the kernel methods and least squares algorithm has achieves comparable classification accuracy to the SVM. SVM performs better when the number of document increase, but require much more support vectors with the size of the training set increasing. CVL requires slightly different number of the support vectors when the training set increase. The most benefit of CVL is that it requires dramatically fewer numbers of support vectors to construct the model. This will improve the prediction efficiency which is particularly useful in online applications.

References

1. Sebastiani, F.: Machine learning in automated text categorisation. ACM Computing Surveys **34** (2002) 1–47
2. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
3. Amasyali, M., Yildirim, T.: Automatic text categorization of news articles. In: Signal Processing and Communications Applications Conference, 2004. Proceedings of the IEEE 12th, Turkish (2004) 224 – 226
4. Basu, A., Walters, C., Shepherd, M.: Support vector machines for text categorization. In: Proceedings of the 36th Annual Hawaii International Conference on System Sciences, Hawaii (2003)
5. Hu, J., Huang, H.: An algorithm for text categorization with svm. In: IEEE Region 10 Conference on Computers, Communications,Control and Power Engineering. Volume 1., Beijin, China (2002) 47 – 50
6. Hu, X.Y.C., Chen, Y., Wang, L., Yun-Fa: Text categorization based on frequent patterns with term frequency. In: International Conference on Machine Learning and Cybernetics. Volume 3., Shanghai, China (2004) 1610 – 1615
7. Rennie, J., Rifkin, R.: Improving multi-class text classification with support vector machine. Technical report, Massachusetts Institute of Technology. AI Memo AIM-2001-026. (2001)
8. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: 10th European Conference on Machine Learning, Springer Verlag (1998) 137–142
9. Tipping, M.E.: Sparse bayesian learning and the relevance vector machine. Journal of Machine Learning Research **1** (2001) 211–244
10. Chen, S.: Locally regularised orthogonal least squares algorithm for the construction of sparse kernel regression models. In: 2002 6th International Conference on Signal Processing. Volume 2. (2002) 1229 – 1232
11. Chen, S., Hong, X., Harris, C.: Sparse kernel regression modeling using combined locally regularized orthogonal least squares and d-optimality experimental design. IEEE Transactions on Automatic Control **48** (2003) 1029 – 1036

12. Chen, S., Hong, X., Harris, C.: Sparse kernel density construction using orthogonal forward regression with leave-one-out test score and local regularization. IEEE Transactions on Systems, Man and Cybernetics, Part B **34** (2004) 1708 – 1717
13. Gao, J., Zhang, L., Shi, D.: Critical vector learning to construct sparse kernel modeling with press statistic. In: International Conference on Machine Learning and Cybernetics. Volume 5., Shanghai, China (2004) 3223 – 3228
14. MacKay, D.: Bayesian interpolation. IEEE Transactions on Neural Networks (1992) 415–447
15. Schlkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press, Cambridge, Massachusetts (2002)
16. Sun, P.: Sparse kernel least squares classifier. In: Fourth IEEE International Conference on Data Mining, Brighton, UK (2004) 539 – 542
17. Nabney, I.: Algorithms for Pattern Recognitions. Springer, London (2001)

Assessing Deduplication and Data Linkage Quality: What to Measure?

http://datamining.anu.edu.au/linkage.html

Peter Christen[⋆] and Karl Goiser

Department of Computer Science,
Australian National University,
Canberra ACT 0200, Australia
{peter.christen,karl.goiser}@anu.edu.au

Abstract. Deduplicating one data set or linking several data sets are increasingly important tasks in the data preparation steps of many data mining projects. The aim of such linkages is to match all records relating to the same entity. Research interest in this area has increased in recent years, with techniques originating from statistics, machine learning, information retrieval, and database research being combined and applied to improve the linkage quality, as well as to increase performance and efficiency when deduplicating or linking very large data sets. Different measures have been used to characterise the quality of data linkage algorithms. This paper presents an overview of the issues involved in measuring deduplication and data linkage quality, and it is shown that measures in the space of record pair comparisons can produce deceptive accuracy results. Various measures are discussed and recommendations are given on how to assess deduplication and data linkage quality.

Keywords: data or record linkage, data integration and matching, deduplication, data mining pre-processing, quality measures.

1 Introduction

With many businesses, government organisations and research projects collecting massive amounts of data, data mining has in recent years attracted interest both from academia and industry. While there is much ongoing research in data mining algorithms and techniques, it is well known that a large proportion of the time and effort in real-world data mining projects is spent understanding the data to be analysed, as well as in the data preparation and pre-processing steps (which may well dominate the actual data mining activity). An increasingly important task in data pre-processing is detecting and removing duplicate records that relate to the same entity within one data set. Similarly, linking or matching records relating to the same entity from several data sets is often required, as information from multiple sources needs to be integrated, combined or linked in

[⋆] Corresponding author

order to allow more detailed data analysis or mining. The aim of such linkages is to match all records relating to the same entity, such as a patient, a customer, a business, a consumer product, or a genome sequence.

Deduplication and data linkage can be used to improve data quality and integrity, to allow re-use of existing data sources for new studies, and to reduce costs and efforts in data acquisition. In the health sector, for example, deduplication and data linkage have traditionally been used for cleaning and compiling data sets for longitudinal or other epidemiological studies [23]. Linked data might contain information that is needed to improve health policies, and which traditionally has been collected with time consuming and expensive survey methods. Statistical agencies routinely link census data [18, 37] for further analysis. Businesses often deduplicate and link their data sets to compile mailing lists, while within taxation offices and departments of social security, data linkage and deduplication can be used to identify people who register for benefits multiple times or who work and collect unemployment benefits. Another application of current interest is the use of data linkage in crime and terror detection. Security agencies and crime investigators increasingly rely on the ability to quickly access files for a particular individual, which may help to prevent crimes by early intervention.

The problem of finding similar entities doesn't only apply to records which refer to persons. In bioinformatics, data linkage helps to find genome sequences in large data collections that are similar to a new, unknown sequence at hand. Increasingly important is the removal of duplicates in the results returned by Web search engines and automatic text indexing systems, where copies of documents – for example bibliographic citations – have to be identified and filtered out before being presented to the user. Comparing consumer products from different online stores is another application of growing interest. As product descriptions are often slightly different, comparing them becomes difficult.

If unique entity identifiers (or keys) are available in all the data sets to be linked, then the problem of linking at the entity level becomes trivial: a simple database join is all that is required. However, in most cases no unique keys are shared by all of the data sets, and more sophisticated data linkage techniques need to be applied. An overview of such techniques is presented in Section 2. The notation used in this paper, and a problem analysis are discussed in Section 3, before a description of various quality measures is given in Section 4. A real-world example is used in Section 5 to illustrate the effects of applying different quality measures. Finally, several recommendations are given in Section 6, and the paper is concluded with a short summary in Section 7.

2 Data Linkage Techniques

Computer-assisted data linkage goes back as far as the 1950s. At that time, most linkage projects were based on *ad hoc* heuristic methods. The basic ideas of probabilistic data linkage were introduced by Newcombe and Kennedy [30] in 1962, and the theoretical statistical foundation was provided by Fellegi and Sunter [16] in 1969. Similar techniques have independently been developed in the 1970s by computer scientists in the area of document indexing and retrieval [13]. However, until recently few cross-references could be found between the statistical and the computer science community.

As most real-world data collections contain noisy, incomplete and incorrectly formatted information, data cleaning and standardisation are important pre-processing steps for successful deduplication and data linkage, and before data can be loaded into data warehouses or used for further analysis [33]. Data may be recorded or captured in various, possibly obsolete, formats and data items may be missing, out of date, or contain errors. Names and addresses can change over time, and names are often reported differently by the same person depending upon the organisation they are in contact with. Additionally, many proper names have different written forms, for example *'Gail'* and *'Gayle'*. The main tasks of data cleaning and standardisation are the conversion of the raw input data into well defined, consistent forms, and the resolution of inconsistencies [7, 9].

If two data sets **A** and **B** are to be linked, the number of possible record pairs equals the product of the size of the two data sets $|\mathbf{A}| \times |\mathbf{B}|$. Similarly, when deduplicating a data set **A** the number of possible record pairs is $|\mathbf{A}| \times (|\mathbf{A}| - 1)/2$. The performance bottleneck in a data linkage or deduplication system is usually the expensive detailed comparison of fields (or attributes) between pairs of records [1], making it unfeasible to compare all record pairs when the data sets are large. For example, linking two data sets with $100,000$ records each would result in ten billion possible record pair comparisons. On the other hand, the maximum number of truly matched record pairs that are possible corresponds to the number of records in the smaller data set (assuming a record can only be linked to one other record). For deduplication, the number of duplicate records will be smaller than the number of records in the data set. The number of potential matches increases linearly when linking larger data sets, while the computational efforts increase quadratically.

To reduce the large number of possible record pair comparisons, data linkage systems therefore employ blocking [1, 16, 37], sorting [22], filtering [20], clustering [27], or indexing [1, 5] techniques. Collectively known as *blocking*, these techniques aim at cheaply removing pairs of records that are obviously not matches. It is important, however, that no potential match is removed by blocking.

All record pairs produced in the blocking process are compared using a variety of field (or attribute) comparison functions, each applied to one or a combination of record attributes. These functions can be as simple as an exact string or a numerical comparison, can take into account typographical errors, or be as complex as a distance comparison based on look-up tables of geographic locations (longitude and latitude). Each comparison returns a numerical value, often positive for agreeing values and negative for disagreeing values. For each compared record pair a *weight vector* is formed containing all the values calculated by the different field comparison functions. These weight vectors are then used to classify record pairs into *matches*, *non-matches*, and *possible matches* (depending upon the decision model used). In the following sections the various techniques employed for data linkage are discussed in more detail.

2.1 Deterministic Linkage

Deterministic linkage techniques can be applied if unique entity identifiers (or keys) are available in all the data sets to be linked, or a combination of attributes can be used to create a *linkage key*, which is then used to match records that have the same key value. Such linkage systems can be developed based on standard *SQL* queries. However, they only achieve good linkage results if the entity identifiers or linkage keys are of high quality. This means they have to be precise, stable over time, highly available, and robust with regard to errors (for example, include a check digit for detecting invalid or corrupted values).

Alternatively, a set of (often very complex) rules can be used to classify pairs of records. Such rule-based systems can be more flexible than using a simple linkage key, but their development is labour intensive and highly dependent upon the data sets to be linked. The person or team developing such rules not only needs to be proficient with the rule system, but also with the data to be deduplicated or linked. In practise, therefore, deterministic rule based systems are limited to ad-hoc linkages of smaller data sets. In a recent study [19], an iterative deterministic linkage system was compared with the commercial probabilistic system *AutoMatch* [25], and empirical results showed that the probabilistic approach achieved better linkages.

2.2 Probabilistic Linkage

As common unique entity identifiers are rarely available in all data sets to be linked, the linkage process must be based on the existing common attributes. These normally include person identifiers (like names and dates of birth), demographic information (like addresses) and other data specific information (like medical details, or customer information). These attributes can contain typographical errors, they can be coded differently, and parts can be out-of-date or even be missing.

In the traditional probabilistic linkage approach [16, 37], pairs of records are classified as matches if their common attributes predominantly agree, or as non-matches if they predominantly disagree. If two data sets **A** and **B** are to be linked, the set of record pairs $\mathbf{A} \times \mathbf{B} = \{(a, b);\ a \varepsilon \mathbf{A},\ b \varepsilon \mathbf{B}\}$ is the union of the two disjoint sets of true matches M and true non-matches U.

$$M = \{(a, b);\ a = b,\ a \varepsilon \mathbf{A},\ b \varepsilon \mathbf{B}\} \tag{1}$$

$$U = \{(a, b);\ a \neq b,\ a \varepsilon \mathbf{A},\ b \varepsilon \mathbf{B}\} \tag{2}$$

Fellegi and Sunter [16] considered ratios of probabilities of the form

$$R = \frac{P(\gamma \ \varepsilon \ \mathit{\Gamma} | M)}{P(\gamma \ \varepsilon \ \mathit{\Gamma} | U)}, \tag{3}$$

where γ is an arbitrary agreement pattern in a comparison space $\mathit{\Gamma}$. For example, $\mathit{\Gamma}$ might consist of six patterns representing simple agreement or disagreement on given name, surname, date of birth, street address, suburb and postcode.

Alternatively, some of the γ might additionally consider typographical errors, or account for the relative frequency with which specific values occur. For example, a surname value *'Miller'* is much more common in many western countries than a value *'Dijkstra'*, resulting in a smaller agreement value. The ratio R, or any monotonically increasing function of it (such as its logarithm) is referred to as a *matching weight.* A decision rule is then given by

if $R > t_{upper}$, then	designate a record pair as *match*,
if $t_{lower} \leq R \leq t_{upper}$, then	designate a record pair as *possible match*,
if $R < t_{lower}$, then	designate a record pair as *non-match.*

The thresholds t_{lower} and t_{upper} are determined by a-priori error bounds on false matches and false non-matches. If $\gamma \; \varepsilon \; \Gamma$ for a certain record pair mainly consists of agreements, then the ratio R would be large and thus the pair would more likely be designated as a match. On the other hand for a $\gamma \; \varepsilon \; \Gamma$ that primarily consists of disagreements the ratio R would be small.

The class of possible matches are those record pairs for which human oversight, also known as *clerical review*, is needed to decide their final linkage status. While in the past (when smaller data sets were linked, for example for epidemiological survey studies) clerical review was practically manageable in a reasonable amount of time, linking today's large data collections – with millions of records – make this process impossible, as tens or even hundreds of thousands of record pairs will be put aside for review. Clearly, what is needed are more accurate and automated decision models that will reduce – or even eliminate – the amount of clerical review needed, while keeping a high linkage quality. Such approaches are presented in the following section.

2.3 Modern Approaches

Improvements [38] upon the classical probabilistic linkage [16] approach include the application of the expectation-maximisation (EM) algorithm for improved parameter estimation [39], the use of approximate string comparisons [32] to calculate partial agreement weights when attribute values have typographical errors, and the application of Bayesian networks [40].

In recent years, researchers have also started to explore the use of techniques originating in machine learning, data mining, information retrieval and database research to improve the linkage process. Most of these approaches are based on supervised learning techniques and assume that training data (i.e. record pairs with known deduplication or linkage status) is available.

One approach based on ideas from information retrieval is to represent records as document vectors and compute the *cosine distance* [10] between such vectors. Another possibility is to use an *SQL* like language [17] that allows approximate joins and cluster building of similar records, as well as decision functions that decide if two records represent the same entity. A generic knowledge-based framework based on rules and an expert system is presented in [24], and a hybrid system which utilises both unsupervised and supervised machine learning techniques is described in [14]. That paper also introduces metrics for determining the quality of these techniques. The authors find that machine learning outperforms probabilistic techniques, and provides a lower proportion of possible matches.

The authors of [35] apply active learning to the problem of lack of training instances in real-world data. Their system presents a representative (difficult to classify) example to a user for manual classification. They report that manually classifying less than 100 training examples provided better results than a fully supervised approach that used 7,000 randomly selected examples. A similar approach is presented in [36], where a committee of decision trees is used to learn mapping rules (i.e. rules describing linkages).

High-dimensional overlapping clustering (as alternative to traditional blocking) is used by [27] in order to reduce the number of record pair comparisons to be made, while [21] explore the use of simple k-means clustering together with a user tunable fuzzy region for the class of possible matches. Methods based on nearest neighbours are explored by [6], with the idea to capture local structural properties instead of a single global distance approach. An unsupervised approach based on graphical models [34] aims to use the structural information available in the data to build hierarchical probabilistic models. Results which are better than the ones achieved by supervised techniques are presented.

Another approach is to train distance measures used for approximate string comparisons. [3] presents a framework for improving duplicate detection using trainable measures of textual similarity. The authors argue that both at the character and word level there are differences in importance of certain character or word modifications, and accurate similarity computations require adapting string similarity metrics for all attributes in a data set with respect to the particular data domain. Related approaches are presented in [5, 12, 29, 41], with [29] using support vector machines for the binary classification task of record pairs. As shown in [12], combining different learned string comparison methods can result in improved linkage classification. An overview of other methods – including statistical outlier identification, pattern matching, and association rules based approaches – is given in [26].

3 Notation and Problem Analysis

The notation used in this paper is presented here. It follows the traditional data linkage literature [16, 37, 38]. The number of elements in a set $\mathbf{X}$ is denoted $|\mathbf{X}|$. A general linkage situation is assumed, where the aim is to link two sets of entities. For example, the first set could be patients of a hospital, and the second set people who had a car accident. Some of the car accidents resulted in people being admitted into the hospital, some did not. The two sets of entities are denoted as $\mathbf{A}_e$ and $\mathbf{B}_e$. $\mathbf{M}_e = \mathbf{A}_e \cap \mathbf{B}_e$ is the intersection set of matched entities that appear in both $\mathbf{A}_e$ and $\mathbf{B}_e$, and $\mathbf{U}_e = (\mathbf{A}_e \cup \mathbf{B}_e) \setminus \mathbf{M}_e$ is the set of non-matched entities that appear in either $\mathbf{A}_e$ or $\mathbf{B}_e$, but not in both. This space of entities is illustrated in Figure 1, and called the *entity space*.

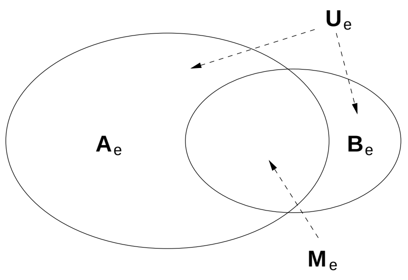


Fig. 1. General linkage situation with two sets of entities $\mathbf{A}_e$ and $\mathbf{B}_e$, their intersection $\mathbf{M}_e$ (the entities that appear in both sets), and the set $\mathbf{U}_e$ which contains the entities that appear in either $\mathbf{A}_e$ or $\mathbf{B}_e$, but not in both

The maximum possible number of matched entities corresponds to the size of the smaller set of $\mathbf{A}_e$ or $\mathbf{B}_e$. This is the situation when the smaller set is a proper subset of the larger one, which also results in the minimum number of non-matched entities. The minimum number of matched entities is zero, which is the situation when no entities appear in both sets. The maximum number of non-matched entities in this situation corresponds to the sum of the entities in both sets. The following equations show this in a more formal way.

$$0 \leq |\mathbf{M}_e| \leq min(|\mathbf{A}_e|, |\mathbf{B}_e|) \quad (4)$$

$$abs(|\mathbf{A}_e| - |\mathbf{B}_e|) \leq |\mathbf{U}_e| \leq |\mathbf{A}_e| + |\mathbf{B}_e| \quad (5)$$

In a simple example, assume the set $\mathbf{A}_e$ contains 5 million entities (e.g. hospital patients), and set $\mathbf{B}_e$ contains 1 million entities (e.g. people involved in car accidents), with 700,000 entities present in both sets (i.e. $|\mathbf{M}_e| = 700,000$). The number of non-matched entities in this situation is $|\mathbf{U}_e| = 4,600,000$, which is the sum of the entities in both sets (6 millions) minus twice the number of matched entities (as they appear in both sets $\mathbf{A}_e$ and $\mathbf{B}_e$). This simple example will be used as a running example in the discussion below.

Records for the entities in $\mathbf{A}_e$ and $\mathbf{B}_e$ are now stored in two data sets (or databases or files), denoted by $\mathbf{A}$ and $\mathbf{B}$, such that there is exactly one record in $\mathbf{A}$ for each entity in $\mathbf{A}_e$ (i.e. the data set contains no duplicate records), and each record in $\mathbf{A}$ corresponds to an entity in $\mathbf{A}_e$. The same holds for $\mathbf{B}_e$ and $\mathbf{B}$. The aim of a data linkage process is to classify pairs of records as matches or non-matches in the product space $\mathbf{A} \times \mathbf{B} = M \cup U$ of true matches M and true non-matches U [16, 37] as given in Equations 1 and 2.

It is assumed that no blocking (as discussed in Section 2) is applied, and that all possible pairs of records are compared. The total number of comparisons equals $|\mathbf{A}| \times |\mathbf{B}|$, which is much larger than the number of entities available in $\mathbf{A}_e$ and $\mathbf{B}_e$ together. In case of the deduplication of a single data set $\mathbf{A}$, the number of record pair comparisons equals $|\mathbf{A}| \times (|\mathbf{A}| - 1)/2$, as each record in the data set must be compared with all others, but not to itself. The space of record pair comparisons is illustrated in Figure 2 and called the *comparison space*.

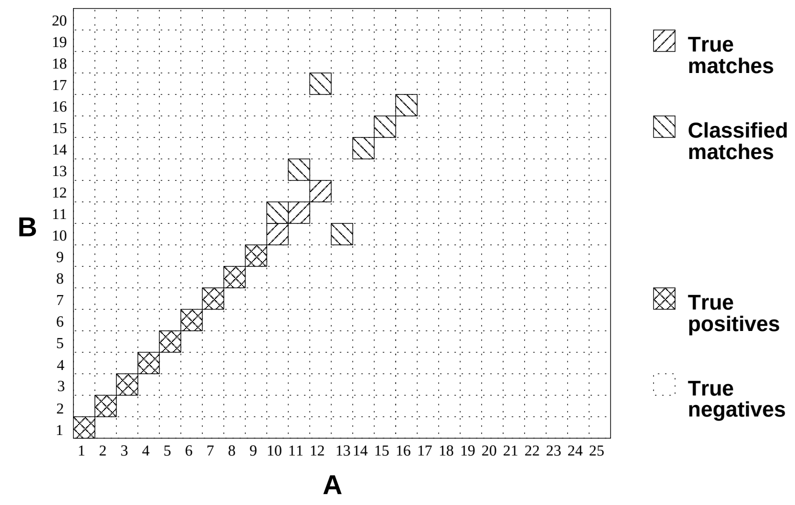


Fig. 2. General record pair comparison space with 25 records in data set **A** arbitrarily numbered on the horizontal axis and 20 records in data set **B** arbitrarily numbered on the vertical axis. The full rectangular area corresponds to all possible record pair comparisons. Assume that record pairs $(A1, B1)$, $(A2, B2)$ up to $(A12, B12)$ are true matches. The linkage algorithm has wrongly classified $(A10, B11)$, $(A11, B13)$, $(A12, B17)$, $(A13, B10)$, $(A14, B14)$, $(A15, B15)$, and $(A16, B16)$ as matches (false positives), but missed $(A10, B10)$, $(A11, B11)$, and $(A12, B12)$ (false negatives)

For the simple example given earlier, the comparison space consists of $|\mathbf{A}| \times |\mathbf{B}| = 5,000,000 \times 1,000,000 = 5 \times 10^{12}$ record pairs, with $|M| = 700,000$ and $|U| = 5 \times 10^{12} - 700,000 = 4.9999993 \times 10^{12}$ record pairs.

A linkage algorithm compares pairs of records and classifies them into $\tilde{M}$ (record pairs considered to be a match by the algorithm) and $\tilde{U}$ (record pairs considered to be a non-match). To keep this analysis simple, it is assumed here that the linkage algorithm does not classify record pairs as possible matches (as discussed in Section 2.2). Both records of a truly matched pair correspond to the same entity in $\mathbf{M}_e$. Un-matched record pairs, on the other hand, correspond to different entities in $\mathbf{A}_e$ and $\mathbf{B}_e$, with the possibility of both records of such a pair corresponding to different entities in $\mathbf{M}_e$. As each record relates to exactly one entity, and there are no duplicates in the data sets, a record in **A** can only be correctly matched to a maximum of one record in **B**, and vice versa. For each record pair, the binary classification into $\tilde{M}$ and $\tilde{U}$ results in one of four possible outcomes [15] as shown in Table 1. As can be seen, $M = TP + FN$, $U = TN + FP$, $\tilde{M} = TP + FP$, and $\tilde{U} = TN + FN$.

When assessing the quality of a linkage algorithm, the general interest is in how many truly matched entities and how many truly non-matched entities have been classified correctly as matches and non-matches, respectively. However, the outcome of the classification is measured in the comparison space (as number

Table 1. Confusion matrix of record pair classification

Actual	Classification	
	Match ($\tilde{M}$)	Non-match ($\tilde{U}$)
Match (M)	True match True positive (TP)	False non-match False negative (FN)
Non-match (U)	False match False positive (FP)	True non-match True negative (TN)

of classified record pairs). While the number of truly matched record pairs is the same as the number of truly matched entities, $|M| = |\mathbf{M}_e|$ (as each truly matched record pair corresponds to one entity), there is however no correspondence between the number of truly non-matched record pairs and non-matched entities. Each non-matched record pair contains two records that correspond to two different entities, and so it not possible to easily calculate a number of non-matched entities.

The maximum number of truly matched entities is given by Equation 4. From this follows the maximum number of record pairs a linkage algorithm should classify as matches is $|\tilde{M}| \leq |\mathbf{M}_e| \leq min(|\mathbf{A}_e|, |\mathbf{B}_e|)$. As the number of classified matches $\tilde{M} = TP + FP$, it follows that $|TP + FP| \leq |\mathbf{M}_e|$. And with $M = TP + FN$, it also follows that both the numbers of FP and FN will be small compared to the number of TN, and they will not be influenced by the multiplicative increase between the entity and the comparison space. The number of TN will dominate, however, as, in the comparison space, the following equation holds:

$$|TN| = |\mathbf{A}| \times |\mathbf{B}| - |TP| - |FN| - |FP|. \tag{6}$$

This is also illustrated in Figure 2. Therefore, any quality measure used in deduplication or data linkage that uses the number of TN will give deceptive results, as will be illustrated and discussed further in Sections 4 and 5.

The above discussion assumes no duplicates in the data sets **A** and **B**. Thus, a record in one data set can only be matched to a maximum of one record in the other data set (often called *one-to-one* assignment restriction). In practise, however, *one-to-many* and *many-to-many* linkages or deduplications are possible. Examples include longitudinal studies of administrative health data, where several records might correspond to a certain patient over time, or business mailing lists where several records can relate to the same customer (this happens when data sets have not been properly deduplicated). While the above analysis would become more complicated, the issue of having a very large number of TN stills hold in one-to-many and many-to-many linkage situations, as the number of matches for a single record will be small compared to the full number of record pair comparisons.

Table 2. Quality measures used in recent deduplication and data linkage publications

Measure	Formula / Description	Used in
Accuracy	$acc = \frac{TP+TN}{TP+FP+TN+FN}$	[21, 35, 36]
Precision	$prec = \frac{TP}{TP+FP}$	[1, 2, 10, 11, 14, 27]
Recall	$rec = \frac{TP}{TP+FN}$	[1, 11, 14, 21, 27]
F-measure	$f{-}measure = 2(\frac{prec \times rec}{prec+rec})$	[1, 11, 27]
False positive rate	$fpr = \frac{FP}{TN+FP}$	[2]
Precision-Recall graph	Plot precision on vertical and recall on horizontal axis	[3, 6, 28]

4 Quality Measures

Given that deduplication and data linkage are classification problems, various quality measures are available to the data linkage researcher and practitioner [15]. With many recent approaches being based on supervised learning, no clerical review process (i.e. no possible matches) is often assumed and the problem becomes a binary classification, with record pairs being classified as either matches or non-matches, as shown in Table 1. A summary of the quality measures used in recent publications is given in Table 2 (a more detailed discussion can be found in [8]).

As presented in Section 2.2, a linkage algorithm is assumed to have a threshold parameter t (with no possible matches $t_{lower} = t_{upper}$), which determines the cut-off between classifying record pairs as matches (with matching weight $R \geq t$) or as non-matches ($R < t$). Increasing the value of t results in an increased number of TN and FP and in a reduction in the number of TP and FN, while lowering t reduces the number of TN and FP and increases the number of TP and FN. Most of the quality measures presented here can be calculated for different values of such a threshold (often only the quality measure values for an optimal threshold are reported in empirical studies). Alternatively, quality measures can be visualised in a graph over a range of threshold values, as illustrated by the examples in Section 5.

Taking the example from Section 3, assume that for a given threshold a linkage algorithm has classified $|\tilde{M}| = 900,000$ record pairs as matches and the rest ($|\tilde{U}| = 5 \times 10^{12} - 900,000$) as non-matches. Of these $900,000$ classified matches $650,000$ were true matches (TP), and $250,000$ were false matches (FP). The number of false non-matched record pairs (FN) was $50,000$, and the number of true non-matched record pairs (TN) was $5 \times 10^{12} - 950,000$. When looking at the entity space, the number of non-matched entities is $4,600,000 - 250,000 = 4,350,000$. Table 3 shows the resulting quality measures for this example in both the comparison and the entity spaces, and as discussed, any measure that includes the number of TN depends upon whether entities or record pairs are counted. As can be seen, the results for accuracy and the false positive rate

Table 3. Quality results for the simple example

Measure	Entity space	Comparison space
Accuracy	94.340%	99.999994%
Precision	72.222%	72.222000%
Recall	92.857%	92.857000%
F-measure	81.250%	81.250000%
False positive rate	5.435%	0.000005%

all show misleading results when based on record pairs (i.e. measured in the comparison space). This issue will be illustrated further in Sections 5 and 6.

The authors of [4] discuss the topic of evaluating deduplication and data linkage systems. They advocate the use of precision-recall graphs over the use of single value measures like accuracy or maximum F-measure, on the grounds that such single value measures assume that an optimal threshold has been found. A single value can also hide the fact that one classifier might perform better for lower threshold values, while another better for higher thresholds.

5 Experimental Examples

In this section the previously discussed issues on quality measures are illustrated using a real-world administrative health data set, the *New South Wales Midwives Data Collection* (MDC) [31]. 175,211 records from the years 1999 and 2000 were extracted, containing names, addresses and dates of birth of mothers giving birth in these two years. This data set has previously been deduplicated (and manually clerically reviewed) using the commercial probabilistic data linkage system *AutoMatch* [25]. According to this deduplication, the data set contains 166,555 unique mothers, with 158,081 having one, 8,295 having two, 176 having three, and 3 having four records (births). The *AutoMatch* deduplication decision was used as the true match (or deduplication) status for this example

A deduplication was then performed using the *Febrl* (Freely extensible biomedical record linkage) [7] data linkage system. Fourteen attributes in the MDC were compared using various comparison functions (like exact and approximate string comparisons), and the resulting comparison values were summed into a matching weight (as discussed in Section 2.2) ranging from -43 (disagreement on all fourteen comparisons) to 115 (agreement on all comparisons). As can be seen in the density plot in Figure 3, almost all true matches (record pairs classified as true duplicates) have positive matching weights, while the majority of non-matches have negative weights. There are, however, non-matches with rather large positive matching weights, which is due to the differences in calculating the weights between *AutoMatch* and *Febrl*.

The full comparison space for this data set with 175,211 records would result in $175,211 \times 175,210/2 = 15,349,359,655$ record pairs, which is infeasible

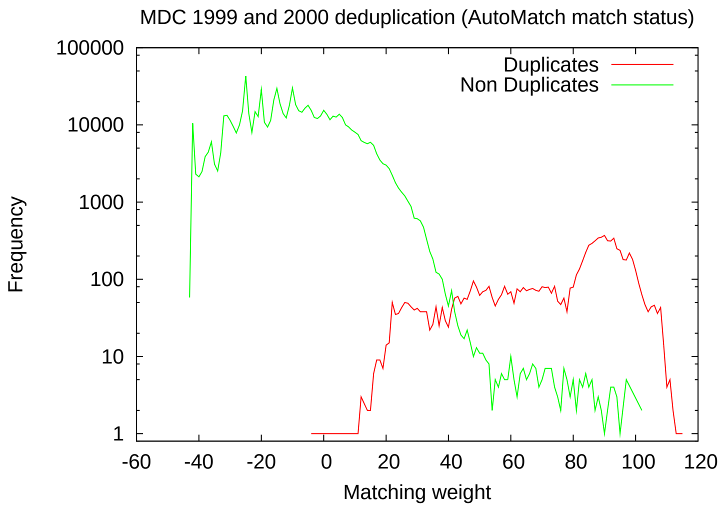


Fig. 3. The density plot of the matching weights for a real-world administrative health data set. This plot is based on record pair comparison weights in a blocked comparison space. The lowest weight is -43 (disagreement on all comparisons), and the highest 115 (agreement on all comparisons). Note that the vertical axis with frequency counts is on a logarithmic scale

to process even with today's powerful computers. Standard blocking was used to reduce the number of comparisons, resulting in $759,773$ record pairs (this corresponds to only around 0.005% of all record pairs in the full comparison space). The total number of truly classified matches (duplicates) was $8,841$ (for all the duplicates as described above), with $8,808$ of the $759,773$ record pairs in the blocked comparison space corresponding to true duplicates (thus, 33 true matches were removed by blocking).

The quality measures discussed in Section 4 applied to this real-world deduplication procedure are shown in Figure 4 for a varying threshold $-43 \leq t \leq 115$. The aim of this figure is to illustrate how the different measures look for a deduplication example taken from the real world. The measurements were done in the blocked comparisons space as described above. The full comparison space ($15,349,359,655$ record pairs) was simulated by assuming that blocking removed mainly record pairs with negative comparison weights (normally distributed between -43 and -10). As discussed previously, this resulted in different numbers of TN between the blocked and the (simulated) full comparison spaces. As can be seen, the precision-recall graph is not affected by the blocking process, and the F-measure differs only slightly. The two other measures, however, resulted in graphs of different shape.

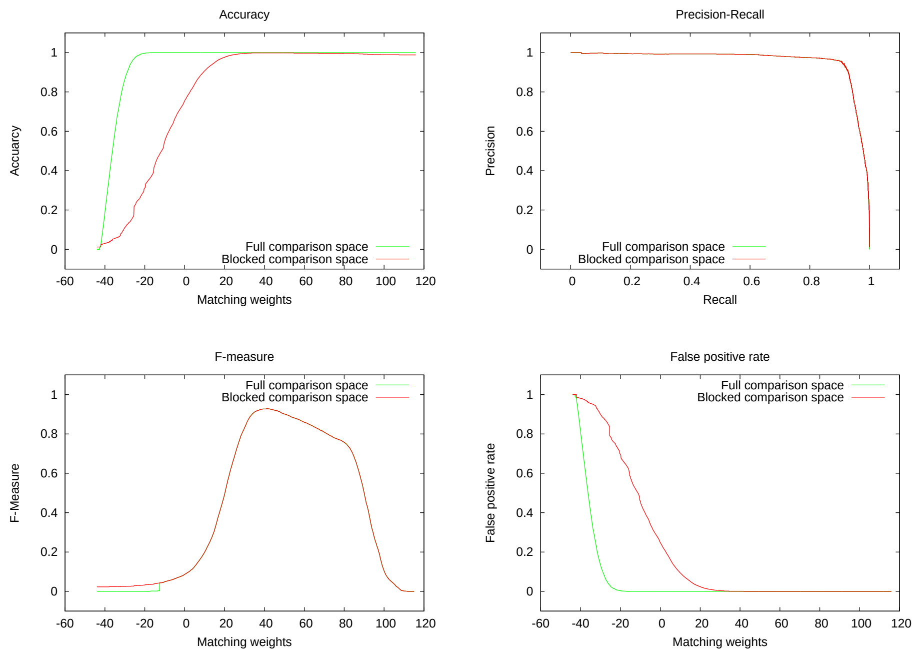


Fig. 4. Quality measurements of a real-world administrative health data set

6 Recommendations

Based on the above discussions, several recommendations for measuring deduplication and data linkage quality can be given. Their aim is to provide both researchers and practitioners with guidelines on how to perform empirical studies on different algorithms, or production deduplication or linkage projects, as well as on how to properly assess and describe the outcome of such linkages.

Record Pair Classification Due to the problem of the number of true negatives in any comparison, quality measures which use that number (for example accuracy or the false positive rate) should not be used. The variation in the quality of a technique against particular types of data means that results should be reported for particular data sets. Also, given that the nature of some data sets may not be known in advance, the average quality across all data sets used in a certain study should be reported. When comparing techniques, precision-recall or F-measure graphs provide an additional dimension to the results. For example, if a small number of highly accurate links is required, the technique with higher precision for low recall would be chosen [4].

Blocking The aim of blocking is to cheaply remove obvious non-matches before the more detailed, expensive record pair comparisons are made. Working perfectly, blocking would only remove record pairs that are true non-matches, thus affecting the number of true negatives, and possibly the number of false positives. To the extent that, in reality, blocking also removes record pairs from the set of true matches, it will also affect the number of true positives and false negatives. Blocking can thus be seen to be a *confounding* factor in quality measurement – the types of blocking procedures and the parameters chosen will potentially affect the results obtained for a given linkage procedure. If computationally feasible, for example in an empirical study using small data sets, it is strongly recommended that all quality measurement results be obtained without the use of blocking. It is recognised that it may not be possible to do this with larger data sets. A compromise would be to publish the blocking approach and resulting number of removed pairs of records, and to make the *blocked* data set available for analysis and comparison by other researchers. At the very least, the blocking procedure and parameters should be specified in a form that can enable other researchers to repeat it.[1]

7 Conclusions

Deduplication and data linkage are important tasks in the pre-processing step of many data mining projects, and also important for improving data quality before data is loaded into data warehouses. An overview of data linkage techniques has been presented, and the issues involved in measuring the quality of deduplication and data linkage algorithms have been discussed. It is recommended that data linkage quality be measured using the precision-recall or F-measure graphs rather than single numerical values, and measures that include the number of true negative matches should not be used due to their large number in the space of record pair comparisons. When publishing empirical studies, researchers should aim to use non-blocked data sets if possible, or otherwise at least detail the blocking approach taken, and report on the number of record pairs being removed by the blocking process.

Acknowledgements

This work is supported by an Australian Research Council (ARC) Linkage Grant LP0453463 and partially funded by the NSW Department of Health. The authors would like to thank Markus Hegland for insightful discussions.

[1] It is acknowledged that the example given in Section 5 doesn't follow the recommendations presented here. It's aim is only to illustrate the presented issues, not the actual results of this deduplication.

References

1. Baxter, R., Christen, P. and Churches, T.: A Comparison of Fast Blocking Methods for Record Linkage. ACM SIGKDD '03 Workshop on Data Cleaning, Record Linkage, and Object Consolidation, August 27, 2003, Washington, DC, pp. 25-27.
2. Bertolazzi, P., De Santis, L. and Scannapieco, M.: Automated record matching in cooperative information systems. Proceedings of the international workshop on data quality in cooperative information systems, Siena, Italy, January 2003.
3. Bilenko, M. and Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. Proceedings of the 9th ACM SIGKDD conference, Washington DC, August 2003.
4. Bilenko, M. and Mooney, R.J.: On evaluation and training-set construction for duplicate detection. Proceedings of the KDD-2003 workshop on data cleaning, record linkage, and object consolidation, Washington DC, August 2003.
5. Chaudhuri, S., Ganjam, K., Ganti, V. and Motwani, R.: Robust and efficient fuzzy match for online data cleaning. Proceedings of the 2003 ACM SIGMOD International Conference on on Management of Data, San Diego, USA, 2003, pp. 313-324.
6. Chaudhuri, S., Ganti, V. and Motwani, R.: Robust identification of fuzzy duplicates. Proceedings of the 21st international conference on data engineering, Tokyo, April 2005.
7. Christen, P., Churches, T. and Hegland, M.: Febrl – A parallel open source data linkage system. Proceedings of the 8th PAKDD, Sydney, Springer LNAI 3056, May 2004.
8. Christen, P. and Goiser, K.: Quality and Complexity Measures for Data Linkage and Deduplication. Accepted for Quality Measures in Data Mining, Springer, 2006.
9. Churches, T., Christen, P., Lim, K. and Zhu, J.X.: Preparation of name and address data for record linkage using hidden Markov models. BioMed Central Medical Informatics and Decision Making, Dec. 2002.
10. Cohen, W.W.: Integration of heterogeneous databases without common domains using queries based on textual similarity. Proceedings of SIGMOD, Seattle, 1998.
11. Cohen, W.W. and Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. Proceedings of the 8th ACM SIGKDD conference, Edmonton, July 2002.
12. Cohen, W.W., Ravikumar, P. and Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. Proceedings of IJCAI-03 workshop on information integration on the Web (IIWeb-03), pp. 73–78, Acapulco, August 2003.
13. Cooper, W.S. and Maron, M.E.: Foundations of Probabilistic and Utility-Theoretic Indexing. Journal of the ACM , vol. 25, no. 1, pp. 67–80, January 1978.
14. Elfeky, M.G., Verykios, V.S. and Elmagarmid, A.K.: TAILOR: A record linkage toolbox. Proceedings of the ICDE' 2002, San Jose, USA, March 2002.
15. Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Researchers, HP Labs Tech Report HPL-2003-4, HP Laboratories, Palo Alto, March 2004.
16. Fellegi, I. and Sunter, A.: A theory for record linkage. Journal of the American Statistical Society, December 1969.
17. Galhardas, H., Florescu, D., Shasha, D. and Simon, E.: An Extensible Framework for Data Cleaning. Proceedings of the Inter. Conference on Data Engineering, 2000.
18. Gill, L.: Methods for Automatic Record Matching and Linking and their use in National Statistics. National Statistics Methodology Series No. 25, London, 2001.
19. Gomatam, S., Carter, R., Ariet, M. and Mitchell G.: An empirical comparison of record linkage procedures. Statistics in Medicine, vol. 21, no. 10, May 2002.

20. Gu, L. and Baxter, R.: Adaptive filtering for efficient record linkage. SIAM international conference on data mining, Orlando, Florida, April 2004.
21. Gu, L. and Baxter, R.: Decision models for record linkage. Proceedings of the 3rd Australasian data mining conference, pp. 241–254, Cairns, December 2004.
22. Hernandez, M.A. and Stolfo, S.J.: Real-world data is dirty: Data cleansing and the merge/purge problem. In Data Mining and Knowledge Discovery 2, Kluwer Academic Publishers, 1998.
23. Kelman, C.W., Bass, A.J. and Holman, C.D.: Research use of linked health data - A best practice protocol. Aust NZ Journal of Public Health, 26:251-255, 2002.
24. Lee, M.L., Ling, T.W. and Low, W.L.: IntelliClean: a knowledge-based intelligent data cleaner. Proceedings of the 6th ACM SIGKDD conference, Boston, 2000.
25. *AutoStan and AutoMatch, User's Manuals*, MatchWare Technologies, 1998.
26. Maletic, J.I. and Marcus, A.: Data Cleansing: Beyond Integrity Analysis. Proceedings of the Conference on Information Quality (IQ2000), Boston, October 2000.
27. McCallum, A., Nigam, K. and Ungar, L.H.: Efficient clustering of high-dimensional data sets with application to reference matching. Proceedings of the 6th ACM SIGKDD conference, pp. 169–178, Boston, August 2000.
28. Monge, A. and Elkan, C.: The field-matching problem: Algorithm and applications. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, August 1996.
29. Nahm, U.Y, Bilenko M. and Mooney, R.J.: Two approaches to handling noisy variation in text mining. Proceedings of the ICML-2002 workshop on text learning (TextML'2002), pp. 18–27, Sydney, Australia, July 2002.
30. Newcombe, H.B. and Kennedy, J.M.: Record Linkage: Making Maximum Use of the Discriminating Power of Identifying Information. Communications of the ACM, vol. 5, no. 11, 1962.
31. Centre for Epidemiology and Research, NSW Department of Health. New South Wales Mothers and Babies 2001. NSW Public Health Bull 2002; 13(S-4).
32. Porter, E. and Winkler, W.E.: Approximate String Comparison and its Effect on an Advanced Record Linkage System. RR 1997-02, US Bureau of the Census, 1997.
33. Rahm, E. and Do, H.H.: Data Cleaning: Problems and Current Approaches. IEEE Data Engineering Bulletin, 2000.
34. Ravikumar, P. and Cohen, W.W.: A hierarchical graphical model for record linkage. Proceedings of the 20th conference on uncertainty in artificial intelligence, Banff, Canada, July 2004.
35. Sarawagi, S. and Bhamidipaty, A.: Interactive deduplication using active learning. Proceedings of the 8th ACM SIGKDD conference, Edmonton, July 2002.
36. Tejada, S., Knoblock, C.A. and Minton, S.: Learning domain-independent string transformation weights for high accuracy object identification. Proceedings of the 8th ACM SIGKDD conference, Edmonton, July 2002.
37. Winkler, W.E. and Thibaudeau, Y: An Application of the Fellegi-Sunter Model of Record Linkage to the 1990 U.S. Decennial Census. RR 1991-09, US Bureau of the Census, 1991.
38. Winkler, W.E.: The State of Record Linkage and Current Research Problems. RR 1999-04, US Bureau of the Census, 1999.
39. Winkler, W.E.: Using the EM algorithm for weight computation in the Fellegi-Sunter model of record linkage. RR 2000-05, US Bureau of the Census, 2000.
40. Winkler, W.E.: Methods for Record Linkage and Bayesian Networks. RR 2002-05, US Bureau of the Census, 2002.
41. Yancey, W.E.: An adaptive string comparator for record linkage RR 2004-02, US Bureau of the Census, February 2004.

Automated Probabilistic Address Standardisation and Verification

http://datamining.anu.edu.au/linkage.html

Peter Christen[*] and Daniel Belacic

Department of Computer Science,
Australian National University,
Canberra ACT 0200, Australia,
{peter.christen,daniel.belacic}@anu.edu.au

Abstract. Addresses are a key part of many records containing information about people and organisations, and it is therefore important that accurate address information is available before such data is mined or stored in data warehouses. Unfortunately, addresses are often captured in non-standard and free-text formats, usually with some degree of spelling and typographical errors. Additionally, addresses change over time, for example when people move, when streets are renamed, or when new suburbs are built. Cleaning and standardising addresses, as well as verifying if they really exist, are therefore important steps in data mining pre-processing. In this paper we present an automated probabilistic approach based on a hidden Markov model (HMM), which uses national address guidelines and a comprehensive national address database to clean, standardise and verify raw input addresses. Initial experiments show that our system can correctly standardise even complex and unusual addresses.

Keywords: Data mining pre-processing, address cleaning and standardisation, hidden Markov model, G-NAF, postal address guidelines.

1 Introduction

Most real world data collections contain noisy, incomplete, incorrectly formatted, or even out-of-date data. Cleaning and standardising such data are therefore important first steps in data pre-processing, and before such data can be stored in data warehouses or used for further data analysis or mining [11, 16]. In most settings it is desirable to be able to detect and remove duplicate records from a data set, in order to reduce costs for business mailings or to improve the accuracy of a data analysis task. The cleaning and standardisation of personal information (like addresses and names) is especially important for data linkage and integration, to make sure that no misleading or redundant information is introduced. Data linkage (also called record linkage) [10] is important in many

[*] Corresponding author

application areas, such as compilation of longitudinal epidemiological studies, census related statistics, or fraud and crime detection systems.

The main tasks of data cleaning [16] are the conversion of the raw input data into well defined, consistent forms, and the resolution of inconsistencies in the way information is represented or encoded. Personal information is often captured and stored with typographical and phonetical variations, parts can be missing or recorded in different (possibly obsolete) formats, or be out-of-order. Addresses and names can change over time, and are often reported differently by the same person depending upon the organisation they are in contact with. Moreover, while for many regular words there is only one correct spelling, there are often different written forms for proper names (which are commonly used as street, locality or institution names), for example *'Dickson'* and *'Dixon'*. For addresses to be useful and valuable, they need to be cleaned and standardised into a well defined format. For example, various abbreviations should be converted into standardised forms, nicknames should be expanded into their full names, and postcodes should be validated using official postcode lists.

In this paper we report on a project that aims to develop techniques for fully automated cleaning, standardisation, as well as verification, of raw input addresses. In Section 2 we introduce the task of address cleaning and standardisation in more detail and present other work that has been done in this area. While traditional approaches have been based on either rules that need to be customised by the user according to her or his data, or manually prepared training data, our system is based on a mainly unsupervised approach. The main contribution of our work is the automated training of a probabilistic address standardisation system using national address guidelines and a comprehensive national address database. We present our approach in Section 3, and discuss the methods used to automatically train our system in Section 4. First experimental results are then presented and discussed in Section 5, and an outlook to future work is given in Section 6.

2 Address Cleaning and Standardisation

The aim of the cleaning and standardisation process is to transform the raw input address records into a well defined and consistent form, as shown in Figure 1. Addresses can be separated into three components, corresponding to the *address site* (containing flat and street number details), *street* (containing street name and type), and *locality* (with locality, state and postcode information). As can be seen from Figure 1, these components are further split into several output fields, each containing a basic piece of information. The standardisation process also replaces different spellings and abbreviations with standard versions. Look-up tables of such standard spellings are often published by national postal services, together with guidelines of how addresses should be written properly on letters or parcels. This information can be used to build an automated address standardiser, as presented in more details in Sections 3 and 4.

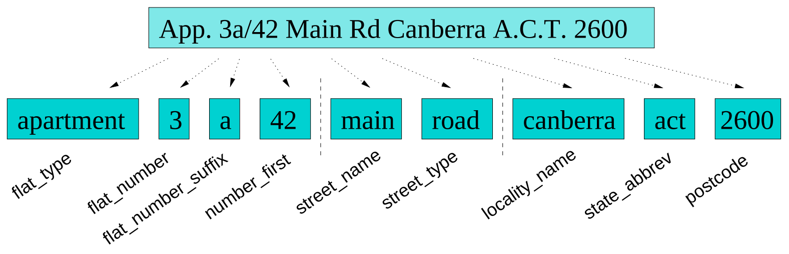


Fig. 1. Example address standardisation. The left four output fields relate to the *address site* level, the middle two to *street level*, and the right three fields to *locality level*

The terms data cleaning (or data cleansing), data standardisation, data scrubbing, data pre-processing, and ETL (extraction, transformation and loading) are used synonymously to refer to the general tasks of transforming source data into clean and consistent sets of records suitable for loading into a data warehouse, or for linking with other data sets. A number of commercial software products are available which address this task. A complete review is beyond the scope of this paper (an overview can be found in [16]). Address (and name) standardisation is also closely related to the more general problem of extracting structured data, such as bibliographic references or name entities, from unstructured or variably structured texts, such as scientific papers or Web pages.

The most common approach for address standardisation is the manual specification of parsing and transformation rules. A well-known example of this approach in biomedical research is *AutoStan* [12], the companion product to the widely-used *AutoMatch* probabilistic record linkage software. *AutoStan* first parses the input string into individual words, and using a re-entrant regular expression parser each word is then mapped to a token of a particular class (determined by the presence of that word in user-supplied look-up tables, or by the type of characters found in the word). This approach requires both an initial and ongoing investment in rule programming by skilled staff. More recent rule-based approaches, which aim at automatically induce rules for information extraction from unstructured text, include *Rapier* [5], which is based on inductive logic programming; *Whisk* [18], which can handle both free and highly structured text; and *Nodose* [1], which is an interactive graphical tool for determining the structure of text documents and for extracting their data.

An alternative to these rule-based, deterministic approaches are probabilistic methods. Statistical models, especially hidden Markov models (HMMs), have widely been used in the areas of speech recognition and natural language processing to help solve problems such as word-sense disambiguation and part-of-speech tagging [15]. More recently, HMMs and related models have been applied to the problem of extracting structured information from unstructured text. An approach using HMMs to find names and other non-recursive entities in free text is described in [3], where word features are used similar to the ones implemented in our system, and experimental results of high accuracy are presented using both English and Spanish test data. HMMs are also used for information extraction by [9], which addresses the problem of lack of training data by applying the statistical techniques of *shrinkage* to improve HMM parameter estimations (different hierarchies of expected similarities are built from a model). The issue of learning the structure of HMMs for information extraction is discussed in [17], where both labelled and un-labelled data is used, and good accuracy results are presented. A supervised approach for segmenting text (including US and Indian addresses) is presented by [4]. Their system *Datamold* uses hierarchical features and nested HMMs, and does allow the integration of external hierarchical databases for improved segmentation. Their results indicate that *Datamold* consistently performs better than the rule-base system *Rapier*. An automatic system that only uses external databases is presented in [2]. The authors describe *attribute recognition models* (ARMs), based on HMMs, which capture the characteristics of the values stored in large reference tables. The topology for an ARM consists of the three states *Beginning*, *Middle*, and *Trailing*. Feature hierarchies are then used to learn the HMM topology as well as transition and emission probabilities. Results presented on various data sets show an up to 50% reduction in segmentation errors compared to *Datamold*.

Earlier work [8] by one of the authors of this paper describes a supervised name and address standardisation approach that uses a lexicon-based tokenisation in combination with HMMs, work that was strongly influenced by [4]. Instead of directly using the elements of the input records for HMM segmentation, a tagging step allocates one or more tags (based on user definable look-up tables and some hard coded rules) to each input element, and sequences of tags are then given to a previously trained (using manually prepared tag sequences) HMM. Results on real world administrative health data showed better accuracy than the rule-based system *AutoStan* for addresses [8]. Training of this system is facilitated by a boot-strapping approach, allowing a reasonable amount of training data to be manually created within a couple of hours.

In this paper we present work which is mainly based on [2] and [8]. The main contribution of our work is the combination of techniques used in these two approaches, with specific application (but not limited) to Australian postal addresses. We use national address guidelines and a large national address database to automatically train a HMM, without the need of any manual preparation of training data. Our system is part of a free, open source data linkage system known as *Febrl* (Freely extensible biomedical record linkage) [6], which is written in the free, open source object-oriented programming language *Python*.

3 Probabilistic Address Standardisation

Our method is based on a probabilistic HMM which is automatically trained using information taken from national address guidelines (which are available in many countries) as well as a comprehensive national address database. The detailed approach on how this HMM is trained using these two sources is discussed in Section 4. Here we present the actual steps involved in the standardisation of raw input addresses, assuming such a trained HMM is available.

We assume that the raw input address records are stored as text files or database tables, and are made of one or more text strings. The task is then to allocate the words and numbers from the raw input into the appropriate output fields, to clean and standardise the values in these output fields, and to verify if an address (or parts of it) really exist (i.e. is available in the national address database). Our approach is based on the following four steps, which will be discussed in more detail in the four sections given below.

1. The raw input addresses are *cleaned*.
2. They are each split into a list of words, numbers and characters, which are then *tagged* using features and look-up tables that were generated using the national address database.
3. These tagged lists are then *segmented* into output fields using a probabilistic HMM.
4. Finally, the segmented addresses are *verified* using the national address database.

3.1 Cleaning

The cleaning step involves converting all letters into lower case, followed by various general corrections of sub-strings using correction lists. These lists are stored in text files that can be modified by the user. For example, variations of *nursing home*, such as `'n-home'` or `'n/home'` are all replaced with the string `'nursing home'`. Various kinds of brackets and quoting characters are replaced with a vertical bar `'|'`, which facilitates tagging and segmenting in the subsequent steps. Correction lists also allow the definition of strings that are to be removed from the input, for example `'n/a'` or `'locked'`. The output of this first step is a cleaned address string ready to be tagged in the next step.

3.2 Tagging

After an address string has been cleaned, it is split at white-space boundaries into a list of words, numbers, punctuation marks and other possible characters. Each of the list elements is assigned one or more *tags*. These tags are based on look-up tables generated using the values in the national address database, as well as more general *features*. For example, a list element `'road'` is assigned the tag `'ST'` (for street type, as `'road'` was found in the street type attribute in the database), as well as the tag `'L4'` (as it is a value of length four characters containing only letters). The tagging does not depend upon the position of a value in the list. The number `'2371'`, for example, will be tagged with `'PC'` (as it is a known postcode) and `'N4'` (as it is also a four digit number), even if it appears at the beginning of an address (where it likely corresponds to a street number). The segmentation step (described below) then assigns this element to the appropriate output field.

Table 1. Example values from the national address database for features used for standardisation. Empty table entries indicate no such values are available in the database

Length	Numbers	Letters	Alpha-numeric	Others
1	3	a		.
2	42	se	b1	.,
3	127	lot	33a	1/7
4	1642	road	672a	3/1a
5	13576	place	lot12	1/23b
6 to 8	2230229	street	rmb1622	lot 1760
9 to 11		jindabyne	coleville2	anderson's
12 to 15		dondingalong	bundanoon305	house no: 2/41
16 or more		stonequarrycreek		armidale-kempsey

Look-up tags specify to the HMM in which attribute(s) of the national address database a list element appears. If it appears in several attributes, more than one look-up tag will be assigned to it. However, if a list element in an input address contains a typographical error, or does otherwise not exactly correspond to any look-up table value, no tag would be assigned to it. Therefore, the features are a more general way of representing the content of the different attributes in the national address database. Features characterise the lengths of an attribute value, as well as its content (if it is made of letters only, numbers only, if it is alpha-numeric, or if it also contains other characters). For example, an attribute value that only contains letters and has a length between 12 and 15 (feature tag `'L12_15'`) is in 73% a locality name, in 26% a street name, and in 1% a building name, as this is the distribution of values with letters only and a length between 12 and 15 in the national address database. A feature tag `'N6_8'`, as another example, corresponds to a number value with length between 6 and 8 digits. Table 1 gives example attribute values from the national address database.

In the tagging step, the look-up tables are searched using a *greedy* matching algorithm, which searches for the longest tuple of list elements that match an entry in the look-up tables. For example, the tuple `('macquarie','fields')` will be matched with an entry in a look-up table with the locality name `'macquarie fields'`, rather than with the single-word entry `'macquarie'` from the same look-up table.

The output of the tagging step is a list of words, numbers and separators, and a corresponding list of look-up and feature tags (as shown in the example given below). As more than one tag can be assigned to a list element (as in the street type example above), different combinations of tag sequences are possible, and the question is which tag sequence is the most likely one, and how should the list elements be assigned to the appropriate output fields? This problem is solved using a probabilistic HMM in the segmentation step as discussed next.

3.3 Segmenting

Having a list of elements (words, numbers and separators) and one or more corresponding tag lists, the task is to assign these elements to the appropriate output fields. Traditional approaches have used rules (such as *"if an element has a tag 'ST' then the corresponding word is assigned to the 'street_type' output field."*). Instead, we use a HMM [15], which has the advantages of being robustness with respect to previously unseen input sequences, and that it can be automatically trained as will be detailed in Section 4.

Hidden Markov models [15] (HMMs) were developed in the 1960s and 1970s and are widely used in speech and natural language processing. They are a powerful machine learning technique, able to handle new forms of data in a robust fashion. They are computationally efficient to develop and evaluate. Only recently have HMMs been used for address standardisation [4, 8, 17].

A HMM is a probabilistic finite state machine made of a set of states, transition edges between these states and a finite dictionary of discrete observation (output) symbols. Each edge is associated with a transition probability, and each state emits observation symbols from the dictionary with a certain probability distribution. Two special states are the *'Start'* and *'End'* state. Beginning from the *'Start'* state, a HMM generates a sequence of length k of observation symbols $O = o_1, o_2, \ldots, o_k$ by making $k - 1$ transitions from one state to another until the *'End'* state is reached. Observation symbol $o_i, 1 \leq i \leq k$ is generated in state i based on this state's probability distribution of the observation symbols. The same output sequence can be generated by many different paths through a HMM with different probabilities. Given an observation sequence, one is often interested in the most likely path through a given HMM that generated this sequence. This path can effectively be calculated for a given observation sequence using the *Viterbi* [15] algorithm, which is a dynamic programming approach. Figure 3 shows a HMM generated by our system for address standardisation.

Instead of using the original words, numbers and other elements from the address records directly, the tag sequences (as discussed in Section 3.2) are used as HMM observation symbols in order to make the derived HMM more general and more robust. Using tags also limits the size of the observation dictionary. Once a HMM is trained, sequences of tags (one tag per input element) as generated in the tagging step can be given as input to the *Viterbi* algorithm, which returns the most likely path (i.e. state sequence) of the given tag sequence through the HMM, plus the corresponding probability. The path with the highest probability is then taken and the corresponding state sequence will be used to assign the elements of the input list to the appropriate output fields.

Example: Let's assume we have the following (randomly created) input address `'42 meyer Rd COOMA 2371'`, which is cleaned and tagged (using both look-up and feature tags) into the following word list and tag sequence:

```
['42', 'meyer', 'road',  'cooma',     '2371' ]
['N2', 'SN/L5', 'ST/L4', 'LN/SN/L5', 'PC/N4' ]
```

with look-up tags `'SN'` for street name, `'ST'` for street type, `'LN'` for locality name, and `'PC'` for postcode; and feature tags for numbers (`'N2'` and `'N4'`) and letter values (`'L4'` and `'L5'`). The number of combinations of the tag sequences is $1 \times 2 \times 2 \times 3 \times 2 = 24$, for example `['N2', 'SN', 'ST', 'LN', 'PC']` or `['N2', 'L5', 'ST', 'SN', 'N4']`. These 24 tag sequences are given to the *Viterbi* algorithm, and using the HMM from Figure 3, the tag sequence with the highest probability that is returned is `['N2', 'SN', 'ST', 'LN', 'PC']`. It corresponds to the following path through the HMM (with the corresponding observation symbols – the output fields – in brackets).

```
Start → number_first (N2) → street_name (SN) → street_type (ST)
      → locality_name (LN) → postcode (PC) → End
```

The values of the input address will be assigned to the output fields as follows.

```
 number_first: '42'
  street_name: 'meyer'
  street_type: 'road'
locality_name: 'cooma'
     postcode: '2371'
```

3.4 Verification

Once segmented an input address can be easily compared to the existing addresses in the national address database. Different techniques can be used for this task, for example inverted indices as described in [7], which allow approximate matching (for example if parts of an address are missing or wrong). Alternatively, hash encodings (like *MD5* or *SHA*) can be used to create a unique *signature* for each address in the national database, allowing to efficiently compare a hash encoded input address with the full database. Similarly, hash encodings of the locality and street (and their combinations) allow the verification of only these parts of an address. This component of our system is currently under development, and more details will be published elsewhere.

4 Automated Hidden Markov Model Training

The automated HMM training approach is based on national address guidelines and a large national address database, and only needs minimal initial manual efforts. Guidelines for correctly addressing letters and parcels are increasingly becoming important as mail is being processed (sorted and distributed) automatically. Many national postal services therefore publish such guidelines[1]. Our system uses these guidelines to build the initial HMM structure, as shown in Figure 2. This is currently done manually, but in the future it is likely that electronic versions of such guidelines (for example as XML schemes) will become available, making the initial manual building of the HMM structure automated

[1] See for example: `http://www.auspost.com.au/correctaddress`

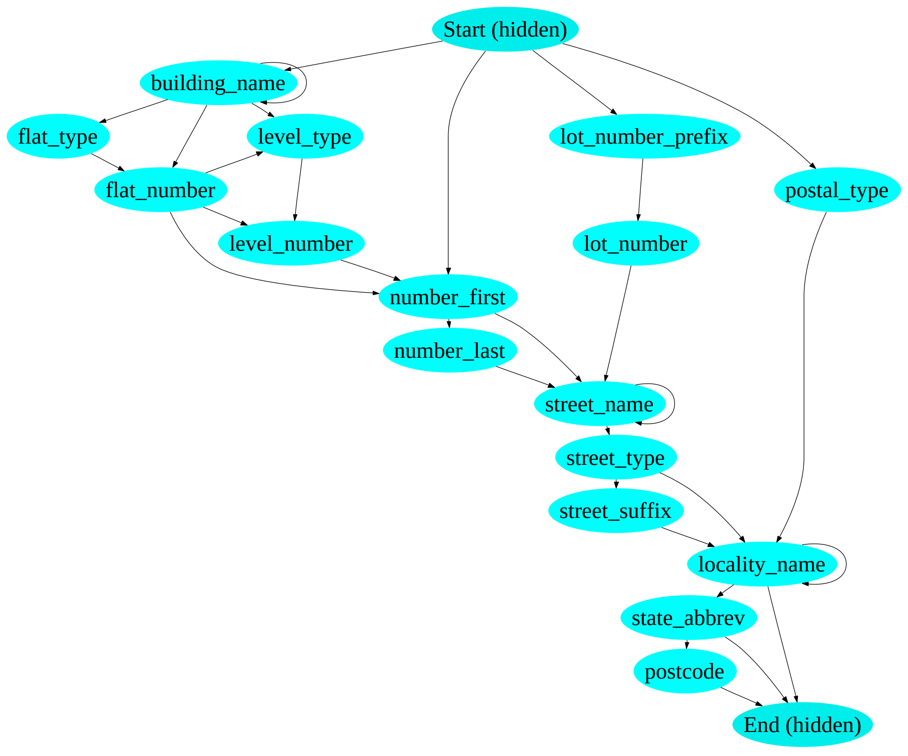


Fig. 2. Initial HMM topology manually constructed from postal address guidelines to support the automated HMM training

as well. The structure is built with the national address database in mind, i.e. the HMM states correspond to the database attributes, and aims to facilitate the automated training process which uses the clean and segmented records in such an address database.

A comprehensive, parcel based national address database has recently become available in Australia: *G-NAF* (the Geocoded National Address File) [13]. Developed mainly for geocoding applications in mind, approximately 32 million address records from several organisations were used in a five-phase cleaning and integration process, resulting in a database consisting of 22 normalised tables. G-NAF is based on a hierarchical model, which stores information about address sites (properties) separately from streets and locations [14]. For our purpose, we extracted 26 address attributes (or output fields) as listed in Table 2. The aim of the standardisation process is to assign each element of a raw user input address to one of these 26 output fields, as shown in the example in Figure 1. Only the G-NAF records covering the Australian state of New South Wales (NSW) were available to us, in total $4,585,707$ addresses. There are two main steps in the set-up and training phase of our address standardisation system as follows.

Table 2. G-NAF address attributes (or fields) used in the standardisation process

	G-NAF fields
Address site	`flat_number_prefix`, `flat_number`, `flat_number_suffix`, `flat_type`, `level_number_prefix`, `level_number`, `level_number_suffix`, `level_type`, `building_name`, `location_description`, `private_road`, `number_first_prefix`, `number_first`, `number_first_suffix`, `number_last_prefix`, `number_last`, `number_last_suffix`, `lot_number_prefix`, `lot_number`, `lot_number_suffix`
Street	`street_name`, `street_type`, `street_suffix`
Locality	`locality_name`, `postcode`, `state_abbrev`

4.1 Generation of Look-up Tables

The look-up tables are generated by extracting all the discrete (string) values for `locality_name`, `street_name` and `building_name` into tables and then combining those tables with manually generated tables containing typographical variations (like common misspellings of suburb names), as well as the complete listing of postcodes and locality names from the national postal services. Other look-up tables are generated using the official G-NAF data dictionary tables (for fields such as `street_type`, `street_suffix`, `flat_type`, or `level_type`). The resulting look-up tables are then cleaned using the same approach as described in Section 3.1, and used in the tagging step to assign look-up tags to address elements.

4.2 HMM Training

The required input data for the training are (1) the initial HMM structure as built using the postal address guidelines and as shown in Figure 2, and (2) the G-NAF database containing cleaned and segmented address records. The distribution of both transition and observation probabilities are learned based on frequency counts of the occurrences of attribute values in the G-NAF database. Each G-NAF record is an example path and observation sequence. Due to minor deficiencies in the data contained in G-NAF, such as the lack of postal addresses, postcodes, or the character slash `'/'` (which is often used to separate flat from street numbers), manually added tweaks must be automatically applied where appropriate to the model during training to account for the lack of observations and transitions, and to account for unusual but legitimate address types, such as corner addresses. A HMM trained using G-NAF is shown in Figure 3. Because training data often does not cover all possible combinations of transitions and observations, during application of a HMM unseen and unknown data is encountered. To be able to deal with such cases, *smoothing* techniques [4] (such as *Laplace* or *absolute discount* smoothing) need to be applied, which enable unseen data to be handled more efficiently. These techniques basically assign small probabilities to all unseen transitions and observations symbols in all states.

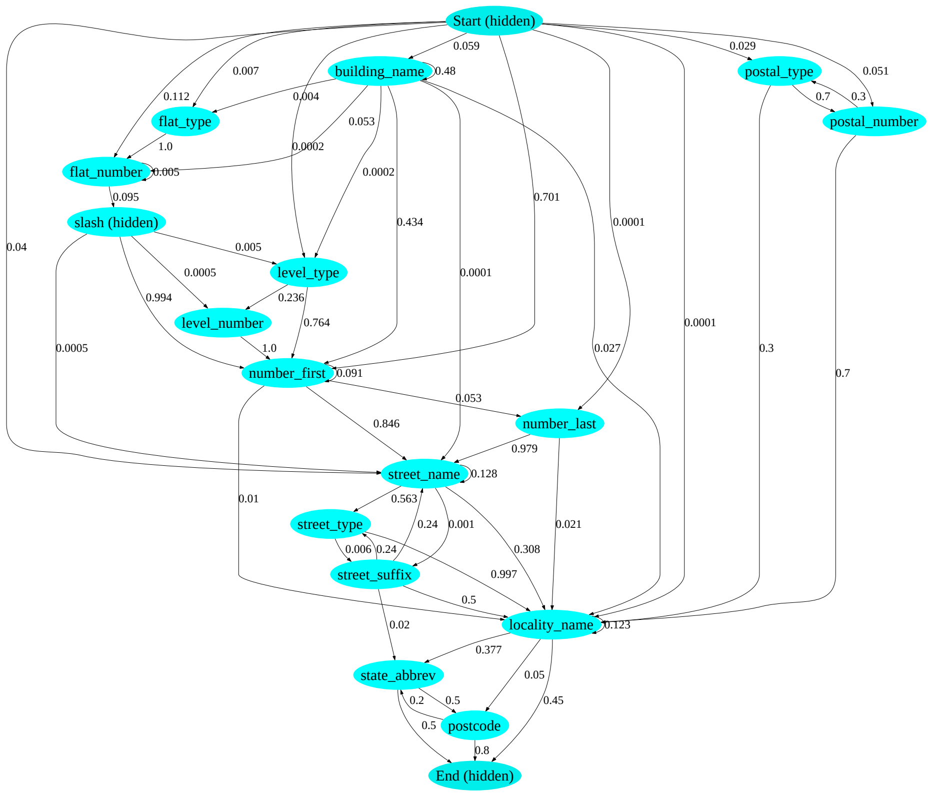


Fig. 3. HMM (simplified) after automated training using the G-NAF national address database (but before smoothing is applied)

5 Experimental Results and Discussion

Special care must be taken when evaluating HMM based systems. If the records used to train a HMM are from the same or similar data set as the records used to evaluate the performance of the same HMM, the model may become over-fitted to the training data and may not accurately reflect the real performance of the HMM. To test the accuracy of our probabilistic standardisation approach raw addresses from three data sets were used. The first contained 500 records with addresses taken from a midwives data collection, the second 600 nursing home addresses, and the third a 150 record sample of unusual and difficult addresses from a large administrative health data set. There are three major variations possible in our system for standardising addresses:

1. **Features and look-up tables (F&LT)**
   During the tagging step of standardisation, each element in the address is assigned one or more tags depending if it can be found in one or more look-

up tables. Once all tables have been checked, the element will also be given a feature tag as described in Section 3.2. However, elements of one character length are only given a feature tag and look-up tables are not searched.

2. **Look-up tables only (LT)**
   This is similar to the supervised system [8] as previously implemented in *Febrl* [6]. An address element is given one or more look-up tags, depending if it can be found in the look-up tables. If it is not assigned any tags, it is given a feature tag. Again, elements of one character length are only given feature tags.
3. **Features only (F)**
   Single address elements are only given feature tags and look-up tables are not used. Any sequence the greedy matching algorithm finds of length two or more elements is assigned a tag from the look-up tables as normal. Unlike the other two options, elements were not placed into their canonical form, since there is no look-up table used to check for original forms.

While HMM's were trained using all three options of smoothing (*no smoothing*, *absolute discount*, and *Laplace*), *no smoothing* was not tested as it is deemed to be highly inflexible and unable to cope with unseen input data. *Laplace* smoothing was tested, but not analysed extensively as initial tests showed a quite poor performance. All results, unless specified, are therefore assumed to be from a HMM with *absolute discount* smoothing applied. Comparison test were also performed using the supervised *Febrl* address standardiser [6, 8].

Records were judged to be accurately standardised if all elements of an input address string were placed into the correct output fields. It was not appropriate to check for correct canonical correction, since feature based tagging will not transform any words. Addresses not fully correct were judged on an individual basis for level of correctness, either *'close'* or *'not close'*, depending upon the criticality of the error. For example, numbers being classified as `number_last` instead of `number_first` were considered *'close'*, whereas street types being judged localities are considered *'not close'*. A second measure of accuracy, called *'could be accuracy'*, was used to show the level of accuracy of the HMM when including *'close'* (but incorrectly standardised) records as correct.

In many data sets the majority of input addresses are of fairly simple structure. We therefore counted the frequency of the following three sequences and included their numbers (labelled as *'Easy addresses'*) in Table 3.

```
(number_first,number_last,street_name,street_type,locality_name,postcode)
(number_first,street_name,street_type,locality_name,postcode)
(street_name,street_type,locality_name,postcode)
```

As expected, the data set with unusual addresses contained much less easy addresses, while for the other two data sets around 90% were easy addresses.

Performance was averaged over 10 runs of the system for each category of execution. All standardisation runs were performed on a moderately loaded Intel Pentium M Centrino 2.0 GHz with 512 MBytes of RAM.

Table 3. Experimental accuracy and standardisation timing results on three test data sets using *absolute discount* HMM smoothing. See text for discussion what *easy addresses* are

	Midwives	Nursing homes	Unusual
Total number of addresses	500	600	150
Easy addresses **(F&LT)**	446	542	31
Easy addresses **(LT)**	438	538	27
Easy addresses **(F)**	445	542	31
Easy addresses *Febrl*	410	529	22
Accuracy **(F&LT)**	97.40%	96.67%	92.67%
Accuracy **(LT)**	95.40%	98.50%	72.67%
Accuracy **(F)**	96.60%	92.67%	79.33%
Accuracy *Febrl*	96.80%	96.00%	96.00%
'Could be' accuracy **(F&LT)**	98.00%	97.80%	94.67%
'Could be' accuracy **(LT)**	97.40%	98.50%	80.00%
'Could be' accuracy **(F)**	97.00%	96.50%	80.67%
'Could be' accuracy *Febrl*	97.60%	98.30%	96.00%
Milli-seconds per record **(F&LT)**	92	445	720
Milli-seconds per record **(LT)**	11	18	37
Milli-seconds per record **(F)**	6	7	7
Milli-seconds per record *Febrl*	7	9	10

5.1 Discussion

As can be seen by the difference between actual accuracy and *'could be'* accuracy in Table 3, not only is the accuracy of the new system quite high, especially when using the **(F&LT)** variation, but quite a large number of the incorrect records were only marginally incorrect in non-critical parts of an address. Perhaps half of the remaining errors were caused by a known deficiency in the greedy tagging system, which has to do with the value `'st'` being a known abbreviation both for *'Saint'* and *'Street'*. Most remaining errors were examined in depth, but in general it was impossible even for a human to determine the exact correct output. Accuracy using our automatically trained system versus a manually trained *Febrl* HMM is equal to or better than in all cases tested. Quite surprisingly, accuracy using the **(F)** HMM was quite comparable to the **(LT)** based HMM.

Also, the *Febrl* address HMM failed on almost all non NSW addresses given, due to them generally being outside the scope of its look-up tables, thus the tagging was ineffective. However the **(F&LT)** and **(F)** HMM's both successfully standardised most non NSW addresses by using the feature information where the look-up tables came up blank. This has promising possibilities for using the HMM to standardise addresses outside the domain of G-NAF without any retraining necessary. There are also possible applications where licensing or other reasons are non permissive for distribution of the G-NAF national address database and corresponding look-up tables generated.

Timing performance using the **(F&LT)** HMM is relatively poor due to the large number of possible combinations of tag sequences, however still quite acceptable, especially since accuracy is generally more highly valued than time taken, and the fact that addresses can be easily standardised in parallel.

6 Outlook and Future Work

In this paper we have presented an automated approach to address cleaning and standardisation based on national postal address guidelines and a comprehensive national address database (G-NAF), and using a probabilistic hidden Markov model (HMM) which can be trained without manual interaction. Standardising addresses is not only an important first step before address data can be loaded into databases or data warehouses, or be used for data mining, but it is also necessary before address data can be linked or integrated with other data.

There are still various improvements possible to our system. Currently corner addresses are implicitly supported, but explicitly creating HMM states such as a second street name and type is a more complete solution. Characters such as dash, brackets, commas, etc. are currently processed in the cleaning step, but handling them in the HMM could improve accuracy. Other minor improvements include training the HMM using corrected G-NAF data, and ways to minimise the number and size of manual tweaks to the HMM. The look-up tables contain some common typographical error correction data, drawn from manually created lists. It should be possible to build far more comprehensive lists automatically by matching between the G-NAF address data and correctly standardised example addresses, in order to find typographical variations.

Each distinct tag sequence given to the HMM will always have the same output states and *Viterbi* probability. This can be used to advantage by *caching* the set of input tags and the resulting probability during execution. Since up to 90% of addresses in some data sets have the same output fields, it is highly likely that there will be a considerable number of addresses with the same tag sequence. These redundant calculations can be eliminated by checking the tag sequence against a cache of sequences. If found in the cache, directly return the probability, otherwise the sequence will be run through the HMM and the resulting probability and input tags will be added to the cache. Using the **(F&LT)** variation, addresses can have dozens of possible tag sequences, thus the caching of results should give considerable performance improvements.

While developed with and using Australian address data, our approach can easily be modified to other countries, or even other domains (for examples names, medical data, etc.) as long as standardisation guidelines and a comprehensive database with standardised records are available.

Acknowledgements

This work is supported by an Australian Research Council (ARC) Linkage Grant LP0453463 and partially funded by the NSW Department of Health.

References

1. Adelberg, B: Nodose: a tool for semi-automatically extracting structured and semistructured data from text documents. In proceedings of ACM SIGMOD International Conference on Management of Data, New York, pp. 283–294, 1998.
2. Agichtein, E. and Ganti, V.: Mining reference tables for automatic text segmentation. In proceedings of the ACM SIGKDD'04, Seattle, pp. 20–29, August 2004.
3. Bikel, D.M., Miller, S., Schwartz, R. and Weischedel, R.: Nymble: a high-performance learning name-finder. In proceedings of ANLP-97, Haverfordwest, Wales, UK, Association for Neuro-Linguistic Programming, pp. 194–201, 1997.
4. Borkar, V., Deshmukh, K. and Sarawagi, S.: Automatic segmentation of text into structured records. In proceedings of the 2001 ACM SIGMOD international conference on Management of data, Santa Barbara, California, 2001.
5. Califf, M.E. and Mooney, R.J.: Relational learning of pattern-match rules for information extraction. In proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), Menlo Park, CA, pp. 328–334, 1999.
6. Christen, P., Churches, T. and Hegland, M.: A Parallel Open Source Data Linkage System. Proceedings of the 8th PAKDD'04 (Pacific-Asia Conference on Knowledge Discovery and Data Mining), Sydney. Springer LNAI-3056, pp. 638–647, May 2004.
7. Christen, P., Churches, T. and Willmore, A.: A Probabilistic Geocoding System based on a National Address File. Proceedings of the 3rd Australasian Data Mining Conference, Cairns, December 2004.
8. Churches, T., Christen, P., Lim, K. and Zhu, J.X.: Preparation of name and address data for record linkage using hidden Markov models. BioMed Central Medical Informatics and Decision Making 2002, 2:9, Dec. 2002. Available online at: `http://www.biomedcentral.com/1472-6947/2/9/`
9. Freitag, D. and McCallum, A.: Information extraction using HMMs and shrinkage. In papers from the AAAI-99 Workshop on Machine Learning for Information Extraction, Menlo Park, CA, pp. 31–36, 1999.
10. Gill, L: Methods for Automatic Record Matching and Linking and their use in National Statistics. National Statistics Methodology Series No. 25, London 2001.
11. Han, J. and Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000.
12. AutoStan and AutoMatch, User's Manuals. MatchWare Technologies, 1998.
13. Paull, D.L.: A geocoded National Address File for Australia: The G-NAF What, Why, Who and When? PSMA Australia Limited, Griffith, ACT, Australia, 2003. Available online at: `http://www.g-naf.com.au/`
14. Paull, D.L. and Marwick, B.: Understanding G-NAF. Proceedings of SSC'2005 (Spatial Intelligence, Innovation and Praxis), Spatial Sciences Institute, Melbourne, September 2005.
15. Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, vol. 77, no. 2, Feb. 1989.
16. Rahm, E. and Do, H.H.: Data Cleaning: Problems and Current Approaches. IEEE Data Engineering Bulletin, 2000.
17. Seymore, K., McCallum, A. and Rosenfeld, R.: Learning Hidden Markov Model Structure for Information Extraction. In proceedings of AAAI-99, workshop on Machine Learning for Information Extraction, 1999.
18. Soderland, S: Learning information extraction rules for semi-structured and free text. Machine Learning, vol. 34, no. 1–3, pp. 233–272, February 1999.

Differential Categorical Data Stream Clustering*

Weiyun Huang[1], Edward Omiecinski[1], Leo Mark[1], and Weiquan Zhao[2]

[1] College of Computing, Georgia Institute of Technology, Atlanta GA, USA
{wyhuang,edwardo,leomark}@cc.gatech.edu
[2] School of Computer and Information Science
University of South Australia, Australia
weiquan.zhao@cs.unisa.edu.au

Abstract. Stream data analysis differs significantly from traditional data processing. To process the data online the algorithm has to work in one pass, incorporating new data into a model maintained in main memory. Categorical stream clustering is especially difficult because of the lack of order and "closeness" properties. We propose a differential categorical stream mining algorithm and demonstrate their good accuracy and fast speed on synthetic and real categorical data. A major step of the algorithm is "data compression", i.e. to store a model or synopsis of processed data in the memory. We propose several data compression schemes that can efficiently generate compact representations of original data, so as to enable the algorithm to process streams at high speed and detect the changes in underlying data.

Keywords: Stream, clustering, categorical, differential, compression schemes

1 Introduction

Stream data is becoming a new and important type of data source. A data stream is a sequence of data points which usually can only be read once and does not support random access. Examples are the observations by distributed sensors, security alarm logs, and events in large-scale scientific computations.

Traditional data clustering tends to get a model from the whole data set, and the order of the data points is often not relevant. However, with stream data, we often want to see the evolution of models. Suppose that we want to see with new data coming in, how the clusters of the data change. What if the data cannot fit in the memory buffer any more after some time? We need to accomodate new data so earlier data have to be moved out of memory. Since we are dealing with stream data, we cannot retrieve those earlier data later, although the clustering algorithm needs them. A stream processing algorithm needs to address this problem. It has to compress the old data in some way and keep the compressed format in memory.

Even if we want to focus on "recent data", or rather, the data falling in a "current window" from some point of time in the past until the current time,

* This research has been partially supported by National Science Foundation Grant CCR-0121643.

as shown in figure 1, it is still possible that the "current window" is too big for the buffer. What's more, if those "windows" are overlapping, compression may be more useful since it will save the effort of processing the overlapped part repeatedly. Suppose that the "current window" slides on the data stream and each step it moves is called a "differential unit". For example, in practice, the current window could be five hours long and each differential unit could be one hour. Every hour we want to obtain the clustering result for the last 5 hours. Without compression we can have to process the overlapped data repeatedly, which is obviously expensive, and if the 5 hour's data cannot fit in memory we are in trouble.

An algorithm that can process data incrementally and decrementally is called a "differential algorithm". It not only can process the data very efficiently, but also can extract models from current data, and compare the models from different time intervals in order to discover the underlying changes.

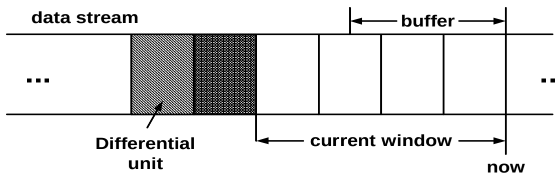


Fig. 1. Data stream, current window and differential unit

Although in the real world most of the data sets contain categorical features, there is less related work on clustering categorical data than on numerical data. However in real life applications, many data features are categorical by nature, such as network protocols/host domains/types of accessed files in network access logs, and color/shape of objects in scientific observations. The clustering of categorical streaming data is even less studied. We address the categorical data clustering algorithm in this paper and use this as a tool to study the evolution of categorical data.

This paper is organized as follows: in section 2, we review the related work. In section 3 we discuss our differential clustering algorithm. Then we present the details of our compression schemes in section 4. In section 5 we provide some experimental results. Section 6 is future work and section 7 concludes the paper.

2 Related Work

More and more attention has been paid to stream data processing [6, 7, 14], such as clustering [11], maintaining histograms [10], approximating certain queries [6], or building decision trees [7, 14] in a stream environment.

Work has been done to cluster data incrementally in one pass [5, 8], which is ideal in the data stream environment. One-pass algorithms often store some model information to represent processed data points or their models. In the case of numerical data, people use “representative points” [12] (a set of well-scattered points in each cluster that could capture the shape and extent of the cluster) or “sufficient statistics” [5] (sums or mean of the points in each cluster, sums of squares of the points in each cluster, standard deviation of each dimension, cardinality of the cluster, and so forth) as a synopsis of the data. Similar mechanism is needed for handling categorical data, and we use our “compression schemes” for this task (see section 4).

Some novel categorical clustering algorithms have been proposed [4, 3]. Huang [13] presented the k-modes algorithm, an extension to the well-known k-means algorithm. The COOLCAT [4] algorithm is an entropy-based algorithm. It incrementally puts the next point into one of the existing clusters where the overall expected entropy of clusters can be minimized.

Recently more and more attention has been put on mining the evolution of the data [1, 2, 17, 15]. For example, Aggarwal [1] uses velocity density estimation concept to diagnose the changes in an evolving stream. Some other research [17] uses an ensemble method to detect concept drift. However, similar to the stream clustering problem, much research work focuses on numerical data, while the evolution of categorical data clusters has not been thoroughly studied. Our algorithm, by clustering categorical data stream differentially, can be used to keep states of data streams and detect the evolution of the data clusters.

3 Differential Categorical Data Clustering

3.1 The Differential Clustering Algorithm

In order to cluster a data stream differentially, our stream clustering algorithm follows the steps in figure 2. The algorithm will make use of a regular categorical clustering algorithm $\mathcal{C}$ (in our paper we use the k-modes algorithm), and maintain a data buffer.

In our discussion, we use k to denote the number of clusters, n as the number of points, d as the number of dimensions, $|w|$ as the current window size, $|u|$ as the differential unit size, and $\mathcal{D}$ as a data set.

The fourth step, “data compression”, stores a representation of data in the buffer. Different compression methods are also called “compression schemes”. They only take a small chunk of memory, and are interchangeable by design, i.e., when time and space allow, we can use a more complicated scheme (which implies better accuracy), and when the stream comes in a burst or memory runs short, it is possible to switch to a simpler scheme. This feature is especially useful in practical stream applications.

This algorithm can produce clusters efficiently. If we don't do step 5, then it's essentially an incremental algorithm that can generate model for the whole data set. Also, by periodically storing $\mathcal{D}_c$ to secondary storage, we can easily go back to a previous time point and do further analysis.

Sometimes, a data set contains both categorical and numerical data. How to combine these two types of data is a domain-related problem. We can use a distance metric which is a linear combination of the distance computed only with

1. Fill the buffer, or put all the available points in the buffer. The set of points in the buffer is $\mathcal{D}$.
2. Choose k starting points from $\mathcal{D}$.
3. Apply the clustering algorithm $\mathcal{C}$ to $\mathcal{D}$ and generate a set of clusters cls.
4. Compress $\mathcal{D}$ based on cls, get a new representation $\mathcal{D}_c$ of data $\mathcal{D}$, Depending on the compression scheme, part of or the entire buffer is freed.
5. If needed, remove the representation of stale data from $\mathcal{D}_c$.
6. Put new data points into the buffer for a new $\mathcal{D}$ (it may contain some points from a prior $\mathcal{D}$). $\mathcal{D}_c$ is also incorporated into this new $\mathcal{D}$. The modes of the clusters in cls are used as the starting points.
7. Go to step 3, until no new points are available.

Fig. 2. The differential algorithm

categorical values and the distance computed only using numerical values [13], or to convert the numerical values by discretizing them according to some rules and to consider the converted values as categorical ones. Provided that we do not have much knowledge about the data set itself, we favor the latter method.

If some of the clusters become empty due to the removal of points, we choose a data point in the buffer which is the farthest from the current non-empty clusters, and then make it the mode of a new cluster.

3.2 The K-modes Algorithm

We take the k-modes algorithm [13](see figure 3) as the $\mathcal{C}$ in figure 2 because it works directly on categorical data efficiently. However, we do not see any reason to restrict our approach to this algorithm.

In the k-modes algorithm, since "mode" (with respect to the "mean" in the numerical scenario) is defined as a vector in which each element is the "most frequently occurring value" (MOV for short) of the corresponding dimension in the cluster, what we are interested in is the MOV of each dimension. Formally, for each dimension d of the data, with respect to a data set D, the MOV $mov(d, D)$ is defined as

$$mov(d, D) = argmax_v freq_d(v, D)$$
$$freq_d(v, D) = \frac{count_{\boldsymbol{x} \in D}(x_d = v)}{|D|}$$

From statistics' point of view MOV is essentially the mode of dimension d. However here we use $mov(d, D)$ to distinguish from the multi-dimensional mode of a cluster. The data set D can be the whole data set $\mathcal{D}$ or any subset of it. And $count_{\boldsymbol{x} \in D}(x_d = v)$ is the number of occurrences of value v along dimension d for the points in D. For the sake of simplicity, sometimes $freq_d(v, D)$ is simplified to $freq(v)$.

One important thing about categorical data clustering is to choose the dissimilarity measure, or the distance metric. Assume we are looking at a data set $\mathcal{D}$, with n data points, m dimensions. One metric proposed by Huang [13] is the Hamming distance: the distance between two data points $\boldsymbol{x} = (x_1, x_2, \cdots, x_m)$

1. Select k initial modes.
2. Allocate every object to the cluster with the nearest mode. Update the modes.
3. After the allocation of all the objects, recompute the distance from each object to the current modes. If an object's nearest cluster changes, reallocate the object to the cluster with the nearest current mode. Update the modes if reallocation happens.
4. Repeat 3 until no object has changed clusters after a full cycle test of the whole data set.

Fig. 3. The k-modes algorithm

and $\boldsymbol{y} = (y_1, y_2, \cdots, y_m) \in \mathcal{D}$ is defined as the number of dimensions along which two points have different values:

$$dist(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{d} \delta(x_j, y_j), \ where \quad \delta(x_j, y_j) = \begin{cases} 0 \ (x_j = y_j) \\ 1 \ (x_j \neq y_j) \end{cases} \tag{1}$$

It is similar to the Jaccard similarity coefficient [16] that is widely used in information retrieval context. This metric is simple and effective, meaningful in the unsupervised learning scenario since we do not have much knowledge about individual dimensions. Hence we adopted it in our study.

Entropy is a good metric when processing categoricdal data, and it is used widely in many algorithms and applications. In our research, we mainly focus on processing high speed streams (since these streams are difficult to buffer and static data processing algorithms cannot be directly applied to them), where processing rate (number of data points processed in unit time) is a major consideration. We notice that entropy related methods usually incur more computation cost than mode/histogram based methods, so we choose the k-mode algorithm. In section 5 we compared our result with an entropy based algorithm.

To an arbitrary point $\boldsymbol{x} = (x_1, x_2, \cdots, x_m) \in \mathcal{D}$, the cluster it belongs to at time t is $cl(\boldsymbol{x}, t)$. For simplicity, we use $cl(\boldsymbol{x})$ to denote the cluster that $\boldsymbol{x}$ belongs to at the time of clustering quality evaluation. The mode of a cluster cl is denoted as $mode(cl)$. The quality of a clustering solution cls can be measured by its deviation, which is defined as:

$$dev(cls) = \frac{1}{n} * \sum_{\boldsymbol{x} \in \mathcal{D}} dist(\boldsymbol{x}, mode(cl(\boldsymbol{x}))) \tag{2}$$

3.3 Initialization of Clusters

Initialization of clusters can be done by randomly selecting points from the first "current window". It is simple to implement but to some extent arbitrary. We use a heuristic called "dissimilarity rings" to find the starting points. We first find out the mode of the data and compute the Hamming distance from each point to the mode. Then the points with same distance are grouped together. We can have at most $d + 1$ groups (these groups are like concentric rings with mode as the center). If number of clusters is smaller than the number of groups, we can select most dissimilar modes of these "rings" as our initial modes.

4 Data Compression

In clustering step we do not ignore any point. Then the compression is implemented by choosing some seed points, assigning rest of the points to them so as to form small groups (subclusters), and for each group recording the group size, mode and histogram for each dimension. The mode of the subcluster may or may not be the initial seed.

We want to record the values present in the cluster for each dimension and the number of times they appeared. This results in a set of histograms: for each dimension, a list of pairs $(value, freq(value))$ has to be recorded. Naturally, we can use "mode" and dimensions' histograms as the "sufficient statistics", and mode can be deduced from dimension's histograms. Usually the domain for each categorical dimension is small (i.e., the number of possible values is not very big). Therefore we assume that the length of the list of pairs is not too long, or rather, the histograms will not take a lot of space and can be efficiently accessed. If this is not the case, we can trim the histogram to a certain length, i.e., keep the high-biased histograms [9] of the old data. If a cluster is "tight", then a small number of most frequently occurring values should be sufficient to represent the histogram fairly well. Our experiments show that 5 most frequently occurring values provide reasonable accuracy.

The compressed data (in the form of subclusters) can be used later with new data points to form a new $\mathcal{D}$ (see figure 2). It is equivalent to add data in groups into $\mathcal{D}$, only these groups cannot be split later in clustering step.

However, in the differential case, we need to remove the effect of a stale block. Obviously, it is difficult to know the contribution of an arbitrary block. We may even have to go back to the original data and repeat the process to generate the "stored models" contributed by the stale block, which is not feasible. In fact, a reasonable assumption could simplify this effort:

Assumption: The starting and ending points of a differential unit can be predicted, and the size of a differential unit is comparable with the size of the "current window".

This assumption implies that the boundary of a data block can be decided by length, time, or some other criteria. Therefore, when we compress the data, we not only need to do this for the clusters, but also for each differential unit. For example, given a live event stream generating events every day, it is rational to set "one day" as a unit of data to remove. Therefore, when we store the models for the clusters, we already know that "one day" will be a unit, so we can store models for the points in each day for each cluster. Since the size of a "differential unit" is comparable with the "current window" we are looking at (which means the "current window" will not contain a large number of differential units), the space required for storing extra information for each differential unit should be reasonable.

Hence, when a differential unit is to be removed, we could simply traverse the clusters, subtract its effect from the stored model of each cluster, to remove its contribution.

4.1 Compression Schemes

We explore several categorical compression schemes in our experiments.

Naive compression In naive compression, cluster modes are the seeds. Hence each cluster is a subcluster.

Sampling based compression The seeds are sampled from the buffer. In order not to miss some small clusters, we choose to sample seeds from every cluster. That is, we use these clusters as strata, and do stratified sampling. We can give each cluster equal sample size (we call this "equal-size sampling", or "E-sampling"); or we can allocate sample sizes based on different criteria, say, according to cluster size (cluster with higher cardinality gets more space), according to area (cluster with larger radius gets more space), or according to the inverse of density (tighter cluster gets less space). We've also tried a "D-sampling" scheme by first grouping points in each cluster by their distance to the mode, then using these distance groups as strata, and allocating spaces according to the inverse of their density.

Dynamic tightness compression In this scheme, we evaluate the "tightness" of each cluster before compression. A cluster's "tightness" is decided by the frequency of most occurring value(MOV) for each dimension and the distances from its members to its mode.

A cluster *cl* is tight given threshold (f, r) iff for each dimension, $freq(MOV) \geq f$ and $radius(cl) \leq r$. The radius of a cluster *cl* is the maximum distance between any point in the cluster and its mode.

$$radius(cl) = \max_{\boldsymbol{x} \in cl}(dist(\boldsymbol{x}, mode(cl))) \tag{3}$$

The "tightness" threshold has to be dynamically changed to free up space for new points. This is done according to two heuristic rules: (1) when the buffer is about 90% full, we compress all the points just like naive compression does, and loosen the tightness threshold; (2) if the buffer is less than 30% full, we tighten the threshold.

Secondary clustering We've also tried another scheme, which involves secondary clustering of the points in the buffer, and compressing the tight secondary clusters. For the data set we used, the number of secondary clusters is about 4 or 5 times the number of final clusters desired. In terms of accuracy this scheme also works, but the computation overhead is significant and in some extreme cases it runs even slower than the regular k-modes. Therefore we did not report details of its results.

4.2 Accuracy Comparison

The error of compression is caused by merging points into subclusters, while they belong to different clusters later.

Suppose two points $\boldsymbol{x}$ and $\boldsymbol{y}$ are compressed into one subcluster, we would like to measure the possibility that $\boldsymbol{x}$ and $\boldsymbol{y}$ belong to different clusters in the future. Intuitively, if their distance is bigger, then the possibility that they should belong to different clusters is also bigger. Therefore, for each subcluster generated by a compression scheme, we can use the sum of distances between any of the two points to measure if it is "good" or not. To avoid computing too many pairwise distances, we can use the sum of distances from each point to its subcluster's seed as an approximated indicator of a scheme's goodness.

Given a data set $\mathcal{D}$, if $\mathcal{D}_c$ is the set of seeds for $\mathcal{D}$ using a certain compression scheme, then define set deviation

$$SDEV(\mathcal{D}, \mathcal{D}_c) = \sum_{\boldsymbol{x} \in \mathcal{D}} dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{D}_c)),$$

while "value mapping" $vmap(\boldsymbol{x}, \mathcal{S})$ is the point in S that is closest to $\boldsymbol{x}$, given a data set $\mathcal{S}$ and a data point $\boldsymbol{x}$:

$$vmap(\boldsymbol{x}, \mathcal{S}) = argmin_{\boldsymbol{y} \in \mathcal{S}}(dist(\boldsymbol{x}, \boldsymbol{y})).$$

Hence if the set of seeds is $\mathcal{D}_c$, then $SDEV(\mathcal{D}, \mathcal{D}_c)$ is our measurement: the smaller it is, the better the compression scheme.

Theorem 1. *If $\mathcal{S}_1$, $\mathcal{S}_2$ are two seed sets for compressing a data set $\mathcal{D}$, And $\mathcal{S}_1$ is a subset of $\mathcal{S}_2$. Then the SDEV caused by compressing using $\mathcal{S}_2$ is no greater than using $\mathcal{S}_1$.*

Proof: For any point $\boldsymbol{x}$ in data set $\mathcal{D}$, $vmap(\boldsymbol{x}, \mathcal{S}_1)$ and $vmap(\boldsymbol{x}, \mathcal{S}_2)$ can be same or different. If they are the same, then $dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{S}_1)) = dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{S}_2))$; if they are different, since $\mathcal{S}_1$ is a subset of $\mathcal{S}_2$ and the subclusters are formed by nearest neighbor search, $dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{S}_2))$ must always be no greater than $dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{S}_1))$. Therefore,

$$\sum_{\boldsymbol{x} \in \mathcal{D}} dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{S}_2)) < \sum_{\boldsymbol{x} \in \mathcal{D}} dist(\boldsymbol{x}, vmap(\boldsymbol{x}, \mathcal{S}_1))$$

This implies that, with high possibility, a scheme with more subclusters should get less compression error than one with less subclusters. Similarly dynamic tightness compression should achieve at least as good accuracy as the naive compression.

4.3 Time Complexity Comparison

Different compression schemes may produce different number of subclusters. This phase involves a pass of nearest neighbor search, which is one of the major components of the execution time. Therefore intuitively naive compression should be the fastest. However, for some data, if we do not trim the histograms, naive compression may take more time than sampling based schemes, although the latter keeps more points in the memory. This happens when some attributes have a relatively big domain. The sampling based schemes keep more points, but the "subclusters" formed around them are tighter, therefore the histogram information kept in each sample can be much less than in a cluster mode in naive compression. However, once histograms are trimmed into fixed length (say, containing 5 most occurring values), the naive scheme always takes less time. Although these compression schemes are based on different heuristics, from the implementation point of view it is not difficult to dynamically switch from one to another. This makes the stream clustering algorithm more flexible so that it can handle the variation of the stream speed or buffer space.

5 Experimental Results

In this section, we discuss our experiments on both real and synthetic data using the differential algorithm, so as to compare the performance of different compression schemes. The results in section 5.1, 5.2 are produced on a Pentium IV 1.5GHz PC with 1GB memory. The operating system is Redhat 7.2. The experiments in section 5.3 are done on a Pentium IV 2.80GHz 1GB memory PC with cygwin[3] on Windows XP, since the COOLCAT algorithm we use runs on Windows platforms.

First we demonstrate the efficiency, accuracy and scalability of the algorithm by doing incremental clustering and comparing with the regular k-modes algorithm (our own implementation). In the second suite of experiments we do differential clustering, by sliding the "current window" along the data stream to produce clusters for each window. At last we compare our incremental results with the COOLCAT algorithm.

If not specified, we run each algorithm 5 times (4 times using randomly selected initial points, one time using the most dissimilar ring modes as initial clusters) for each parameter setting and compare average value.

5.1 Incremental Clustering

In this section our algorithm runs in incremental mode(step 5 of the algorithm is omitted). We use the 10% KDD Cup 1999 data[4], an ascii data set with 494021 points and 41 features (continuous features are discretized).

To compare with K-mode we set k=8. The sampling based schemes both use 200 points as the sample size. The D-sampling scheme divides each cluster into 3 groups.

Figure 4 shows the performance of different compression schemes. Because even the optimal clustering result will have a positive $dev(cls)$,we use $dev(cls) - dev(optimal)$ to measure compression error(it may still contain errors due to other reasons such as a bad initialization, but it is a reasonable approximation). Since we do not know the optimal modes, we use the best result by regular k-modes to approximate them. We can see in the upper plot that E-sampling scheme achieves least average compression error since more subclusters keep more information. D-sampling has same number of subclusters, but during clustering the ever-changing densities cause the redistribution of sample spaces. Some dissimilar points may have to be merged at one time, and when the cluster gets more space, they cannot be split so the usage of the space is not optimal. Therefore, in some cases, the smallest space a cluster gets in the whole process decides the accuracy. The error for 5k (points) buffer is smaller than the 10k case in that the number of subclusters is the same (e.g., 200 for E-sampling) for both cases, so 10k buffer implies larger compression ratio.

Also important in stream processing is the rate at which points are processed. In our experiments we are able to process more than 60000 points per second. From figure 4 we can see that the naive compression achieves faster processing

[3] Available at *http://www.cygwin.com*
[4] Available at *http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html*

rate, but larger error. This matches our analysis in section 4.2 and 4.3. Accuracy and processing rate are a trade-off. Therefore when data flow rate is high we can switch to naive scheme to process all the data, and when flow rate drops we can switch back to more complicated schemes for better accuracy. The figure does not show the effect of compression spaces (the sample size is the same for 10k and 5k buffer). However our other experiments shows that for equal-size sampling, larger buffer space tends to get better accuracy.

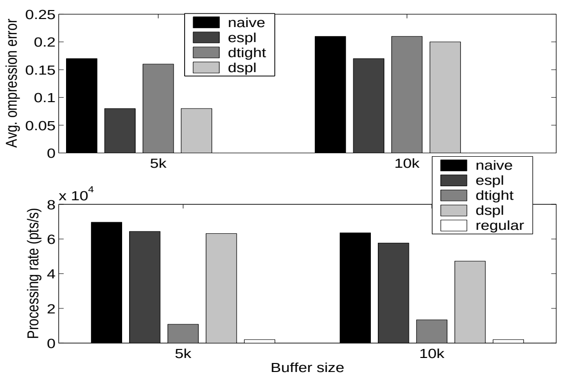


Fig. 4. Incremental clustering, comparison for kdd data: compression error and processing rates (records per second).

With this data set, our stream-based algorithm finds clusters with good quality in much less time than the regular k-modes algorithm. The accuracy measurement of best result achieved by the equal-size sampling is only 0.7% worse than that achieved by the regular k-modes. Naive and sampling based schemes all achieve significant speedup. The dynamic tightness scheme doesn't provide much speedup. One reason is that the scheme is probing for an appropriate tightness threshold (0.60 initially, which means the frequency of the MOV value for each dimension is greater than 40%), and that it might need to do compression multiple times. Also, testing if each cluster is tight and picking points to compress introduces additional overhead.

5.2 Differential Clustering

Although the data sets reside on local disk, we treat them as if they come from a stream. To do differential clustering, we divide the data into many differential units, and set the size of one "current window" as five differential units. We slide the window by removing the effect of one oldest differential unit and adding a new differential unit.

Synthetic Data To show that our compression schemes and algorithm can detect the changes in a data stream, we apply our differential algorithm to a synthetic data set.

The synthetic data generator we use is similar to the one in [5] and [8], but it outputs categorical data. The generator first chooses a set of means and variances from the given range, then randomly draws points from multiple Gaussian distributions, and finally rounds numerical values to closest integers, so as to make a "categorical" data set. We also convert the original means of the distributions into integer values and use them as the "true" modes. Because of the conversion, these modes may not be the optimal modes of the data set. However, they should give us guidance.

Obviously, our stream based algorithm will perform well on uniformly distributed data. However, in real life, a stream data source may not be uniformly distributed, so we created a "skewed" data set. Basically we generated 4 data sets with 4 sets of means and variances. The ranges of these 4 sets of means partly overlap. Then we concatenate the 4 data sets together to get a "hybrid" data set. This data set contains 100000 points and the number of features is 20. Each sub data set has 10 clusters. And we set the number of clusters as 10 when we do clustering. The current window size is 10000 points, and each differential unit contains 2000 points. The two sampling based schemes uses 200 points as sample pool size.

Our algorithm outputs a set of modes for each current window (first window's result is generated after processed 10000 points. After that, a result is generated when the next 2000 points are received). Since it is very difficult to visualize the 20-dimensional modes S, we take the four sets of "true" modes S_1, S_2, S_3 and S_4, and compute $SDEV(S, S_i), i = 1, 2, 3, 4$. Figure 5 (left) shows four curves generated by one run of the algorithm based on the D-size sampling scheme, and each curve is a $SDEV(S, S_i)$ w.r.t. the end of the current window. Figure 5 (right)shows a different presentation of the generated results: each point in the plot is the average distance from a current window's modes to its previous window's modes. For other compression schemes, we get similar results.

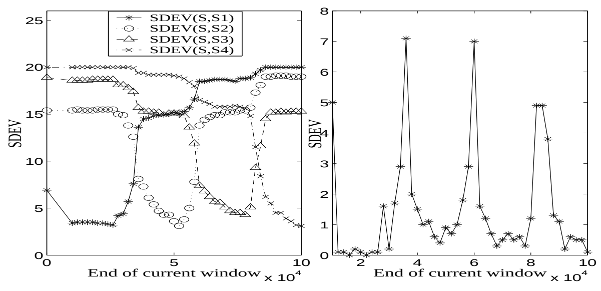


Fig. 5. Evolution detection using D-size sampling. (Left) comparing result modes with predefined modes; (Right) pairwise distance from current modes to previous modes

We can see that in the left figure, from 10000th point position to about 24000th point position, the clusters generated are pretty close to S_1, while $SDEV(S, S_2)$, $SDEV(S, S_3)$ and $SDEV(S, S_4)$ are much larger. Then there is a "mixed phase" where the current window contains data from both the first and the second distribution. The clusters become dissimilar to both "true" mode sets, and then gradually go close to S_2. Similarly, the clusters become close to S_3 and S_4 in later phases. In the right plot, the peaks approximates the position where a window containing data purely from a new set of modes. It is obvious that our algorithm can detect the evolution in underlying data and generate clusters based on the current window. Plots like the one to the right can visualize the changes of modes and are more useful in practical applications (while we do not know the "true" modes). We can also generates plots on modes changes with longer steps (say, distance across 10 current windows), changes of tightness of clusters, changes of the sizes of clusters, and so forth.

Kdd Data For the KDD 10% data set, the differential unit size is set to 4000 points, and the current window size is 20000 points. Number of clusters is 10, and sample size is 200 points.

In figure 6, the regular k-modes algorithm only runs on the last current window of the data set and chooses initial points from it, while the differential algorithm has to use modes from previous current window. The differential algorithm based on our compression schemes achieves worse accuracy compared with the regular k-modes algorithm, however, the regular k-modes algorithm has to select starting points from each current window and uses multiple iterations, hence is not practical in a stream environment.

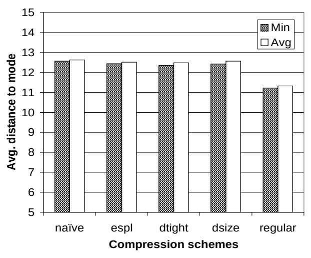


Fig. 6. Differential clustering error comparison, kdd %10 data. Compression schemes (from left to right): naive; equal-size sampling; dynamic tightness; D-size sampling; regular k-modes

5.3 Comparison with COOLCAT

COOLCAT [4] is an entropy-based algorithm. It is similar to the LIMBO algorithm [3] in that they optimize the same objective function, and it "exhibits average clustering quality that is close to that of LIMBO" [3], although LIMBO is more stable. In this section we compare our incremental algorithm with COOLCAT.

Interestingly, although the k-modes algorithm and COOLCAT both tend to put "similar" points into the same clusters, their assignment of new points into existing clusters can be different. In brief, the reason for the disagreement is that k-modes only compares new point with the MOV, while entropy is also affected by the distribution of other possible values. This makes COOLCAT to incur much heavier computation cost than the k-modes. On the other hand, k-modes guarantees that most dimensions' MOV occurrences get increased, which decrease the part of entropy generated from these dimensions. Our experiment shows that k-modes can generate pretty good expected entropy of all the clusters.

We adopted the error measurements used in [4]. In brief, the expected entropy of the clusters is defined as

$$\bar{E}(cls) = \sum_{k} \frac{|C_k|}{|D|} (\sum_{j=0}^{d-1} \sum_{l \in domain(j)} P_{kjl} log P_{kjl})$$

where P_{kjl} is the probability in cluster k that j^{th} dimension takes value l. The smaller $\bar{E}(cls)$ is, the better the clustering results.

Table 1. Results on kdd 10% data (avg. of 5 runs)

Setting	Coolcat				E-spl
m	0%	20%	40%	40%	N/A
buffer (points)	10k	300	300	10k	10k
sample (points)	100	100	100	100	200
Time(seconds)	4913	337	411	15282	89
Entropy	9.039	9.039	9.208	4.129	4.183

Table 1 shows the results on a binary version of KDD 10% data. Our algorithm uses the equal-size sampling scheme. Due to the entropy computation overhead, COOLCAT runs more efficiently with binary data. Therefore we convert the numerical values into binary categorical values according to [4]: for each numerical attribute, a median value (across the whole data set) is taken and any value lower than average is recorded as 0, otherwise as 1. The number of clusters is 2. The COOLCAT executable randomly reorders the points in its each run, and each run of our algorithm processes the points according to the order generated by COOLCAT setting one (first data column in table 1). To be fair to COOLCAT, the time to compute the expected entropy at the end is included in the execution time of our algorithm. We use the dissimilarity ring model to initialize the clusters. All the starting points are selected from the first window of data. The parameter m from the COOLCAT algorithm is the percentage of reprocessed points. The sample used by COOLCAT is for initialization: it draws a sample from the whole dataset, and the most dissimilar points are selected from this sample as the initial modes (so it scans the data twice).

Since we only have the COOLCAT executable, we can only measure the execution time of COOLCAT by recording the process running time, and it may not be completely accurate. However, from the table we can see our algorithm uses significantly less time to process the data set (the timer error thus can be ignored). The COOLCAT algorithm seems to be faster when buffer size is small rather than big. Hence we present results for different settings.

The COOLCAT algorithm runs significantly slower than our algorithm in all the settings. Although the goal of our algorithm is to minimize the average distance from each point to the corresponding mode, it does achieve pretty good expected entropy compared with the COOLCAT algorithm. For COOLCAT, bigger m and larger buffer size make cluster quality better, but the execution time is longer. Only in the last setting does the COOLCAT algorithm achieve an entropy slightly better than ours, but the execution time is about 170 times longer than ours. We believe that speed is very important in processing data streams. Since our algorithm runs in much shorter time, it can be used to process data arriving at a much higher rate.

6 Future Work

We plan to use the differential algorithm to analyze real data sets and try to find the time-related changes of models in an efficient way.

Another interesting problem is how a stream mining algorithm should react when a stream gets turned "off" and "on", or rather, when there is a long period of time that no data arrive. The old model (compressed before this long pause) may get less important, or have different meanings, or expire completely. A stream mining algorithm should be able to handle these situations.

The stream processing also requires DBMS level supports for stream querying, data caching, incoming rate variance flattening, and so forth.

7 Conclusion

This paper proposes a differential categorical stream clustering algorithm based on several compression schemes. Categorical data stream clustering is a challenging problem due to the inherent complexity of categorical data processing. To process high speed streams, algorithms need to achieve high processing rate. Our algorithm can generate an approximate clustering result on the fly with reasonable accuracy. The differential clustering function is very effective in the periodic monitoring of a stream containing a large amount of data. Accuracy and processing rate are a trade-off to our compression schemes and one can switch among them to satisfy the application's requirements.

8 Acknowledgment

We would like to thank Professor Daniel Barbara and his group for providing us the executable of the COOLCAT algorithm.

References

1. Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proc. of SIGMOD 2003*, pages 575–586.
2. Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proc. of VLDB 2003*, pages 81–92.
3. Periklis Andritsos, Panayiotis Tsaparas, Renee J. Miller, and Kenneth C. Sevcik. Limbo: Scalable clustering of categorical data. In *Proc. of EDBT 2004*.
4. Daniel Barbara, Yi Li, and Julia Couto. COOLCAT: an entropy-based algorithm for categorical clustering. In *CIKM Conference*, 2002.
5. Paul S. Bradley, Usama M. Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In *Proc. of SIGKDD 1998*, pages 9–15.
6. A. Dobra, M. Garofalakis, J. E. Gehrke, and R. Rastogi. Processing complex aggregate queries over data streams. In *Proc. of SIGMOD*, 2002.
7. P. Domingos and G. Hulten. Mining high-speed data streams. In *Proc. of SIGKDD 2000*, pages 71–80.
8. Fredrik Farnstrom, James Lewis, and Charles Elkan. Scalability for clustering algorithms revisited. *SIGKDD Explorations*, 2(1):51–57, 2000.
9. Phillip B. Gibbons, Yossi Matias, and Viswanath Poosala. Fast incremental maintenance of approximate histograms. In *Proc. of VLDB 1997*, pages 466–475.
10. Sudipto Guha, Nick Koudas, and Kyuseok Shim. Data-streams and histograms. In *ACM Symposium on Theory of Computing*, pages 471–475, 2001.
11. Sudipto Guha, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams: Theory and practice. *IEEE TKDE*, 15:515–528, 2003.
12. Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: an efficient clustering algorithm for large databases. In *Proc. of SIGMOD 1998*, pages 73–84.
13. Zhexue Huang. Clustering large data sets with mixed numeric and categorical values. In *Proc. of PAKDD 1997*, pages 21–34.
14. G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *Proc. of ACM SIGKDD 2001.*
15. Spiros Papadimitriou, Anthony Brockwell, and Christos Faloutsos. Adaptive, hands-off stream mining. In *Proc. of VLDB 2003*, pages 560–571.
16. G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, 1983.
17. Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proc. of SIGKDD*, pages 226–235, 2003.

S-Monitors: Low-Cost Change Detection in Data Streams ⋆

Weiyun Huang[1], Edward Omiecinski[1], Leo Mark[1], and Weiquan Zhao[2]

[1] College of Computing,Georgia Institute of Technology, Atlanta GA, USA
{wyhuang,edwardo,leomark}@cc.gatech.edu
[2] School of Computer and Information Science
University of South Australia, Australia
weiquan.zhao@cs.unisa.edu.au

Abstract. Change detection in continuous data streams is very useful in today's computing environment. However, high computation overhead prevents many data mining algorithms from being used for online monitoring. We formalize the change detection problem and propose several metrics to evaluate change detection algorithms. We then present a novel low-cost approach to detect changes of models in streams and demonstrate the advantages of this approach using subspace cluster monitoring as an example. Our experiments on both synthetic and real world data show that this approach can catch more changes in a more timely manner with lower cost. The same approach can be applied to different models in various applications, such as monitoring live weather/environmental data, stock market fluctuations and network traffic streams.

Keywords: Stream, change detection, low-cost, s-monitor

1 Introduction

With the development of network, data management and ubiquitous computing technology, data streams have become an important type of data source and attracted many people's attention [2, 6, 9, 8, 11]. A data stream is a sequence of data points which usually can only be read once and does not support random access. Generally the data points are time ordered. Change detection in data streams has become a popular research topic in the data mining community [10, 12, 13, 3, 4, 16]. Weather changes monitoring and stock quotes watching are obvious examples.

Although stream data change detection is very useful, very often its difficulty prevents it from being widely used. Change detection usually involves two steps: model generation and model comparison. We call this a "compute-and-compare" approach, or "C&C" for short. This approach repeatedly generates models from the data stream and then compares them to see if there is any change among these models. This process incurs more effort than mining one model. In some cases, because of the inherent complexity of the models, the algorithms to generate them cannot process data at a sufficiently high rate. Every time we check

⋆ This research has been partially supported by National Science Foundation Grant CCR-0121643.

the stream for a possible change, we have to first call the algorithm (possibly with the help of a huge buffer) and then do the model comparison. Therefore we may fail to detect any short-lasting change, and even if we do discover the change, it may be too late.

In this paper, we propose a new approach to tackle this difficult problem. By putting "monitors" into the stream of the data, we try to avoid the expensive step of model generation as much as possible.

An analog of our approach is the deployment of sensors for environmental research. People put sensors in a river to monitor the water temperature. Although it might be interesting to know the temperature changes in any location, only a few sensors are placed in the most representative locations. Hopefully, these sensors can reveal most of the knowledge people want to discover, although they do not cover all the locations in the river.

Our approach treats a data stream like a river. We make use of any knowledge to put "monitors" at the locations that most likely reflect the change. We call this type of "monitor" *stream-monitor*, or *s-monitor*. An *s-monitor* with respect to an expensive data model is a simple model that costs much less to compute, and its change reflects the change of the expensive model.

In brief, our contribution in this paper includes the following:

1. We define the concept of change and propose metrics for the evaluation of change detection algorithms.
2. We propose a low-cost approach for change detection for expensive models: putting s-monitors into data streams, i.e., when model generation is expensive, we can limit our focus to "important" aspects of data only and avoid frequently generating new models for all the data.
3. As an example, we argue that subspace clustering can provide essential insight to the data and is a good tool to analyze streams; and we can place s-monitors into the data stream to handle the complexity issue.
4. We provide the data structure and the algorithm to efficiently detect the subspace cluster changes.
5. Our experiments show the effectiveness of our algorithm for real world data.

This paper is organized as follows. Section 2 reviews the related work on change detection. In section 3 we define the problem of change detection and propose the framework of our method. Section 4 discusses our approach in detail using subspace change detection as an example. Section 5 presents experimental results on synthetic and real world data. Section 6 addresses the future work and section 7 concludes the paper.

2 Related Work

Recently more and more attention has been paid on mining the evolution of the data [13, 3, 4, 16]. Aggarwal [3] uses the velocity density estimation concept to diagnose the changes in an evolving stream. Wang et al. [16] use ensemble methods to detect concept drift. Aggarwal et al. [4] also propose a framework for clustering evolving numerical data streams with the use of both an on-line algorithm and an off-line processing component. Kifer et al [13] lay theoretical foundation by designing statistical tests for one dimensional data. Our approach is different from previous research in that we try to tackle the complexity problem by strategically choosing part of data to process and/or doing low-cost model processing.

Subspace clustering algorithms [5, 14, 15] are very effective in high dimensional data sets. The MAFIA algorithm is one such algorithm. Like many other subspace clustering algorithms, it divides each dimension into bins and find dense 1-D bins first, then uses them to generate 2-D grids and find dense ones, then goes up to three dimensions, and so forth. This process requires multiple passes of data. Theoretical results [7] show that in high dimensional space (could be as few as 10-15 dimensions) "nearest neighbor" can be meaningless, which causing difficulties for many clustering algorithms. High dimensional data sets often contain many outliers, which can result in poor accuracy for partitional clustering algorithms. Also, in many cases it is unrealistic to assume a certain shape for the clusters, which some algorithms do. Subspace clustering algorithms can capture arbitrary shapes of clusters, and the results do not depend on the initialization of the clusters. The generated model is easy to interpret.

For stream analysis, subspace clustering's high complexity is a major problem. We use subspace cluster monitoring as an example to demonstrate how to integrate low-cost s-monitors with expensive model generation algorithms.

Aggarwal et al. [1] propose a framework to maintain "fading cluster structures" for data streams and compute projected clusters for current and historical data. This work examines subsets of dimensions as we do, but it is different from our research. We focus on detecting changes from sliding windows of data, rather than designing a clustering algorithm. Our subspace clustering example uses low-cost s-monitors together with an off-the-shelf algorithm to detect the changes in the data.

3 Detecting Changes without Model Generation

3.1 Definitions and Metrics for Change Detection

Although there has been some interesting work on change detection, basic questions, such as what is a change and how to measure a change, have not been formally addressed by previous work. We believe that the effort of formalizing the problem and proposing corresponding metrics to evaluate change detection algorithms are very important for the long term development of this domain.

We discuss this problem in the context of a data stream with a sliding window (figure 1). A "current window"(the shaded rectangle in the figure) slides on the data stream. Here a window is the portion of data in the stream that enters the system within a certain range. This range can be defined by a time interval, or a certain number of points. As time goes by, the current window is changing, and any two windows W_i and W_j may or may not overlap. The C&C approach generates models for different windows and if they are different, then the monitoring system will report a change and invoke change handling routines. The end of each current window is essentially the point where the stream is being checked (i.e., checkpoint). To be general, we do not require the set of "current windows" to cover the whole stream. The models to be compared do not have to be models generated from consecutive windows – models from any two windows can be compared as long as the application requires. However in the rest of the paper we assume that we always compare a model with the one generated from its previous window. We call the distance between the end of these two windows as the "check interval".

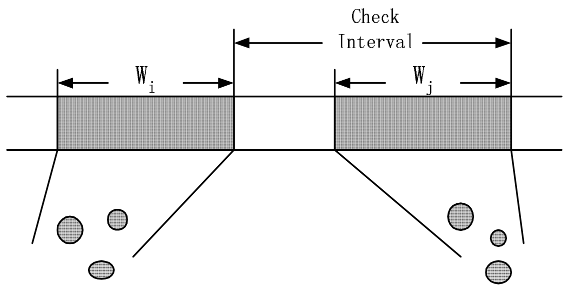


Fig. 1. Data stream, current window and check interval

When we talk about changes, we refer to the difference between an earlier state and a later state. In data mining applications, the state is essentially the model people are interested in. Therefore, a change is always associated with a type of model.

Definition 1. *Change: A change is defined with respect to a certain model. A change of model implies that there are sufficient differences in the model before the change and the one after the change, so that the descriptions of these two models are different.*

The "model" in the definition includes not only the type of the model but also the corresponding parameters, which decide the accuracy, error bounds, and so forth.

Definition 2. *Detection Delay: the time from the point when change happens to the point when the change detection algorithm recognizes the change.*

Definition 3. *Detection Rate: the percentage of changes that are actually captured by the change detection algorithm.*

Check interval, detection delay and detection rate are correlated. A long check interval results in long detection delay, and possibility of missing some changes. Obviously, a good change detection algorithm should have short detection delay and high detection rate.

Definition 4. *Sensitivity: The minimal size of change that can be captured by the algorithm.*

While change, detection delay and detection rate can be defined for general problems, sensitivity is an application specific concept. The exact meaning of sensitivity varies with the applications even for the same model. Let's consider change of two-dimensional clusters. If we are more interested in the area of the clusters, then can define the size of change as the sum of non-overlapped areas between old clusters and new clusters; if what we are interested is the location of clusters, the size of change can be defined as the sum of distances between old cluster centers to their closest new cluster centers.

Once the metric of change has been decided, we can use the corresponding sensitivity metric to evaluate change detection algorithms. For instance, we can tellthat an algorithm A is superior to algorithm B if it guarantees shorter detection delay with same performance on other metrics.

There are many problems to study in the change detection domain. Here we did not consider the "false positive" case, i.e., we assumed that all the changes reported by an algorithm are actually changes. What we discussed above are the "first order" concepts, such as the metric of change. There are higher order concepts such as the velocity of the changes, or even the acceleration of the changes. These higher order concepts can be defined in many different ways, but are beyond the scope of this paper.

3.2 Change Detection without Knowing Current Model

Table 1 contains some notations that are used in the rest of the paper.

M	the expensive model
ms	the set of s-monitors
CI_l	the check interval for M
CI_s	the check interval for ms
C_M	the cost of generating model M and check it for changes
C_{ms}	the cost of checking s-monitors.
C_c	the cost of creating monitors.
$\mathcal{C}$	the algorithm to generate M for a window

Table 1. Notations

While a stream is flowing, our change detection approach analyzes the sliding windows in the following steps. We use variable nms to denote the number of s-monitor checkpoints since last generation of M. Here CI_l, CI_s and r are user-adjustable parameters. Their functions will be explained later in this section.

1. Run $\mathcal{C}$ to the current window and get its corresponding model M.
2. If there is a previous value of M, compare the current M with the old one. If there is a significant change (meaning the difference between two models reaches a threshold), go to step 6.
3. With the insights provided by M, define s-monitors and deploy them into the data stream. These s-monitors are easy to compute, and the change of ms will reflect the change of M (however, a change of M may not necessarily cause the change of ms). Set $nms = 0$.
4. If $nms > r$ and the time since last M checkpoint reaches CI_l, go to step 1.

5. If the time since last checkpoint (either M or ms) reaches CI_s, recompute ms and compare with the previous s-monitor model. If a significant change of ms is detected, report the change and go to step 7; otherwise set $nms = nms + 1$ and go to step 4.
6. Execute change handling routine and then go to step 3.
7. Execute change handling routine and then go to step 1.

In the last step, when a change is detected, some predefined steps are taken to handle the change. These steps can include starting $\mathcal{C}$ as early as possible (since the system resource may not allow the immediate launch of the algorithm) to generate new models, or other diagnostic routines.

Figure 2 shows two example scenarios in which we can benefit from our approach. Suppose we are watching a one-dimensional data stream. The C&C approach generates expensive models $M1$ and $M2$, but our approach can check the stream more frequently using ms. The arrows in figure 2 indicate the checkpoints for both models. In figure 2(b), after getting $M2$, the C&C approach detects the change, while we can do after $ms3$. In figure 2(a), because the stream changes back to the initial state before $M2$ is obtained, no changes can be detected by C&C, while we can still report a change after $ms3$.

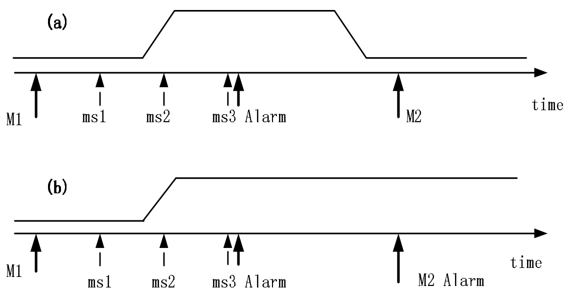


Fig. 2. Change detection examples

Obviously, to best discover the underlying changes, the third step is very important. What are the requirements of ms? To what extent can we benefit from those s-monitors? Below we show some analysis.

3.3 Detection Delay

Assume that changes can happen at any time point, with uniform probabilities. We test ms for r times after run C once. For the sake of simplicity we assume that there is no "failure to detect", i.e., if a change happens, the following checkpoint (no matter ms or M) will capture it. We will discuss "failure to detect" a little later. Obviously, $CI_l >= C_M$.

For the C&C approach, let's look at the period of time CI_l starting from right after one checkpoint (figure 3(a)). A change can happen any time within this period and it will be captured at the end of the period. Thus the average detection delay is

$$\int_0^{CI_l} \frac{1}{CI_l}(CI_l - x)dx = \frac{1}{2}CI_l \tag{1}$$

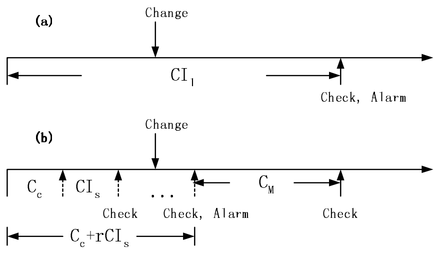


Fig. 3. Detection Delay

For the s-monitor approach, let's look at the time interval of length $C_c + rCI_s + C_M$, starting from right after one checkpoint, as shown in figure 3(b). First C_c time is to create the s-monitors. A change will be captured at the end of the next sub-intervals of length CI_s. On the other hand, if it happens after the r^{th} s-monitor checkpoint, then it cannot be detected until a new model M is generated. Therefore the average detection delay is

$$\begin{aligned}
&\int_0^{C_c+CI_s} \frac{C_c + CI_s - x}{rCI_s + C_c + C_M} dx + (r-1)\int_0^{CI_s} \frac{CI_s - x}{rCI_s + C_c + C_M} dx \\
&+ \int_0^{C_M} \frac{C_M - x}{rCI_s + C_c + C_M} dx \\
&= \frac{(C_c + CI_s)^2 + (r-1)CI_s^2 + C_M^2}{2(rCI_s + C_c + C_M)} \qquad (2)
\end{aligned}$$

Now we show that by choosing appropriate r and CI_s, we can achieve much smaller average detection delay than the $C\&C$ approach.

Our pre-condition is that it costs a lot less to create or check a set of s-monitors than to compute M. Based on this it is safe to say $C_c << \frac{1}{2}C_M$ and $C_m << \frac{1}{2}C_M$. Then we can set $CI_s = max(C_c, C_m)$ and it satisfies $CI_s << \frac{1}{2}C_M$. Now if we choose $r > 3$, then we can have $CI_s(1 + \frac{3}{r}) << C_M$. We can use this conclusion in equation 2 and get:

$$\begin{aligned}
\frac{(C_c + CI_s)^2 + (r-1)CI_s^2 + C_M^2}{2(rCI_s + C_c + C_M)} &< \frac{(2CI_s)^2 + (r-1)CI_s^2 + C_M^2}{2(rCI_s + C_c + C_M)} \\
&< \frac{(r+3)CI_s^2 + C_cC_M + C_M^2}{2(rCI_s + C_c + C_M)} \\
&<< \frac{rC_MCI_s + C_cC_M + C_M^2}{2(rCI_s + C_c + C_M)} \\
&= \frac{1}{2}C_M <= \frac{1}{2}CI_l \qquad (3)
\end{aligned}$$

Therefore, the average detection delay of the s-monitor approach is significantly less than the $C\&C$ approach.

3.4 Detection Rate

In the detection delay analysis we assume that every change (once happened) can be captured by the following checkpoint. This ideal case will not happen in most of the applications. Sometimes it is difficult to find all the possible changes using s-monitors, and the $C\&C$ approach may also miss a lot of changes due to the effect shown in figure 2(a). We can only say that in some cases the s-monitor approach will have a guaranteed high detection rate, while in some other cases the $C\&C$ approach can have superior detection rate.

The scenario we describe is as follows. The average change rate is x. Here the "change" is with respect to the state at the last M checkpoint. And the average lifetime for a change is μ. Assume that at the checkpoint, if there is a change present (meaning it happened and its lifetime did not expire), M can definitely capture it, while ms has $\delta\%$ probability of detecting it. Here $\delta\% <= 1$.

Consider the situation when $\mu < C_M$ and $\mu > 2CI_s$(although generally C_M can be either time cost or space cost, here we consider it as a time cost). Then the performance of the s-monitor approach will depend on $\delta\%$. When $\delta\% > \frac{\mu}{C_M}$, we can prove that the s-monitor approach can capture at least as many changes as the $C\&C$ approach.

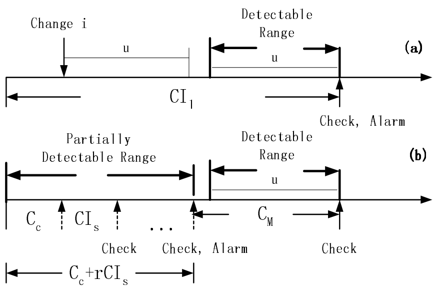


Fig. 4. Detection Rate

For the $C\&C$ approach, as in figure 4(a), if a change (change i for instance) happens too early, it cannot be detected at the end of the CI_l interval. Therefore the detectable range for the $C\&C$ approach is of length μ. While in the period of time CI_l, there should be altogether xCI_l changes that happened. Therefore the detection rate is

$$\frac{x\mu}{xCI_l} = \frac{\mu}{CI_l}$$

.

For the s-monitor approach, we can say that during the first $C_c + rCI_s$ time, $\delta\%$ changes can be detected, while at the last μ time, 100% of the changes can be detected. Therefore the detection rate is

$$\frac{x(C_c + rCI_s)\delta\% + x\mu}{x(C_c + rCI_s + C_M)} > \frac{(C_c + rCI_s)\frac{\mu}{C_M} + \mu}{C_c + rCI_s + C_M}$$
$$= \mu\frac{C_c + rCI_s + C_M}{C_M(C_c + rCI_s + C_M)}$$

$$= \frac{\mu}{C_M} > \frac{\mu}{CI_l} \tag{4}$$

This analysis shows us that when the changes are relatively short-lived, the s-monitor approach has a better chance to achieve a higher detection rate than the $C\&C$ approach. This is consistent with our intuition.

Detection delay, detection rate and sensitivity are three important metrics. In practice we need to find good instances of these metrics with respect to the application, so as to evaluate change detection algorithms.

3.5 Define S-monitors from Expensive Models

The relationship between s-monitors and expensive models is similar to the relationship between a sample set and the original data population. However, defining s-monitors with respect to an expensive model is not trivial. Below we provide several starting points to define s-monitors.

Define s-monitors from the building blocks of the model. Some expensive models are generated layer by layer through multiple iterations. A layer at a low level, or the intermediate result in an earlier iteration can be a set of s-monitors.

Define s-monitors from attributes of the model. Although some models are expensive to compute, they may have some attributes that are easy to check and a changed attribute implies a changed model. This kind of attribute may serve as s-monitors.

Define s-monitors from statistics/summary of data. There are some well-known statistics that are often used to describe the data distribution, such as mean, standard deviation, and histograms. If such simple statistics can reflect the change of the model, then they can be candidate s-monitors.

Use domain knowledge or historical results. Domain knowledge or historical information can often tell us where changes are most likely to happen, and what kind of changes are more important. The analog of the former case is that people usually place surveillance cameras in the doorway where people usually enter/exit the building, while few like to watch solid walls.

4 Detecting the Changes of Subspace Clusters

In this section, we demonstrate our approach by applying it to change detection for subspace clusters.

4.1 Sensitivity Definition

We first want to define what sensitivity is in our application scenario. We are looking for dense subspaces in data windows (a subspace with density higher than threshold is called "dense", otherwise "sparse"). If between two checkpoints, a subspace turns from dense to sparse or vice versa, then it is a change.Therefore the sensitivity of the change can be defined using the size of the smallest subspace. For example, for a d-dimensional data set, if a subspace clustering algorithm divides each dimension into bins so that it divides the whole data space into d-dimensional grids, then the change within one d-dimensional unit is the smallest change the algorithm can detect.

4.2 Monitor Selection

We use the MAFIA algorithm [14] as the model generation algorithm $\mathcal{C}$. The subspace clusters make up the high cost model M.

Our goal is to discover a change, even though there may be 10 changes happening altogether. As long as we can detect one, our goal is achieved, because it will cause the change handling routines to be executed. Therefore we focus on the building blocks of subspace clustering – the candidate subspaces, and choose some of them as our s-monitors. Dimensions and bins essentially divide the whole data spaces into multidimensional grids and subspaces are no more than grid cells or a collection of grid cells. In the extreme, if we can set each grid cell as an s-monitor and watch each s-monitor in time, then we can detect any change. However, that could be more expensive than $\mathcal{C}$ itself. We need to do things fast, and also with limited system resources (such as memory). Therefore we only choose a subset of the grids/collection of grids, and try to choose the ones that are most likely to change.

Our heuristic is that **changes of the subspace clusters are most likely to happen around previous clusters**. This assumption implies that changes gradually happen, i.e. it is more likely to see clusters moving a short distance, or expanding/shrinking from original size, than emerging from a totally sparse area. This assumption is often true in the real world, especially when the check interval is short. Based on this assumption, when we define s-monitors for subspace clustering, we keep track of the candidate subspaces generated by MAFIA and record their density counts. When the number of such spaces is too large, we sample them. When we do the sampling, we allocate more spaces to highest dimensional subspaces (so that the sensitivity of the detection can be higher) and lowest dimensional subspaces (so as to catch more "big" changes), while the subspaces whose dimensionality is in the middle get less space. We call this the "two-ends" strategy. By doing so, the detection sensitivity of our approach is bounded by the highest number of dimensions that M has checked last time. We also tried other sampling strategies such as random sampling, sampling in favor of high or low dimensional subspaces, and choosing subspaces whose density are the closest to the threshold. Our experiments show that the "two-ends" strategy works the best to the data sets we used. Due to page limitations we do not report the results for other strategies.

For the selected subspaces, we record how far their densities are from the threshold. Then for the next window, we can see whether there is an s-monitor density change by scanning the data once.

Since we know the candidate subspaces in every dimension, we use a prefix tree to accelerate counting. Each node's key is a pair of values, i.e. dimension number and bin number. Each node has two pointers, descendant and sibling. Figure 5 shows an example. This prefix tree contains two 4-dimensional subspaces $\{(1,3)\ (8,4)\ (15,4)\ (18,1)\}$, $\{(1,3)\ (9,2)\ (15,4)\ (18,1)\}$ and one 5-dimensional subspace $\{(1,3)\ (8,4)\ (9,2)\ (15,4)\ (18,1)\}$. Dashed lines point to sibling and the solid lines point to descendants. Each path from root to leaf is an s-monitor.

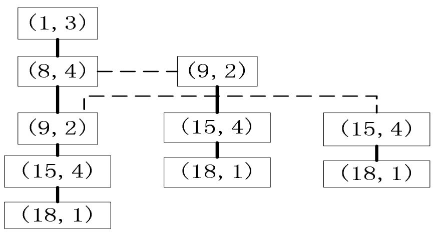


Fig. 5. Prefix tree containing three subspaces

4.3 Complexity Analysis

Using a prefix tree can substantially reduce the number of comparisons in the counting process. In the above example, without the prefix tree, the given subspaces require 26 comparisons altogether (for each bin we have to compare the two boundaries to know if the point falls within it). With the prefix tree, since the nodes are sorted by the order of dimension number and bin number, we usually do not have to reach the leaf level of the tree to know that a point does not fit in any of the subspaces. Therefore, suppose that a point can stop at any node in the tree with equal chance, the average number of comparisons is only 10. This is a luxury MAFIA or other exhaustive subspace clustering algorithms do not have since they do not know a priori what subspaces they are watching. Another reason our s-monitors can save a lot of time is that they only need to scan the data once, while MAFIA has to run multiple iterations and scan data multiple times.

In our experiments (results reported below), the number of s-monitors is set to 50. The number of s-monitors is essentially bounded by the time and space the end user wants to spend on s-monitors. For a window with n d-dimensional points and m s-monitors, the time cost of s-monitor is $O(nmd)$, and the space cost is bounded by md_{max}, where d_{max} is the maximum number of dimensions an s-monitor has. The maximum number of s-monitors is thus bounded by the resources, and users can choose a number they feel is sufficient. In our experiments, we choose 50 s-monitors because they work well in terms of detecting changes, and they incur little cost. When necessary the number of s-monitors may be dynamic.

5 Experiments

We use the MAFIA algorithm as our subspace clustering algorithm. The only change we made is to let it record some intermediate results generated by MAFIA and use those to help our processing.

Here we apply our subspace change detection algorithm to a synthetic data set and a real data set to demonstrate its performance. The experiments are performed on a Pentium IV 2.80GHz PC with 1GB memory, running Redhat 7.2. Both data sets are in binary format.

5.1 Synthetic Data

We create a synthetic data set to demonstrate the effectiveness of our approach. We fix a density threshold $\tau = 10\%$ and set the number of dimensions to 20.

Then we create 10 subsets of data with the following predefined subspace cluster sets (we call them "seed sets"):

1. Base set: seed set 1 contains 4 5-dimensional clusters Cl_1 to Cl_4.
2. Change of one dimension: seed set 2 contains Cl_2 to Cl_4 from set 1, and another cluster Cl'_1 obtained by changing one dimension of Cl_1.
3. New cluster: seed set 3 contains all the four clusters from seed set 2, and a new cluster Cl_5.
4. New cluster with higher dimensionality: seed set 4 contains all the five clusters from seed set 3, and a 6-dimensional cluster Cl_6.
5. Removing a cluster: seed set 5 contains Cl'_1,Cl_2,Cl_4 and Cl_6.

We use each seed set to generate 2 data subsets, each with 100000 points. The number of points in each predefined clusters is above 15%. Within each subset, data points in subspace clusters and not in clusters are randomly mixed together. Then we concatenate these 10 subsets into a data set and use it to simulate a stream.

In practice we need to set the check interval according to the stream rate. One extreme is to check the stream whenever it is possible, that is, once a check is done immediately start the next one. But in many cases, the check is done with certain idle time – i.e., the check interval CI is larger than the execution time of the check process and there is some delay between every two checkpoints. Therefore the performance value of the algorithm depends on the value of CI_l and CI_s. We arrange an M checkpoint at the end of each of the 10 subsets, both for the $C\&C$ approach and our s-monitor approach; and we also set r=4, so that we can put 4 s-monitor checkpoints before generating and checking M, and the distance between two consecutive checkpoints is 20000 points. Once an ms detects a change, we will not perform the next ms checks; instead we just wait until it is time to run M and generate the models. We do so because we want to fix the position of those checkpoints so that the comparison between the s-monitor approach and the $C\&C$ approach is fair. Even with the same data set, different checkpoint positions may cause an algorithm to detect different number of changes. If we arrange an M checkpoint immediately after one s-monitor detects a change, then the rest of the M checkpoints are moved and it is unfair to compare the number of changes our algorithm detected with the number obtained by $C\&C$, because they did not check the same windows of the data. We report the processing time C_M, C_{ms} and C_c.

Table 2 shows the result of one execution of our algorithm. The first column shows the range of points within which each window **begins**. For example, in the first row of the table, M is generated from window 0-100k and ms_1 is generated from window 20k-120k, and ms_4 is from window 80k-180k. Therefore according to the creation of the synthetic data, we expect changes to happen in the intervals described by the 100-200k, 300-400k, 500-600k and 700-800k rows in the table. The MAFIA algorithm confirms this, i.e., the corresponding M checkpoints of both s-monitor and C&C approaches detect those changes. The right 5 columns in the table show whether each checkpoint has detected changes. A "-" in the table means that no check is done. The first entry is "-" because there is nothing to compare with.

The data points in each window have been randomly reordered, so that clustered data and non-clustered data are mixed. This allows, for a change from sparse to dense in subspace A, A's density count to reach the threshold at any time during this window. As we mentioned earlier, we consider the time that A's density count reaches the threshold as the time when the change happens. Table 2 shows that s-monitors have successfully detected 3 out of 4 changes before M does. In the row of "300-400k", none of the s-monitor checkpoints report the "new cluster" change. A closer look reveals that it is because the location of the change is not covered by the s-monitors.

Multiple runs of our algorithm shows similar results to that reported in table 2.

Points range	M	ms_1	ms_2	ms_3	ms_4	Points range	M	ms_1	ms_2	ms_3	ms_4
0-100k	-	N	N	N	N	500-600k	N	N	N	Y	-
100-200k	N	N	Y	-	-	600-700k	Y	N	N	N	N
200-300k	Y	N	N	N	N	700-800k	N	N	Y	-	-
300-400k	N	N	N	N	N	800-900k	Y	N	N	N	N
400-500k	Y	N	N	N	N	900k-end	N	N	N	N	N

Table 2. Effectiveness: one execution of the s-monitor algorithm

Now let's look at how much extra cost the s-monitor approach incurs (table 3). We run our algorithms 5 times and compute the average C_M, C_{ms}, and C_c. The column "No. M" contains the number of times M is executed in each run, and "No. ms" is the number of times s-monitors are checked. For example, in the first row, 0.00122 is the total time in the first run spent on 10 times of s-monitor creation. $\sum C_M$ is obtained by $\sum(C_M + C_c) - \sum C_c$. Since we fix the number of s-monitors, C_{ms} has relatively small variation.

Runs	$\sum(C_M + C_c)$	No. M	$\sum C_c$	$\sum C_M$	$\sum C_{ms}$	No.ms
1	14.157401	10	0.00122	14.156181	3.988114	34
2	14.294491	10	0.001212	14.293279	4.172361	36
3	14.076228	10	0.001223	14.075005	4.093249	36
4	14.03696	10	0.001204	14.035756	3.72233	33
5	14.089386	10	0.001193	14.088193	3.711843	33
Total	-	50	0.006052	70.648414	19.687897	172
Avg. time(s)	-	-	0.000121	1.4130	0.1145	-

Table 3. Cost of s-monitor (average of 5 runs)

We can see that the average running time of ms is only 8% of the cost of M, and the space requirement is also much smaller. By adding $r = 4$ s-monitor checkpoints before each M checkpoints, we only increased the processing time by 28%, while we can detect changes much earlier. Therefore it is beneficial to use s-monitors.

5.2 Real Data

For the experiment with real data we use the sea surface time series data from the TAO project [3] of the Pacific Marine Environmental Laboratory (PMEL). We selected data for the year 2003, from 17 sensors located between 155 and 180 degrees west longitude, and 8 degrees north and 8 degrees south latitude. The data are sampled every 10 minutes (the highest frequency PMEL now provides). This data set is not a very large data set and our algorithm can handle a much higher data rate. We use this data set to demonstrate that our algorithm can work well under real world situations.

We put every two months' data into one current window and apply our algorithm. We choose 50 s-monitors and set $r = 4$. It turns out that the subspace clusters in this data set are rapidly changing and the changes are big. Therefore every time we run ms, a change is detected (and the change is confirmed by MAFIA). Table 4 shows the average execution time of 5 runs of this experiment. Each row corresponds to a two-month window. The "No. spaces" column shows the number of subspaces (2-dimensional and above) checked by MAFIA (while our algorithm checks 50 subspaces every time). For example, the first row means that to process the data from January to February, MAFIA checks 4891 subspaces, and it takes 1.2011 seconds in average, while s-monitor creation takes 0.0009 seconds and to check s-monitors takes 0.00382 seconds. Our s-monitor creation step selects subspaces from all the subspaces checked by MAFIA. Therefore C_c increases when MAFIA checks more subspaces. As shown by the table, C_c and C_{ms} are significantly less than the execution time of MAFIA. Hence we can detect changes early with little overhead.

C_M(s)	C_c(s)	C_{ms}(s)	No. spaces	C_M/C_{ms}
1.2011	0.0009	0.00382	4891	314
5.186	0.0025	0.00276	14657	1879
0.92512	0.0007	0.00354	3863	261
1.56046	0.0010	0.00298	5720	524
0.89138	0.0007	0.0032	3769	279
1.02864	0.0008	0.00412	4290	250

Table 4. Two-month window experiment

6 Future Work

The idea of deploying low-cost s-monitors into the data space can be applied to various applications. People often think of sampling when large amounts of data need to be processed, similarly, s-monitors can help in many occasions when expensive models are needed for stream change detection. We intend to apply our approach to other types of models and to look for good s-monitors with theoretically guaranteed accuracy.

7 Conclusion

In this paper, we first propose metrics for evaluating change detection algorithms, and then present a new approach for data stream change detection, with

[3] http://www.pmel.noaa.gov/tao/index.shtml

respect to the widely used "compute-and-compare" approach. When the stream monitoring task requires the repeated generation of complicated models, we can interleave the model generation with low-cost s-monitor checking, so that we can detect more changes in a more timely manner. We take subspace cluster monitoring as an example and demonstrate the effectiveness and performance of our approach.

8 Acknowledgment

We would like to thank Professor Alok Choudhary and his group for providing us the source code of the MAFIA algorithm.

References

1. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for projected clustering of high dimensional data streams. In *Proc. of VLDB*, 2004.
2. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. On demand classification of data streams. In *Proc. of the ACM SIGKDD*, 2004.
3. Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proc. of ACM SIGMOD*, pages 575–586, 2003.
4. Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proc. of VLDB*, pages 81–92, 2003.
5. R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of ACM SIGMOD*, pages 94–105, 1998.
6. B. Babcock and C. Olston. Distributed top-k monitoring. In *Proc. of ACM SIGMOD*, pages 28–39, 2003.
7. K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbor meaningful. In *Proc. of ICDT*, pages 217–235, 1999.
8. Sirish Chandrasekaran and Michael J. Franklin. Streaming queries over streaming data. In *Proc. of VLDB*, pages 203–214, 2002.
9. Donald Carney et al. Monitoring streams - a new class of data management applications. In *Proc. of VLDB*, 2002.
10. Venkatesh Ganti, Johannes Gehrke, and Raghu Ramakrishnan. Mining data streams under block evolution. *SIGKDD Explorations*, 3(2):1–10, 2002.
11. Sudipto Guha, Nick Koudas, and Kyuseok Shim. Data-streams and histograms. In *Proc. of STOC*, pages 471–475, 2001.
12. G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *Proc. of ACM SIGKDD 2001.*
13. Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *Proc. of VLDB*, pages 180–191, 2004.
14. H. Nagesh, S. Goil, and A. Choudhary. MAFIA: Efficient and scalable subspace clustering for very large data sets. *Technical Report 9906-010, Northwestern University*, 1999.
15. Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explorations*, 6(1):90–105, 2004.
16. Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proc. of ACM SIGKDD*, pages 226–235, 2003.

Domain-Driven In-Depth Pattern Discovery: A Practical Methodology [1]

[1]Longbing Cao, [2]Rick Schurmann, [1]Chengqi Zhang

[1]*Faculty of Information Technology, University of Technology Sydney, Australia*
[1]*{lbcao, chengqi}@it.uts.edu.au*
[2]*Business Integrity Strategy Division, Centrelink, Australia*
[2]*rick.rt.schurmann@centrelink.gov.au*

Abstract. Traditional data mining is a data-driven trial-and-error process. The patterns discovered via predefined models in the above process are *generic patterns*. Generally, they are often not really interesting to constraint-based real business. In order to work out patterns that are of interest and actionable to the real world, *in-depth patterns* are often essential. This type of pattern discovery is more likely to be a business or industry domain-driven human-machine-cooperated process. The use of in-depth patterns requires the development of a more practical methodology, than is presently available for guiding real-world data mining. This paper proposes such a practical data mining methodology, referred to as *domain-driven in-depth pattern discovery* (DDID-PD). The main idea of the DDID-PD methodology is to mine in-depth patterns through domain-driven iterative human-machine interaction in a constraint-based context. Using this methodology as a basis, we demonstrate some of our work in mining in-depth correlations in Australian Stock Exchange (ASX) data and preliminary research on developing a quality knowledge base for Centrelink interventions. The deployment of DDID-PD to ASX data mining tasks has shown that the methodology is practical and has potential for further improving the analysis of large quantities of data to identify patterns for practical use by industry and business.
Keywords. Data mining methodology, domain-driven, constraint, interactive mining.

1 Introduction

Traditional data mining is a data-driven trial-and-error process [1] where data mining algorithms extract patterns from data via some predefined models. It targets fully automated mining processes, algorithms and tools [1]. A data mining system is expected to be an automated tool without human involvement and the capability to adapt to external environment constraints.

However, data mining in the real world, for instance, financial data mining and data mining in Centrelink, is highly constraint-based [2]. The constraints on real-world data mining are represented in terms of data constraints, domain constraints,

[1] This project is sponsored by CMCRC (Capital Market CRC, www.cmcrc.com) Australia and UTS Research Fund.

interestingness constraints, and rule constraints. Real-world patterns that are interesting to business are often hidden in a large quantity of data with complex data structures and source distribution (*data constraints*). The real-world business process, problems and requirements are often tightly embedded in domain-specific information and expertise (*domain constraints*). Often mined patterns are not interesting or actionable to business even though they are sensible to research, or there exists interestingness conflicts between academia and business (*interestingness constraints*). Furthermore, the rules automatically discovered from domain-specific data often do not make sense to business process or regulations, or they must be integrated with other business rules so that they can be deployed into real life (*rule constraints*).

To deal with the above-mentioned constraints in the real world, it is essential to cast off the superficial and capture the essential information from the data mining. Some real experience and lessons learned in artificial intelligence and pattern recognition [3], and integrated business intelligence for Telecom customer relationship management and fraud control [4, 14] have taught us the involvement of domain knowledge and even domain experts can assist with filtering subtle concerns while capturing incisive issues and driving a practical design. Similarly, in order to effectively mine and deploy interesting patterns from the aforementioned constraint-based context, the involvement of domain knowledge and experts and the consideration of constraints are essential for knowledge discovery on a neatly definable domain problem. Combining these aspects together, a sleek data mining methodology can be developed to find the distilled core of a problem and build a deep domain model for advising the process of real-world data analysis and preparation, the selection of features, the design and fine-tuning of algorithms, and the evaluation and refinement of mining results in a more effective way. This leads to the *domain-driven in-depth pattern discovery* (DDID-PD) framework.

The key ideas of the DDID-PD framework include:

(i) dealing with constraint-based context;
(ii) mining in-depth patterns;
(iii) supporting human-machine-cooperated interactive knowledge discovery; and
(iv) viewing data mining as a loop-closed iterative refinement process.

Dealing with the constraint-based context can improve the quality and effectiveness of data mining by extracting and transforming the domain-specific datasets in terms of guides taken from domain experts and their knowledge. In-depth pattern mining can discover more interesting and actionable patterns from a domain-specific perspective. In this framework, data mining and domain experts complement each other in regard to in-depth granularity via an interactive interface.

The involvement of domain experts and knowledge can assist in developing highly effective data mining techniques and reduce the complexity of the knowledge producing process in the real world. A system following the DDID-PD framework can embed effective supports for domain knowledge and expert feedback, and refine the lifecycle of data mining in an iterative manner. Therefore, DDID-PD can benefit the real-world knowledge discovery in a more effective and efficient manner, and support the discovery of more interesting and actionable patterns compared with a current data-driven data mining methodologies such as CRISP-DM [5].

Taking data mining in stock markets and Centrelink as instances, this paper introduces some preliminary work for mining deep correlations in stock markets and developing a quality knowledge base on interventions for Centrelink through the application of the DDID-PD methodology. These real-world in-depth analyses further show that the DDID-PD methodology has potential for discovering a deep-core domain model that adapts to complex and dynamic business processes and requirements.

The remainder of this paper is organised as follows. Section 2 outlines the DDIP-PD framework. Section 3 demonstrates some examples on mining in-depth correlations in real stock markets. Section 4 explains the development of a quality knowledge base for compliance purposes. It includes the articulation of a possible deployment of the DDIP-PD framework within Centrelink to undertake in-depth analysis on compliance related customer interventions. Section 5 concludes this paper and presents future work.

2 Related Work

Many data mining methodologies have been developed in academia and industry. They generally advocate the idea of directly mining patterns from data. Typical theories include constraint-based data mining [2], human-centered data mining [1,7], human-involved or guided data mining [6], and interactive mining [10, 1], and so forth. These theories to varying degree highlight different aspects and factors, for instance, the role of human being, in the pattern discovery process. In industry, a typical standard is the CRISP-DM [5]. It basically views data mining as an autonomous pattern discovery process directly from data. Its beauty is the introduction of domain knowledge and business understanding.

DDID-PD, on the other hand, discovers knowledge from data via the involvement and assistance of business, domain experts and their knowledge, and the interaction between mining system and experts. It takes data mining as a kind of human-machine-cooperated interaction in an iterative loop-closed manner. In this interaction, domain knowledge and the involvement of business experts are essential in the overall mining process rather than only in the steps of business and data understanding and/or mining result evaluation. It highlights the significance of mining in-depth patterns which are not only interesting to technicians but also business decision-makers. To make the domain-driven data mining effective, some friendly and intelligent human-machine interaction interfaces are essential for human-machine cooperation. In addition, appropriate system support mechanisms are required for dealing with multiform constraints and domain knowledge.

3 DDID-PD Framework

3.1 Fundamental concepts

In the DDID-PD framework, a collection of concepts are proposed in terms of practical requirements from the real world. These concepts attempt to bring new ideas and deep thinking into the existing data mining framework, and enhance the efficiency and effectiveness of real-world data mining.

DEFINITION 1: Generic Pattern. Refers to patterns automatically discovered by data mining models while taking little consideration of business requirements and interestingness.

For instance, in association rule mining, a large number of rules are often found while most of them might not make sense to business. These rules are called *generic patterns*. Another instance is stock trading strategies (also called trading rules), for example, Moving Average (MA), discovered by financial experts. MA actually represents a huge quantities of trading strategies. These strategies are generic rules since they are neither specifically developed for handling certain cases nor as effective as possible for daily trading decisions. Generic patterns may be interesting to data miners while not generally interesting to business for taking action.

DEFINITION 2: In-depth Pattern – Refers to patterns which are highly interesting and actionable in business decision-making. These patterns are created through refining models or tuning parameters to optimise generic patterns. They may also be directly discovered from data with sufficient consideration of business requirements and constraints.

In-depth patterns are not only interesting to data miners, but also to business decision-makers. For instance, in the afore-mentioned trading strategies, more actionable trading strategies can be found via model refinement or parameter tuning. We also call them *optimised strategies* compared with underlying strategies.

DEFINITION 3: Human-Machine Cooperation [3] – This is the in-depth pattern discovery conducted with the cooperation of business analysts and data analysts. Section 3.5 fits the human-machine cooperation concept into the data mining context.

DEFINITION 4: Domain-Driven Data Mining – In-depth pattern discovery is not only a data-driven trial-and-error process, rather it is highly domain-dependent. Domain-dependent doesn't mean that a pattern is specific to a domain. Rather it indicates that the in-depth pattern discovery process highlights the involvement of domain expertise and constraints in a human-machine cooperation context.

3.2 DDID-PD process model

The components of the DDID-PD framework are shown in Figure 1. The lifecycle of DDID-PD is as follows, but be aware the sequence is not rigid, some phases may be bypassed or moved back and forth in solving a real problem.

P1. Problem understanding and definition;
P2. Data understanding;
P3. Data preprocessing;
P4. Modeling;
P5. Results evaluation;
P6. Based on feedback and progress of the phases from P2 to P5, it is quite possible that each phase may be iteratively reviewed starting from P1 via the interaction with domain experts in a back-and-forth manner for the refinement of mining results;
P7. Results post-processing; or
P7a.: In-depth modeling on the mined results where applicable; then going to P7;

P8. Going back and reviewing phases from P1 may be required;
P9. Deployment;
P10. Knowledge and report delivery.

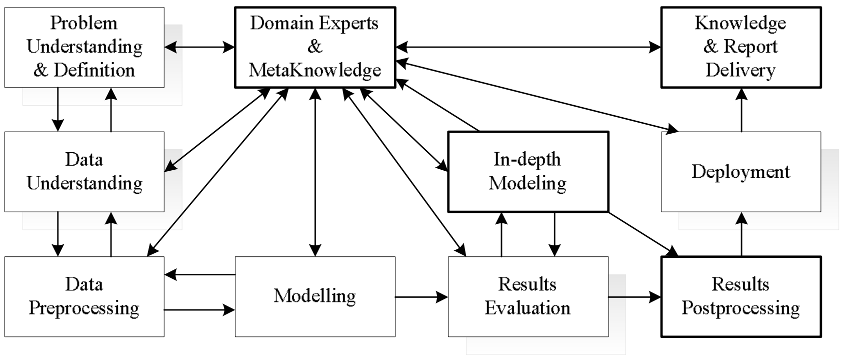


Fig. 1. DDID-PD process model

The DDID-PD process highlights four highly correlated ideas that are critical for the success of a data mining process in the real world. They are:

(i) *constraint-based context*, multiple types of constraints widely exist in the domain problem and its analysis objectives,

(ii) *in-depth pattern mining*, it could be through another round of modeling on the first-round results for mining patterns really interesting and actionable to business,

(iii) *human-machine-cooperated interactive knowledge discovery*, the involvement of domain experts and their knowledge and the interaction between experts and mining system in the whole process are important for effective execution of the mining, and

(iv) *a loop-closed iterative refinement process*, patterns that can be deployed and adopted for smart business decision-making are the outcome of iterative refinement.

The following sub-sections outline them individually.

3.3 Constraint-based context

In human society, everyone is constrained by either social (environmental) regulations or personal situations. Similarly, advanced knowledge discovery and smart decision-making need to consider real-world aspects such as environmental reality, expectations and constraints in the whole process. More specifically, the following four kinds of constraints play important roles in building effective and efficient data mining from requirements engineering to evaluation and refinement engineering. They consist of domain-specific, functional and environmental constraints, and form a constraint-based data mining context [2].

- Domain Constraints: it involves domain type, characteristics (eg privacy), business process and workflow, domain knowledge, human capability and role, qualitative and quantitative hypothesis and conditions, etc;

- Data Constraints: this is related to data quantity, data structures, data distribution, data semantic complexity, etc;
- Interestingness Constraints: this is driven either by academic objectives or business goals, or interestingness metrics covering both aspects, as well as problem requirements and analytical goals, etc;
- Rule Constraints: involved in rule representation, rule interestingness to analytical goals, rule explanation, rule deployment in the integration with real-world business process and environment, etc.

All the above constraints must be, to varying degrees, considered in real-world data mining. They are involved in the whole process of domain-specific data mining. In the development of data mining process and algorithms, we need to think of what they will bring to the improvement of traditional data mining, and what techniques or system supports can be used for utilising, analysing and avoiding these constraints. They must be closely connected to specific modeling methods, business environment, and analytical objectives in a systematic manner, i.e., *constraint-based data mining*.

3.4 Mining in-depth patterns

Existing data mining methods, for example association rule mining, often generate a huge number of patterns (or rules), but a majority of them are either redundant or do not reflect true interestingness from a business perspective. This has hindered the deployment and adoption of data mining in real applications. Taking trading rules in finance as an instance, a trading rule, for example MA(sr, lr, σ), usually implies millions of parameter combinations, i.e., rules. However, most of them are not actionable for a specific business environment. Therefore it is essential to further refine these rules so that more interesting and actionable rules can be discovered and recommended for smarter and more effective decision-making. To overcome this obstacle in deploying data mining into the real world, we need to discover more interesting and actionable rules highlighting business requirements and objectives. This leads to *in-depth mining*.

In-depth mining leads to deep models either through a further-round mining on existing (mined) patterns/rules or in selected/refined datasets. Obviously, the involvement of domain knowledge and constraints is often necessary for conducting in-depth mining. More importantly, some appropriate in-depth mining techniques should be developed in response to both technical and business objectives. For instance, in Section 4, we illustrate some of our work results in mining in-depth correlated patterns in the stock market environment.

3.5 Human-machine-cooperated interactive knowledge discovery

Real-world data mining should be a human-machine-cooperated interactive knowledge discovery process rather than an autonomous system. Domain experts are the centre of and an essential constituent of the data mining process via dynamic expert-model interaction. In fact, they and their knowledge play significant role in the whole data mining process such as business and data understanding, features selection, hypotheses proposal, model selection and learning, the evaluation and refinement of algorithms and resulting outcomes. For instance, domain experts can narrow down the selection of features and models, and create high quality hypotheses and efficient constraints based on their domain knowledge (especially their experience and imaginary thinking), which will effectively accelerate the mining process.

Instead of producing patterns or knowledge directly from data, the domain-driven data mining methodology allows domain experts and/or their knowledge to be at the front and centre of the mining process, and to interact with data and business via friendly interfaces and system supports to maximize the power of domain expert knowledge and capability in complex problem solving. For instance, domain experts can incorporate their knowledge (especially their qualitative experience and imaginary thinking) into data and feature selection, model analysis and building by generating effective qualitative hypothesis and constraints on business data and problems. This point may also be called *human-centered* [1, 7], *human-involved*, *supervised* or *guided* [6] data mining.

As discussed above, domain-driven in-depth data mining supports in-depth analysis with the assistance of domain knowledge. Furthermore, the mining is actually an interaction between domain-experts and the mining system. To support the dynamic interaction, user-friendly human-machine interfaces are necessary. The interface needs to support domain expert-mining system dialogue, so that knowledge from domain experts can be online and instantly embedded into the mining system and knowledge base on demand. This will allow for the refinement, fine tuning and improvement to the quality of the final mined rules. This makes the data mining process and tool highly interactive and dynamic rather than fully automated as previously imagined. For this commitment, the knowledge base including Expert Systems, Artificial Intelligence, Pattern Recognition and Cognitive Science needs to be involved. A good option would be to build an intelligent agents-based data mining platform [8, 9] to support user modeling, user interaction, and the like. This is similar to *interactive mining* [10, 1]. Essentially, issues such as how to support business-oriented user-friendly and personalized interaction, knowledge representation should be investigated in the interface and knowledge base design.

3.6 Loop-closed iterative refinement

The data mining process and its system is closed rather than open because it encloses iterative refinement and feedback of hypotheses, features, models, evaluation and explanations in a human-involved context. This real-world mining process is iterative because the evaluation and refinement of features, models and outcomes cannot be completed once, rather it is based on iterative feedback and interaction during the whole process. It iteratively evaluates and tunes features and models based on feedback from and the involvement of domain experts and their knowledge, and the interaction with the domain problem.

To support the loop-closed iterative refinement, some appropriate human-computer interaction interfaces and system supports should be designed. Again, to this end intelligent agents [8] can play a competitive role.

4 Mining in-depth correlations in real stock markets

Financial data mining [11] is practical but challenging in the business world, for example stock markets are very complex. Taking the ASX as an example, there are more than 1000 listed companies in this small market. Even though financial data analysis has been researched for decades, more actionable data mining on this data is still a research challenge with increasing interests from industry.

In the Data Mining Program (DMP) of Australian Capital Markets Cooperative Research Center (CMCRC) [12], the DDID-PD framework has been developed for mining in-depth correlations of Stock Market data. Its main function consisted of:

(i) high dimension reductions to generate a small quantity of data or rule representatives from a huge data set or rule combinations,
(ii) human-machine-cooperated interactive refinement to refine correlation coefficients based on domain-specific knowledge and objectives, and
(iii) in-depth pattern discovery to obtain the interesting correlations.

The correlation pattern mining in the stock order stream targets patterned interesting and actionable information for stock traders. Specially, we observer the in-depth correlation mining in stock market data to aim at finding correlations between stocks, searching correlated patterns from existing trading rules developed by financial experts in order to develop more actionable trading rules, and discovering correlated relations between trading rules and stocks.

In the following, we will present some results in utilising the DDID-PD framework to mine correlated stocks, in-depth trading rules, and trading rule-stock correlations based on ASX stock data.

4.1 Mining correlated stocks

In stock markets, for instance the ASX, brokers and retailers trade hundreds of stocks every trading day. It is a common hypothesis that there may be some form of correlation existing among stocks from the same or similar sectors, or belonging to a shared production chain. A typical example is pairs trading strategy. Pairs trading involves the purchase of one security while simultaneously selling (or selling short) another security when a pair of highly correlated securities deviates from the normal relationship between them. Correlation metrics as well as business constraints are considered in analyzing finding the paired stocks in a stock market. The following outlines the basic idea of the correlated stock mining algorithm.

```
ALGORITHM: Mining Correlated Stocks
C1. Calculating the coefficient $\rho$ of two stock;
C2. Determining the scope of $\rho$ interesting to real trading through cooperation
    between miners and traders via considering other domain-specific aspects;
C3. Evaluating the correlation between stocks via some additional domain-specific
    elements;
C4. Recommending correlated stocks.
```

In order to testify to the effectiveness of mined correlated stock pairs, a Pairs Trading strategy was developed and used to trade in the historical market orderbook.

Using the ASX as instance, we targeted 32 stocks with quality data from January 1997 to June 2002. 13 stocks were found to be highly correlated. In 78 pairs of combinations, nine pairs are found to be actionable to real trading. For instance, it was found that there was a high correlation in the pair of stock CBA and GMF. Without considering the market impact, the return on the pair CBA-GMF was 40.51% in average on historical data from 1 January 1997 to 19 June 2002.

In this exercise, we found the following interesting points:

- The analytics showed that the correlated stocks actionable to traders cannot just be specified by the coefficient;
- The correlated stocks mined in the ASX market all come from different sectors, meaning that correlated stocks are not necessary from the same industry as assumed by financial researchers; and
- The analytical results showed that the profit of trading a correlated pair is greatly affected by the liquidity and the volatility of stocks. Therefore, an actionable (profitable) stock pair is based on correlation, liquidity and volatility of the parties.

4.2 Mining in-depth trading rules

In stock markets, since a long time ago, both data mining and financial researchers have developed many trading rules to support traders' decision-making. These rules actually indicate possible patterns hidden in stock markets. For instance, the trading strategy MA actually indicates a correlated pattern between two features namely *short-run moving average* (*sr*) and *long-run moving average* (*lr*). The pattern MA (*sr*, *lr*, δ) is defined as follows (whereδis the fix band for the difference between *sr* and *lr*):

```
IF sr *(1-δ) >= lr THEN Buy
IF sr *(1+δ) <= lr THEN Sell
```

This pattern actually consists of a large number of rules (we call them *generic patterns*) from a finance perspective, for instance MA(2, 50, 0.1) and MA(10, 50, 0.1) represent two different MA rules. However, traders do not know which rule is actionable for their specific trading decision.

The in-depth pattern mining on existing trading rules aims to mine more actionable rules (i.e., *in-depth patterns*) which can better serve traders' objectives.

To discover in-depth rules from generic rules, a robust genetic algorithm [13] and a human-machine interaction interface were developed so that financial experts could dynamically and iteratively supervise and evaluate the training. Interfaces are developed so that business analysts can supervise the construction of some features, fine tune the parameters, and set evaluation criteria for the business requirements and objectives. Technical analysts can advise the above process as well as refining technical factors to narrow down the search space. Taking the MA as an instance, an in-depth rule MA(4, 19, 0.033) is found in the training data from 1 January 2000 to 31 December 2000 and testing set between 1 January 2001 and 31 December 2001. The number of trading signals generated by this rule is much more than other generic rules. Its Sharpe Ratio[2], as shown in Figure 2 (b), has a greatly improved positive scope compared with (a) the generic results. This demonstrates that the in-depth correlation mining with the involvement of domain knowledge can lead to more interesting and actionable rules for trading support.

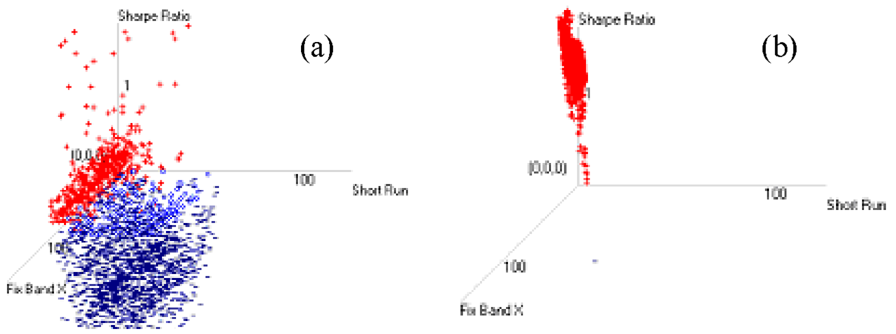


Fig.2. Improved business objective by in-depth rules
(a) Sharpe ratio with generic MA rules (b) Sharpe ratio with in-depth MA rules

4.3 Mining in-depth rule-stock correlations

It is assumed that some trading rules are suitable for a class of stock, while others are more effective to guide the trading of other stocks in the market. This hypothesis actually indicates whether there are correlations between trading rules and stocks. If yes, and if we can discover the correlation, then it would be very helpful for guiding the trading.

Based on this hypothesis, algorithms can be developed to search for the in-depth correlations between trading rules and stocks in stock market data. The basic ideas of the rule-stock correlation mining algorithms are as follows.

1) Mining in-depth rules for individual stock

For each ASX security, a set of in-depth rules are discovered for each class of trading rules by the algorithm described in Section 4.1. Furthermore, in-depth patterns can be discovered from all classes of rules for all stocks respectively. As a result, a rule-stock set is found in which a trading rule is matched with one or multiple stocks.

2) *Mining the highly correlated rule-stock pairs*

In the above step, multiple in-depth rules from different rule classes may be found suitable for one stock. It is necessary to discover a highly correlated rule for a specific stock from the above resulting set. This leads to the most suitable rule for a stock, and forms a correlated rule-stock pair.

3) *Refining and evaluating the rule-stock pairs*

In order to find the interesting and profitable rule-stock pair, the assistance of domain experts and their suggestions are essential for the refinement and evaluation of pairs found in the above steps.

[2] It is taken as a business interestingness benchmark for evaluating the performance of a trading rule in the real world.

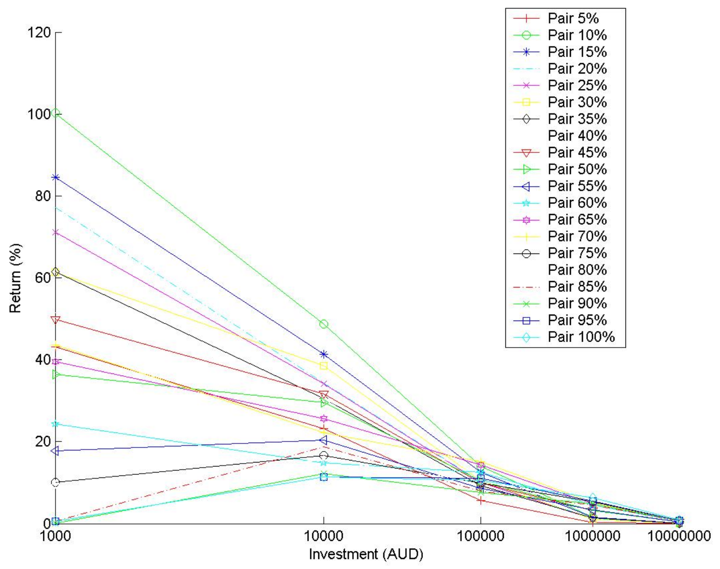


Fig.3. Return on investment with in-depth rule-stock pairs

In the analysis of the rule-stock correlations in ASX markets, three classes of trading rules were identified. They were: MA, Filter Rule and Channel Breakout [6]. 26 ASX stocks were chosen for the experiments. The data for training was from 1 January 2001 to 31 January 2001, and the testing set was from 1 February 2001 to 28 February 2001. Five different investment plans were conducted on these rules and stocks. In order to organize the pairs, we ranked them based on return, and generated 5% pairs, 10% pairs, and so forth from the whole pair population. The 5% pair means that the pairs are the top 5% based on the return.

Figure 3 illustrates returns we have found for different investment plans on different pairs. These graphs are interesting to traders, allowing them to make smarter trading decision using these mined rule-stock pairs.

5 Developing quality knowledge base for compliance purposes

The DDIP-PD can be deployed to develop a quality knowledge base on interventions, such as for Telecom fraud control [14] and Centrelink compliance. This would allow for more targeted customer interventions and less debt to the customer and debt holding to the organisation. The following illustrates two possible scenarios for developing a quality knowledge base on compliance.

- Correlation analysis of customer and interventions; and
- Developing and evaluating a smart knowledge base.

Obviously, DDIP-PD could be deployed against many other industry and business interests.

5.1 Correlation analysis of customer and interventions

This study could investigate the existence and potential association and correlation between specific customers and interventions such as in Centrelink. It would lead to finding some interesting relationships between customer characteristics or behaviour and associated interventions, and answering business questions, such as "Which customer characteristics are more likely to be associated with non-compliance". It would help understand the impact of and improve current customer contacts (interventions). It could also allow Centrelink to reduce the number of lesser impact interventions thereby allowing resources to be diverted to higher impact and more useful interventions. It could also inform more targeted proactive measures to improve prevention and deterrence (front end) as opposed detection driven (back end) compliance.

To discover association, both positive and negative association mining techniques can be investigated. Existing association rule mining may find a lot of rules but a majority of them are not informative and would not discover the true correlation relationship. Therefore, new correlation mining methods and measures such as those based on all-confidence and coherence [15], and confidence-closed correlated patterns [16] can be developed to mine strongly correlated patterns that are informative and interesting to business without information loss. Additionally, negative and independent correlated patterns can also be investigated.

High dimensional data mining techniques can be studied to reduce the number of dimensions and to efficiently search correlations from a customer-intervention pair space. Taking the customer contacts and interventions as sequences, we may divide them into segments, and develop a dimension reduction technique to every segment, and then keep higher coefficients for more recent data. Effective generic algorithm techniques may also be built for the reduction of search space and dimensions in selecting strongly correlated rules.

5.2 Developing and evaluating a smart knowledge base

Knowledge, such as behaviour patterns predictors and evidences, which inform interventions, need to be interpreted and integrated to build a quality knowledge base and predictive models for improved interventions. An interactive system can provide the capability of deploying smart knowledge, and aid the testing of a knowledge base using live and large-scale data for the activities in an organisation. Feedback from this system could allow for further refining of the knowledge base and consolidate the domain-driven data mining methodology. This could be conducted by:

- interpretation and visualisation of knowledge, including evidences, rules, models, predicators, indicators, etc.;
- building a quality knowledge base and user-friendly interactive interface, and building predictive models for improved interventions based on the quality knowledge base;
- testing and deploying the smart knowledge on real data and business; and

- feedback and refinement on the knowledge base. The resulting outputs would encompass a practical knowledge base with a set of effective evidences and predictors for certain real-world decision-support systems.

This knowledge base would be very useful to improve the quality of data mining for an organisation. For instance, in studying Telecom fraud control [14], the features were iteratively fine-tuned for classifying fraudulent customers, and a feature base was built to effectively detect and predict customers who would be more likely to dishonestly interact with Telecom. In this feature base, some attributes which were originally presumed to be necessary, for instance, the number of suspending services requested by a customer, were found unnecessary for this specific analysis.

6 Conclusions and future work

In the real world, correlated patterns that are interesting to business are often hidden in constraint-based context. This often leads to the scenario where many rules are mined while few of them are interesting to business. Therefore, to improve data mining outcomes, in-depth pattern discovery should be conducted within a constraint-based context. To this end, the domain-driven in-depth pattern discovery (DDID-PD) framework has been developed to guide improved real-world data mining. The DDID-PD framework has been outlined in this paper, which provides methodology for dealing with constraint-based context, mining in-depth patterns and supporting interactive mining in a loop-closed iterative refinement process.

The main phases and components of the DDID-PD framework (as shown in Figure 1) include almost all phases of the well-known industrial data mining methodology CRISP-DM. There are three main differentiations from the CRISP-DM:

(i) some new essential components highlighted by thick rims, such as *results post-processing* and *in-depth modeling*, are taken into account in designing the lifecycle of the DDID-PD process;

(ii) in the DDID-PD, the phases of CRISP-DM highlighted by shadow are enhanced via dynamic interaction with domain experts and the consideration of constraints and domain knowledge; and

(iii) the lifecycle of the DDID-PD is actually different from that of CRISP-DM. These differences are key to mine in-depth patterns in the real world.

In real DDID-PD deployments, we have demonstrated some of our work such as mining in-depth correlations in stock markets, and developing a quality knowledge base for compliance. The experiments on real data have shown that the mined in-depth results guided by the DDID-PD framework are more interesting and actionable to business after considering constraints and domain expert cooperation. Finally, the DDID-PD has potential in mining real-world patterns that are interesting and actionable to business in an effective and efficient manner.

Our further work will include developing detailed data mining process management supports and interfaces for real-world data mining on top of DDID-PD. We are developing a specification as well so that business people can easily follow this methodology.

Acknowledgements

We are grateful to Mr. Li Lin for his supports in system implementation. Thanks also are given to Dr Jie Chen at CISRO and Mr. Zhenxing Qin for the discussion of DDID-PD in the ARC Linkage grant development.

References

[1] Panel members. The perfect data mining tool: Automated or interactive?, in Panel at ACM SIGKDD02', Edmonton, Canada, 2002.
[2] R. Ng, Lakshmanan, L., Han, J. & Pang, A. Exploratory mining and pruning optimizations of constrained association rules, in 'Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems', ACM Press, Seattle, Washington, pp. 13–24, 1998.
[3] L. Cao, R. Dai. "Human-Computer Cooperated Intelligent Information System Based on Multi-Agents", ACTA AUTOMATICA SINICA, 29(1):86-94, 2003, China (in English)
[4] L.B. Cao, et al. Ontology-Based Integration of Business Intelligence. Int. J. on Web Intelligence and Agent Systems, Vol.4 No 4, 2006.
[5] http://www.crisp-dm.org
[6] S. Ryan, Allan, T., Halbert, W., Data-snooping, Technical Trading Rule Performance, and the Bootstrap. *The Journal of Financial*, 1999. 54, (5):1647-1692.
[7] J. Han. *Towards Human-Centered, Constraint-Based, Multi-Dimensional Data Mining.* An invited talk at Univ. Minnesota, Minneapolis, Minnesota, Nov. 1999.
[8] C. Zhang, Z. Zhang, L. Cao. Agents and Data Mining: Mutual Enhancement by Integration, LNCS 3505, 2005
[9] F-Trade. http://datamining.it.uts.edu.au/f-trade
[10] Ankerst, M. (2001), Human involvement and interactivity of the next generation's data mining tools, in 'ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery', Santa Barbara, CA.
[11] B. Kovalerchuk and E. Vityaev. Data Mining in Finance: Advances in Relational and Hybrid Methods, Kluwer Acad. Publ, 2000
[12] www.cmcrc.com
[13] L. Lin, et al. Genetic algorithms for robust optimization in financial applications. Proceedings of 4th IASTED conf. on Computational Intelligence Canada 2005.
[14] D. Luo, W. Liu, C. Luo, L. Cao, R. Dai. Hybrid Analyses and System Architecture for Telecom Frauds. Journal of Computer Science, Vol.32 No.5, pp17-22, 2005.
[15] E. Omiecinski. Alternative Interest Measures for Mining Associations. *IEEE Transactions on Knowledge and Data Engineering*, 15:57-69, 2003.
[16] W. Kim, Y. Lee, and J. Han. CCMine: Efficient Mining of Confidence-Closed Correlated Patterns. In *Proceedings of PAKDD04*, published by Springer-Verlag publisher in LNAI-3056, pp. 569-579, 2004.

Modeling Microarray Datasets for Efficient Feature Selection

Chia Huey Ooi, Madhu Chetty, Shyh Wei Teng

Gippsland School of Information Technology
Monash University, Churchill, VIC 3842, Australia
{chia.huey.ooi, madhu.chetty, shyh.wei.teng}@infotech.monash.edu.au

Abstract. Modeling multiclass gene expression datasets for the purpose of classification is still largely an unexplored area of research. In this paper, we propose two approaches that can be used to model such microarray datasets. We established the usefulness of the artificially generated datasets by demonstrating how they can improve the efficiency of a known feature selection technique. In this study, the proposed models enable predetermination of parameters in feature selection based on characteristics of the dataset alone. This precludes the need for parameter tuning in inner cross-validation loops, radically reducing the computational costs of the predictor set search. Our microarray dataset simulator can also be used with any other supervised machine learning techniques for microarray data analysis.

Keywords. molecular classification, microarray data analysis, feature selection, artificial datasets

1 Introduction

Compared to datasets in other domains, real-life microarray datasets can be considered scarce in view of the complexities and costs involved in conducting large-scale microarray experiments. Therefore, in the area of microarray data analysis, artificial datasets present an attractive solution to this problem. Practical considerations aside, the use of artificial datasets to more precisely test the strengths and weaknesses of an algorithm has also been recommended in [1].

The most palpable advantage of artificial datasets over real-life datasets is that they can be easily mass-produced requiring considerably less cost and amount of time than those involved in generating real-life datasets. Another important advantage is the control the researcher exercises over parameters governing dataset characteristics such as number of classes and noise levels. Moreover, unlike the case of real-life datasets, the amount of noise in artificial datasets is always a known quantity, since noise level is a parameter fed into the microarray dataset simulator. This can facilitate the task of determining how dataset characteristics influence the optimal values of parameters in the data analysis techniques used.

Although not fully explored for simulating microarray datasets so far, artificial datasets have been widely used in traditional machine learning and data mining areas.

The most well-known are the MONK dataset and the artificial characters database [2]. More recently, in a feature selection challenge, four semi-artificial and one artificial datasets have been made publicly available [3]. Here it is to be noted that datasets are considered semi-artificial in the sense that randomly perturbed features are added to the original features in the dataset. The limitation presented by these extant artificial datasets is that none of these simultaneously possesses the following two traits common to multiclass microarray datasets: 1) more than two classes, and 2) large number of features (high dimensionality).

An adequate microarray dataset simulator should be able to produce artificial datasets with the aforementioned traits since these are the common characteristics of any multiclass microarray datasets. Although several microarray dataset simulators have been proposed previously, all of them are for use with one or combinations of the following: clustering, normalization or noise-elimination techniques [4, 5, 6]. None of these are devised specifically for use with feature selection or classifier techniques. Therefore, to address these inadequacies, we have devised a novel method for generating artificial datasets to best simulate microarray datasets. The two models behind our microarray dataset simulator are the one-vs.-all (OVA) and the pairwise (PW) models. The ability of each model to realistically simulate microarray datasets is investigated in this paper.

The two objectives of the study reported in this paper are as follows:

1. Establish a suitable model for representing microarray datasets for use with feature selection or classifier techniques.
2. Provide support to the hypothesis regarding the influence dataset characteristics have on the optimal value of the parameter(s) in a feature selection technique.

The impact of the first objective is wide-reaching. Although there are plenty of microarray datasets available publicly, many are the results of microarray experiments conducted for the purpose of class discovery. This means that the class labels for samples in this kind of datasets are derived from the information in the datasets themselves. This type of datasets will be useful for testing the performance of clustering and other unsupervised machine learning techniques, but not that of supervised techniques such as classification and feature selection – the focus of our study. Therefore, establishing a model which represents multiclass microarray datasets accurately will assist researchers in those two fields in overcoming the hurdle currently faced (i.e. the small number of available datasets), at least until the microarray technology catches up and the microarray version of the UCI repository is made possible. Even then, the use of artificial datasets will still continue to be highly attractive because of the control over dataset characteristics (noise levels, number of features, samples or classes and class sizes) it affords researchers.

The most likely of researchers to benefit from our microarray dataset simulator are those in the field of feature selection or classification, who, having tested their methods on various (but limited) real-life microarray datasets, have come to the conclusion that dataset characteristics are influencing the optimal parameters in their methods, and the ability to predict those optimal parameters based on the dataset characteristics will improve the performance of their feature selection or classifier techniques.

The outcome of the second objective will add great value to any feature selection or classifier technique. Instead of testing for the whole range of possible values (for example, 0 to 1) for a parameter of the feature selection or classifier technique during internal cross-validation or tuning stage, knowing how (and which) dataset characteristics influence the optimal value of the parameter will make it possible for us to predict a much narrower range for the parameter (e.g. 0.2 to 0.3) to focus on. This will bring about definite savings in terms of computational power and time.

Starting with a detailed presentation of the modeling method, we then provide a brief background regarding feature selection for microarray datasets and an overview of the **d**egree of **d**ifferential **p**rioritization-based (DDP-based) feature selection technique that will be applied to study the viability of the microarray dataset simulator. This is then followed by classification results from real-life and artificial datasets. The paper ends with model validation, discussion regarding the strengths and shortcomings of the study, and the conclusion.

2 Modeling Microarray Datasets

It is widely accepted that over-expression or under-expression (suppression) of genes causes the difference in phenotype among samples of different classes. The categorization of gene expression is given as follows.

- A gene is over-expressed: if its expression value is above baseline.
- A gene is under-expressed: if its expression value is below baseline.
- Baseline interval: the normal range of expression value.

Usually the mean of the expression of a gene across samples is taken as the middle of the baseline interval. Multiples of the standard deviation (SD) of expression across samples are used as boundaries of the baseline interval. For example, expression values between −1.5SD and 1.5SD may be considered as baseline values. With this categorization we next employ two well-known paradigms (OVA and PW) leading to the OVA and PW models respectively, which are then used to generate two different sets of artificial microarray datasets.

Table 1. The OVA Model ([a]OX = over-expressed, [b]UX = under-expressed, [c]BL = baseline)

Groups of marker genes	Samples of class			
	1	2	…	*K*
Group 1	OX[a]	BL[c]	…	BL
Group 2	BL	OX	…	BL
⋮	⋮	⋮	⋮	⋮
Group *K*	BL	BL	…	OX
Group *K*+1	UX[b]	BL	…	BL
Group *K*+2	BL	UX	…	BL
⋮	⋮	⋮	⋮	⋮
Group 2*K*	BL	BL	…	UX

1. *OVA model:* The crux of the OVA concept has gained wide, albeit tacit, acceptance among microarray and tumor gene expression researchers. The fact that particular marker genes are only over-expressed in tissues of certain type of cancer, and not any other types of cancer or normal tissues [7], is part of the entrenched domain

knowledge. Hence the term 'marker' – for genes that mark the particular cancer associated with them. We develop this concept into a model for use with the microarray dataset simulator. In the OVA model (Table 1), certain groups of genes, also called the 'marker genes' are only over-expressed (or under-expressed) in samples belonging to a particular class and never in all samples of other classes. This model emphasizes that a group of marker genes is specific to one class. Therefore for a K-class dataset, there will be $2K$ different groups of marker genes.

2. *PW Model:* In the PW model (Table 2), for a given pair of classes, a group of marker genes is over-expressed (or under-expressed) in samples of one class of the pair but under-expressed (or over-expressed) in samples of the other class. As implied by its name, this model represents the 1-vs.-1 paradigm as opposed to the 1-vs.-others of the OVA model. For a K-class dataset, there are $2({}^{K}C_2)$ different groups of marker genes in the PW model.

Table 2. The PW Model ([a]OX = over-expressed, [b]UX = under-expressed, [c]BL = baseline)

Groups of marker genes	Samples of class					
	1	2	3	…	K–1	K
Group 1 (Classes 1 vs. 2)	OX	UX	BL	…	BL	BL
Group 2 (Classes 1 vs. 3)	OX	BL	UX	…	BL	BL
⋮	⋮	⋮	⋮	⋮	⋮	⋮
Group ${}^{K}C_2$ (Classes $K-1$ vs. K)	BL	BL	BL	…	OX	UX
Group ${}^{K}C_2 + 1$ (Classes 1 vs. 2)	UX	OX	BL	…	BL	BL
Group ${}^{K}C_2 + 2$ (Classes 1 vs. 3)	UX	BL	OX	…	BL	BL
⋮	⋮	⋮	⋮	⋮	⋮	⋮
Group $2({}^{K}C_2)$ (Classes $K-1$ vs. K)	BL	BL	BL	…	UX	OX

Due to the high-throughput nature of microarray experiments and the hybridization tendencies of certain mRNA probe-target pairs, microarray datasets are inherently noisy, although the level of noise differs from dataset to dataset. Hence, noise needs to be added to make the artificial datasets more realistic. Following [5, 6], we use Gaussian noise to simulate hybridization noise in real-life datasets. A percentage of noise, r_v, is added for each data entry. The expression of gene i in sample j ($x_{i,j}$) is perturbed in the following manner.

$$\tilde{x}_{i,j} = x_{i,j} \cdot (1 + r_v) \tag{1}$$

where r_v is a random number picked from a Gaussian distribution having a mean 0 and variance v. The variance v is also referred to as the noise level of the dataset.

3 Experiments to Validate the Models

For microarray datasets, the term *gene* and *feature* may be used interchangeably. The objective of feature selection is to form a subset of features (the predictor set), which would yield the optimal estimate of classification accuracy. The importance of feature selection prior to classification for microarray datasets has been proven in previous studies [8, 9, 10]. From techniques as simple as rank-based techniques [10, 11, 12] to those as sophisticated as wrapper-based methods [13], various feature selection techniques have been proposed for microarray datasets, with mixed results.

For filter-based feature selection, one or both of two criteria, *relevance* and *redundancy*, have often been used in the search for the predictor set. For microarray datasets, while relevance alone has been employed early on in rank-based techniques [10, 11, 12, 14], it is only recently that redundancy is included as the second criterion in forming the predictor set [8, 9]. Extending from here, in [15], we have demonstrated that aside from relevance and redundancy, a third criterion, called the **d**egree of **d**ifferential **p**rioritization (DDP), is necessary for the optimal performance of filter-based feature selection for multiclass microarray datasets. During the search for the predictor set, DDP compels the search method to prioritize the optimization of one of the criteria (of relevance and redundancy) at the cost of the optimization of the other.

In order to validate the models and the usefulness of artificial datasets, any feature selection or classifier technique can be used. However, in this study the DDP-based feature selection technique is chosen for the following reason: The empirical results from this technique suggested that the optimal value of DDP (i.e. the value of DDP leading to the best estimate of accuracy) is dataset-specific [15], leading us to hypothesize on the superficial characteristics of a dataset most likely to influence the value of its optimal DDP. Superficial characteristics refer to dataset characteristics which are discernible through direct inspection, such as the total number of features, samples or classes. However, due to the limited number of available real-life multiclass microarray datasets, it is not possible to verify the hypothesis suggested by the findings in [15] without the use of artificial datasets. Before validating the models, we briefly review the DDP-based feature selection technique.

3.1 Overview of the DDP-based Feature Selection Technique

The training set upon which feature selection is to be implemented, T, consists of N genes and M_t training samples. Sample j is represented by a vector, $\mathbf{x}_j$, containing the expression of the N genes $[x_{1,j},\ldots, x_{N,j}]^T$ and a scalar, y_j, representing the class the sample belongs to. The target class vector $\mathbf{y}$ is defined as $[y_1, \ldots, y_{Mt}]$, $y_j \in [1,K]$ in a K-class dataset. From the N genes, the objective is to form the subset of genes, called the predictor set S, which would give the optimal estimate of classification accuracy.

The relevance of S, V_S, is the average individual relevance of its members.

$$V_S = \frac{1}{|S|}\sum_{i\in S} F(i) \qquad \textbf{(2)}$$

The score of relevance for gene i, $F(i)$, indicates the correlation of gene i to $\mathbf{y}$. A popular parameter for computing $F(i)$ is the BSS/WSS ratios used in [8, 11].

U_S is the measure of the antiredundancy of S.

$$U_S = \frac{1}{|S|^2}\sum_{i,j\in S} 1 - |R(i,j)| \qquad \textbf{(3)}$$

$R(i,j)$ is the Pearson product moment correlation coefficient between genes i and j. Larger U_S indicates lower average pairwise similarity among the members of S, and hence, smaller amount of redundancy in S.

The score of goodness for predictor set S is given as follows.

$$W_{A,S} = (V_S)^{\alpha} \cdot (U_S)^{1-\alpha} \qquad (4)$$

where the power factor $\alpha \in (0,1]$ denotes the value of the DDP. In other words, $\alpha/(1-\alpha)$ represents the ratio of the priority in maximizing V_S to the priority in maximizing U_S. The significance of α has been elaborated in [15].

The linear incremental search method is employed, where the first member of S is chosen by selecting the gene with the highest $F(i)$ score. To find the second and the subsequent members of the predictor set, the remaining genes are screened one by one for the gene that would give the maximum $W_{A,S}$. This search method, with a much lower computational complexity of $O(NP_{\max})$ than that of exhaustive search $(O(N^{P_{\max}}))$, has been applied in previous feature selection studies [8, 9]. $P_{\max}$ is the upper limit of the predictor set size we wish to search.

The DDP-based feature selection technique is applied to both real-life and artificial datasets. Results from both categories of datasets are examined to validate the viability of the proposed microarray dataset simulator and to then determine the better model (between OVA and PW models) for realistic simulation of microarray datasets.

Table 3. Descriptions of real-life datasets. N is the number of features after preprocessing

Dataset	Type	N	K	Training:Test set size
GCM	Affymetrix	10820	14	144:54
NCI60	cDNA	7386	8	40:20
PDL	Affymetrix	12011	6	166:82
Lung	Affymetrix	1741	5	135:68
SRBC	cDNA	2308	4	55:28
MLL	Affymetrix	8681	3	48:24
AML/ALL	Affymetrix	3571	3	48:24

3.2 Real-life Microarray Datasets

The characteristics of seven real-life microarray datasets: the GCM [7], NCI60 [16], lung [17], MLL [18], AML/ALL [14], PDL [19] and SRBC [20] datasets, are listed in Table 3. For NCI60, only 8 tumor classes are analyzed; the 2 samples of the prostate class are excluded due to the small class size. Datasets are preprocessed and normalized based on the recommended procedures in [11] for Affymetrix and cDNA microarray data. Details regarding the performance of the DDP-based feature selection method on the first five datasets are available in [15]. With the exception of the GCM dataset, where the original ratio of training to test set size used in [7] is maintained to enable comparison with previous studies, for all other datasets we employ the standard 2:1 split ratio.

3.3 Artificial Datasets

Artificial datasets are generated from both OVA and PW models presented in Section 2. For both models, several levels of noise have been incorporated by setting the value of v ranging from 0 to $v_{\max}$ with equal intervals of 0.05 ($v_{\max}$ being arbitrarily set to 0.35 in this study). The range of K is from 3 to 15, producing 13 datasets for each level of noise. All remaining parameters aside from v and K involved in generating the artificial datasets are kept fixed. They are described below.

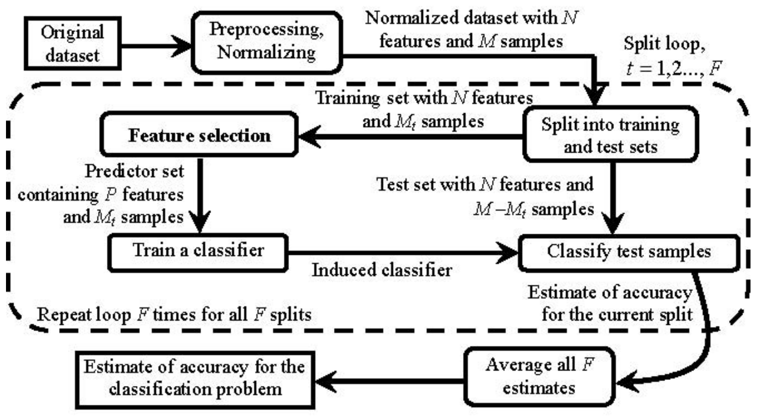

Original dataset
Preprocessing, Normalizing
Normalized dataset with N features and M samples
Split loop, t = 1,2..., F
Training set with N features and M_t samples
Feature selection
Split into training and test sets
Predictor set containing P features and M_t samples
Test set with N features and $M - M_t$ samples
Train a classifier
Induced classifier
Classify test samples
Estimate of accuracy for the current split
Repeat loop F times for all F splits
Estimate of accuracy for the classification problem
Average all F estimates

Fig. 1. F-splits procedure

The size of each group of marker genes is fixed at 2 genes per group. Therefore, there is a total of $4K$ (or $4({}^{K}C_2)$) marker genes in a K-class dataset generated using the OVA (or the PW) model. N being set to 2000, the remaining features are irrelevant genes containing random expression values. The class size, 12 samples per class, is kept equal for all classes. The bounds for over-expression are [0.5,2], for under-expression [–2,–0.5] and for baseline [–0.5,0.5]. These bounds are in accordance with the standard microarray dataset preprocessing and normalizing procedures recommended in [11], where expressions are log-transformed and normalized to have mean 0 across features; and data entries with values above a maximal threshold or below a minimal threshold are eliminated.

3.4 Evaluation Procedure – Looking for α*

Various values of α have been employed in the experiment, in the range of [0.1,1] with equal intervals of 0.1. The range of the predictor set size, P, analyzed is from 2 to $P_{\max}$=100. The F-splits procedure (Figure 1) is used to evaluate the classification performance of a predictor set of a certain size P derived from a particular value of α. We set F to 10 in this study. The DAGSVM classifier is used for all performance evaluation. The DAGSVM is an SVM-based multiclassifier which uses substantially less training time compared to either the standard algorithm or Max Wins, and has been shown to produce accuracy comparable to both of these algorithms [21].

Since our feature selection technique does not explicitly predict the best P from the range of $[2, P_{\max}]$, in order to determine the value of α likely to produce the optimal accuracy, we use a parameter called *size-averaged accuracy*, which is computed as follows. For all predictor sets found using a particular value of α, we plot the estimate of accuracy obtained from the procedure outlined in Figure 1 against the value of P of the corresponding predictor set (Figure 2). The size-averaged accuracy for that value of α is the area under the curve in Figure 2 divided by the number of predictor sets, $(P_{\max} - 1)$. The value of α associated with the highest size-averaged accuracy is deemed the empirical estimate of α^* (the empirical optimal value of the DDP). If there is a tie in terms of the highest size-averaged accuracy between different values of α, the empirical estimate of α^* is taken as the average of those values of α.

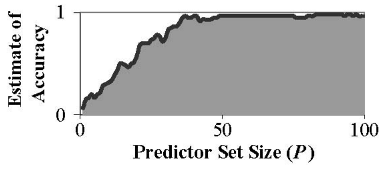


Fig. 2. Area under the accuracy-predictor set size curve

3.5 Classification Results

Real-life Datasets. The size-averaged accuracy vs. α plots (Figure 3a) leads to the surmise that α^* is strongly influenced by K. The peak in the size-averaged accuracy plot moves towards the left as K increases.

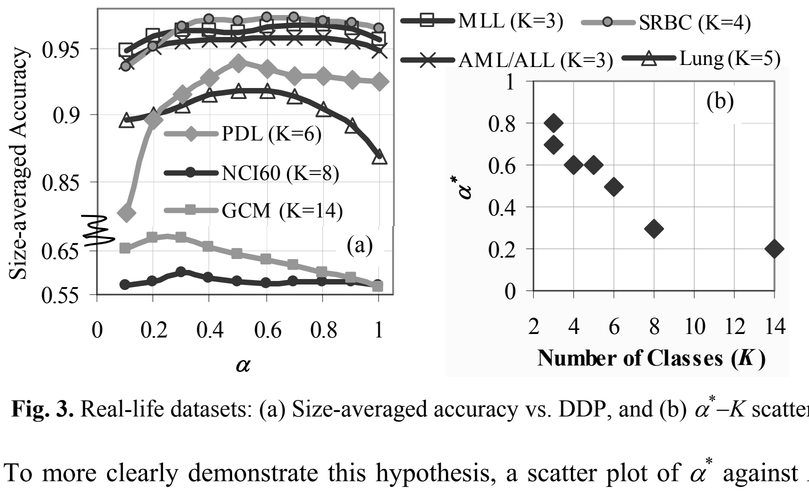


Fig. 3. Real-life datasets: (a) Size-averaged accuracy vs. DDP, and (b) α^*–K scatter plot

To more clearly demonstrate this hypothesis, a scatter plot of α^* against K is depicted in Figure 3b. With just seven points of data, even the best curve-fitting method would not be able to give us an equation governing the value of α^* with sufficiently high confidence. However, with the proposed microarray dataset simulator, this difficulty can be easily surmounted.

Another observation from Figure 3a is the tendency of accuracy to deteriorate as K increases. This is not surprising since a 3-class problem is considerably easier than a 15-class problem. What is surprising is the gap between the accuracy from the 8-class NCI60 dataset and the 6-class PDL dataset (the latter possessing merely two classes less than the former). However, we will be able to explain this phenomenon with the aid of artificial datasets.

Artificial Datasets. Results for artificial datasets are divided into 2 sections below, each section focusing on datasets generated from each of the OVA and PW models.

OVA model. The size-averaged accuracy vs. α plots from datasets with selected K values (3 to 15, with equal intervals of 3) for 3 out of 8 noise levels tested ($v = 0, 0.2, 0.35$) are depicted in Figure 4a. Due to space constraint, we have not shown all datasets from the complete tested range of K (3 to 15) and v (0 to 0.35).

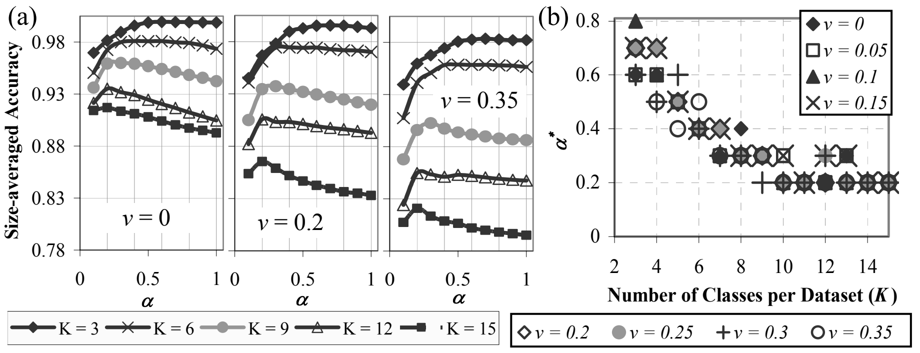


Fig. 4. Artificial datasets derived from OVA model: (a) Size-averaged accuracy vs. DDP for 3 different noise levels $v = 0$, 0.2 and 0.35, and (b) α^*–K scatter plot for various noise levels v

For all noise levels, datasets with larger K produce lower accuracy than datasets with smaller K (Figure 4a). In irrefutable support of Figure 3b, Figure 4b shows that the influence of K on α^* is indubitable. We say 'irrefutable', because the experiment settings are such that all parameters concerning the superficial dataset characteristics except K have been fixed for all of our artificially generated datasets. Therefore, for each noise level, the value of α^* must have been influenced by K alone.

Using Figure 4, we can make some additional observations based on artificial microarray datasets which are not possible using real-life data sets. Firstly, in terms of accuracy, datasets with larger K are more susceptible to rising level of noise than datasets with smaller K (Figure 4a). That is, given the same amount of increase in noise level, accuracy from datasets with larger K (>6) decreases at a more drastic rate compared to accuracy from datasets with smaller K. Secondly, while for each value of K, minor aberrations in α^* do occur due to different noise levels (Figure 4b), clearly the effect of the noise level, v, on α^* is not as prevailing that of the number of classes per dataset, K. For the OVA-based artificial datasets, the biggest difference between α^* values for the same K from different v values occur at the smallest K value, 3. The difference tapers off as K increases. At the largest K values, 14 and 15, the difference in α^* due to varying noise levels have actually disappeared.

There are 2 similar trends between Figures 3 (real-life datasets) and 4 (OVA-based artificial datasets): 1) accuracy decreases as K increases, and 2) α^* decreases as K increases. Hence, we can say that in terms of the relationships among K, accuracy and α^*, the OVA model portrays real-life microarray datasets satisfactorily at this stage.

PW Model. Figure 5a depicts the size-averaged accuracy vs. α plots from datasets with selected K values (3 to 15, with intervals of 3) for 3 out of 8 noise levels tested ($v = 0, 0.2, 0.35$). The α^*–K scatter plot is shown in Figure 5b.

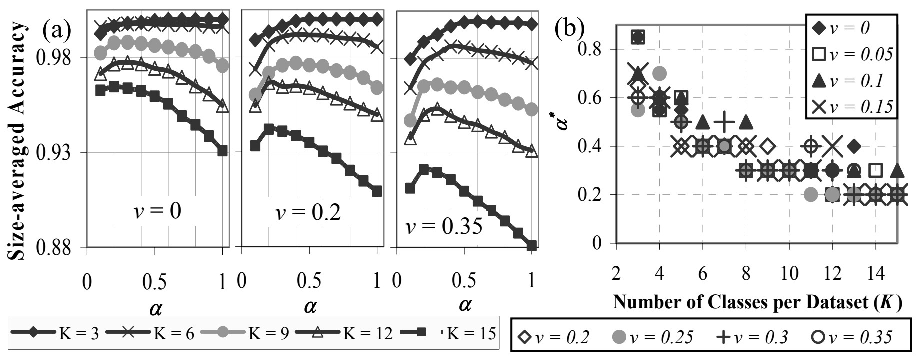


Fig. 5. Artificial datasets derived from PW model (a) Size-averaged accuracy vs. DDP for 3 different noise levels $v = 0$, 0.2 and 0.35, and (b) α^*–K scatter plot for various noise levels v

The overall trend shown by PW-based artificial datasets is similar to the trend produced by their OVA-based counterparts, with two important exceptions.

Firstly, in general the resulting size-averaged accuracy is higher in PW-based datasets than accuracy from OVA-based datasets. This is especially obvious in datasets with larger K (Figure 5a). Secondly, for each particular value of K, the aberrations in the values of α^* due to varying noise levels are larger for the PW model than the OVA model (Figure 5b). For the PW-based artificial datasets, the largest noise-induced difference in α^* does occur at $K = 3$ as in case of the OVA-based counterparts. However, as K increases the aberrations in α^* do not deteriorate as rapidly as in case of the OVA-based datasets. Even at $K = 15$, there is still a difference of 0.1 in α^* among the 15-class PW-based datasets with varying noise levels.

4 Model Validation

We now investigate how satisfactorily artificial datasets from OVA and PW models represent real-life datasets. Since the DDP-based feature selection technique is chosen as the microarray data analysis technique of interest, we validate the models in terms of the behavior of α^* against K. For any technique in general, the models are validated by investigating how closely the behavior of a parameter in that technique (against dataset characteristics) in results from artificial datasets resembles its behavior in results from real-life datasets.

4.1 Curve-Fitting

For each level of noise, we fit a curve to the α^*–K scatter plot from the artificial datasets (Figures 4b and 5b). Three equations are considered in describing the α^*–K relationship: exponential, power and rational fit of constant numerator and linear polynomial denominator. Based on the average of adjusted R^2 values from all levels of noise, for both models, the best fit is the rational function:

$$\alpha^* = \frac{b}{K + q} \tag{5}$$

The values of the constants b and q differ depending on the level of noise.

To investigate how well each model fits real-life datasets in terms of the behavior of α^* w.r.t. K, we implemented a deductive fit analysis:

1. For each noise level, $v = 0, 0.05, \ldots, v_{\max}$
   1.1. Apply curve-fitting to α^*–K data points from noise level ranging from v to $v_{\max}$ (inclusive).
   1.2. Using the parameters b and q obtained from 1.1, fit the curve to α^*–K data points from the seven real-life datasets (Figure 3b).
   1.3. Record the adjusted R^2 value for this fit as $R^2(v \rightarrow v_{\max})$. Larger $R^2(v \rightarrow v_{\max})$ indicates better fit to the real-life datasets.
2. Plot $R^2(v \rightarrow v_{\max})$ against corresponding values of v.

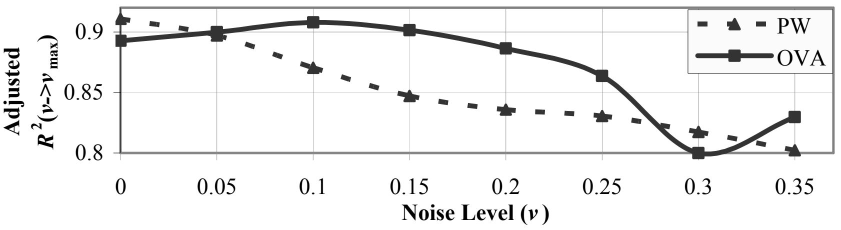


Fig. 6. $R^2(v \rightarrow v_{\max})$ against v for both OVA and PW models

The results of this analysis (Figure 6) indicate that the OVA model portrays the real-life datasets more realistically than the PW model. The PW model fits the real-life datasets best when zero noise level ($v = 0$) is included in the curve fitting exercise, but as v is increased up to $v_{\max}$, the fit as measured by $R^2(v \rightarrow v_{\max})$ declines. This is far from convincing, since real-life microarray datasets are not likely to contain zero noise. Therefore, the initial high R^2 value in case of the PW model may be dismissed as a case of overfitting the real-life datasets. The case against the PW model is also strengthened by observations in the previous section: namely, the too-optimistic size-averaged accuracies and the larger noise-induced aberrations in α^*.

Whereas for the OVA model, the best fit occurs when noise levels of $v = 0.1$ up to $v_{\max}$ are included in the curve fitting exercise. It is only by eliminating α^*–K data points from the lower noise levels ($v = 0$ and $v = 0.05$) that the fit to real-life datasets is improved. Consequent removal of data points from noise levels higher than $v = 0.1$ causes deterioration in the fit (as indicated by the decrease in $R^2(v \rightarrow v_{\max})$). This is possibly due to the fact that among themselves, the seven real-life datasets actually contain noise levels equivalent to the noise levels between $v = 0.1$ and $v = 0.35$ of the artificial datasets.

While determining the location of a real-life dataset in K-space is straightforward, it is not so in case of the v-space. If classification has been performed on the real-life dataset, as is the case in our study, an alternative is to compare the best size-averaged accuracy of the real-life dataset to those of artificial datasets of varying noise levels, but of the same K. However, the purpose of this study is to devise a way to predict a parameter in a feature selection technique based purely on the characteristics of the real-life dataset of interest *before* feature selection or classifier induction is actually conducted. Hence, classifying test sets from all F-splits in order to determine the noise level the dataset contains defeats the very purpose of the study itself.

One way to overcome this difficulty is the class variance analysis, where a number of real-life datasets are ranked in terms of noisiness. This method might be useful in giving us an idea of the *relative* noise levels among a group of datasets, despite its two weaknesses: Firstly, the ranking is only a surmise at best, and secondly, it does not provide any *absolute* quantitative information regarding the level of noise in each dataset. The class variance analysis is conducted as follows:

```
1. For each split, $f$ = 1, 2, …, $F$
   1.1. For each of the top 100 genes ranked using BSS/WSS ratio
        1.1.1. For each class, $k$ = 1, 2, …, $K$
               1.1.1.1. Compute the variance among samples belonging to class $k$.
        1.1.2. Average the variance for all classes. This is the class-averaged variance for the gene.
   1.2. Pick the largest class-averaged variance from all top 100 genes.
2. Average the largest class-averaged variance from all $F$ splits. This is a measure of the relative noise level for a dataset.
```

Based on the class variance analysis, this is how the seven real-life datasets rank in terms of relative noise level, arranged in order of descending noise level:

$$\text{AML/ALL} \rightarrow \text{NCI60} \rightarrow \text{GCM} \rightarrow \text{Lung} \rightarrow \text{MLL} \rightarrow \text{PDL} \rightarrow \text{SRBC}$$

Allowing for the influence of K on accuracy, this ranking makes sense – for instance, the AML/ALL has lower accuracy than the MLL, which has lower noise level, although both comprise of 3 classes. More importantly, this also explains the low accuracy rate of the 8-class NCI60 dataset compared to the 14-class GCM dataset, or the aforementioned discrepancy between the 8-class NCI60 dataset and the 6-class PDL dataset. The NCI60 dataset contains higher noise level than the other two datasets.

4.2 Classification Performance from Predicted α^*

According to the best realistic fit of the OVA model to the real-life datasets, the values of the constants in equation (5), which governs the relationship between α^* and K, are: $b = 2.85 \pm 0.253$ and $q = 0.9334 \pm 0.4789$.

Using the values of α^* predicted from equation (5) with b and q set to the aforementioned values, we re-run the DDP-based feature selection technique on the seven real-life datasets and evaluated the resulting predictor sets. We then compare the best estimate of accuracy obtained from the predicted α^* to the best accuracy obtained from the empirical α^* (Table 4). The greatest difference between accuracies from predicted α^* and empirical α^* is no larger than 3%. As expected, the biggest deviation of –3% occurs in the dataset with the second highest estimated relative noise level and the second largest numbers of classes, the NCI60 dataset. Therefore, without having to conduct feature selection for values of α from the full domain of 0 to 1, by using equation (5) we could simply focus on the much smaller predicted range or value of α^* and emerge with similar classification accuracy.

Table 4. Comparing accuracies obtained from empirical and predicted α^*

Dataset	Empirical α^*	Predicted α^*	Best accuracy from empirical α^*	Best accuracy from predicted α^*
GCM	0.2	0.191	0.806	0.802
NCI60	0.3	0.319	0.740	0.710
PDL	0.5	0.411	0.990	0.988
Lung	0.6	0.480	0.953	0.954
SRBC	0.6	0.578	0.996	0.996
MLL	0.7	0.725	0.992	0.992
AML/ALL	0.8	0.725	0.979	0.979

By comparing the best size-averaged accuracy of various datasets, real-life or artificial, it is clear that the measure of relevance being used in the DDP-based feature selection technique is not efficient enough to capture the relevance of a feature when K is larger than 6. The subsequent decrease in α^* as K increases implies that placing more emphasis on maximizing antiredundancy (rather than relevance) produces better accuracy for large-K datasets. On a more cautious note, the effect of noise on α^* might be more profound than observed from the results at hand. Extending the noise levels from the current $v_{\max}$ value of 0.35 to a higher value, or using interval size smaller than 0.05, will provide better understanding on the influence of noise on the optimal DDP value. Moreover, candidate models for simulating microarray datasets are not limited to the two presented in this paper. Other models might emerge that portray microarray datasets more accurately than the OVA model.

This study demonstrates the usefulness of modeling microarray datasets in helping reduce the time and computational cost needed to determine the optimal value or range of one parameter in one particular feature selection technique. However, the fit established by the OVA model to the seven real-life datasets proves the robustness of the modeling technique. Therefore, we believe that this modeling technique will work as well (as it does for our DDP-based feature selection technique) when applied to any other feature selection or classifier technique for microarray datasets, regardless of the number of parameters involved. Furthermore, instead of the number of classes in the dataset, there are other dataset characteristics which can be varied – depending on which characteristic(s) the researcher suspects is affecting the optimal value of the parameter(s) in his own technique.

In cases where the optimal range of multiple parameters in the feature selection or classifier technique needed to be determined, one experiment will have to be conducted for each parameter. In each experiment, all other parameters are to be fixed while the optimal range of the parameter in question is being determined.

5 Conclusion

We have presented a novel method for simulating microarray datasets by employing two models (OVA and PW models), which can be used in conjunction with any feature selection or classifier technique for the analysis of microarray data. In case of the DDP-based feature selection, we have demonstrated that the OVA model simulates real-life microarray datasets better than the PW model in explaining the relationship between K and α^*.

It would not have been possible to describe in more detail the relationship between a characteristic of the dataset and a parameter in a feature selection technique without the use of artificial datasets. The capability to predict a narrow range of the optimal parameter for the dataset of interest is extremely useful in helping us achieve the optimal classification performance. Savings are achieved in terms of computational costs and time, since with this capability the need to conduct feature selection and parameter tuning for the **whole** domain of the parameter is eliminated.

References

1. Salzberg, S.: On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery 1 (1997) 317–328
2. Murphy, P.M., Aha, D.W.: UCI repository of machine learning databases.
3. Guyon, I.: Design of experiments for the NIPS 2003 variable selection benchmark. http://clopinet.com/isabelle/Projects/NIPS2003/ (2003)
4. Newton, M.A., Kendziorski, C.M., Richmond, C.S., Blattner, F.R., Tsui, K.W.: On differential variability of expression ratios: Improving statistical inference about gene expression changes from microarray data. J. Computat. Biol. 8 (2001) 37–52
5. Zhao, Y., Li, M.C., Simon, R.: An adaptive method for cDNA microarray normalization. BMC Bioinformatics 6(28) (2005) doi:10.1186/1471-2105-6-28
6. Huang, J.C., Dueck, D., Morris, Q.D., Frey, B.J.: Iterative analysis of microarray data. In: Proc. 42nd Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, Sep 29– Oct 1, 2004 (2004)

7. Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C.H., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J.P., Poggio, T., Gerald, W., Loda, M., Lander, E.S., Golub, T.R.: Multi-class cancer diagnosis using tumor gene expression signatures. Proc. Natl. Acad. Sci. 98 (2001) 15149–15154
8. Ding, C., Peng, H.: Minimum Redundancy Feature Selection from Microarray Gene Expression Data. In: Proc. 2nd IEEE Computational Systems Bioinformatics Conference. IEEE Computer Society (2003) 523–529
9. Yu, L., Liu, H.: Efficiently Handling Feature Redundancy in High-Dimensional Data. In: Domingos, P., Faloutsos, C., Senator, T., Kargupta, H., Getoor, L. (eds.): Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, New York (2003) 685–690
10. Chai, H., Domeniconi, C.: An Evaluation of Gene Selection Methods for Multi-class Microarray Data Classification. In: Proc. 2nd European Workshop on Data Mining and Text Mining in Bioinformatics (2004) 3–10
11. Dudoit, S., Fridlyand, J., Speed, T.: Comparison of discrimination methods for the classification of tumors using gene expression data. JASA 97 (2002) 77–87
12. Li, T., Zhang, C., Ogihara, M.: A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. Bioinformatics 20 (2004) 2429–2437
13. Guyon, I., Weston, J., Barnill, S.: Gene Selection for Cancer Classification Using Support Vector Machines. Machine Learning 46 (2002) 389–422
14. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. Science 286 (1999) 531–537
15. Ooi, C.H., Chetty, M., Teng, S.W.: Relevance, redundancy and differential prioritization in feature selection for multiclass gene expression data. In: Oliveira, J.L. et al. (eds.): Proc. 6th Int'l Symposium on Biological and Medical Data Analysis (ISBMDA-05), in press.
16. Ross, D.T., Scherf, U., Eisen, M.B., Perou, C.M., Spellman, P., Iyer, V., Jeffrey, S.S., Van de Rijn, M., Waltham, M., Pergamenschikov, A., Lee, J.C.F., Lashkari, D., Shalon, D., Myers, T.G., Weinstein, J.N., Botstein, D., Brown, P.O.: Systematic variation in gene expression patterns in human cancer cell lines, Nature Genetics 24(3) (2000) 227–234
17. Bhattacharjee, A., Richards, W.G., Staunton, J., Li, C., Monti, S., Vasa, P., Ladd, C., Beheshti, J., Bueno, R., Gillette, M., Loda, M., Weber, G., Mark, E.J., Lander, E.S., Wong, W., Johnson, B.E., Golub, T.R., Sugarbaker, D.J., Meyerson, M.: Classification of Human Lung Carcinomas by mRNA Expression Profiling Reveals Distinct Adenocarcinoma Subclasses. Proc. Natl. Acad. Sci. 98 (2001) 13790–13795
18. Armstrong, S.A., Staunton, J.E., Silverman, L.B., Pieters, R., den Boer, M.L., Minden, M.D., Sallan, S.E., Lander, E.S., Golub, T.R., Korsmeyer, S.J.: MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. Nature Genetics 30 (2002) 41–47
19. Yeoh, E.-J., Ross, M.E., Shurtleff, S.A., Williams, W.K., Patel, D., Mahfouz, R., Behm, F.G., Raimondi, S.C., Relling, M.V., Patel, A., Cheng, C., Campana, D., Wilkins, D., Zhou, X., Li, J., Liu, H., Pui, C.-H., Evans, W.E., Naeve, C., Wong, L., Downing, J. R.: Classification, subtype discovery, and prediction of outcome in pediatric lymphoblastic leukemia by gene expression profiling. Cancer Cell 1 (2002) 133–143
20. Khan, J., Wei, J.S., Ringner, M., Saal, L.H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C.R., Peterson, C., Meltzer, P.S.: Classification and diagnostic prediction of cancers using expression profiling and artificial neural networks. Nature Medicine 7 (2001) 673–679
21. Platt, J.C., Cristianini, N., Shawe-Taylor, J.: Large Margin DAGs for Multiclass Classification. Advances in Neural Information Processing Systems 12 (2000) 547–553

Predicting Intrinsically Unstructured Proteins Based on Amino Acid Composition

Pengfei Han[1], Xiuzhen Zhang[1], Raymond S. Norton[2], and Zhiping Feng[2]

[1] School of Computer Science and IT, RMIT University
Melbourne, Victoria 3001, Australia
{phan, zhang}@cs.rmit.edu.au
[2] Structural Biology Division
The Walter and Eliza Hall Institute of Medical Research
Parkville, Victoria 3050, Australia
{ray.norton, feng}@wehi.edu.au

Abstract. Intrinsically Unstructured or Disordered Proteins (IUPs) exist in a largely disordered structural state. The automated prediction of IUPs provides a first step towards high-throughput analysis of IUPs. The problem of predicting IUPs given training data of ordered proteins and IUPs can be mapped to the classification problem. In this paper, we propose to convert the original primary sequence database into an amino acid composition database and build a decision tree model. The system derives concise and biologically meaningful amino acid composition (AAC) classification rules. Cross-validation tests estimate that for predicting IUPs that contain long disordered regions or are completely disordered, the AAC-rule classifier achieves a recall of 77.3% and precision of 81.4%.

Predicting Intrinsically Unstructured Proteins Based on Amino Acid Composition

Abstract. Intrinsically Unstructured or Disordered Proteins (IUPs) exist in a largely disordered structural state. The automated prediction of IUPs provides a first step towards high-throughput analysis of IUPs. The problem of predicting IUPs given training data of ordered proteins and IUPs can be mapped to the classification problem. In this paper, we propose to convert the original primary sequence database into an amino acid composition database and build a decision tree model. The system derives concise and biologically meaningful amino acid composition (AAC) classification rules. Cross-validation tests estimate that for predicting IUPs that contain long disordered regions or are completely disordered, the AAC-rule classifier achieves a recall of 77.3% and precision of 81.4%.

1 Introduction

Proteins are linear chains composed of 20 amino acids. The amino acids are linked together by polypeptide bonds and folded into complex three-dimensional (3D) structures. The global fold of linear chains has been believed to be essential for protein function for a long time. Great efforts have been made to determine the three-dimensional structures of proteins by experimental and computational methods. Experimental methods such as X-ray diffraction and Nuclear Magnetic Resonance (NMR) spectroscopy are used to determine the coordinates of all atoms in a protein and thus its 3D structure.

Recent sequence analysis and experimental data show that a number of proteins contain extended disordered or flexible regions [1]. Such disordered regions (DRs) have little or no ordered structure under physiological conditions but nonetheless carry out important functions [2–5]. The flexibility of DRs often leads to difficulties in protein expression, purification and crystallisation [6]. NMR can provide valuable structural and dynamic information on such proteins but this technique is complex and time-consuming. As it is known that sequence determines structure, sequence should determine lack of structure as well. Sequence analysis has shown that the amino acid composition (AAC) is biased in IUPs [7] and it is feasible to predict DRs and IUPs from protein sequences.

Predicting IUPs can be cast as the binary classification problem in machine learning and data mining. Given unstructured protein sequences and protein sequences of known structures as training data, the unstructured sequences are supposedly tagged with the label P and the structured sequences are tagged with label N. A classification model can be learnt from the training data. The classification model can then classify an unseen protein sequence as belonging to class P or N. In other words, it can predict whether a sequence is an IUP.

There have been 14 published DR predictors. Comparing the predictors is difficult because of the lack of a precise definition of disordered residues. Several definitions for disorder have been proposed, including loop/coil regions where the carbon alpha (C_a) on the protein backbone has a high temperature factor and residues whose coordinates are missing in crystal structures. In this paper, following the definition of lacking a fixed 3D structure, we build a disordered database curated from DisProt [8].

Most of the 14 published DR predictors are based on complex models. VL-XT and XL1 from PONDR [1], RONN [9], VL2, VL3, VL3H and VL3E from DisProt [10], NORSp [11, 12], DISpro [13], and COILS, HOT-LOOPS and REMARK465 from DisEMBL [14] are all based on the neural networks. DISOPRED [15] and DISOPRED2 [16] are based on the support vector machine. These sophisticated modelling techniques are black box learning systems that produce models from the input and gives no basis for how such a model is derived.

In this paper, a predictor based on decision tree learning is developed, based on the amino acid composition information from protein sequences. A set of rules is produced from our curated IUP database for predicting IUPs. These rules confirmed that IUPs have low overall hydrophobicity, high net charge and low sequence complexity [17]. More importantly they present complex amino acid composition information that is previously unknown. Our work is also distinguished from previous work in that protein sequences are predicted as IUPs or otherwise structured proteins.

2 Preliminaries

A protein sequence comprises *amino acids* or *residues*. There are big proteins defined by thousands of residues as well as small peptides of several residues. Early work in structural biology has established that a protein sequence folds spontaneously and reproducibly to a unique 3D structure in order to be functional. The Protein Data Bank (PDB) [1] has over 32,000 proteins with solved structures and it grows larger every day.

Predicting IUPs from primary sequences is a binary classification problem. Training instances are presented to classification systems and classification models or classifiers are developed from the training data. In terms of estimating the predictive accuracy of a classifier, self-test tends to underestimate the error rate, as error rate is estimated on the training instances where the classifier is developed. Cross-validation is a reliable approach. By leaving out some training instances as the test instances, a classifier is developed on the remaining training instances and tested on the leave-out test instances. The error rate is estimated from the misclassified test instances.

Leave-one-out cross-validation test or the jackknife test has been widely used in evaluating protein structure class prediction [18, 19]. Each instance in turn of

[1] http://www.rcsb.org/pdb/

the training dataset is singled out and a classifier is developed on the remaining training instances and tested on the singled-out instance. The error rate of a classifier is estimated as the misclassified instances out of the whole population of the training dataset.

Traditionally in machine learning, the performance of classification systems is measured with overall accuracy. However overall accuracy can be misleading when class distribution is very unequal. With a two-class problem of the positive (P) and negative (N) classes, the recall and precision for each class provide a much clearer description of the performance of a classifier on each class.

The recall for the class P is the percentage of the number of instances correctly predicted as P (TP) compared to the number of actual instances of class P: Recall$(P) = \frac{TP}{TP+FN}$, where FN denotes the number of false negatives, the P class instances misclassified as class N. The precision for class P is the percentage of the number of instances predicted correctly in relation to the number of residues predicted as class P: Precision$(P) = \frac{TP}{TP+FP}$, where FP denotes the number of false positives, the N class instances misclassified as class P.

The Receiver Operating Characteristics (ROC) curve is a plot of the true positive rate against the false positive rate for the different parameters of a classifier. ROC curves have long been used in signal detection theory. They are also used extensively in medical and biological studies. There has been an increase in the use of ROC graphs in the machine learning and data mining communities. In addition to being a generally useful performance graphing method, they have properties that make them especially useful for domains with skewed class distribution and unequal classification error costs.

An ROC graph depicts relative trade-offs between benefits (true positives) and costs (false positives). The horizontal axis of the ROC space is the false positive rate and the vertical axis of the ROC space is the true positive rate. The diagonal line $y = x$ represents the strategy of randomly guessing a class. Generally a good classifier should produce a large area under its ROC curve at low false positive rate.

3 Predicting IUPs with Amino Acid Composition Rules

We first describe how the amino acid composition training dataset is constructed. We then show how decision tree learning is applied on the training data, and then present the classification rules obtained.

3.1 Training Data

The disordered training set in our study was extracted from DisProt (release 2.1), a curated published database of IUPs. DisProt was established by searching the relevant literature and biological databases. This database is exceptional in that it includes molten globule-like proteins [8] for the definition of the absence of a fixed 3D structure. Since the major information in DisProt is based on literature-derived descriptions of DRs, some observations have certain overlap,

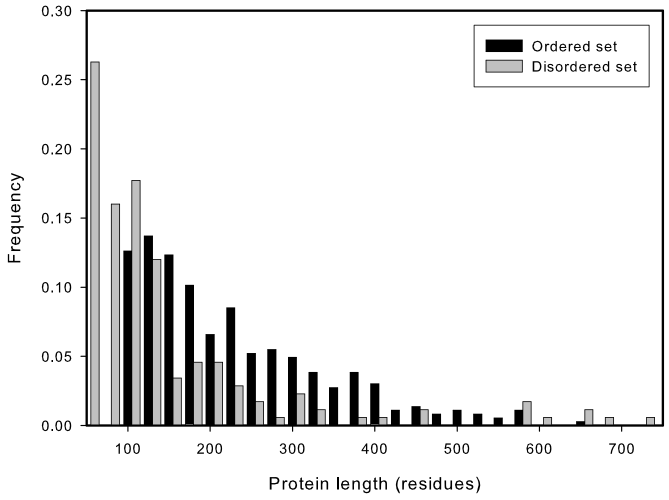


Fig. 1. The distribution of ordered vs. disordered segments

which enlarges the overall evaluated accuracy. To eliminate this impact, only the longest DR from all the overlapping DRs was chosen in each case. Furthermore only DRs of more than 30 residues were extracted in order to reduce the random noise from protein structure and focus on the AAC features of long disordered regions or completely disordered proteins. As a result, 176 DRs including 25261 residues were used as our disordered training set, which is designated as D-DisProt hereafter.

Our ordered training set was extracted from PDB_Select_25, [2] a subset of structures obtained from PDB [20] that shows less than 25% amino acid sequence homology. From the 2485 protein sequences from the PDB_Select_25, 366 higher resolution crystal structures (<2Å) that are free from missing backbone or side chain coordinates, free from non-standard amino acid residues and with sequence length larger than 80 residues were finally selected, which included 80324 residues and is designated as O-PDBselect25 hereafter. All the PDB code of our training sets are available upon request.

The contrasting distribution of disordered and ordered segments of different lengths is plotted in Fig. 1. We can see that there are more shorter segments in D-DisProt than in O-PDBselect25. Specifically more than 40% of disordered segments contain less than 100 residues, of which about 25% contain less than 80 residues. Ordered segments usually contain less than 700 residues. In contrast, disordered segments can contain thousands residues.

[2] ftp://ftp.embl-heidelberg.de/pub/databases/protein_extras/pdb_select/recent.pdb_select.

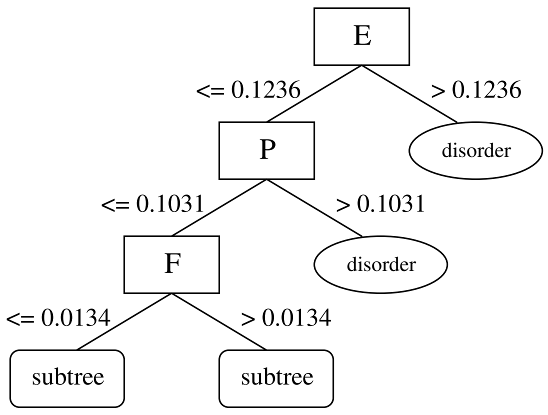


Fig. 2. A decision tree

Each protein sequence in the training data sets was represented by its amino acid composition (AAC). By a single scan of the given protein sequence database, the composition database was constructed. With our training data, the IUP database was converted into a 176 × 20 matrix and the ordered database was converted into a 366 × 20 matrix. The amino acid composition for each sequence of the IUP database is tagged with the label P and each sequence of the ordered sequence database was tagged with the label N.

3.2 Learning amino acid composition rules by decision trees

A decision tree is constructed for the amino acid composition dataset. Each node of the decision tree is a test on an amino acid X, "*freq*$(X) > \alpha$?". A sample decision tree formed from our training data is illustrated in Figure 2. The root node of the tree is on residue E with test "$freq(E) > 12.36\%$?": If "$freq(E) > 12.36\%$" the tree reaches a decision of "disorder"; otherwise the frequency of P has to be tested.

Every path from the root of an unpruned tree to a leaf gives one initial rule. The left-hand side of the rule contains all the conditions established by the path, and the right-hand side specifies the class at the leaf. Each such rule is simplified by removing conditions that do not seem helpful for discriminating the nominated class from other classes, using a pessimistic estimate of the accuracy of the rule. For each class in turn, all the simplified rules for that class are sifted to remove rules that do not contribute to the accuracy of the set of rules as a whole. The sets of rules for the classes are then ordered to minimise false positive errors and a default class is chosen. This process leads to a production rule classifier that is usually about as accurate as a pruned tree, but more understandable.

Using the popular C4.5 decision tree system [21] with default settings for parameters, only 12 rules are learned for the D-DisProt, and 4 rules for the O-PDBselect25. These rules are listed in Tables 1 and 2, respectively. Rules

Table 1. Disordered rules from D-DisProt

#	*Rule*	*Accuracy*	*#*	*Rule*	*Accuracy*
1	F≤0.013; L≤0.118; Y≤ 0.025	97.6%	7	E>0.124	94.7%
2	F≤0.007	97.4%	8	S>0.090; V≤0.034	94.6%
3	H ≤0.069; K>0.122; Y≤0.045	96.6%	9	G>0.093; I≤0.022	90.5%
4	P>0.103	95.9%	10	D>0.101; I≤0.035	89.9%
5	I≤0.022; K≤0.122; W≤0.003	95.5%	11	D>0.101; S>0.081; V>0.034	89.1%
6	C≤0.004; R≤0.011; Y≤0.039	95.3%	12	C>0.027; H>0.042; K≤0.122	79.4%

Table 2. Ordered rules from O-PDBselect25

#	*Rule*	*Accuracy*
1	D≤0.101; E≤0.124; F>0.013; H≤0.042; I>0.022; K≤0.122; P≤0.103; V>0.033	95.5%
2	E≤0.124; F>0.013; G≤0.104; K≤0.122; P≤0.103; S≤0.090; W>0.003	94.5%
3	E≤0.124; F>0.013; P≤0.103; Y>0.045	94.4%
4	F>0.013; H>0.069	85.7%

are listed in decreasing order of estimated predictive accuracy (the last column of Tables 1 and 2). All rules are very concise and understandable. The rules that describe the disordered state are much simpler than those describing the ordered state. This is a result of the biased composition and lower sequence complexity of the sequences in D-DisProt dataset. On the other hand, the AAC in the sequences of the O-PDBselect25 dataset are more uniform and sequence complexities are much higher. As a result, there are fewer rules and they tend to be more complicated.

From the biological standpoint, some rules in Table 1 are extremely explicit, such as rules 2, 4 and 7. They indicate that sequences extremely depleted in Phe ($F \leq 0.70\%$) or extremely enriched in Pro ($P > 10.3\%$) or Glu ($E > 12.4\%$) are very likely to be in a disordered state. Rule 1 shows that if a sequence lacks Phe(F), Leu(L) and Tyr(Y) at the same time, it most likely is in a disordered state. Most of the others rules listed in Table 1 are the combination of abundance in polar or hydrophilic residues and dearth of hydrophobic residues. Interestingly, positively charged residues His(H), Lys(K), and the sulphur-containing residue Cys(C) are environment-dependent in their state. For example, the sequence could be in disordered state if the content of Lys(K) is greater than 12.2%, but that of His(H) less than 6.9%, and that of Tyr(Y) less than 4.5% at the same time (rule 3). On the other hand, the sequence could also be in the disordered state if the content of Lys(K) is less than 12.2% but the content of Ile(I) is less than 2.2% and that of Trp(W) is less than 0.3% (rule 5), or the content of Cys(C) is larger than 0.3% and that of His(H) is larger than 2.7% (rule 12).

Our rules not only confirm that residues Phe, Tyr, Trp, Ile, Leu and Val are ordered promoters and Pro, Glu, Gln, Ser and Gly are disordered promoters

Table 3. The self- and jackknife- test results

	Overall accuracy	IUP Recall : Precision	Ordered Recall: Precision
Self-test	97.2%	92.6% : 98.8%	99.5% : 96.6%
Jackknife test	86.9%	77.3% : 81.4%	91.5% : 89.3%

as indicated by others [3, 7, 17], they also describe the detailed and complicated impact from the combinations of different AACs.

3.3 Predicting IUPs with AAC rules

To predict whether an unseen protein sequence is likely to be an IUP, the AAC of the sequence is first computed. The AAC of the sequence is then checked against the classification rules for each class learnt from training data, the rule with the highest accuracy is used to predict. The estimated accuracy of the rule is the probability of the prediction.

4 Performance Evaluation

With D-DisProt and O-PDBselect25, the self- and jackknife cross-validation tests were used to study the overall accuracy, recall, and precision of our AAC-rule predictor. The results are shown in Table 3. With both the self- and jackknife tests the recall and precision on the ordered proteins are better than those on IUPs. This is because of the imbalance distribution of 3:1 for number of residues of the ordered proteins to those of IUPs for training. The jackknife cross-validation test is indicating good predictive accuracy of our AAC-rule predictor on IUPs, with a recall of 77.3% and precision of 81.4%.

The ROC curve of the AAC-rule classifier derived from the jackknife test is shown in Figure 3. The figure shows that the classifier has good performance. With the default settings, at the very low false positive rate of 8.5%, the classifier achieves a true positive rate of 77.3%. This implies that our classifier can achieve good predictive accuracy at the low cost of not introducing too many errors. On the other hand, the classifier reaches a high true positive rate of 91.5% at the cost of a false positive rate of 22.7%.

5 Conclusions

Intrinsically Unstructured Proteins (IUPs) are becoming increasingly interesting because they are common and functionally important. Experimental studies of IUPs are expensive and time consuming. An effective computational tool is helpful for structural biologists to understand protein structure and related function. In this paper we have proposed an approach for deriving amino acid composition (AAC) rules for predicting IUPs. The AAC rules derived are consistent with the

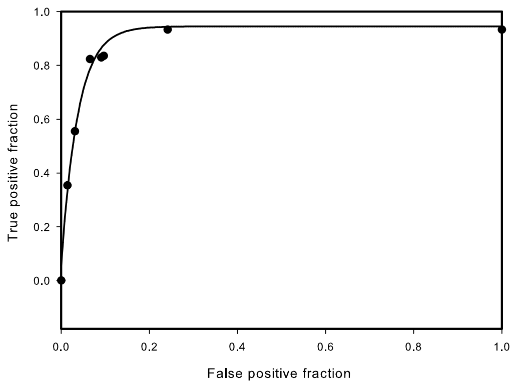


Fig. 3. The ROC curve of the AAC-rule classifier

biological findings [2, 7, 17] and also quantitatively specify the combined effect of amino acid compositions. Cross-validation tests have shown that the our amino acid composition rules have high accuracy for predicting IUPs. A user-friendly interface for our predictor is under development. Further work includes elaborating the system and a complicated tree model for predicting disordered regions in IUPs.

Acknowledgements

Z.P. Feng is supported by an APD award from Australian Research Council. The authors thank Dr Marc Cortese for his explanation of the DisProt database.

References

1. X. Li, P. Romero, M. Rani, A. K. Dunker, and Z. Obradovic. Predicting protein disorder for N-,C-,and internal regions. *Genome Informatics*, 10:30–40, 1999.
2. H. J. Dyson and P. E. Wright. Intrinsically unstructured proteins and their functions nature. *Nature Reviews Molecular Cell Biology*, 6:197–208, 2005.
3. A. K. Dunker, C. J. Brown, J D. Lawson, L. M. Iakoucheva, and Z. Obradovic. Intrinsic disorder and protein function. *Biochemistry*, 41:6573–6582, 2002.
4. L. M. Iakoucheva, C. J. Brown, J. D. Lawson, Z. Obradovic, and A. K. Dunker. Intrinsic disorder in cell-signaling and cancer-associated proteins. *J. Mol. Biol.*, 323:573–584, 2002.
5. C. Szasz P. Tompa and L. Buday. Structural disorder throws new light on moonlighting. *TRENDS in Biochem. Sci.*, 30:484–489, 2005.
6. K. W. Plaxco and M. Gross. Unfolded, yes, but random? never! *Nat.Struct.Biol.*, 8:659–670, 2001.
7. V. N. Uversky, J. R. Gillespie, and A. L. Fink. Why are natively unfolded proteins unstructured under physiologic conditions? *Proteins: Structure, Function, and Genetics*, 41:415–427, 2000.

8. S. Vucetic, Z. Obradovic, V. Vacic, P. Radivojac, K. Peng, L. M. Iakoucheva, M. S. Cortese, J. D. Lawson, C. J. Brown, J. GSikes, C. D. Newton, and A. K. Dunker. DisProt: A database of protein disorder. *Bioinformatics*, 21:137–140, 2005.
9. R. Thomson and R. Esnouf. Prediction of natively disordered regions in proteins using a bio-basis function neural network. In *LNCS 3177*, pages 108–116, 2004.
10. Z. Obradovic, K. Peng, S. Vucetic, P. Radivojac, C. Brown, and A. K. Dunker. Predicting intrinsic disorder from amino acid sequence. *Proteins: Structure, Function and Bioinformatics*, 53:566–572, 2003.
11. J. Liu, H. Tan, and B. Rost. Loopy proteins appear conserved in evolution. *J. Mol. Biol.*, 332:53–64, 2002.
12. J. Liu and B. Rost. Norsp: predictions of long regions without regular secondary structure. *Nucleic Acids Research*, 31:3833–3835, 2003.
13. J. Cheng, M. Sweredoski, and P. Baldi. Accurate prediction of protein disordered regions by mining protein structure data. *Data Mining and Knowledge Discovery*, in press (2005).
14. R. Linding, L. J. Jensen, F. Diella, P Bork, T J Gibson, and R. B. Russell. Protein disorder prediction: implications for structural proteomics. *Structure*, 11:1453–1459, 2003.
15. J. J. Ward, L. J. McGuffin, K. Bryson, B. F. Buxton, and D. T. Jones. The disopred server for the prediction of protein disorder. *Bioinformatics*, 20:2138–2139, 2004.
16. D. T. Jones and J. J. Ward. Prediction of disordered regions in proteins from position specific score matrices. *Proteins: Structure, Function, and Genetics,*, 53:573–578, 2003.
17. K. N. UVERSKY. Natively unfolded proteins: A point where biology waits for physics. *Protein Sci.*, 11:739–756, 2002.
18. P. Klein. Prediction of protein structural class by discriminant analysis. *Biochem. Biophys.*, Acta 874:205–215, 1986.
19. K. V. Madia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis.* Academic Press, London, 1979.
20. H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P.E. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.
21. J. Ross Quinlan. *C4.5: programs for machine learning.* Morgan Kaufman Publishers, 1993.

A Comparative Study of Semi-naive Bayes Methods in Classification Learning

Fei Zheng and Geoffrey I. Webb

Clayton School of Information Technology
Monash University
Melbourne VIC 3800, Austrailia
{feizheng, Geoff.Webb}@infotech.monash.edu.au

Abstract. Numerous techniques have sought to improve the accuracy of Naive Bayes (NB) by alleviating the attribute interdependence problem. This paper summarizes these semi-naive Bayesian methods into two groups: those that apply conventional NB with a new attribute set, and those that alter NB by allowing inter-dependencies between attributes. We review eight typical semi-naive Bayesian learning algorithms and perform error analysis using the bias-variance decomposition on thirty-six natural domains from the UCI Machine Learning Repository. In analysing the results of these experiments we provide general recommendations for selection between methods.

Keywords

Naive Bayes, Semi-naive Bayes, attribute independence assumption, bias-variance decomposition

1 Introduction

Supervised classification is a basic task in data mining, predicting a discrete class label for a previously unseen instance $I = \langle a_1, \ldots, a_n \rangle$ from a labelled training sample, where a_i is the value of the i^{th} attribute A_i. There are numerous approaches to produce classifiers, functions that map an unlabelled instance to a class label, such as decision trees, neural networks and probabilistic methods. The Bayesian classifier [1] is a well known probabilistic induction method. It predicts the class label for I by selecting

$$\underset{c_i}{\operatorname{argmax}} \left(P(c_i \mid a_1, \ldots, a_n) \right) \propto \underset{c_i}{\operatorname{argmax}} \left(P(c_i) P(a_1, \ldots, a_n \mid c_i) \right), \tag{1}$$

where $c_i \in \{c_1, \ldots, c_k\}$ is the i^{th} value of the class variable C.

However, accurate estimation of $P(a_1, \ldots, a_n \mid c_i)$ is non-trivial. Naive Bayes (NB) [2–4] gets round this problem by making the assumption that the attributes are independent given the class. Although the assumption is unrealistic in many practical scenarios, NB has exhibited competitive accuracy with other learning algorithms, especially in domains without highly related attributes. There are many attempts to explain NB's impressive performance, and to develop techniques that further improve its accuracy by alleviating the attribute interdependence problem [4–19]. Collectively, we call these methods *semi-naive Bayesian methods*. Domingos and Pazzani [20] argue that the interdependence between attributes will not affect NB's accuracy performance, so long as it can generate the correct ranks of conditional probabilities for the classes. However, the success of semi-naive Bayesian methods suggest that weakening the attribute independence assumption is effective.

Gaining a better understanding of the strengths and limitations of different semi-naive Bayesian algorithms motivates our comparative study. In this paper, we broadly classify semi-naive Bayesian algorithms into two groups, and examine eight representative semi-naive Bayesian algorithms, including a detailed time and space complexity analysis. We compare these algorithms on thirty-six natural domains from the UCI Machine Learning Repository [21] by using the bias-variance decomposition [22], a key tool for understanding machine learning algorithms. We also give some general recommendations for selecting appropriate semi-naive Bayesian methods.

2 Naive Bayes (NB)

Naive Bayes (NB) [2–4] simplifies probabilistic induction by making the assumptions that the attributes are independent given the class and all the probability estimations from the training sample are accurate. Hence, NB classifies I by selecting

$$\underset{c_i}{\operatorname{argmax}} \left(P(c_i) \prod_{j=1}^{n} P(a_j \mid c_i) \right) . \tag{2}$$

Due to the independence assumption, NB is simple, and computationally efficient. Although the attribute independence assumption is often violated, previous research [3, 12, 20] has shown that NB behaves well across many domains. As it uses a fixed formula to classify, there is no model selection.

At training time NB generates a one-dimensional table of class probability estimates, indexed by the classes, and a two-dimensional table of conditional attribute-value probability estimates, indexed by the classes and attribute-values. The time complexity of calculating the estimates is $O\big(tn\big)$, where t is the number of training examples. The resulting space complexity is $O\big(knv\big)$, where v is the mean number of values per attribute. At classification time, to classify a single example has time complexity $O\big(kn\big)$ using the tables formed at training time with space complexity $O\big(knv\big)$.

3 Semi-naive Bayes Methods

Previous semi-naive Bayesian methods can be roughly subdivided into two groups. The first group establishes NB with a new attribute set, which can be generated by deleting attributes [4, 5, 9] and joining attributes [6, 9]. The second group adds explicit links between attributes, which represent attribute interdependencies. Sahami [10] introduces the notion of the x-dependence Bayesian classifier, which allows each attribute to depend on the class and at most x other attributes. NB is a 0-dependence classifier, and the methods that add explicit links between attributes can be classified into those that establish 1-dependence classifiers [12, 14, 19] and those that establish x-dependence classifiers ($x \geq 1$) [8, 16]. In addition, these methods can be classified into eager learning methods [4, 5, 8, 9, 12, 14, 19], which perform learning at training time, and lazy learning methods [16], which defer learning until classification time. The following Sections present these methods in more details.

3.1 Backwards Sequential Elimination (BSE) and Forward Sequential Selection (FSS)

In Naive Bayes, all the attributes are utilised for classification. When two attributes are strongly related, NB may overweight the influence from these two attributes, and reduce the influence of the other attributes, which can result in prediction bias. Deleting one of these attributes may have the effect of alleviating the problem.

Backwards Sequential Elimination (BSE) [5] and Forward Sequential Selection (FSS) [4] select a subset of attributes using leave-one-out cross validation error as a selection criterion and establish a NB with these attributes. Starting from the full set of attributes, BSE successively eliminates the attribute whose elimination most improves accuracy, until there is no further accuracy improvement. FSS uses the reverse search direction, that is iteratively adding the attribute whose addition most improves accuracy, starting with the empty set of attributes. The subset of selected attributes is denoted as $Atts = \{A_{g_1}, \dots, A_{g_h}\}$. Independence is assumed among the resulting attributes given the class. Hence, BSE and FSS classify I by selecting

$$\underset{c_i}{\operatorname{argmax}} \left(P(c_i) \prod_{j=g_1}^{g_h} P(a_j \mid c_i) \right) . \tag{3}$$

At training time BSE and FSS generate a one-dimensional table of class probability estimates and a two-dimensional table of conditional attribute-value probability estimates, as NB does. As they perform leave-one-out cross validation to select the subset of attributes, they must also store the training data, with additional space complexity $O\big(tn\big)$. The resulting space complexity is $O\big(tn + knv\big)$. Deleting attributes for BSE and adding attributes for FSS have time complexity of $O\big(tkn^2\big)$, as leave-one-out cross validation will be performed at most $O\big(n^2\big)$ times. They have identical time and space complexity with NB at classification time.

3.2 Backward Sequential Elimination and Joining (BSEJ)

Creating new compound attributes when inter-dependencies between attributes are detected is another approach to relaxing the attribute independence assumption. Semi-naive Bayesian classifier [6] uses exhaustive search to join attribute values iteratively based on a statistical method. However, the experimental results are somewhat disappointing.

Backward Sequential Elimination and Joining (BSEJ) [9] uses predictive accuracy as a merging criterion to create new Cartesian product attributes. The value set of a new compound attribute is the Cartesian product of the value sets of the two original attributes. As well as joining attributes, BSEJ also considers deleting attributes. BSEJ repeatedly joins the pair of attributes or deletes the attribute that most improves predictive accuracy using leave-one-out cross validation. This process terminates if there is no accuracy improvement. The resulting Cartesian product attribute set is denoted as $JoinAtts = \{Join_{g_1}, \ldots, Join_{g_h}\}$. The remaining original attributes that have not been either deleted or joined are indicated as $\{A_{l_1}, \ldots, A_{l_q}\}$. Hence, BSEJ classifies I by selecting

$$\underset{c_i}{\operatorname{argmax}} \left(P(c_i) \prod_{j=g_1}^{g_h} P(join_j \mid c_i) \prod_{r=l_1}^{l_q} P(a_r \mid c_i) \right), \tag{4}$$

where $join_j$ is the value of attribute $Join_j$.

At training time BSEJ generates a one-dimensional table of class probability estimates, a two-dimensional table of conditional attribute-value probability estimates, as NB does. It also generates two-dimensional tables of conditional joined attribute-value probability estimates, indexed by the classes and compound attribute-values. In the worst case, the new Cartesian product attribute has v^n values. Therefore, the space complexity is $O\big(tn + kv^n\big)$. BSEJ considers at most $O\big(n^3\big)$ Cartesian product attributes. The time complexity of joining and deleting attributes is $O\big(tkn^3\big)$. At classification time, to classify a single example has time complexity $O\big(kn\big)$ and space complexity $O\big(kv^n\big)$.

3.3 Tree Augment Naive Bayes (TAN) and SuperParent TAN (SP-TAN)

Friedman et al. [12] compared NB with unrestricted Bayesian networks. The observation that unrestricted Bayesian networks did not usually result in accuracy improvement and sometimes lead to reduction in accuracy motivated them to use an intermediate solution that allows each attribute to depend on at most one non-class attribute, called the parent of the attribute. Based on this representation, they utilised conditional mutual information to efficiently find a maximum spanning tree as a classifier. As each attribute depends on at most one other non-class attribute, TAN is a 1-dependence classifier. The parent of each attribute

A_i is indicated as $\pi(A_i)$. Hence, TAN classifies by selecting

$$\underset{c_i}{\operatorname{argmax}} \left(P(c_i) \prod_{j=1}^{n} P(a_j \mid c_i, \pi(a_j)) \right). \tag{5}$$

At training time TAN generates a one-dimensional table of class probability estimates, and a three-dimensional table of probability estimates for each attribute-value, conditioned by each other attribute-value and each class, with space complexity $O\big(k(nv)^2\big)$. The time complexity of forming the three dimensional probability table is $O\big(tn^2\big)$, as it requires each entry for every combination of the two attribute-values for every instance to be updated. Creating the conditional mutual information matrix requires each pair of attributes, every pairwise combination of their respective values in conjunction with each class to be considered, resulting in time complexity $O\big(kn^2v^2\big)$. The parent function is then generated by establishing a maximal spanning tree, with time complexity $O\big(n^2 \log n\big)$. At classification time, to classify a single example has time complexity $O\big(kn\big)$. The three dimensional conditional probability table formed at training time can be compressed by storing probability estimates for each attribute-value conditioned by the parent selected for that attribute and the class. Hence, the space complexity is $O\big(knv^2\big)$.

SuperParent TAN (SP-TAN) [14], a variant of TAN, uses a different approach to construct the parent function. It uses the same representation as TAN, but utilises leave-one-out cross validation error as a criterion to add a link. The SuperParent is the attribute that is the parent of all the other orphans, the attributes without a non-class parent. There are two steps to add a link: first selecting the best SuperParent that improves accuracy the most, and then selecting the best child of the SuperParent from orphans. SP-TAN stops adding links when there is no accuracy improvement. As TAN and SP-TAN use different criteria to establish the parent function, TAN tends to add $N-1$ links, while SP-TAN may have fewer links between attributes. Another difference is the direction of links. TAN chooses the direction randomly, while SP-TAN makes the direction from SuperParents to their favorite children. SP-TAN also uses Equation 5 to classify an unseen instance.

At training time SP-TAN needs additional space complexity $O\big(tn\big)$ for storing the training data compared with TAN. The time complexity of forming the parent function is $O\big(tkn^3\big)$, as the selection of a single SuperParent is order $O\big(tkn^2\big)$, the selection of the favorite child of the SuperParent is order $O\big(tkn\big)$, and parent selection is performed repeatedly at most $O\big(n\big)$ times. SP-TAN has identical classification time complexity and space complexity to TAN.

3.4 NBTree

NBTree [8] is a hybrid approach combining NB and decision tree learning. It partitions the training data using a tree structure and establishes a local NB in each leaf. It uses 5-fold cross validation accuracy estimate as the splitting criterion. A split is defined to be significant if the relative error reduction is greater than 5% and the splitting node has at least 30 instances. When there is no significant improvement, NBTree stops the growth of the tree. As the number of splitting attributes is greater than or equals one, NBTree is a x-dependence classifier. The classical decision tree predicts the same class for all the instances that reach a leaf. In NBTree, these instances are classified using a local NB in the leaf, which only considers those non-tested attributes. Let $S = \{S_1, \ldots, S_g\}$ be the set of the test attributes on the path leading to the leaf, and let $R = \{R_1, \ldots, R_{n-g}\}$ be the set of the remaining attributes, we have

$$P(C, \mathbf{I}) = P(S)P(C \mid S)P(R \mid C, S) \propto P(C \mid S)P(R \mid C, S). \tag{6}$$

Therefore, NBTree classifies I by selecting

$$\operatorname*{argmax}_{c_i} \left(P(c_i \mid s) \prod_{j=1}^{n-g} P(r_j \mid c_i, s) \right), \tag{7}$$

where s is a value of S and r_j is a value of R_j.

In NBTree, the number of leaves possible is $O\big(t\big)$, and the height of the tree is $O\big(log_v t\big)$. Therefore, there are $O\big(t/v\big)$ internal nodes in the tree. At the root, NBTree performs 5-fold cross validation on each attribute to select the best one to split, time complexity of $O\big(tkn^2\big)$. Less time is required for the other internal nodes. Hence, the time complexity of building the tree is $O\big(t^2kn^2/v\big)$. Each leaf has $O\big(n - log_v t\big)$ attributes and stores a two-dimensional table of conditional attribute-value probability estimates. The space complexity is $O\big(tk(n - log_v t)v\big)$. At classification time, to classify a single example has time complexity $O\big(kn\big)$, and space complexity $O\big(tk(n - log_v t)v\big)$.

3.5 Lazy Bayesian Rules (LBR)

Zheng and Webb [16] developed Lazy Bayesian Rules (LBR), which adopts a lazy approach, and generates a new Bayesian rule for each test example. The antecedent of a Bayesian rule is a conjunction of attribute-value pairs, and the consequent of the rule is a local NB, which uses those attributes that do not appear in the antecedent to classify. LBR stops adding attribute-value pairs into the antecedent if the outcome of a one-tailed pairwise sign test of error difference is not better than 0.05. As the number of the attribute-value pairs in the antecedent is greater than or equals one, LBR is a x-dependence classifier. Let $s = \{s_1, \ldots, s_g\}$ be the set of attribute values in the antecedent, and let $r = \{r_1, \ldots, r_{n-g}\}$ be the set of remaining attribute values, LBR classifies I by selecting

$$\operatorname*{argmax}_{c_i} \left(P(c_i \mid s) \prod_{j=1}^{n-g} P(r_j \mid c_i, s) \right). \tag{8}$$

The Bayesian rule generated by LBR can be described as a branch of a tree built by NBTree. LBR generates a rule for each unseen instance, while NBtree builds a single model according to all the examples in the training data. If examples are not evenly distributed among branches in NBTree, small disjuncts, which cover only few training samples, will result in poor prediction performance. As LBR uses lazy learning, it may avoid this problem. LBR is efficient when few examples are to be classified. However, the computational overhead of LBR may be excessive when large numbers of examples are to be classified.

At training time, the time and space complexity of LBR are $O\big(tn\big)$, as it only stores the training data. At classification time, LBR adds attribute-value pairs to the antecedent with time complexity of $O\big(tkn^3\big)$, as the selection of an attribute-value pair for the antecedent is order $O\big(tkn^2\big)$ and this selection is performed repeatedly until there is no significant improvement on accuracy. The space complexity is $O\big(tn + knv\big)$.

3.6 Averaged One-Dependence Estimators (AODE)

To avoid model selection and retain the efficiency of 1-dependence classifiers, Webb et al. [19] proposed AODE, which averages the predictions of all qualified 1-dependence classifiers. In each 1-dependence classifier, all attributes depend on the class and a single attribute. For any attribute value a_j,

$$P(c_i, I) = P(c_i, a_j)P(I \mid c_i, a_j). \tag{9}$$

This equality holds for every a_j. Therefore,

$$P(c_i, I) = \frac{\sum_{j:1\leq j\leq n\wedge F(a_j)\geq m} P(c_i, a_j)P(I \mid c_i, a_j)}{|\{j : 1 \leq j \leq n \wedge F(a_j) \geq m\}|}, \tag{10}$$

where $F(a_j)$ is the frequency of a_j in the training sample.

AODE classifies by selecting:

$$\underset{c_i}{\operatorname{argmax}} \left(\sum_{j:1\leq j\leq n\wedge F(a_j)\geq m} P(c_i, a_j) \prod_{h=1}^{n} P(a_h \mid c_i, a_j) \right). \tag{11}$$

If $P(a_j)$ is small, the estimate of $P(I|c_i, a_j)$ may be unreliable. Hence, AODE averages models where the frequency of the parent attribute is larger than $m = 30$, a widely used minimum sample size in statistics.

At training time AODE generates a three-dimensional table of probability estimates for each attribute-value, conditioned by each other attribute-value and each class. The resulting space complexity is $O(k(nv)^2)$. Forming this table is of time complexity $O(tn^2)$. Classification requires the tables of probability estimates formed at training time of space complexity $O(k(nv)^2)$. The time complexity of classifying a single example is $O(kn^2)$ as we need to consider each pair of qualified parent and child attribute within each class.

4 Algorithm Comparisons

In this study, we compare eight representative semi-naive algorithms and NB. These semi-naive Bayesian algorithms are BSE, FSS, BSEJ, TAN, SP-TAN, NBTree, LBR and AODE.

4.1 Experimental Domains and Methodology

The thirty-six data sets from the UCI Machine Learning Repository used in our experiments are shown in Table 1. The experiments were performed in the Weka workbench [23] on a dual-processor 1.7 GHz Pentium 4 Linux computer with 2 Gb RAM, and all data were discretized using MDL discretization [24].

Table 1. Data sets

No.	Domain	Case	Att	Class	No.	Domain	Case	Att	Class
1	Adult	48,842	14	2	19	Labor negotiations	57	16	2
2	Annealing	898	38	6	20	LED	1,000	7	10
3	Balance Scale	625	4	3	21	Letter Recognition	20,000	16	26
4	Breast Cancer (Wisconsin)	699	9	2	22	Liver Disorders (bupa)	345	6	2
5	Chess	551	39	2	23	Lung Cancer	32	56	3
6	Credit Screening	690	15	2	24	Mfeat-mor	2,000	6	10
7	Echocardiogram	131	6	2	25	New-Thyroid	215	5	3
8	German	1,000	20	2	26	Pen Digits	10,992	16	10
9	Glass Identification	214	9	3	27	Postoperative Patient	90	8	3
10	Heart	270	13	2	28	Primary Tumor	339	17	22
11	Heart Disease (cleveland)	303	13	2	29	Promoter Gene Sequences	106	57	2
12	Hepatitis	155	19	2	30	Segment	2,310	19	7
13	Horse Colic	368	21	2	31	Sign	12,546	8	3
14	House Votes 84	435	16	2	32	Sonar Classification	208	60	2
15	Hungarian	294	13	2	33	Syncon	600	60	6
16	Hypothyroid	3,163	25	2	34	Tic-Tac-Toe Endgame	958	9	2
17	Ionosphere	351	34	2	35	Vehicle	846	18	4
18	Iris Classification	150	4	3	36	Wine Recognition	178	13	3

As bias-variance decomposition provides a valuable insight into the aspects that affect the performance of a learning algorithm, we use Weka's bias-variance decomposition utility which utilised the experimental method proposed by Kohavi and Wolpert [22] to compare the performance of the nine algorithms. Bias denotes the systematic component of error, and variance describes the component of error that stems from sampling [22]. There is a bias-variance tradeoff such that bias typically increases when variance decreases and vice versa.

In Kohavi and Wolpert's method, the training data are divided into a training pool and a test pool randomly. Each pool contains 50% of the data. 50 local training sets, each containing half of the training pool, are sampled from the training pool. Classifiers are generated from each local training set, which is 25% of the full data set. Bias, variance and error are estimated from the performance of the classifiers on the test set.

4.2 Experimental Results

The mean error, bias and variance across all the thirty-six data sets for the nine algorithms are shown in Table 2, 3 and 4 respectively. The pairwise win/draw/loss

Table 2. Error

No.	Domain	NB	AODE	NBTree	LBR	TAN	SP-TAN	BSEJ	BSE	FSS
1	Adult	0.168	0.152	0.144	0.140	0.147	0.147	0.141	0.146	0.144
2	Annealing	0.082	0.065	0.085	0.064	0.067	0.067	0.070	0.076	0.123
3	Balance Scale	0.303	0.302	0.304	0.302	0.303	0.300	0.301	0.304	0.319
4	Breast Cancer (Wisconsin)	0.030	0.027	0.031	0.030	0.050	0.030	0.030	0.030	0.050
5	Chess	0.143	0.140	0.151	0.141	0.128	0.137	0.133	0.142	0.186
6	Credit Screening	0.171	0.163	0.174	0.172	0.177	0.172	0.172	0.172	0.167
7	Echocardiogram	0.389	0.382	0.388	0.392	0.388	0.388	0.382	0.386	0.389
8	German	0.268	0.262	0.283	0.269	0.277	0.268	0.270	0.269	0.288
9	Glass Identification	0.300	0.299	0.299	0.303	0.300	0.295	0.298	0.300	0.313
10	Heart	0.215	0.216	0.232	0.215	0.236	0.218	0.224	0.221	0.269
11	Heart Disease (cleveland)	0.174	0.176	0.191	0.174	0.176	0.178	0.185	0.188	0.246
12	Hepatitis	0.139	0.140	0.144	0.140	0.143	0.138	0.138	0.140	0.153
13	Horse Colic	0.221	0.219	0.228	0.210	0.213	0.219	0.222	0.218	0.218
14	House Votes 84	0.086	0.054	0.064	0.069	0.068	0.082	0.082	0.083	0.039
15	Hungarian	0.169	0.173	0.176	0.173	0.179	0.172	0.176	0.172	0.196
16	Hypothyroid	0.024	0.021	0.018	0.016	0.025	0.018	0.015	0.015	0.014
17	Ionosphere	0.119	0.102	0.121	0.119	0.099	0.118	0.114	0.113	0.137
18	Iris Claasification	0.058	0.058	0.061	0.058	0.056	0.058	0.057	0.058	0.060
19	Labor negotiations	0.150	0.150	0.151	0.196	0.168	0.154	0.154	0.150	0.249
20	LED	0.255	0.258	0.272	0.257	0.271	0.259	0.265	0.265	0.271
21	Letter Recognition	0.292	0.193	0.238	0.220	0.212	0.210	0.250	0.287	0.288
22	Liver Disorders (bupa)	0.424	0.424	0.424	0.424	0.424	0.424	0.424	0.424	0.424
23	Lung Cancer	0.556	0.556	0.556	0.557	0.562	0.555	0.556	0.550	0.619
24	mfeat-mor	0.317	0.311	0.320	0.313	0.312	0.314	0.322	0.317	0.322
25	New-Thyroid	0.074	0.074	0.077	0.074	0.077	0.075	0.075	0.075	0.108
26	Pen Digits	0.132	0.037	0.071	0.065	0.066	0.055	0.078	0.124	0.125
27	Postoperative Patient	0.366	0.366	0.366	0.364	0.383	0.386	0.380	0.354	0.319
28	Primary Tumor	0.559	0.572	0.603	0.571	0.593	0.571	0.573	0.567	0.649
29	Promoter Gene Sequences	0.130	0.130	0.130	0.132	0.315	0.134	0.134	0.133	0.248
30	Segment	0.112	0.071	0.081	0.092	0.082	0.090	0.090	0.092	0.084
31	Sign	0.362	0.302	0.279	0.280	0.292	0.297	0.287	0.362	0.364
32	Sonar Classification	0.274	0.275	0.286	0.274	0.293	0.279	0.280	0.282	0.301
33	Syncon	0.069	0.059	0.095	0.069	0.058	0.069	0.068	0.068	0.115
34	Tic-Tac-Toe Endgame	0.296	0.261	0.254	0.291	0.294	0.295	0.265	0.294	0.293
35	Vehicle	0.444	0.383	0.375	0.385	0.382	0.428	0.421	0.433	0.420
36	Wine Recognition	0.040	0.042	0.049	0.040	0.053	0.040	0.044	0.043	0.158
	Mean	0.220	0.206	0.214	0.211	0.219	0.212	0.213	0.218	0.241

summary of error, bias and variance for all the algorithms on thirty-six data sets are presented in Table 5, 6 and 7. The win/draw/loss record in each table entry compares the algorithm with which the row is labelled (L) against the algorithm with which the column is labelled (C). The number of wins is the number of data sets for which L achieved a lower mean value for the metric than C. Losses represent higher mean values and draws represent values that are identical for 3 decimal places. The algorithms are sorted in ascending order on the mean metric in each win/draw/loss table. As no specific prediction about relative performance has been made, the p value is the outcome of a two-tailed binomial sign test. We assess a difference as significant if $p \leq 0.05$.

Considering first the error outcomes, AODE achieves the lowest mean error, its mean error being substantially (0.010 or more) lower than that of BSE, TAN, NB and FSS. The mean error of FSS is substantially higher than that of all the other algorithms. The win/draw/loss record indicates that AODE has a significant advantage over all the other algorithms, except LBR and SP-TAN. The advantage of LBR, SP-TAN and BSE is significant compared to NBTree

Table 3. Bias

No.	Domain	NB	AODE	NBTree	LBR	TAN	SP-TAN	BSEJ	BSE	FSS
1	Adult	0.156	0.139	0.123	0.127	0.129	0.119	0.114	0.115	0.122
2	Annealing	0.053	0.045	0.048	0.041	0.043	0.043	0.042	0.046	0.092
3	Balance Scale	0.175	0.172	0.177	0.173	0.172	0.172	0.174	0.172	0.181
4	Breast Cancer (Wisconsin)	0.028	0.025	0.025	0.028	0.027	0.028	0.026	0.026	0.030
5	Chess	0.104	0.101	0.078	0.097	0.062	0.090	0.081	0.095	0.112
6	Credit Screening	0.147	0.138	0.117	0.147	0.130	0.143	0.138	0.172	0.124
7	Echocardiogram	0.249	0.247	0.248	0.253	0.246	0.250	0.248	0.245	0.246
8	German	0.203	0.195	0.183	0.202	0.174	0.196	0.185	0.197	0.217
9	Glass Identification	0.169	0.168	0.160	0.167	0.164	0.166	0.161	0.161	0.153
10	Heart	0.156	0.156	0.153	0.156	0.165	0.154	0.152	0.146	0.143
11	Heart Disease (cleveland)	0.127	0.127	0.119	0.127	0.117	0.126	0.124	0.123	0.124
12	Hepatitis	0.098	0.096	0.082	0.094	0.078	0.095	0.088	0.094	0.083
13	Horse Colic	0.188	0.179	0.158	0.177	0.170	0.183	0.177	0.183	0.174
14	House Votes 84	0.077	0.043	0.028	0.046	0.044	0.071	0.071	0.070	0.028
15	Hungarian	0.156	0.156	0.144	0.157	0.134	0.155	0.151	0.150	0.158
16	Hypothyroid	0.021	0.018	0.012	0.013	0.022	0.014	0.012	0.012	0.013
17	Ionosphere	0.077	0.068	0.070	0.077	0.063	0.076	0.075	0.073	0.075
18	Iris Claasification	0.037	0.037	0.038	0.037	0.034	0.038	0.039	0.039	0.039
19	Labor negotiations	0.046	0.046	0.047	0.068	0.057	0.048	0.045	0.044	0.088
20	LED	0.209	0.211	0.209	0.208	0.221	0.208	0.207	0.211	0.212
21	Letter Recognition	0.230	0.133	0.102	0.103	0.124	0.110	0.142	0.226	0.223
22	Liver Disorders (bupa)	0.292	0.292	0.292	0.292	0.292	0.292	0.292	0.292	0.292
23	Lung Cancer	0.311	0.311	0.312	0.312	0.375	0.309	0.310	0.306	0.311
24	mfeat-mor	0.246	0.240	0.212	0.231	0.235	0.234	0.218	0.227	0.217
25	New-Thyroid	0.039	0.040	0.037	0.039	0.028	0.039	0.039	0.039	0.039
26	Pen Digits	0.111	0.023	0.025	0.025	0.035	0.025	0.045	0.097	0.094
27	Postoperative Patient	0.299	0.299	0.300	0.300	0.315	0.306	0.307	0.298	0.305
28	Primary Tumor	0.346	0.348	0.330	0.352	0.370	0.342	0.330	0.331	0.354
29	Promoter Gene Sequences	0.043	0.043	0.044	0.044	0.134	0.044	0.045	0.048	0.080
30	Segment	0.075	0.044	0.034	0.047	0.039	0.056	0.053	0.055	0.043
31	Sign	0.324	0.260	0.206	0.218	0.245	0.235	0.214	0.310	0.311
32	Sonar Classification	0.181	0.180	0.172	0.181	0.169	0.182	0.172	0.175	0.178
33	Syncon	0.046	0.037	0.027	0.046	0.027	0.046	0.045	0.045	0.033
34	Tic-Tac-Toe Endgame	0.234	0.191	0.107	0.207	0.195	0.199	0.134	0.214	0.192
35	Vehicle	0.315	0.255	0.225	0.248	0.231	0.300	0.299	0.306	0.267
36	Wine Recognition	0.015	0.016	0.014	0.015	0.017	0.016	0.016	0.016	0.063
	Mean	0.155	0.141	0.129	0.140	0.141	0.142	0.138	0.149	0.150

and FSS. All the algorithms, except NB, have a significant advantage over FSS. It is notable that AODE is the only algorithm to have a significant advantage in error over NB.

With respect to bias, NBTree exhibits the lowest mean bias, its mean bias being substantially lower than that of all the remaining algorithms but BSEJ. The win/draw/loss record shows that NBTree has a significant advantage over the other algorithms, except TAN. The advantage of BSEJ is significant compared with SP-TAN and NB. All the algorithms except FSS have significant advantage over NB.

Turning to variance, the mean variance of NB and AODE is substantially lower than that of BSEJ, TAN, NBTree and FSS. The win/draw/loss record indicates that NB has a significant advantage over the other algorithms, but AODE. AODE shares similar levels of variance with NB and LBR, and has a significant advantage over the other algorithms. LBR and SP-TAN have a significant advantage over BSEJ, TAN, NBTree and FSS. The advantage of BSE

Table 4. Variance

No.	Domain	NB	AODE	NBTree	LBR	TAN	SP-TAN	BSEJ	BSE	FSS
1	Adult	0.011	0.012	0.020	0.013	0.018	0.027	0.027	0.031	0.021
2	Annealing	0.028	0.020	0.036	0.023	0.023	0.024	0.027	0.030	0.031
3	Balance Scale	0.125	0.128	0.125	0.126	0.128	0.126	0.125	0.129	0.135
4	Breast Cancer (Wisconsin)	0.002	0.002	0.006	0.002	0.023	0.002	0.004	0.004	0.020
5	Chess	0.038	0.038	0.072	0.043	0.065	0.046	0.051	0.046	0.074
6	Credit Screening	0.024	0.025	0.056	0.024	0.047	0.028	0.034	0.037	0.042
7	Echocardiogram	0.137	0.133	0.138	0.137	0.139	0.135	0.131	0.138	0.140
8	German	0.063	0.066	0.098	0.066	0.101	0.071	0.084	0.071	0.070
9	Glass Identification	0.129	0.129	0.136	0.134	0.134	0.127	0.134	0.136	0.156
10	Heart	0.058	0.058	0.078	0.058	0.070	0.063	0.070	0.074	0.123
11	Heart Disease (cleveland)	0.046	0.048	0.071	0.046	0.059	0.051	0.059	0.063	0.119
12	Hepatitis	0.040	0.043	0.061	0.044	0.064	0.043	0.049	0.045	0.069
13	Horse Colic	0.032	0.040	0.068	0.033	0.042	0.035	0.043	0.034	0.043
14	House Votes 84	0.009	0.010	0.035	0.022	0.024	0.011	0.011	0.013	0.011
15	Hungarian	0.013	0.017	0.032	0.016	0.044	0.016	0.025	0.021	0.038
16	Hypothyroid	0.003	0.003	0.006	0.002	0.003	0.004	0.003	0.003	0.002
17	Ionosphere	0.041	0.033	0.050	0.041	0.036	0.041	0.039	0.039	0.061
18	Iris Claasification	0.021	0.021	0.022	0.021	0.021	0.019	0.018	0.019	0.020
19	Labor negotiations	0.102	0.102	0.102	0.126	0.109	0.104	0.107	0.104	0.157
20	LED	0.045	0.046	0.062	0.048	0.049	0.050	0.057	0.052	0.058
21	Letter Recognition	0.061	0.058	0.133	0.114	0.086	0.098	0.106	0.060	0.064
22	Liver Disorders (bupa)	0.130	0.130	0.130	0.130	0.130	0.130	0.130	0.130	0.130
23	Lung Cancer	0.240	0.240	0.240	0.240	0.184	0.241	0.241	0.239	0.302
24	mfeat-mor	0.070	0.070	0.106	0.081	0.075	0.079	0.101	0.088	0.103
25	New-Thyroid	0.034	0.034	0.038	0.034	0.049	0.035	0.036	0.036	0.068
26	Pen Digits	0.020	0.014	0.045	0.039	0.030	0.029	0.033	0.026	0.031
27	Postoperative Patient	0.065	0.065	0.065	0.064	0.067	0.078	0.071	0.056	0.013
28	Primary Tumor	0.210	0.219	0.268	0.215	0.218	0.225	0.238	0.232	0.290
29	Promoter Gene Sequences	0.085	0.085	0.085	0.086	0.177	0.088	0.088	0.084	0.165
30	Segment	0.036	0.026	0.047	0.044	0.043	0.034	0.036	0.037	0.040
31	Sign	0.037	0.041	0.072	0.060	0.045	0.061	0.072	0.051	0.052
32	Sonar Classification	0.092	0.093	0.112	0.092	0.122	0.095	0.106	0.105	0.120
33	Syncon	0.022	0.022	0.067	0.022	0.030	0.022	0.023	0.023	0.081
34	Tic-Tac-Toe Endgame	0.061	0.068	0.145	0.083	0.097	0.094	0.129	0.079	0.099
35	Vehicle	0.126	0.126	0.147	0.134	0.148	0.125	0.120	0.124	0.149
36	Wine Recognition	0.024	0.025	0.034	0.024	0.036	0.024	0.027	0.026	0.093
	Mean	0.063	0.064	0.083	0.069	0.076	0.069	0.074	0.069	0.089

and BSEJ compared with NBTree and FSS is significant. TAN, NBTree and FSS share similar levels of variance.

4.3 Analysis

Bias describes how closely the learner is able to describe the decision surfaces for a domain, while variance reflects the sensitivity of the learner to variations in the training sample. In general, the better the learner is able to fit the training data, the lower the bias. However, closely fitting the training data may result in greater changes in the model formed from sample to sample, and hence higher variance. There is a tension between bias and variance. However, variance is expected to decrease with increasing training sample size, as the differences between the different samples decrease [25]. Therefore, bias may come to dominate error for problems with large training samples.

NB uses a fixed formula to classify, and hence there is no model selection, which results in relatively low variance. Weakening the attribute independence

Table 5. Win/Draw/Loss Records of Error on 36 Datasets

W/D/L p of W/D/L	AODE	LBR	SP-TAN	BSEJ	NBTree	BSE	TAN	NB	FSS
AODE									
LBR	12–6–18								
	0.3616								
SP-TAN	11–3–22	13–7–16							
	0.0802	0.7110							
BSEJ	8–3–25	13–3–20	11–9–16						
	0.0046	0.2962	0.4420						
NBTree	5–5–26	10–1–25	9–3–24	11–3–22					
	0.0002	**0.0166**	**0.0136**	0.0802					
BSE	7–4–25	10–7–19	12–6–18	12–7–17	24–2–10				
	0.0022	0.1360	0.3616	0.4582	**0.0244**				
TAN	8–2–26	12–1–23	13–4–19	13–1–22	15–3–18	15–3–18			
	0.0030	0.0896	0.3770	0.1754	0.7284	0.7284			
NB	8–7–21	11–10–15	11–6–19	15–3–18	21–4–11	13–7–16	17–3–16		
	0.0242	0.5572	0.2004	0.7284	0.1102	0.7110	1.0000		
FSS	5–1–30	6–1–29	9–1–26	7–2–27	7–2–27	8–2–26	7–3–26	11–2–23	
	<0.0001	**0.0002**	**0.0090**	**0.0008**	**0.0008**	**0.0030**	**0.0014**	0.0580	

assumption may make semi-naive Bayesian methods fit the training sample better. Consequently, they may have lower bias, but higher variance compared with NB. AODE reduces variance successfully by aggregating all the qualified 1-dependence classifiers. It delivers competitive variance with NB. NBTree has relatively low bias, but high variance. Brain and Webb [25] hypothesized that the low variance algorithms tend to enjoy lower relative error on small training sets, while low bias algorithms enjoy lower relative error on large training sets. Therefore, the Weka bias-variance estimation method used in this study, which produces small training sets, might put NBTree at a disadvantage. We believe that this also accounts for why AODE was the only algorithm to achieve a significant advantage over NB with respect to error in our experiments, given the low variance of these two algorithms.

As has been discussed above, bias tends to dominate error for large training samples. Therefore, for large training data we recommend use of the lowest bias semi-naive Bayesian method whose complexity satisfies the computational constraints of the application context. For small training data we recommend the lowest variance semi-naive Bayesian method that has suitable computational complexity. For intermediate size training samples, an appropriate trade-off between bias and variance should be sought within the prevailing computational complexity constraints. AODE has very low variance, relatively low bias, and low training time and space complexity. In consequence, it may prove competitive over a considerable range of classification tasks. For extremely small data

Table 6. Win/Draw/Loss Records of Bias on 36 Datasets

W/D/L *p* of W/D/L	NBTree	BSEJ	LBR	AODE	TAN	SP-TAN	BSE	FSS	NB
NBTree									
BSEJ	7–5–24								
	0.0034								
LBR	4–5–27	11–3–22							
	<0.0001	0.0814							
AODE	10–2–24	13–3–20	18–4–14						
	0.0244	0.2962	0.5966						
TAN	12–2–22	19–1–16	21–1–14	21–2–13					
	0.1214	0.7358	0.3106	0.2294					
SP-TAN	5–4–27	7–4–25	15–7–14	16–3–17	14–3–19				
	0.0001	**0.0022**	1.0000	1.0000	0.4868				
BSE	8–2–26	10–8–18	19–3–14	16–4–16	14–2–20	19–5–12			
	0.0030	0.1850	0.4868	1.2734	0.3916	0.2810			
FSS	5–2–29	12–5–19	16–3–17	15–2–19	13–2–21	17–2–17	13–3–20		
	<0.0001	0.2810	1.0000	0.6076	0.1754	1.2734	0.2962		
NB	6–2–28	4–2–30	7–11–18	4–9–23	9–1–26	7–4–25	5–2–29	13–3–20	
	0.0002	**<0.0001**	**0.0432**	**0.0004**	**0.0060**	**0.0022**	**<0.0001**	0.2962	

NB may prove better and for large data NBTree, BSEJ and LBR may have an advantage if their computational profiles are appropriate to the task.

Admittedly these guidelines are imprecise, as the relevant data size is relative to the complexity of the decision surfaces that must be approximated, and in most applications this is unknown. Nonetheless, we believe that they provide a useful framework within which to operate when choosing between semi-naive Bayesian methods.

5 Conclusion

A number of techniques have developed to improve Naive Bayes's accuracy performance by relaxing the attribute independence assumption. We study eight typical semi-naive Bayesian algorithms, and give details of the time and space complexity of these methods. BSEJ, NBTree and SP-TAN have relatively high training time complexity, while LBR has high classification time complexity. BSEJ has very high space complexity. We performed extensive experimental evaluation of the relative error, bias and variance of these algorithms. For the experimental data sets investigated, AODE shares similar levels of error with LBR and SP-TAN, and has a significant advantage over the other algorithms. NBTree has a significant advantage over all the other algorithms, except TAN. All the other algorithms, except TAN and FSS have a significant advantage over NBTree. As bias tends to be a larger portion of error when training set size

Table 7. Win/Draw/Loss Records of Variance on 36 Datasets

W/D/L *p* of W/D/L	NB	AODE	LBR	SP-TAN	BSE	BSEJ	TAN	NBTree	FSS
NB									
AODE	6–15–15								
	0.0784								
LBR	3–13–20	10–8–18							
	0.0004	0.1850							
SP-TAN	6–5–25	7–4–25	11–7–18						
	0.0008	**0.0022**	0.2650						
BSE	7–2–27	6–2–28	13–1–22	11–5–20					
	0.0008	**0.0002**	0.1754	0.1496					
BSEJ	5–4–27	4–2–30	10–2–24	7–5–24	12–6–18				
	0.0001	**<0.0001**	**0.0244**	**0.0034**	0.3636				
TAN	3–3–30	2–4–30	8–4–24	11–1–24	12–2–22	13–5–18			
	<0.0001	**<0.0001**	**0.0070**	**0.0410**	0.1214	0.4732			
NBTree	0–6–30	1–5–30	3–2–31	6–1–29	3–3–30	5–3–28	13–1–22		
	<0.0001	**<0.0001**	**<0.0001**	**0.0001**	**<0.0001**	**<0.0001**	0.1754		
FSS	3–1–32	3–1–32	7–2–27	6–2–28	5–1–30	8–3–25	12–1–23	15–1–20	
	<0.0001	**<0.0001**	**0.0008**	**0.0002**	**<0.0001**	**0.0046**	0.0896	0.4996	

increases, we suggest using low bias methods for large data sets, and low variance methods for small data sets, within the further constraints on applicable algorithms implied by the computational constraints of the given application. Computation cost and the trade-off between bias and variance should be considered for intermediate size data.

References

1. Duda, R.O., Hart, P.E.: Pattern classification and scene analysis. John Wiley and Sons, New York (1973)
2. Kononenko, I.: Comparison of inductive and naive Bayesian learning approaches to autpmatic knowledge acquisition. In Wielinga, B., Boose, J., B.Gaines, Schreiber, G., van Someren, M., eds.: Current Trends in Knowledge Acquisition. Amsterdam: IOS Press (1990)
3. Langley, P., Iba, W., Thompson, K.: An analysis of Bayesian classifiers. In: Proc. 10th Nat. Conf. Artificial Intelligence, AAAI Press and MIT Press (1992) 223–228
4. Langley, P., Sage, S.: Induction of selective Bayesian classifiers. In: Proc. Tenth Conf. Uncertainty in Artificial Intelligence, Morgan Kaufmann (1994) 399–406
5. Kittler, J.: Feature selection and extraction. In Young, T.Y., Fu, K.S., eds.: Handbook of Pattern Recognition and Image Processing. Academic Press, New York (1986)
6. Kononenko, I.: Semi-naive Bayesian classifier. In: Proc. 6th European Working Session on Machine Learning, Berlin: Springer-Verlag (1991) 206–219

7. Langley, P.: Induction of recursive Bayesian classifiers. In: Proc. 1993 European Conf. Machine Learning, Berlin: Springer-Verlag (1993) 153–164
8. Kohavi, R.: Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid. In: Proc. 2nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining. (1996) 202–207
9. Pazzani, M.J.: Constructive induction of Cartesian product attributes. ISIS: Information, Statistics and Induction in Science (1996) 66–77
10. Sahami, M.: Learning limited dependence Bayesian classifiers. In: Proc. 2nd Int. Conf. Knowledge Discovery in Databases, Menlo Park, CA: AAAI Press (1996) 334–338
11. Singh, M., Provan, G.M.: Efficient learning of selective Bayesian network classifiers. In: Proc. 13th Int. Conf. Machine Learning, Morgan Kaufmann (1996) 453–461
12. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning **29** (1997) 131–163
13. Webb, G.I., Pazzani, M.J.: Adjusted probability naive Bayesian induction. In: Proc. 11th Australian Joint Conf. Artificial Intelligence, Berlin:Springer (1998) 285–295
14. Keogh, E.J., Pazzani, M.J.: Learning augmented Bayesian classifers: A comparison of distribution-based and classification-based approaches. In: Proc. Int. Workshop on Artificial Intelligence and Statistics. (1999) 225–230
15. Zheng, Z., Webb, G.I., Ting, K.M.: Lazy Bayesian rules: A lazy semi-naive Bayesian learning technique competitive to boosting decision trees. In: Proc. Sixteenth Int. Conf. Machine Learning (ICML 1999), Morgan Kaufmann (1999) 493–502
16. Zheng, Z., Webb, G.I.: Lazy learning of Bayesian rules. Machine Learning **41** (2000) 53–84
17. Webb, G.I.: Candidate elimination criteria for lazy Bayesian rules. In: Proc. Fourteenth Australian Joint Conf. Artificial Intelligence. Volume 2256., Berlin:Springer (2001) 545–556
18. Xie, Z., Hsu, W., Liu, Z., Lee, M.L.: Snnb: A selective neighborhood based naive Bayes for lazy learning. In: Advances in Knowledge Discovery and Data Mining, Proc. Pacific-Asia Conference (PAKDD 2002), Berlin:Springer (2002) 104–114
19. Webb, G.I., Boughton, J., Wang, Z.: Not so naive Bayes: Aggregating one-dependence estimators. Machine Learning **58** (2005) 5–24
20. Domingos, P., Pazzani, M.J.: Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In: Proc. 13th Int. Conf. Machine Learning, Morgan Kaufmann (1996) 105–112
21. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998) [http://www.ics.uci.edu/∼mlearn/MLRepository.html]. Irvine, CA: University of California, Dept. Information and Computer Science.
22. Kohavi, R., Wolpert, D.: Bias plus variance decomposition for zero-one loss functions. In: Proc. 13th Int. Conf. Machine Learning, San Francisco: Morgan Kaufmann (1996) 275–283
23. Witten, I.H., Frank, E.: Data mining : practical machine learning tools and techniques with Java implementations. San Francisco, CA: Morgan Kaufmann (2000)
24. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Proc. 13th Int. Joint Conf. Artificial Intelligence (IJCAI-93), Morgan Kaufmann (1993) 1022–1029
25. Brain, D., Webb, G.I.: The need for low bias algorithms in classification learning from large data sets. In: Proc. 16th European Conf. Principles of Data Mining and Knowledge Discovery (PKDD2002), Berlin:Springer-Verlag (2002) 62–73

A Statistically Sound Alternative Approach to Mining Contrast Sets

Robert J. Hilderman and Terry Peckham

Department of Computer Science
University of Regina
Regina, Saskatchewan, Canada S4S 0A2
{hilder, peckham}@cs.uregina.ca

Abstract. One of the fundamental tasks of data analysis in many disciplines is to identify the significant differences between classes or groups. Contrast sets have previously been proposed as a useful tool for describing these differences. A contrast set is a conjunction of (association rule-like) attribute-value pairs for which the conjunction is true for some group. The intuition is that comparing the support for a contrast set across groups may provide some insight into the fundamental differences between the groups. In this paper, we compare two contrast set mining methods that rely on different statistical philosophies: the well-known STUCCO approach, and CIGAR, our proposed alternative approach. We survey and discuss the statistical measures underlying the two methods using an informal tutorial approach. Experimental results show that both methodologies are statistically sound, representing valid alternative solutions to the problem of identifying potentially interesting contrast sets.

1 Introduction

One of the fundamental tasks of data analysis in many disciplines is to identify the significant differences between classes or groups. For example, an epidemiological study of self-reported levels of stress experienced by health care providers could be used to characterize the differences between those who work in rural and urban communities. The differences could be conveniently described using pairs of contrasting conditional probabilities, such as `P(Stress=high` $\wedge$ `Income=low | Location=rural) = 32%` and `P(Stress=high` $\wedge$ `Income=low | Location= urban) = 25%`. The conditional probabilities shown here are equivalent to rules of the form `Location=rural` $\Rightarrow$ `Stress=high` $\wedge$ `Income=low (32%)` and `Location = urban` $\Rightarrow$ `Stress=high` $\wedge$ `Income=low (25%)`, known as *association rules* [1], where the *antecedents* (i.e., `Location=rural` and `Location=urban`) describe distinct groups that share a common *consequent* (i.e., `Stress=high` $\wedge$ `Income=low`), and the percentages represent the number of examples in each group for which the association rule is true (called *support*). The common consequent is called a *contrast set* [2].

Contrast set mining is an association rule-based discovery technique that was originally introduced as emerging pattern mining [4], a temporal pattern mining problem that is essentially a special case of the more general contrast set mining problem. An excellent bibliography can be found in [6].

More generally applicable work in contrast set mining can be found in [2], [3], and [8]. In [2], contrast set mining is studied within the context of an association rule-based technique called STUCCO (Searching and Testing for Understandable Consistent COontrasts). For an extensive description and evaluation, see [3]. The fundamental characteristic of this approach is that it utilizes a canonical ordering of nodes in the search space, such that any node that cannot be pruned is visited only once. STUCCO also utilizes χ^2 testing of two-dimensional contingency tables, along with a modified Bonferroni method to control Type I error, to determine whether differences between rules in a contrast set are statistically significant.

Group differences are also studied in [8] within the context of an association rule-like technique called Magnum Opus, a commercial exploratory rule discovery tool. However, the statistical reasoning used by Magnum Opus actually performs a within-groups comparison rather than a between-groups comparison [7], finding only a subset of the contrast sets generated by STUCCO, so we do not discuss it further in this work.

Here, we discuss STUCCO in detail, and introduce CIGAR (ContrastIng, Grouped Association Rules), a contrast set mining technique that relies on an alternative statistical philosophy to the discovery of statistically significant contrast sets, yet still adheres to sound and accepted practices. CIGAR not only considers whether the difference in support between two groups is significant, it also considers whether the attributes in a contrast set are correlated, and a correlational pruning technique is utilized to reduce the size of the search space.

2 The Contrast Set Mining Framework

In this section, we describe how the contrast set mining problem generalizes the association rule mining problem from binomial or transactional data types to multinomial, grouped categorical data.

2.1 The Association Rule Mining Problem

Mining contrast sets is based upon the problem of mining association rules [1]. The problem of association rule mining is typically studied within the context of discovering buying patterns from retail sales transactions (i.e., market basket data), and is formally defined as follows. Let $A = \{\mathtt{A}_1, \mathtt{A}_2, \ldots, \mathtt{A}_\mathtt{m}\}$ be a set of attributes called *items*. Let D be a set of *transactions*, where each transaction T is described by a vector of m attribute-value pairs $\mathtt{A}_1 = \mathtt{V}_1, \mathtt{A}_2 = \mathtt{V}_2, \ldots, \mathtt{A}_\mathtt{m} = \mathtt{V}_\mathtt{m}$, and each $\mathtt{V}_\mathtt{j}$ is selected from the set $\{1, 0\}$ (i.e., $\mathtt{V}_\mathtt{j} = 1$ ($\mathtt{V}_\mathtt{j} = 0$) indicates that item $\mathtt{A}_\mathtt{j}$ was purchased (not purchased)). The collection of purchased items contained in transaction T is an *itemset*. Transaction T *contains* $\mathtt{X}$, a set of purchased items, if $\mathtt{X} \subseteq T$. An *association rule* is an implication of the form $\mathtt{X} \Rightarrow \mathtt{Y}$, where $\mathtt{X} \subset A$, $\mathtt{Y} \subset A$, and $\mathtt{X} \cap \mathtt{Y} = \emptyset$. *Confidence* c in $\mathtt{X} \Rightarrow \mathtt{Y}$ is the percentage of transactions in D containing $\mathtt{X}$ that also contain $\mathtt{Y}$. *Support s* for $\mathtt{X} \Rightarrow \mathtt{Y}$ is the percentage of transactions in D containing $\mathtt{X} \cup \mathtt{Y}$. The confidence in an association rule $\mathtt{X} \Rightarrow \mathtt{Y}$ measures the conditional probability of $\mathtt{Y}$ given $\mathtt{X}$, denoted $\mathtt{P(Y|X)}$. The goal of association rule mining is to identify rules whose support and confidence exceed some user-defined thresholds.

2.2 The Contrast Set Mining Problem

In the problem of contrast set mining [2], [3] transaction set D is generalized to a set of multinomial *examples*, where each example E is described by a vector of m attribute-value pairs $\mathtt{A_1} = \mathtt{V_1}, \mathtt{A_2} = \mathtt{V_2}, \ldots, \mathtt{A_m} = \mathtt{V_m}$, and each $\mathtt{V_i}$ is selected from the finite set of discrete domain values in the set $\{\mathtt{V_{i_1}}, \mathtt{V_{i_2}}, \ldots, \mathtt{V_{i_n}}\}$ associated with $\mathtt{A_i}$. One attribute $\mathtt{A_j}$ in D is a distinguished attribute whose value $\mathtt{V_{j_k}}$ in example E is used to assign E into one of $\mathtt{n}$ mutually exclusive groups $G_1, G_2, \ldots, G_{\mathtt{n}}$. A *contrast set* $\mathtt{X}$ is a conjunction of attribute-value pairs defined on $G_1, G_2, \ldots, G_{\mathtt{n}}$, such that no $\mathtt{A_i}$ occurs more than once. From these conjunctions, we have rules of the form $\mathtt{A_j} = \mathtt{V_{j_k}} \Rightarrow \mathtt{X}$, where the antecedent contains the distinguished attribute and determines group membership, and the consequent describes a contrast set. *Support s* for association rule $\mathtt{A_j} = \mathtt{V_{j_k}} \Rightarrow \mathtt{X}$ is the percentage of examples in $G_{\mathtt{j}}$ containing $\mathtt{X}$. The goal of contrast set mining is to identify all contrast sets for which the support is significantly different across groups.

3 STUCCO

The objective of STUCCO is to find contrast sets from grouped categorical data, where a dataset D can be divided into n mutually exclusive groups, such that for groups G_i and G_j, $G_i \cap G_j = \emptyset$, for all $i \neq j$. Specifically, we want to identify all contrast sets, such that the conditions

$$\exists ij \mathtt{P}(\mathtt{X}|G_i) \neq \mathtt{P}(\mathtt{X}|G_j)$$

and

$$max_{ij}|\mathtt{support}(\mathtt{X},G_i) - \mathtt{support}(\mathtt{X},G_j)| \geq \delta$$

are satisfied, where $\mathtt{X}$ is a contrast set, G_k is a group, and δ is the user-defined minimum support difference (i.e., the minimum support difference between two groups). Contrast sets satisfying the first condition are called *significant*, those satisfying the second condition are called *large*, and those satisfying both conditions are called *deviations.*

3.1 Finding Deviations

The search space consists of a canonical ordering of nodes, where all possible combinations of attribute-value pairs are enumerated. The rule contained at each node is called a *candidate set* until it has been determined that it meets the criteria required to be called a contrast set (i.e., it is significant and large).

Determining Support for a Candidate Set. The search for contrast sets follows a breadth-first search strategy, and is based upon support for a candidate set. For example, a sample contingency table is shown in Table 1. In Table 1, `Support(Location=urban` $\wedge$ `Stress= high)` $= 194 / 554 = 0.35$ (or 35%) and `Support(Location=rural` $\wedge$ `Stress= high)` $= 355 / 866 = 0.41$ (or 41%).

Table 1. An example contingency table

	`Location=urban`	`Location=rural`	$\sum$ *Row*
`Stress=high`	194	355	549
$\neg$ `(Stress=high)`	360	511	871
$\sum$ *Column*	554	866	1420

Determining Whether a Candidate Set is Large. As mentioned above, two rules whose support difference exceeds some user-defined threshold are called large rules. For example, in Table 1, `|Support(Location=urban` $\wedge$ `Stress=high ) - Support(Location=rural` $\wedge$ `Stress=high)|` = |0.35 - 0.41| = 0.06 (or 6%). If we assume that $\delta = 0.05$ (or 5%), then the rules `Location=urban` $\wedge$ `Stress=high` and `Location=rural` $\wedge$ `Stress=high` are large.

Determining Whether a Candidate Set is Significant. To determine whether support for rules are significantly different across groups, two-dimensional contingency table analysis and the χ^2 statistic are used. A $2 \times n$ contingency table is constructed, where the rows represent the truth of the contrast set and the columns represent the groups. The χ^2 statistic tests the null hypothesis that row and column totals are not related (i.e., are *independent*), and is given by

$$\chi^2 = \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{(O_{ij} - E_{ij})^2}{E_{ij}},$$

where O_{ij} is the observed frequency at the intersection of row i and column j, $E_{ij} = (O_{i.} \times O_{.j})/O_{..}$, $O_{i.}$ is the total in row i, $O_{.j}$ is the total in column j, and $O_{..}$ is the sum of the row and column totals (i.e., the total number of examples). The number of *degrees of freedom* for χ^2 is given by $df = (r-1) \times (c-1)$, where r and c are the number of rows and columns, respectively. A sufficiently large χ^2 value will cause rejection of the null hypothesis that row and column totals are not related. For example, $\chi^2 = 5.08$ for the contingency table in Table 1. At the 5% significance level (i.e., $\alpha = 0.05$), $\chi^2 = 3.84$. Since $5.08 > 3.84$, we reject the null hypothesis. That is, we conclude that `Location` and `Stress` in the rules `Location=urban` $\wedge$ `Stress=high` and `Location=rural` $\wedge$ `Stress=high` are dependent.

However, we have failed to consider the effects of multiple hypothesis tests. The α level is used to control the maximum probability of falsely rejecting the null hypothesis in a single χ^2 test (i.e., known as a Type I error or a false positive error in statistical and non-statistical parlance, respectively). In the above example, we used $\alpha = 0.05$. But since STUCCO performs multiple hypothesis tests, a modified Bonferroni statistic is employed to limit the total Type I error rate for all χ^2 tests to α. The modified Bonferroni statistic uses a different α for contrast sets being tested at different levels of the search space. That is, at level i in the search space,

$$\alpha_i = min((\alpha/2^i)/|C_i|, \alpha_{i-1}),$$

where $|C_i|$ is the number of candidates at level i. The net effect, then, is that as we descend through the search space, α_i is half that of α_{i-1}, so a significant difference is increasingly restrictive as we descend. In the case of the contingency table in Table 1, the rules being tested are found at level two of the search space. If we assume there are 10 nodes at level two, then $i = 2$ and $\alpha_2 = ((0.05/2^2)/|10|) = 0.00125$. For $\alpha = 0.00125$, $\chi^2 \approx 10.83$. Since $5.08 < 10.83$, we accept the null hypothesis. That is, we conclude that `Location` and `Stress` are independent (i.e., no relationship exists between the two attributes). And since the rules are not significantly different, they are not significant. Finally, since the rules are not both large and significant, they are not deviations, and therefore, do not constitute a contrast set.

3.2 Pruning the Search Space

Conceptually, the basic pruning strategy is simple: a node in the search space can be pruned whenever it fails to be significant and large.

Effect Size Pruning. When the maximum support difference, δ_{max}, between all possible pairs of groups has been considered and $\delta_{max} < \delta$, then the corresponding nodes can be pruned from the search space. This ensures that the effect size is large enough to be considered important by the domain expert.

Statistical Significance Pruning. The accuracy of the χ^2 test depends on the expected frequencies in the contingency table. When an expected frequency is too small, the validity of the χ^2 test may be questioned. However, there is no universal agreement on what is appropriate, so frequencies ranging anywhere from 1 (liberal) to 5 (conservative) are considered acceptable. Thus, nodes are pruned whenever an expected frequency is considered unacceptable.

Maximum χ^2 Pruning. As we descend through the search space, the number of attribute-value pairs in a contrast set increases (i.e., itemsets are larger as the rules become more specific), and at each successive lower level in the search space, the support for a contrast set at level i is bounded by the parent at level $i - 1$. For example, given the rule `Location=rural` $\Rightarrow$ `Stress=high (54%)`, then the rule `Location=rural` $\Rightarrow$ `Stress=high` $\wedge$ `Income=low (65%)` cannot possibly be true. That is, any specialization of this rule cannot be any more than 54%. Consequently, the support for the parent rule `Stress=high` becomes an upper bound for all descendants in the search space. Similarly, as we ascend through the search space, the support for a contrast set at level i is bounded by the child at level $i+1$. That is, the support for the child rule becomes a lower bound for all ancestors in the search space. Within the context of a contingency table, the observed frequencies in the upper (lower) row decrease (increase) as the contrast set becomes more specialized.

Since the support is bounded, the maximum possible χ^2 value for all specializations at the next level can be determined and used to prune the specialization if it cannot meet the χ^2 cutoff for α at that level. Let u_i and l_i represent the upper and lower bounds, respectively, of the observed values in position i of row one across all specializations. For example, if we have three specializations, say, and the observed values at position $i = 2$ of the three specializations are 4, 2, and 5, then $u_2 = 5$ and $l_2 = 2$. The maximum χ^2 value possible for any specialization of a rule is given by

$$\chi^2_{max} = max_{o_i \in \{u_i, l_i\}} \chi^2(o_1, o_2, \ldots, o_n),$$

where $\chi^2(o_1, o_2, \ldots, o_n)$ is the value for a contingency table with $\{o_1, o_2, \ldots, o_n\}$ as the observed values in the first row. The rows that we use to determine the maximum χ^2 value are based upon the n upper and lower bounds from the specializations. For example, if the first rows of our three specializations are {5, 4, 9}, {3, 2, 10}, and {8, 5, 6}, then the upper and lower bounds are {8, 5, 10} and {3, 2, 6}, respectively. We generate all 2^n possible first rows from combinations of the values in the upper and lower bounds. For example, from the upper and lower bounds given previously, the 2^3 unique first rows that we can generate are {8, 5, 10}, {3, 5, 10}, {8, 2, 10}, {3, 2, 10}, {8, 5, 6}, {3, 5, 6}, {8, 2, 6}, and {3, 2, 6}. These rows actually correspond to the extreme points (i.e., corners) of a feasible region where the maximum χ^2 value can be found. Since the values in the second row of each contigency table are determined by the values in the first row (since the column totals are fixed), then each contingency table is unique. For example, if the column totals are {15, 7, 13}, then the second row corresponding to {8, 5, 10} is {7, 2, 3}. We then simply determine the χ^2 value for each of the generated contingency tables and take the maximum. If χ^2_{max} exceeds the α cutoff, then none of the specializations can be pruned.

Interest Based Pruning. Specializations with support identical to the parent are not considered interesting by STUCCO. Similarly, when the support for one group is much higher than other groups, it will sometimes remain much higher regardless of the nature of any additional attribute-value pairs that are added to the rule. Specializations of the rule are pruned from the search space.

Statistical Surprise Pruning. When the observed frequencies are statististically different from expected frequencies (i.e., statistically surprising), a contrast set is considered interesting. For cases involving two variables, the expected frequency can be determined by multiplying the respective observed frequencies.

For example, if `P(Stress=high | Location=rural) = 40%` and `P(Income=low | Location=rural) = 65%`, then `P(Stress=high` $\wedge$ `Income=low | Location =rural) = 26%`. If the product is within some threshold range, the contrast set is considered uninteresting and pruned from the search space. For more complicated cases (i.e., more than two variables), iterative proportional fitting can be used [5].

4 CIGAR: A Statistically Sound Alternative

Whereas STUCCO answers the question whether a difference exists between contrast sets in two or more groups through the analysis of $2 \times n$ contingency tables, CIGAR seeks a more fine grained approach by breaking the $2 \times n$ contingency tables down into a series of 2×2 contingency tables to try to explain where these differences actually occur. So, while we still want to identify all contrast sets such that the conditions

$$\exists ij \texttt{P}(\texttt{X}|G_i) \neq \texttt{P}(\texttt{X}|G_j)$$

and

$$max_{ij}|\texttt{support}(\texttt{X},G_i) - \texttt{support}(\texttt{X},G_j)| \geq \delta$$

used by STUCCO are satisfied (i.e., to find the significant and large contrast sets), CIGAR also utilizes three additional constraints. That is, we also want to identify all contrast sets such that the conditions

$$\texttt{support}(\texttt{X},G_i) \geq \beta,$$

$$\texttt{correlation}(\texttt{X},G_i) \geq \lambda,$$

and

$$|\texttt{correlation}(\texttt{X},G_i) - \texttt{correlation}(\texttt{child}(\texttt{X},G_i))| \geq \gamma$$

are satisfied, where `X` is a contrast set, G_k is a group, β is the user-defined minimum support threshold, λ is the user-defined minimum correlation threshold, and γ is the user-defined minimum correlation difference. Contrast sets satisfying the third condition are called *frequent*. We believe a support threshold can aid in identifying outliers. Since outliers can dramatically affect the correlation value, the minimum support threshold provides an effective tool for removing them. Contrast sets satisfying the fourth condition are called *strong*. This measures the strength of any linear relationship between the contrast set and group membership. Contrast sets satisfying the first four conditions are called *deviations*. Those deviations that fail to satisfy the last condition are called *spurious* and pruned from the search space.

4.1 Finding Deviations

With CIGAR, before a candidate set can become a contrast set it must meet more restrictive criteria than STUCCO. That is, it must not only be significant and large, it must also be frequent and strong.

Determining Support for a Candidate Set. CIGAR determines support for a candidate set in the same way as STUCCO, but CIGAR also utilizes a minimum support threshold. This threshold is useful for two reasons. First, the domain expert may not be interested in low support rules. Consequently, the nodes for these rules and all the descendant nodes can be pruned from the search space. Second, if the rule support is 0% or 100%, then the rule is pruned since a conjunction of this rule with any other does not create any new information.

Determining Whether a Candidate Set is Large and/or Significant. CIGAR determines whether a candidate set is large and/or significant in the same way as STUCCO.

Determining Whether a Candidate Set is Correlated. In CIGAR, correlation is calculated using the Phi correlation coefficient. The Phi correlation coefficient is a measure of the degree of association between two dichotomous variables, such as those contained in a 2×2 contingency table, and is conveniently expressed in terms of the observed frequencies. For example, given the generic 2×2 contingency table shown in Table 2, the Phi correlation coefficient is given by

$$\phi = \frac{O_{11}O_{22} - O_{12}O_{21}}{\sqrt{(O_{11}+O_{21})(O_{12}+O_{22})(O_{11}+O_{12})(O_{21}+O_{22})}}.$$

Table 2. A generic contingency table

	G_1	G_2	$\sum$ *Row*
`Contrast Set`	O_{11}	O_{12}	$O_{11} + O_{12}$
`¬ (Contrast Set)`	O_{21}	O_{22}	$O_{21} + O_{22}$
$\sum$ *Column*	$O_{11} + O_{12}$	$O_{12} + O_{22}$	$O_{11} + O_{12} + O_{21} + O_{22}$

The Phi correlation coefficient compares the diagonal cells (i.e., O_{11} and O_{22}) to the off-diagonal cells (i.e., O_{21} and O_{12}). The variables are considered positively associated if the data is concentrated along the diagonal, and negatively associated if the data is concentrated off the diagonal. To represent this association, the denominator ensures that the Phi correlation coefficient takes values between 1 and -1, where zero represents no relationship. However, the calculation for the Phi correlation coefficient can be expressed in terms of the χ^2 value (which we have to calculate anyway), and is given by

$$r = \sqrt{\chi^2/N},$$

where $N = O_{11} + O_{12} + O_{21} + O_{22}$. So, for the example in Section 3.1, and using $\chi^2 = 5.08$ and $\alpha = 0.05$, we have $r = \sqrt{5.08/1420} = 0.06$. Now a general rule of thumb is that $0.0 \leq r \leq 0.29$ represents little or no association, $0.3 \leq r \leq 0.69$ represents a weak positive association, and $0.7 \leq r \leq 1.0$ represents a strong positive association. Consequently, although we have previously determined that a significant relationship exists between `Location` and `Stress` (i.e., prior to considering the effects of multiple hypothesis tests), at $r = 0.06$, this relationship is very weak.

4.2 Pruning the Search Space

CIGAR provides a powerful alternative strategy for reducing the number of results that must be considered by a domain expert. Conceptually, the basic pruning strategy is that a node in the search space is pruned whenever it fails to be significant, large, frequent, and strong.

Look-Ahead χ^2 Pruning. The χ^2 look-ahead approach calculates the χ^2 value for each specialization of a rule. If no specialization is found to be significant, then all the specializations are pruned from the search space. If at least one specialization is found to be significant, all the specializations are considered candidate sets at the next level of the search tree.

Statistical Significance Pruning. As mentioned in Section 3.2, the validity of the χ^2 test may be questioned when the the expected frequencies are too small. To address this problem, Yates' correction for continuity has been suggested. Although there is no universal agreement on whether this adjustment should be used at all, there does seem to be some consensus that indicates the correction for continuity should be applied to all 2×2 contingency tables and/or when at least one expected frequency is less than five (liberal) or 10 (conservative). Either way, Yates' correction provides a more conservative estimate of the χ^2 value that is, hopefully, a more accurate estimate of the significance level. Yates' correction for continuity is given by

$$\chi^2 = \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{(|O_{ij} - E_{ij}| - 0.5)^2}{E_{ij}}.$$

For example, Yates' $\chi^2 = 4.84$ for the contingency table in Table 1.

Minimum Support Pruning. The minimum support threshold utilized by CIGAR is the first line pruning strategy. For example, when determining correlation between `Location` and `Stress`, if one group happens to have very low support, it will likely affect the correlation for all pairwise group comparisons. Consequently, a domain expert may decide to exclude the low support group.

Minimum Correlation Pruning. The Phi correlation coefficient provides a basis for determining whether a contrast set is worth further consideration. For example, the lower the correlation, the higher the likelihood that no relationship actually exists between a rule and the group. That is, even if a rule is considered significant, if the correlation is zero, then the probability that the rules is simply a statistical artifact is high. Therefore, the removal of rules that do not meet the minimum correlation criteria eliminates the likelihood of reporting statistical artifacts. For example, we determined in the previous section that the relationship between `Location` and `Stress` is weak at $r = 0.06$ and the rule should be removed from further consideration.

When a high minimum correlation threshold is used, many significant rules may be pruned, resulting in an increase in Type II error. Similarly, when a low minimum correlation threshold is used, many spurious rules may not be pruned. This is analogous to the problem of setting support thresholds in the classic association rule mining problem.

CIGAR is different from STUCCO in that it tends to report more specialized contrast sets rather than generalized contrast sets. The assumption behind approaches that report more generalized rules is that more general rules are better for prediction. However, the complex relationships between groups can often be better explained with more specialized rules.

Minimum Correlation Difference Pruning. CIGAR calculates the difference between the correlation of a rule and the correlations of specializations of that rule. If the difference in correlation between a rule and a specialization is less than the minimum correlation difference threshold, the specialization is pruned from the search space. That is, if the addition of a new attribute-value pair to a contrast set does not add any new information that directly speaks to the strength of the relationship, then the contrast set is spurious. For example, assume that $r = 0.70$ for the rule `Location=rural` $\wedge$ `Income=low`. If $\gamma = 0.05$, and $r = 0.67$ for the specialization `Location=rural` $\wedge$ `Income=low` $\wedge$ `Stress=high`, then the specialization is pruned from the search space because $|0.70 - 0.67| = 0.03$ and $\gamma > 0.03$.

Generally, as we descend through the search space, the support for rules at lower levels decreases. As a result, the χ^2 value and r generally decrease, as well. The decision on whether to prune a rule from the search space is then a fairly easy one, as the previous example showed (i.e., it failed to exceed the minimum correlation difference threshold). However, it is possible that as we descend through the search space the χ^2 value and/or r can increase. It is also possible the χ^2 value and/or r can decrease and then increase again. If the correlation difference between a rule and one of its specializations is less than the minimum correlation difference, regardless of whether the difference represents a decrease or an increase, the domain expert has to be careful when deciding whether to prune the specialization. That is, pruning a specialization that fails to meet the minimum correlation difference criteria at level i could result in the loss of a specialization at level $i + 1$ that does meet the minimum correlation difference criteria. So, some statistical judgment may be required on the part of the domain expert to ensure that only unproductive and spurious rules can be pruned.

5 Experimental Results

In this section, we present the results of our experimental evaluation and comparison of STUCCO and CIGAR. STUCCO was supplied by the original authors [2], [3]. STUCCO, implemented in C++ and compiled using gcc (version 2.7.2.1), was run on a Sun Microsystems Enterprise 250 Model 1400 with two UltraSparc-II 400 MHz processors and 1 GB of memory. CIGAR was implemented by the authors of this paper in Java 1.4.1 and was run under Windows XP on an IBM compatible PC with a 2.4 GHz AMD Athlon processor and 1 GB of memory. The performance of the two software tools was compared by generating contrast sets from publicly available datasets.

5.1 The Datasets

Discovery tasks were run on three datasets: Mushroom, GSS Social, and Adult Census. The Mushroom dataset, available from the UCI Machine Learning Repository (`www.ics.uci.edu/ mlearn/MLRepository.html`), describes characteristics of gilled mushrooms. The GSS Social dataset is a survey dataset from Statistics Canada that contains the responses to the General Social Survey of Canada (1986 - Cycle 2): Social Activities and Language Use. The Adult Census dataset is a subset of the Adult Census Data: Census Income (1994/1995) dataset, a survey dataset from the U.S. Census Bureau.

The characteristics of the three datasets are shown in Table 3. In Table 3, the *Tuples* column describes the number of tuples in the dataset, the *Attributes* column describes the number of attributes, the *Values* column describes the number of unique values contained in the attributes, and the *Groups* column describes the number of distinct groups defined by the number of unique values in the grouping attribute.

Table 3. Characteristics of the Four Datasets

Dataset	*Tuples*	*Attributes*	*Values*	*Groups*
Mushroom	8,142	23	130	2
GSS Social	179,148	16	2,026	7
Adult Census	826	13	129	2

5.2 The Effect of Error Control

STUCCO seeks to control Type I (or false positive) error, whereas CIGAR seeks to control Type II (or false negative) error. In this section, we compare the error control philosophies of STUCCO and CIGAR to evaluate the impact on the number of candidate sets and contrast sets generated.

The number of candidate sets generated from the Mushroom, GSS Social, and Adult Census datasets is shown in Table 4. Table 4 shows for the Mushroom, GSS Social, and Adult Census datasets that CIGAR generated approximately 9.1, 1.3, and 2.8 times more candidate sets, respectively, than STUCCO. For example, for the Mushroom dataset, CIGAR generated 128,717 candidate sets containing up to 13-itemsets, while STUCCO generated 14,089 candidate sets containing up to 8-itemsets.

Table 4. Summary of Candidate Sets Generated

	Mushroom		*GSS Social*		*Adult Census*	
k-Itemsets	*STUCCO*	*CIGAR*	*STUCCO*	*CIGAR*	*STUCCO*	*CIGAR*
1	103	53	11,965	3,009	97	44
2	951	694	13,994	5,980	877	419
3	3,470	3,912	6,670	8,620	2,011	1,680
4	6,025	10,496	4,897	13,168	3,033	3,545
5	3,054	21,006	792	10,298	826	4,806
6	485	28,427	117	5,356	36	4,357
7	1	27,995	6	1,524	0	2,755
8	0	20,189	0	236	0	1,184
9	0	10,545	0	20	0	342
10	0	3,870	0	9	0	60
11	0	939	0	0	0	5
12	0	133	0	0	0	0
13	0	8	0	0	0	0
14	0	0	0	0	0	0
Total	14,089	128,717	38,411	48,220	6,880	19,197

Up to the 2-itemset level, STUCCO generates more candidate sets than CIGAR. The primary reason for this is that when STUCCO is determining whether a candidate set is large, it includes groups for which the support is zero. CIGAR uses the minimum support threshold to remove contrast sets with low support from further consideration. In addition, at the 3-itemset level, the significance level calculated by the modified Bonferroni statistic used in STUCCO starts to become more restrictive than the significance level used in CIGAR.

5.3 The Effect of 2 × 2 Contingency Tables

The number of contrast sets generated from the Mushroom, GSS Social, and Adult Census datasets by STUCCO and CIGAR is shown in Table 5. Table 5 shows that CIGAR generated significantly more contrast sets than STUCCO. For the datasets that contain only two groups (i.e., Mushroom and Adult Census), the number of contrast sets generated is somewhat similar until the modified Bonferroni statistic becomes more restrictive at the 4-itemset level. Essentially, the number of groups contained in a dataset affect the number and size of the contingency tables used. For example, the Mushroom and Adult Census datasets contain two groups, so both STUCCO and CIGAR use 2×2 contingency tables. But the GSS Social dataset contains seven groups. In this case, STUCCO uses a 2×7 contingency table, while CIGAR uses a series of 2×2 contingency tables, one for each possible combination of group pairs.

Table 5. Summary of Contrast Sets Generated

	Mushroom		*GSS Social*		*Adult Census*	
k-Itemsets	*STUCCO*	*CIGAR*	*STUCCO*	*CIGAR*	*STUCCO*	*CIGAR*
1	71	46	83	566	22	23
2	686	548	466	3,081	139	202
3	2,236	2,721	1,292	7,645	353	843
4	2,531	7,577	1,155	10,930	341	1,972
5	714	13,899	199	9,368	64	2,929
6	102	18,293	22	4,852	0	2,920
7	0	17,915	0	1,504	0	2,011
8	0	13,124	0	249	0	943
9	0	7,077	0	20	0	286
10	0	2,715	0	11	0	53
11	0	697	0	0	0	5
12	0	106	0	0	0	0
13	0	7	0	0	0	0
14	0	0	0	0	0	0
Total	6,340	84,725	3,217	38,226	919	12,187

The 2×2 contingency tables used by CIGAR have the potential to provide more information about the differences between groups than the 2×7 contingency table used by STUCCO. For example, a 2×7 contingency table for the contrast set `Activity Code = everyday shopping` generated by STUCCO for the seven groups in the GSS Social dataset is shown in Table 6. The χ^2 value and degrees of freedom calculated for this table are $\chi^2 = 386.38$ and $df = 6$, respectively. From this information, STUCCO reports that a significant difference exists between groups and generates the rule `All Groups` $\Rightarrow$ `Activity Code = everyday shopping`. But other than pointing out that the relationship between the contrast set `Activity Code = everyday shopping` and group is not randomly causal, it does not provide any details as to where the differences actually occur. That is, it does not provide any details as to which groups are different.

In contrast, CIGAR is able to provide details as to which groups are different. For example, CIGAR generates a series of 21 2×2 contingency tables, one for each possible combination of group pairs. From these contingency tables, CIGAR determines that for the contrast set `Activity Code = everyday shopping`, there are significant differences between G_2 and G_6, G_2 and G_7, G_3 and G_6, G_3 and G_7, and G_4 and G_7. The other sixteen combinations of group pairs failed to meet the minimum support and minimum support difference thresholds. Consequently, not only do we know that a significant difference exists between some groups, we have a fine grained breakdown of the groups involved.

Table 6. Contingency Table for `Activity Code = everyday shopping`

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	$\sum$ *Row*
`Activity Code = everyday shopping`	164	555	558	650	481	619	718	3,745
$\neg$ `(Activity Code = everyday shopping)`	17,278	32,655	31,627	31,815	20,078	20,685	21,264	175,402
$\sum$ *Column*	17,442	33,210	32,185	32,465	20,559	21,304	21,982	179,147

5.4 The Effect of a Minimum Support Threshold

Recall that one of the constraints utilized by CIGAR in contrast set mining, and not utilized by STUCCO, is a minimum support threshold. To aid in making this discussion clear, we discuss the 1-itemset results generated by STUCCO and CIGAR for the Mushroom and Adult Census datasets. These results are shown in Table 7. In Table 7, the *Zero Itemsets* row describes the number of contrast sets that were generated where at least one of the groups had zero support. The *Below Minimum Support* row describes the number of contrast sets where at least one of the groups had support below the minimum support threshold. The *Unmatched Contrast Sets* row describes the number of contrast sets that are found by STUCCO (CIGAR) but not by CIGAR (STUCCO). The *Matched Contrast Sets* row describes the number of contrast sets found.

Table 7. Summary of 1-itemset Results

	Mushroom		*Adult Census*	
	STUCCO	*CIGAR*	*STUCCO*	*CIGAR*
Zero Itemsets	15	0	2	0
Below Minimum Support	10	0	1	0
Unmatched Contrast Sets	0	0	0	4
Matched Contrast Sets	46	46	19	19

Table 7 shows that for the Mushroom and Adult Census datasets, STUCCO generates 25 and 3 contrast sets, respectively, whose support is below the minimum support threshold. These contrast sets represent 35% and 14%, respectively, of the total number of contrast sets generated. On the Mushroom dataset, this represents 100% of the difference between the contrast sets generated by STUCCO and CIGAR. On the Adult Census dataset, four (or 17%) of the contrast sets generated by CIGAR did not have a corresponding contrast set in those generated by STUCCO. These four contrast sets were pruned by STUCCO because they did not meet the significance level cutoff of the modified Bonferroni statistic.

5.5 The Effect of Correlational Pruning

The minimum correlation threshold utilized by CIGAR can significantly reduce the number of contrast sets that need to be considered by a domain expert by focusing attention on only those contrast sets where the relationship between variables is strong. The number of contrast sets generated by CIGAR from the Mushroom dataset is shown in Table 8. In Table 8, the k-Itemset column is as previously described. The *No Prune* and *Prune* columns describe the number of contrast sets generated without and with correlational pruning, respectively, for each of the specified minimum correlation threshold values (i.e., $r = 0.00$ to $r = 0.70$). The minimum correlation difference threshold was set at 2%.

Table 8. Contrast Sets Generated With and Without Correlational Pruning

	$r = 0.00$		$r = 0.25$		$r = 0.50$		$r = 0.60$		$r = 0.70$	
k-Itemsets	*No Prune*	*Prune*	*No Prune*	*Prune*	*No Prune*	*Prune*	*No Prune*	*Prune*	*No Prune*	*Prune*
1	46	46	21	21	9	9	0	0	0	0
2	548	531	226	226	53	50	11	11	7	7
3	2,721	2,506	949	882	188	148	36	35	17	14
4	7,577	6,290	2,377	1,956	394	257	53	36	21	4
5	13,899	10,183	4,104	2,838	536	332	35	15	15	0
6	18,293	11,897	5,359	3,063	508	318	10	2	6	0
7	17,915	10,305	5,433	2,562	345	208	0	0	0	0
8	13,124	6,531	4,232	1,671	167	86	0	0	0	0
9	7,077	2,964	2,466	835	55	20	0	0	0	0
10	2,715	931	1035	309	11	2	0	0	0	0
11	697	191	295	80	0	0	0	0	0	0
12	106	23	51	13	0	0	0	0	0	0
13	7	0	4	0	0	0	0	0	0	0
Total	84,725	52,398	26,552	14,456	2,266	1,430	145	99	66	25

Clearly, the choice of minimum correlation threshold can affect the quantity and validity (i.e., quality) of the contrast sets generated. For example, when $r = 0.00$ and with no pruning and pruning, 84,725 and 52,398 contrast sets were generated, respectively. Contrast sets containing up to 13-itemsets and 12-itemsets were generated with no pruning and pruning, respectively. The number of contrast sets generated with pruning is 62% of the number generated without pruning. Similarly, when $r = 0.25$, 0.50, 0.60, and 0.70, the number of contrast sets generated with pruning is 54%, 63%, 68%, and 38% of the number generated without pruning. The number of contrast sets generated is also significantly reduced as the minimum correlation threshold increases. For example, the number of contrast sets generated with pruning when $r = 0.70$ (i.e., strong positive correlation by most standards) is 0.00048% of the number generated when $r = 0.00$.

Finally, we describe a situation where contrast sets at level $i + 1$ in the search space have higher correlation than those at level i, a situation that is possible, as was described in Section 4.2.2. The situation occurs frequently in practice. For example, the rules `Bruise=no (r=0.501)`, `Bruise=no` $\wedge$ `Gill Space=close (r=0.735)`, and `Bruise=no` $\wedge$ `Gill Space=close` $\wedge$ `Veil Color=white (r=0.787)` were generated from the Mushroom dataset. Recall that according to the general rule of thumb previously described, a Phi correlation coefficient in the range $0.7 \leq r \leq 1.0$ represents a strong positive association. If we set the minimum correlation threshold to $\lambda = 0.7$, then the more general rule `Bruise=no (r=0.501)` would have been pruned and the two specializations never would have been generated. This highlights a problem in setting the minimum correlation threshold and shows how it can affect results.

6 Conclusion

We have discussed and demonstrated two alternative approaches to the contrast set mining problem. Essentially, STUCCO and CIGAR are based upon different statistical philosophies and assumptions: STUCCO seeks to control Type I error, while CIGAR seeks to control Type II error. However, experimental results showed that even though the underlying statistical assumptions are different, both approaches can be used to generate potentially interesting contrast sets.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data (SIGMOD'93)*, pages 207–216, Washington, D.C., U.S.A., May 1993.
2. S.D. Bay and M.J. Pazzani. Detecting change in categorical data: Mining contrast sets. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*, pages 302–306, San Diego, U.S.A., August 1999.
3. S.D. Bay and M.J. Pazzani. Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery*, 5(3):213–246, 2001.
4. G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*, pages 43–52, San Diego, U.S.A., August 1999.
5. B.S. Everitt. *The Analysis of Contingency Tables.* Chapman and Hall, 1992.
6. J. Li, T. Manoukian, G. Dong, and K. Ramamohanarao. Incremental maintenance on the border of the space of emerging patterns. *Data Mining and Knowledge Discovery*, 9(1):89–116, 2004.
7. Terry Peckham. Contrasting interesting grouped association rules. Master's thesis, University of Regina, 2005.
8. G.I. Webb, S. Butler, and D. Newlands. On detecting differences between groups. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, pages 256–265, Washington, D.C., U.S.A., August 2003.

Classification of Music Based on Musical Instrument Timbre

Peter Somerville and Alexandra L. Uitdenbogerd

School of Computer Science and Information Technology, RMIT University
GPO Box 2476V, Melbourne 3001, Australia
{pcsomerv,alu}@cs.rmit.edu.au

Abstract. Many people have access to large collections of music in digital form. However, their methods of access are mostly based on meta-data and directory browsing. One content-based method of access may involve using instrument samples to retrieve pieces possessing similar timbres. In this paper, feature selection and classification of six instrument families is explored to help identify and characterise an instrument using its timbre. Single instrument note samples plus short song excerpts were examined. Classification accuracy of single instrument note samples using a KNN classifier was close to 90% for both Fourier-based spectral audio features and Mel-Frequency Cepstral Coefficients. However, distinguishing pianos was the most difficult classification task, achieving only 56% accuracy.

1 Introduction

The growing field of content-based music information retrieval researches a range of techniques for finding music. These include retrieval based on melodies, harmony, audio sample and musical taste. One of the aims is to make audio-based retrieval possible, for example retrieving pieces of the same genre, artist, or mood as a given sample. Related to this is the idea of retrieving pieces given a sample of a particular musical instrument — the focus our work. This type of query well matches the modern aesthetic of enjoying pieces of music not so much for the melodic or harmonic structure but for the timbres that it contains.

Most work on audio-based retrieval is currently still focused on classification and the determination of good features for the task at hand. The task that we attempt here — musical instrument classification given a sequence of notes — has only fairly recently been examined [3]. Previous studies have built musical instrument classifiers that use as their training and test data recordings of musical instruments playing a single note.

The intention behind this paper is to help answer the following questions. 'How can an instrument be characterised using its timbre in order to retrieve songs of similar timbre?' and, 'How can we best classify music based on the instrument's timbre?'. Eventually an application will be produced that allows the retrieving of songs based on an instrument's timbre. Part of the process is to examine the features of the samples and to use classification techniques to classify the instruments.

In our experiments reported here we tested both one-note instrument samples and short song segments of digital audio containing a single instrument. Audio features extracted from the musical instrument sample files were used to differentiate one instruments. We used six instrument families, each comprising six different types of digital instruments. Many of the instruments used in our experiments were software-based.

This paper concentrates on the following extracted features: Spectral Centroid, Rolloff, Flux, Zerocrossings, Low Energy and Mel-Frequency Cepstral Coefficients (MFCC) which are applied to the samples. We used decision trees (J48), OneR and k-nearest neighbor (KNN) to classify the instrument samples based on these features.

Throughout the work covered in this paper, the best performing classifier was KNN. The piano instrument family was more difficult to classify than the other instrument families.

This paper first covers related work. It then explains the method and approach used, feature extraction and classification and then the source data. Experiments and results are then included, finishing with the conclusion and possible future work.

2 Related Work

Most of the instrument classification research has been undertaken using single-note instrument samples. Very little research has used multi-note samples. In our survey we discuss some of the work published on instrument classification as well as other related work in audio classification and segmentation.

A range of different features have been tried for instrument classification, with common ones including brightness (spectral centroid) and Mel-Frequency Cepstrum Coefficients (MFCCs). Herrera et. al. [6] provides a very thorough account of feature selectors and classification techniques for this classification task. Reference is made to many other authors who have explored feature extraction and classification.

Jensen and Arnspang [7] included the amplitude of odd partials and inharmonicity, and used 1500 sounds from seven instruments for their work. Kostek used features derived from the Wavelet Transforms rather than the FFT-based ones [10]. Herrera et. al. [6] indicate that there are some features that work particularly well with certain types of instruments and the features considered should cover temporal, spectral and temporal evolution.

Essid and David [3] indicate that there has not been any real consensus on what features should be chosen when trying to identify musical instruments. Class pairwise feature selection was used to determine the most efficient features used to compare two instruments. When combined with the Gaussian Mixture Model Classification strategy, the results are commendable. Ten musical instruments were considered in the experiments conducted.

A number of instrument classification techniques are listed by Herrera et. al. [5] where the emphasis is to segment musical audio streams to achieve tasks like locating a solo in the middle of a song. The KNN algorithm was reported to perform very well on small datasets [9]. In this case, the Root Mean Square (RMS - the average value of a part waveform) descriptor was used with four instruments having a restricted note range of one octave. On some occasions, the KNN classifier has been combined with other classifiers [2] or enhanced [4] particularly when larger data sets were used.

The techniques outlined by Herrera et. al. [6] that cover sound classification are K-Nearest Neighbours, Naive Bayesian Classifiers, Discriminant Analysis, binary (or decision) trees, Artificial Neural Networks(ANN), Support Vector Machines (SVMs), Rough Sets and Hidden Markov Models. A SVM was used for classification of eight instruments playing musical scores by Marques [12]. When using MFCCs and 200msec sound segments, an accuracy of 70% was acheived. When applied to longer segments of sound, an improvement to 83% resulted. The instruments which proved difficult to classify were trombone and harpsichord.

Vincent and Rodet [16] presented a method used for instrument identification, based on using Independent Subspace Analysis (ISA). Tests using five instruments and some song data from commercial CDs gave promising results. The results showed some advantages over using Gaussian Mixture Models (GMM), SVMs or linear ISA. This approach appeared to be quite successful when applied to polyphonic music, whereas the other approaches seemed to work better with monophonic music.

Zhang and Kuo [19] looked at content-based classification and retrieval of audio, where recordings were classified and segmented into music, speech and other environmental sounds. The sound collection consisted of 1000 environmental sound clips, 100 excerpts of music played with ten kinds of instruments, other styles of music sung by males and females, speech from different languages and speech with music in the background. They achieved an accuracy of over 90% with their collection.

The work carried out by Tzanetakis and Cook [14] covered musical genre classification of audio signals, revealing a classification of 61% where ten musical genres were concerned. The datasets included 20 musical genres and three speech genres with 100 excerpts being used for training. The excerpts were taken from radio, compact disc and MP3 compressed audio files.

Other studies have also been undertaken in exploring musical instrument classification. Eronen and Klapuri identified musical instruments using various feature extraction techniques and classifying them into musical instrument families based on the similarity of features found. Features covering the spectral range and temporal properties of sounds were investigated. The database of sounds consisted of 1498 solo tones from 30 orchestral instruments covering several styles including pizzicato, bowed and muted [2]. An accuracy of 94% was achieved for identifying the correct instrument family and 80% when identifying individual instruments.

Kaminskyj [8] looked at a multi-feature instrument sound classifier, and achieved an accuracy of 96% when identifying instrument families. Nineteen musical instruments were used providing 604 recordings, of which 517 were non-vibrato and the rest, vibrato recordings.

The recognition of musical instruments in polyphonic audio was tackled by Eggink and Brown [1]. When using 27 different instruments, the system achieved a recognition accuracy of 57%. When frequency regions were muddled and cluttered with excessive tones, the consequence was to omit such regions from the classification process.

A content-aware sound browser known as SoundFisher has been developed by Musclefish [18]. It is a sound effects database management system, featuring retrieval and content-based recognition. The structures, determined by the system, contain statistical data, describing aspects such as pitch, brightness, bandwidth and loudness. Soundfisher provides a useful application for storing, categorizing and retrieving sounds.

3 Method and approach used

To identify characteristics of an instrument's timbre, it was necessary to examine appropriate features. Features extracted including Spectral Centroid, Spectral Rolloff, Spectral Flux, Zero Crossings, Low Energy and MFCCs helped to define characteristics that can distinguish instruments. Spectral Centroid refers to the spectral brightness; Spectral Rolloff refers to the measure of spectral shape; Spectral Flux refers to the measure of spectral change; Zero Crossings is the number of time domain zero-crossings of the signal. Low energy indicates the amount of quiet time, which is a good discriminator for speech against music. Throughout this paper, Centroid, Rolloff, Flux, Zerocrossings and Low Energy will be referred to as SPEC-CHA (spectral characteristics). Centroid, Rolloff and Flux are based on the Short Time Fourier Transform. An in depth coverage of the SPEC-CHA based features was covered by Tzanetakis et. al. [15]. MFCCs are used in speech recognition and some music based applications. They represent a compact form of the audio spectrum. Further details concerning MFCCs are covered extensively by Logan [11].

In our notation of the SPEC-CHA attributes used in the experiments, we include a number representing the time slice on which the attribute is based, and we prefix it with Mean or Std, for mean and standard deviation respectively. Thus the features are annotated MeanCentroid1, MeanRolloff1, MeanFlux1, MeanZeroCrossings1, StdCentroid1, StdRolloff1, StdFlux1, Stdzerocrossings1, Lowenergy1 through to MeanCentroid25, MeanRolloff25, MeanFlux25, MeanZeroCrossings25, StdCentroid25, StdRolloff25, StdFlux25, Stdzerocrossings25, Lowenergy25 for a 500msec length sample. MeanCentroid5 for instance, referred to the 5th Mean Spectral Centroid value calculated from the sample. This measured a value at approximately 5 x 20msec = 100msec from the beginning of the sample. The nine attributes repeated 25 times across the one note sample provided readings at all stages of the instrument sample. This gave

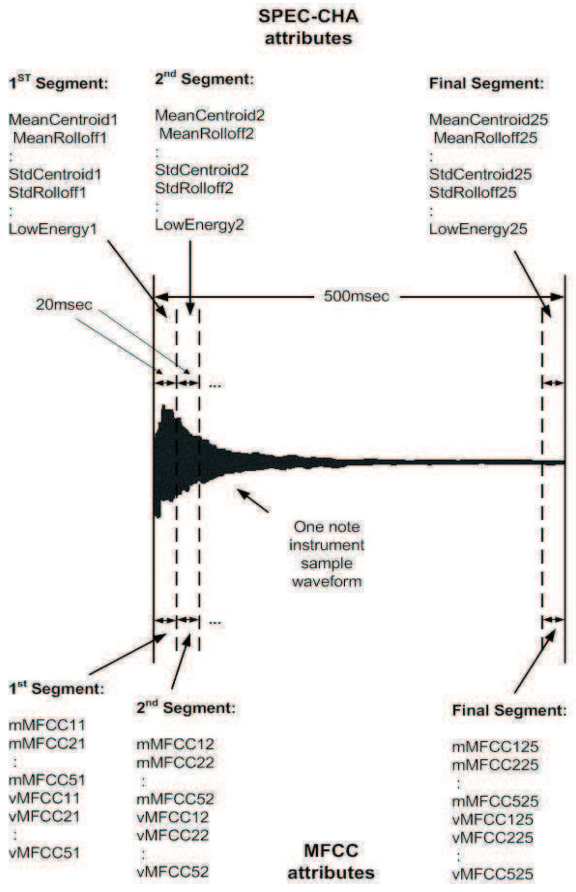


Fig. 1. SPEC-CHA and MFCC attributes spanning a one note instrument sample

225 attributes spanning the one note sample. MFCC attribute values ranged from mMFCC11 to mMFCC525 and vMFCC11 to vMFCC525, indicating firstly, mean attributes followed by variance attributes. The first digit in the attribute name, ranged from 1 to 5 and the next digit entries ranged from 1 to 25. This gave 5 x 25 = 125 mean and 125 variance coefficients spanning from the beginning to the end of the instrument sample. Similar to the SPEC-CHA situation, 250 attributes were used for MFCC which span the instrument sample. For one-note samples lasting longer than 500msec, extra attributes were used and therefore the numbers attached to the attributes also increased. Figure 1 indicates the coverage of the attributes over the entire instrument sample.

After extracting features, classification techniques including OneR, Decision Trees (J48) and K-Nearest Neighbour classifier were applied. These revealed more in-depth details of the audio samples explored [1, 8].

To help with the sound analysis tasks, MARSYAS (Musical Analysis and Retrieval Systems for Audio Signals version Marsyas-0.1) was used for its feature extraction ability. MARSYAS provides access to feature extraction selectors including Spectral Centroid, Rolloff, Flux, Zero Crossing, Low Energy and MFCCs. These approaches would provide a suitable starting point when applied to single instrument samples and short song segments.

The Weka Data Mining toolkit was used to classify instrument collections [13]. Within Weka, to help to provide classification accuracy, ten-fold stratified cross-validation was used. The stratified approach is where Weka attempts to properly represent each instrument class in both training and test sets. Ten-fold cross-validation is where the data is split into ten similar sized partitions, and in turn, each is used for testing while the rest is used for training. This procedure is repeated until every instance has been used once for testing [17]. In general, when using the classification methods within Weka, default values were used.

All sound files were stored as 22050Hz, 16 bit, mono audio files. Forty analysis windows of 20 milliseconds were used with 512 samples per window [15]. All features were calculated every 20 milliseconds.

Four types of experiments were carried out. The first three involved one shot instrument samples, and the last one, five second segments of digital audio generated from midi files. The first involved running feature extraction on the six individual instrument groups. For each, this included three volume levels, low, medium and high and five different octave notes, C1 through to C5 where C4 represents middle C, which is at 440Hz. Since the piano category results were inferior and very low compared to the remaining categories, experiment two was carried out on just the piano, exploring different groupings of the five octaves used. The third major test combined all six instruments to determine any differences between the instrument families. The final test combined all six instruments whilst using five second segments of audio generated from six midi files.

4 Data Sources

The data sources used samples from software and hardware based instruments. Figure 2 shows the categories of musical instrument families used in the experiments. The instruments came from an array of sources. These included synthesizers (Korg Trinity and Yamaha CS2X), digital piano (Roland EP85), Soundfonts, Gigasamples and VST instruments. The first three instruments were hardware-based, whereas the final three were generated in software. The emphasis in the experiments was to use software and digitally based instruments rather than sampling from real acoustic instruments. Six categories of instruments were used, which cover pianos, strings, organs, brass, flutes and violins. The strings category differed to violins in that the string instrument samples were built from

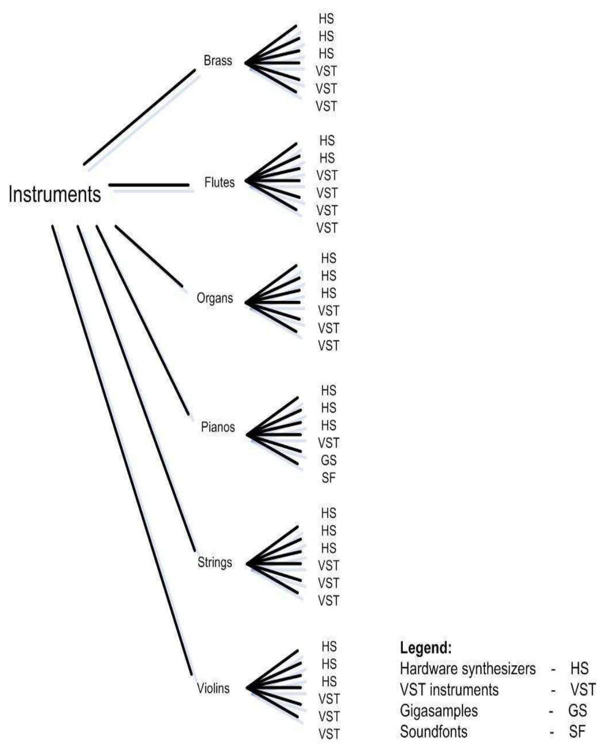


Fig. 2. Instrument tree

layers of strings sounds whereas the violins were just individual violin samples. Six different types of instruments were used within each category. These can also be seen in Figure 2.

The experiments used one note instrument samples, lasting up to two seconds in length and also short song pieces lasting five seconds. The short song pieces were digital songs files generated from midi files. The segments of midi files came from the following six pieces of music, Tchaikovsky - Swanlake - prelude, BirdLand full band. Handel Water Music, a Reggae piece, a Hardrock piece and a Latin piece.

5 Experiments & Results

The experiments aimed to extract features in order to characterize the timbre of instrument samples. Feature extraction of single note instrument samples was undertaken and then classified, based on some popular classification methods. These experiments were then extended to include songs containing multiple in-

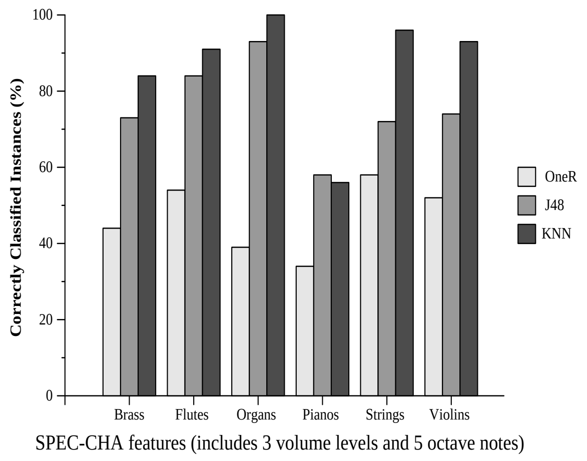


Fig. 3. SPEC-CHA Features - Mean and Stddev.

strument notes. A further task undertaken in the experiments, was to identify features that are prominent. The OneR classifier helped to achieve this.

5.1 Experiment 1

To help answer the first question concerning the characterising of an instrument using its timbre, the first experiment treated each instrument group separately, extracts the relevant features and performs classification on them. More specifically, this experiment endeavored to answer another question: 'Given six different instrument family groups and relevant feature extractors, can adequate classification occur using the OneR, KNN and J48 classifiers?'. This experiment dealt primarily with one note instrument samples. To help answer this, the following approach was undertaken. Short, one note instrument samples were used which last between 500msec and 800msec, depending on the category of instrument. The samples covered three different levels of volume and five octave notes which were C1, C2, C3, C4 and C5. Features, including a Low Energy value, plus four mean and standard deviation values of Centroid, Rolloff, Flux and Zero Crossings, were used. Comparisons can be seen in Figure 3. The KNN classification technique returned the highest correctly classified number of instances with most instruments recording above 90%. The piano category only returned 56% in this instance. In the particular case of the organ category, SPEC-CHA features returned very high results for both J48 and KNN. Correctly classified instances were 93% and 100% respectively.

Figure 4 which refers to the MFCC feature selectors, also showed quite highly classified instances for the KNN classification approach, with most instruments recording above 90%. However, the piano category only delivered 67%. For both categories of feature selectors, the decision tree classification method generally returned poorer results than the KNN technique.

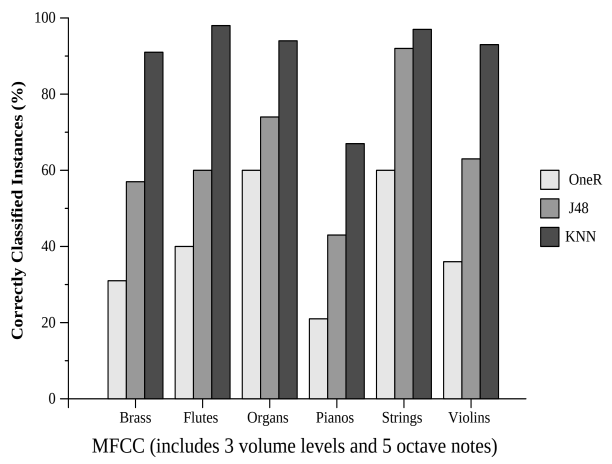


Fig. 4. MFCC features - 5 mean MFCCs and 5 std dev. MFCCs

Table 1. OneR classifier showing significant features

Instrument Category	*OneR attributes (SPEC − CHA)*	*OneR attributes (MFCCs)*
Brass	StdFlux13	mMFCC324
Flutes	StdFlux8	mMFCC44
Organs	StdCentroid29	vMFCC130
Pianos	MeanFlux19	mMFCC21
Strings	MeanFlux4	vMFCC139
Violins	MeanFlux12	mMFCC12

Witten and Frank indicated that the OneR classifier aimed to express a set of rules that test one particular attribute [17]. Table 1 indicates the significant features identified by WEKA using the OneR classifier.

Flux values which refer to the spectral change were prevalent with the SPEC-CHA based features. Coefficients registering early, and near the middle of the sample, were indicated using the MFCC approach.

The confusion matrices in Table 2 shows the piano instrument family, which includes instruments that were difficult for the given classifiers to identify. Piano2ep85, a Roland digital piano was particularly difficult to classify and The-Grand, a Vst instrument and puii002wiredcs2x, a hardware synthesizer, posed difficulties in numerous cases. Interestingly, piano2ep85 and puii002wiredcs2x represented two instrument patches coming from two of the most expensive instruments of the ones used. These two were hardware instruments whilst most of the others are all created in software.

Table 2. Confusion Matrices for pianos - using one note samples

```
SPEC-CHA Features:
Pianos
J48                                             KNN
=== Confusion Matrix ===                        === Confusion Matrix ===
  a  b  c  d  e  f   <-- classified as            a  b  c  d  e  f   <-- classified as
 13  2  0  0  0  0 |  a = SCC1                   15  0  0  0  0  0 |  a = SCC1
  2  8  3  1  1  0 |  b = TheGrand                3  2  2  1  3  4 |  b = TheGrand
  1  2 10  0  1  1 |  c = gigapiano               0  1 13  0  1  0 |  c = gigapiano
  2  2  1  9  0  1 |  d = pa127isntitgrandkorg    0  0  1 12  2  0 |  d = pa127isntitgrandkorg
  1  0  3  0  6  5 |  e = piano2ep85              1  4  3  4  0  3 |  e = piano2ep85
  1  3  1  0  4  6 |  f = puii002wiredcs2x        2  1  1  1  2  8 |  f = puii002wiredcs2x

MFCC features
Pianos
J48                                             KNN
=== Confusion Matrix ===                        === Confusion Matrix ===
  a  b  c  d  e  f   <-- classified as            a  b  c  d  e  f   <-- classified as
 10  1  0  3  0  1 |  a = SCC1                   14  0  1  0  0  0 |  a = SCC1
  3  3  4  1  2  2 |  b = TheGrand                1  6  1  3  1  3 |  b = TheGrand
  0  4  8  0  3  0 |  c = gigapiano               1  0 12  0  1  1 |  c = gigapiano
  2  2  1  9  1  0 |  d = pa127isntitgrandkorg    0  1  1 13  0  0 |  d = pa127isntitgrandkorg
  2  4  0  2  6  1 |  e = piano2ep85              2  1  2  2  4  4 |  e = piano2ep85
  3  3  2  4  0  3 |  f = puii002wiredcs2x        0  2  1  1  0 11 |  f = puii002wiredcs2x
```

5.2 Experiment 2

Since the piano instrument family in experiment one returned far poorer classification results than any other instrument family, the second experiment involved close examination of pianos. The question relevant to this experiment is 'Are there significant differences of timbre characteristics within the piano instrument family?'. To answer this, further experiments were conducted involving different octave groupings. The previous entries in the confusion matrices for the piano category were conducted across the five octaves, C1 to C5. Different groupings of the octaves were then explored to determine any significant variations across the span of the five octaves. Firstly, tests were carried out on individual octaves C1, C2, C3, C4 and C5, followed by groupings of two adjacent octaves C1 & C2, C2 & C3,C3 & C4 and C4 & C5 and then three adjacent octaves C1 & C2 & C3, C2 & C3 & C4 and C3 & C4 & C5. There was no significant outcome arising from the tests, except for a marginal difference between lower octaves and higher octaves. Classification was generally more successful when working with lower octaves and middle range octaves rather than higher ones. This, however, was not conclusive in all cases. The largest difference was evident with the grouping of three octaves C1 & C2 & C3, C2 & C3 & C4 and C3 & C4 & C5. The details can be seen in Figure 5.

Classifying the octave category of C3 & C4 & C5 for both SPEC-CHA and MFCC features, was difficult compared to that of C1 & C2 & C3 and C2 & C3 & C4, but the differences were only minor and less than 10% in most cases.

To identify which particular piano instruments were of concern, the confusion matrices in Table 3 are given.

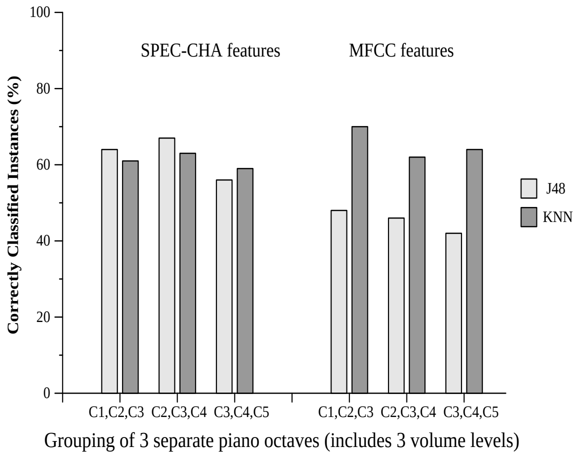


Fig. 5. Pianos - general features including 3 volumes

Table 3. Confusion Matrices for piano octaves: C3, C4 & C5

```
SPEC-CHA features:
Pianos
J48                                             KNN
=== Confusion Matrix ===                        === Confusion Matrix ===
 a b c d e f   <-- classified as                 a b c d e f   <-- classified as
 8 0 0 1 0 0 | a = SCC1                          9 0 0 0 0 0 | a = SCC1
 1 5 2 1 0 0 | b = TheGrand                      2 4 2 1 0 0 | b = TheGrand
 0 2 5 1 1 0 | c = gigapiano                     0 0 9 0 0 0 | c = gigapiano
 2 0 1 6 0 0 | d = pa127isntitgrandkorg          1 0 1 7 0 0 | d = pa127isntitgrandkorg
 0 0 0 0 5 4 | e = piano2ep85                    1 0 2 1 2 3 | e = piano2ep85
 1 0 0 0 7 1 | f = puii002wiredcs2x              2 0 1 1 4 1 | f = puii002wiredcs2x

MFCC features
Pianos
J48                                             KNN
=== Confusion Matrix ===                        === Confusion Matrix ===
 a b c d e f   <-- classified as                 a b c d e f   <-- classified as
 7 1 0 1 0 0 | a = SCC1                          8 0 0 1 0 0 | a = SCC1
 0 3 2 1 2 1 | b = TheGrand                      0 4 1 2 1 1 | b = TheGrand
 1 3 3 1 0 1 | c = gigapiano                     0 0 7 1 0 1 | c = gigapiano
 1 1 0 5 0 2 | d = pa127isntitgrandkorg          0 0 1 8 0 0 | d = pa127isntitgrandkorg
 1 3 0 1 1 3 | e = piano2ep85                    1 1 0 1 2 4 | e = piano2ep85
 0 2 1 0 2 4 | f = puii002wiredcs2x              0 1 1 1 0 6 | f = puii002wiredcs2x
```

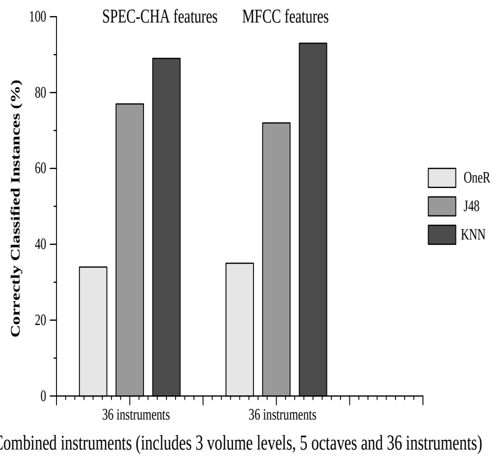


Fig. 6. Combined instruments - includes 36 instruments, 3 volumes and 5 octave notes

The piano2ep85 and puiio002wiredcs2x again failed to score highly in most cases. Generally the two instruments were difficult to classify from one another. In particular, using the SPEC-CHA features, the puii002wiredcs2x piano was difficult to classify, whereas, applying the MFCC features, piano2ep85 scored rather low.

5.3 Experiment 3

Experiment three tackled the question of 'Is there a notable difference in timbre between the different instrument group families?'. To assist in answering this question, the third experiment combined all six instruments, so that the general categories of brass, flutes, organs, pianos, strings and violins were used. The first 500msec of the one note samples were examined for the combined instruments. All three classifiers performed reasonably well, considering the variables included: three volumes, five octave notes and 36 different instruments. Figure 6 shows that the KNN classifier achieved close to 90% whereas the decision tree classifier resulted in correctly classified instance percentages near to 70. The most significant aspect to result from the confusion matrices was that the violin category of instruments was confused as strings on numerous occasions. As stated earlier, the string instrument samples comprised layers of string sounds whereas the violins were just purely individual violin samples. The confusion matrices in Table four indicate that for the OneR classifier, the piano category was classified far better than any other instrument category.

When considering different instrument categories, of all the instrument families considered, the OneR classifier had the least trouble in classifying the piano instrument family. Classification values of 49 and 43 for SPEC-CHA and

Table 4. Confusion Matrices for OneR classifier of combined instruments

```
SPEC-CHA features:
Combined instruments
OneR
=== Confusion Matrix ===
  a  b  c  d  e  f   <-- classified as
 26  6 11 14 16 17 |  a = organs
  8 25  6 18 19 14 |  b = brass
 10 10 32  5  9 24 |  c = flutes
 12 13 10 22 21 12 |  d = strings
 10 19  7 13 28 13 |  e = violins
  3  9 12  4 13 49 |  f = pianos
```

```
    MFCC features:
Combined instruments
OneR
=== Confusion Matrix ===
 a  b  c  d  e  f   <-- classified as
41 14  8  9  8 10 |  a = organs
15 26  5 14 13 17 |  b = brass
19 12 29 10 15  5 |  c = flutes
18 11 15 24 16  6 |  d = strings
14  9 13 14 28 12 |  e = violins
14 10  5  6 12 43 |  f = pianos
```

MFCC respectively were achieved, whereas most of the other instrument families recorded figures in the twenties or thirties. There was a total of 90 instances provided for the experiment. Here again, the piano instrument family was being identified as appearing separate to the other groups. The OneR classifier identified the MeanRolloff17 attribute as significant for SPEC-CHA and the variance of MFCC11 for MFCC. Rolloff referred to the measure of spectral shape.

5.4 Experiment 4

The final experiment expanded on experiment three, to include samples spanning multiple notes rather than a single one note instrument sample . The question 'Is it possible to distinguish timbre characteristics between instruments when longer segments of digital samples are used?' was examined in the final experiment. The experiment combined all six instruments and used five seconds of a song which was based on a monophonic midi file piece. Six different midi files were used which all possessed a different style. Figure 7 indicates the classification outcomes. The following confusion matrices in Table five indicate that the violins, and on some occasions, the flutes were the instrument families most difficult to classify, using all six midi files. Part of the reason for the violins being difficult to classify, may have been due to some of the midi file pieces being classically based. The violins were sometimes being confused with the string category of instruments.

6 Conclusion and future work

The questions concerning characterising an instrument using its timbre, and classifying music based on the instrument's timbre were posed and examined and the following observations resulted from our experiments. When the instruments were considered separately, using one note samples, distinguishing different piano timbres was difficult. On the contrary, the organ category performed particularly well when classified with both J48 and KNN. When the pianos were examined more closely, some of the hardware based instruments were difficult to classify. Further examination of the pianos was carried out and a grouping of three higher octaves C3, C4 & C5 revealed slightly inferior results to that of lower octaves.

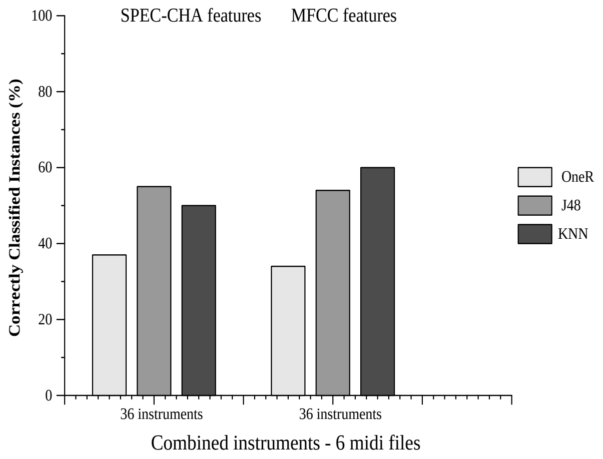


Fig. 7. Combined instruments (6 midi files) - includes 36 instruments, 3 volumes and 5 octave notes

Table 5. Confusion matrices for all instrument categories using 6 midi files

```
SPEC-CHA features:
6 midi files
J48                                             KNN
=== Confusion Matrix ===                        === Confusion Matrix ===
  a  b  c  d  e  f   <-- classified as            a  b  c  d  e  f   <--classified as
 19  2  3  6  3  3 |  a = organs                 12  4 14  3  3  0 |  a = organs
  3 20  4  2  5  2 |  b = brass                   2 12  5  3  9  5 |  b = brass
  8  8 13  1  2  4 |  c = flutes                  7  9  9  0  3  8 |  c = flutes
  2  1  1 22  8  2 |  d = strings                 3  0  1 27  2  3 |  d = strings
  2  8  1  2 18  5 |  e = violins                 1  4  3  6 19  3 |  e = violins
  0  3  4  1  2 26 |  f = pianos                  1  0  3  1  2 29 |  f = pianos

MFCC features
J48                                             KNN
=== Confusion Matrix ===                        === Confusion Matrix ===
  a  b  c  d  e  f   <-- classified as            a  b  c  d  e  f   <-- classified as
 22  2  2  4  2  4 |  a = organs                 22  2  2  3  1  6 |  a = organs
  2 18  5  2  6  3 |  b = brass                   1 27  0  3  5  0 |  b = brass
  4  2 24  1  2  3 |  c = flutes                  3  0 21  5  4  3 |  c = flutes
  1  2  5 20  4  4 |  d = strings                 3  0  1 17 13  2 |  d = strings
  3  3  3  7 15  5 |  e = violins                 2  3  0 14 15  2 |  e = violins
  5  2  4  6  1 18 |  f = pianos                  2  1  0  2  4 27 |  f = pianos
```

This may be due to piano samples differing more within the one instrument's samples than with other pianos. We also hypothesise that distinguishing pianos may be more difficult with lower sampling rates.

Using combined instrument categories of brass, flutes, organs, pianos, strings and violins, revealed that violins (solo) and strings (ensemble) were sometimes confused with one another. Another notable outcome was the high performance of the KNN classifier of close to 90% for this task. Surprisingly the OneR classifier had the least trouble in classifying pianos from other instrument categories. A trend seemed to be emerging where the pianos were being considered as different to the other instruments. The differences can be seen within the piano group itself, as well as when it was compared with other instrument groups.

When considering longer segments of instrument samples, the violin family group was generally the most difficult to classify and classification accuracy fell.

A common theme ran through many of the experiments. Experiments applied to MFCC features, resulted in KNN classification results that were better than when using SPEC-CHA features. Experiments using SPEC-CHA features and the decision tree classifier resulted in outcomes being varied.

In the future, when examining the single note piano samples, it may be worthwhile examining higher octaves than the ones tested. The range could be taken from C2 to C6 or even C7. It will also be interesting to test the effect on classification accuracy of varying the resolution of the sample. Currently we have considered each specific instrument separately within each class, leading to separate classifiers for violins, pianos and so on. We also tested a general classifier that decides on the general category of the instrument. A further step would be to test a multi-level classifier made from these components. It would be worthwhile investigating the KNN classification approach and determining significant reasons why it gave higher results than the other approaches. Other approaches such as Support Vector Machines, Neural Networks and the random forest classifer could also be explored. The random forest classifier is based on using a large number of individual decision trees. Ultimately, however, we wish to apply instrument classification to pieces containing multiple instruments, probably after segmentation has been applied.

In conclusion, we have demonstrated the possibility of classifying single-instrument musical snippets according to broad instrument classes, albeit with mixed success at this stage. The work here is still preliminary, but leads us to be optimistic regarding our goal of developing a query by instrument timbre system.

References

1. J. Eggink and G. Brown. Application of missing feature theory to the recognition of musical instruments in polyphonic audio. In *Proc. International Conference on Music Information Retrieval*, pages 125–131, Washington DC, USA, 2003.
2. A. Eronen and A. Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 753–756, 2000.

3. S. Essid, G. Richard, and B. David. Musical instrument recognition based on class pairwise feature selection. In *5th International Conference on Music Information Retrieval*, Bardelona, Spain, October 2004.
4. I. Fujinaga and K. MacMillan. Realtime recognition of orchestral instruments. In *Proc. International Computer Music Conference*, 2000.
5. P. Herrera, X. Amatriain, E. Batlle, and X. Serra. Towards instrument segmentation for music content description: a critical review of instrument classification techniques. In *Proc. International Symposium on Music Information Retrieval*, Plymouth, 2000.
6. P. Herrera, G. Peeters, and S. Dubnov. Automatic classification of musical instrument sounds. In *Journal of New Music Research*, 2002.
7. K. Jensen and J.Arnspang. Binary decision tree classification of musical sounds. In *Proc. International Computer Music Conference*, San Francisco, CA, 1999.
8. I. Kaminskyj. Multi-feature musical instrument sound classifier w/user determined generalisation performance. In *Proc. Aust. Comp. Music Assoc. Conf.*, pages 53–62, Melbourne, July 2002.
9. I. Kaminskyj and A. Materka. Automatic source identification of monophonic musical instrument sounds. In *Proc. of the IEEE International Conference on Neural Networks*, volume 1, pages 189–194, 1995.
10. B. Kostek. *Soft computing-based recognition of musical sounds.* L. Polkowski & A. Skowron (Eds.), Heidelberg: Physica-Verlag, 1998.
11. B. Logan. Mel frequency cepstral coefficients for music modeling. In *Proc. International Symposium on Music Information Retrieval*, 2000.
12. J. Marques. An automatic annotation system for audio data containing music. In *Unpublished master's thesis, Massachussetts Institute of Technology*, Cambridge, MA, 1999.
13. G. Tzanetakis. Marsyas: a software framework for computer audition. *Web Page - (http://opihi.cs.uvic.ca/marsyas/)*, 2005.
14. G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.
15. G. Tzanetakis, G. Essl, and P. Cook. Automatic musical genre classification of audio signals. In *Proc. Int. Symposium on Music Information Retrieval*, pages 205–215, USA, October 2001.
16. E. Vincent and X. Rodet. Instrument identification in solo and ensemble music using independent subspace analysis. In *Proc. International Symposium on Music Information Retrieval*, 2004.
17. I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques with Java implementations.* Academic Press, San Diego, CA, 2000.
18. E. Wold, T. Blum, D. Keislar, and J. Wheaton. A content-aware sound browser. In *Proc. International Computer Music Conference*, pages 457–459, 1999.
19. T. Zhang and K. Jay. Content-based classification and retrieval of audio. In *Proc. SPIE's 43rd Annual Meeting-Conference on advanced Signal Processing Algorithms, Architectures, and Implementations VII*, pages 432–443, San Diego, July 1998.

A Comparison of Support Vector Machines and Self-Organizing Maps for e-Mail Categorization

Helmut Berger[1] and Dieter Merkl[2]

[1] Electronic Commerce Competence Center – ec3
Donau-City-Straße 1, A–1220 Wien, Austria
helmut.berger@ec3.at
[2] Institut für Rechnergestützte Automation, Technische Universität Wien
Karlsplatz 13/183, A–1040 Wien, Austria
dieter.merkl@inso.tuwien.ac.at

Abstract. This paper reports on experiments in multi-class document categorization with support vector machines and self-organizing maps. A data set consisting of personal e-mail messages is used for the experiments. Two distinct document representation formalisms are employed to characterize these messages, namely a standard word-based approach and a character n-gram document representation. Based on these document representations, the categorization performance of both machine learning approaches is assessed and a comparison is given.

1 Introduction

The task of automatically sorting documents into categories from a predefined set, is referred to as *text categorization.* Text categorization is applicable in a variety of domains such as document genre identification, authorship attribution, survey coding, to name but a few [13]. One particular application is categorizing e-mail messages into legitimate and spam messages, i.e. *spam filtering.* The fact that spam has become a ubiquitous problem with e-mail has lead to considerable research and development of algorithms to efficiently identify and filter spam or unsolicited messages. In [1] a comparison between a *Naïve Bayes* classifier and an *Instance-Based* classifier to categorize e-mail messages into spam and legitimate messages is reported. The data for this study is composed of sample spam messages received by the authors as well as messages distributed through a linguist mailing list. The latter messages are regarded as legitimate. The authors conclude that the learning-based classifiers clearly outperform simple anti-spam keyword approaches. The data used in the experiments, however, does not reflect the typical mix of messages encountered in personal mailboxes. In particular, the exclusive linguistic focus of the mailing list should be regarded as rather atypical.

In [11] a related approach aims at authorship attribution and topic detection. In this paper, the performance of a *Naïve Bayes* classifier combined with n-gram language models is evaluated. The authors state that the n-gram-based approach showed better classification results than the word-based approach for topic detection in newsgroups messages. Their interpretation is that the character-based approach captures regularities that the word-based approach is missing.

A completely different approach to e-mail categorization is presented in [6]. In this work, e-mail filtering is based on a reputation network of e-mail users. The reputation represents the "trust" in the relevance of e-mails from various people. By means of transitive closures, reputation values can be assigned even to people with whom an individual had no contact before. Hence, the reputation network resembles the ideas immanent in collaborative filtering.

The study presented herein compares the performance of two text classification algorithms in a multi-class setting. More precisely, the performance of support vector machines (SVMs) trained with sequential minimal optimization and self-organizing maps (SOMs) in categorizing e-mails into a predefined set of multiple classes is evaluated. Besides categorizing messages into categories, we aim at providing a visual representation of document similarities in terms of the spatial arrangement obtained with the self-organizing map.

By nature, e-mail messages are short documents containing misspellings, special characters and abbreviations. This entails the additional challenge for text classifiers to cope with noisy input data. To classify e-mail in the presence of noise, a method used for language identification is adapted in order to statistically describe e-mail messages. Specifically, character-based n-grams as proposed in [4] are used as features that represent each particular e-mail message. A performance comparison of e-mail categorization based on an n-gram document representation vs. a word-based representation is provided.

Besides the content contained in the body of an e-mail message, the e-mail header holds valueable information that might impact classification results. The study presented in this paper explores the influence of header information on classification performance thoroughly. Two different representations of each e-mail message were generated. The first set consists of the information extracted from the textual data as found in the e-mail body. The second set additionally contains *all* the information of the e-mail header. So, the impact on classification results when header information is discarded can be assessed.

This paper is structured as follows. Section 2 reviews document representation approaches as well as the feature selection metric used for this study. The algorithms applied for text categorization are presented in Section 3. A description of the experiments for multi-class e-mail categorization is provided in Section 4. Finally, some conclusions are given in Section 5.

2 Document Representation

One objective of this study is to determine the influence of document representation methods on the performance of different text categorization approaches. To this end, a character n-gram document representation [4] is compared with a word-based document representation. For both document representation methods we rely on binary weighting, i.e. the presence or absence of a word (n-gram) in the document is recorded. The rationale behind this decision is that in our previous work [2] binary weighting resulted in superior categorization accuracy as compared to frequency-based weighting for this particular corpus. No stemming is applied to the word-based document representation, basically because of the multilinguality of the corpus that would require automatic language detection in order to apply the correct stemming rules.

2.1 n-Grams as Features

An n-gram is an n-character slice of a longer character string. When dealing with multiple words in a string, the blank character indicates word boundaries and is usually retained during the construction of the n-grams. However, it might get substituted with another special character. As an example for $n = 2$, the character *bi*-grams of *"have a go"* are $\{$*ha, av, ve, e_, _a, a_, _g, go*$\}$. Note that the "space" character is part of the alphabet and is represented by "_".

Formally, let $\mathcal{A}$ be an alphabet of characters. If $|\mathcal{A}|$ is the cardinality of $\mathcal{A}$ and $\mathcal{A}(n)$ the number of unique n-grams over $\mathcal{A}$, then $\mathcal{A}(n) = |\mathcal{A}|^n$. In case of $|\mathcal{A}| = 27$, i.e. the Latin alphabet including the blank character, we obtain 27 possible sub-sequences for *uni*-grams, already 729 possible sub-sequences for *bi*-grams and as many as $19,683$ possible sub-sequences for *tri*-grams. Note that these numbers refer to the hypothetical maximum number of n-grams. In practice, however, the number of distinct n-grams extracted from natural language documents will be considerably smaller than the mathematical upper limit due to the characteristics of the particular language. As an example consider the *tri*-gram *"yyz"*. This *tri*-gram will usually not occur in English or German language documents, except for the reference to the three letter code of Toronto's international airport.

Using character n-grams for describing documents has a number of advantages. First, it is *robust* with respect to spelling errors, second, the token alphabet is known in advance and is, therefore, *complete*, third, it is topic *independent*, fourth, it is very *efficient* and, finally, it does not require linguistic knowledge and offers a *simple* way of describing documents. Nevertheless, a significant problem is the number of n-grams obtained, if the value of n increases. Most text categorization algorithms are computationally demanding and not well suited for analyzing very high-dimensional feature spaces. For that reason, it is necessary to reduce the feature space using feature selection metrics.

2.2 Feature Selection

Generally, the initial number of features extracted from text corpora is very large. Many classifiers are unable to perform their task in a reasonable amount of time, if the number of features increases dramatically. Thus, appropriate feature selection strategies must be applied to the corpus. Another problem emerges if the amount of training data in proportion to the number of features is very small. In this particular case, classifiers produce a large number of hypothesis for the training data. This might lead to *overfitting* [8]. So, it is important to reduce the number of features while retaining those that are potentially useful. The idea of feature selection is to score each feature according to a feature selection metric and then take the top-ranked m features. A survey of different feature selection metrics for text classification is provided in [5].

For this study the *Chi-squared (χ^2)* feature selection metric is considered. The χ^2 statistic measures the lack of independence between a particular feature f and a class c of instances. The χ^2 metric has a natural value of zero if a particular feature and a particular class are independent. Increasing values of the χ^2 metric indicate increasing dependence between the feature and the class.

For the exact notation of the χ^2 metric, we follow closely the presentation given in [16]. Let f be a particular feature and c be a particular class. Let further A be the number of times f and c co-occur, B be the number of times f occurs without c, C be the number of times c occurs without f, D be the number of times neither f nor c occurs, and N be the total number of instances. We can then write the χ^2 metric as given in Equation 1.

$$\chi^2(f,c) = \frac{N(AD - CB)^2}{(A+C)(B+D)(A+B)(C+D)} \tag{1}$$

3 Text Categorization Algorithms

For the text categorization experiments an unsupervised and a supervised learning technique was selected. In particular, self-organizing maps as a prominent representative of unsupervised learning was chosen because of its capability of visual representation of document similarities. Support vector machines are chosen as the representative of supervised learning techniques because they have been identified in a number of studies as highly effective for text categorization.

3.1 Self-organizing Maps

The self-organizing map is a general unsupervised tool for ordering of high-dimensional data in such a way that similar instances are grouped spatially close to one another [7]. The model consists of a number of neural processing elements, i.e. units. These units are arranged according to some topology where the most common choice is marked by a two-dimensional grid. Each of the units i is assigned an n-dimensional weight vector m_i, $m_i \in \mathbf{R}^n$. It is important to note that the weight vectors have the same dimensionality as the instances, i.e. the document representations in our application.

The training process of self-organizing maps may be described in terms of instance presentation and weight vector adaptation. Each training iteration t starts with the random selection of one instance x, $x \in \mathbf{X}$ and $\mathbf{X} \subseteq \mathbf{R}^n$. This instance is presented to the self-organizing map and each unit determines its activation. Usually, the Euclidean distance between the weight vector and the instance is used to calculate a unit's activation. In this particular case, the unit with the lowest activation is referred to as the *winner*, c. Finally, the weight vector of the *winner* as well as the weight vectors of selected units in the vicinity of the *winner* are adapted. This adaptation is implemented as a gradual reduction of the difference between corresponding components of the instance and the weight vector, as shown in Equation (2). Note that we use a discrete-time notation with t denoting the current training iteration.

$$m_i(t+1) = m_i(t) + \alpha(t) \cdot h_{ci}(t) \cdot [x(t) - m_i(t)] \tag{2}$$

The weight vectors of the adapted units are moved slightly towards the instance. The amount of weight vector movement is guided by the learning rate, α, which decreases over time. The number of units that are affected by adaptation as well as the strength of adaptation depending on a unit's distance from the *winner* is determined by the neighborhood function, h_{ci}. This number of units also decreases over time such that towards the end of the training process only the *winner* is adapted. The neighborhood function is unimodal, symmetric and monotonically decreasing with increasing distance to the winner, e.g. Gaussian.

The movement of weight vectors has the consequence that the Euclidean distance between instances and weight vectors decreases. So, the weight vectors become more similar to the instance. Hence, the respective unit is more likely to win at future presentations of this instance. The consequence of adapting not only the *winner* but also a number of units in the neighborhood of the *winner* leads to a spatial clustering of similar instances in neighboring parts of the self-organizing map. Existing similarities between instances in the n-dimensional input space are reflected within the two-dimensional output space of the self-organizing map. In other words, the training process of the self-organizing map describes a topology preserving mapping from a high-dimensional input space onto a two-dimensional output space. Such a mapping ensures that instances, which are similar in terms of the input space, are represented in spatially adjacent regions of the output space.

3.2 Support Vector Machines

A support vector machine (SVM) is a learning algorithm that performs binary classification (pattern recognition) and real value function approximation (regression estimation) tasks. The idea is to non-linearly map the n-dimensional input space into a high-dimensional feature space. This high-dimensional feature space is classified by constructing a linear classifier. The basic SVM creates a *maximum-margin hyperplane* that lies in this transformed input space. Consider a training set consisting of labelled instances: A maximum-margin hyperplane splits the training instances in such a way that the distance from the closest instances to the hyperplane is maximized.

The training data is labelled as follows: $S = \{(x_i, y_i) | i = 1, 2, ..., N\}, y_i \in \{-1, 1\}, x_i \in \mathbf{R}^d$. Consider a hyperplane that separates the positive from the negative examples: $w \cdot x + b = 0$ is satisfied from those points x which lie on the hyperplane. Moreover, w is orthogonal to the hyperplane, $|b|/||w||$ represents the perpendicular distance from the hyperplane to the origin and $||w||$ is the Euclidean norm of w. Let d^+ (d^-) be the shortest distance from the separating hyperplane to the closest positive (or negative) example. Define the *margin* of a separating hyperplane to be $d^+ + d^-$. If the examples are linearly separable, the SVM algorithm looks for the separating hyperplane with the largest margin, i.e. *maximum-margin hyperplane*. In other words, the algorithm determines exactly this hyperplane, which is most distant from both classes. For a comprehensive exposition of support vector machines we refer to [3, 9].

For the study presented herein, the *sequential minimal optimization* (SMO) training algorithm for support vector machines is used. During the training process of a SVM the solution of a very large quadratic programming optimization problem has to be found. The larger the number of features which describe the data, the more time and resource consuming the calculation process becomes. For a detailed report on the functionality of the SMO training algorithm for SVMs we refer to [12].

4 Empirical Validation

4.1 Experimental Setting

The document collection consists of 1,811 e-mail messages. These messages have been collected during a period of four months commencing with October 2002 until January 2003. The e-mails have been received by a single e-mail user account at the *Institut für Softwaretechnik*, Vienna University of Technology, Austria. Beside the noisiness of the corpus, it contains messages of different languages.

Messages containing confidential information were removed from the corpus. The corpus was manually classified according to the categories outlined in Table 1. Due to the manual classification of the corpus, some of the messages may have been misclassified. Some of the introduced classes might give the impression of a more or less arbitrary separation. Introducing similar classes was intentionally done for assessing the performance of classifiers on closely related topics. Consider, for example, the *position* class that comprises 66 messages mainly posted via the *dbworld* and *seworld* mailinglists. In particular, it contains 38 *dbworld* messages, 23 *seworld* messages, 1 *isaus* message and 4 messages from sources not otherwise categorized. In contrast to standard *dbworld* or *seworld* messages, *position* messages deal with academic job announcements rather than academic conferences and alike. Yet they still contain similar header and signature information as messages of the *dbworld* or *seworld* classes. Hence, the difference between these classes is based on the message content only.

Two representations of each message were generated. The first representation consists of all data contained in the e-mail message, i.e. the complete header as well as the body. However, the e-mail header was not treated in a special way. All non-Latin characters, apart from the blank character, were discarded. Thus, all HTML-tags remain part of this representation. Henceforth, we refer to this representation as *complete* set. Furthermore, a second representation retaining only the data contained in the body of the e-mail message was generated. In addition, HTML-tags were discarded. Henceforth, we refer to this representation as *cleaned* set. Due to the fact, that some of the e-mail messages contained no textual data in the body besides HTML-tags and other special characters, the corpus of the *cleaned* set consists of less e-mails than the *complete* set. To provide

Table 1. Corpus statistics (e-mails per category).

category	***complete* set**	***cleaned* set**	**description**
admin	32	32	administration
dbworld	260	259	mailinglist
department	30	29	department issues
dilbert	70	70	"daily dilbert"
ec3	20	19	project related
isaus	24	22	mailinglist
kddnuggets	6	6	mailinglist
lectures	315	296	lecturing issues
michael	27	25	unspecific
misc	69	67	unspecific
paper	15	14	publications
position	66	66	job announcements
seworld	132	132	mailinglist
spam	701	611	spam messages
talks	13	13	talk announcements
technews	31	31	mailinglist
totals	**1,811**	**1,692**	

the total figures, the *complete* set consists of $1,811$ messages whereas the *cleaned* set comprises $1,692$ messages, cf. Table 1. Subsequently, both representations were translated to lower case characters.

Starting from these two message sets, the document representations are built. For each message in both sets a character n-gram representation with $n \in \{2, 3\}$ was generated. For the *complete* set we obtained $20,413$ distinct features and for the *cleaned* set $16,362$. Next, we generated the word-based representation for each set and obtained $32,240$ features for the *complete* set and $20,749$ features for the *cleaned* set. Note that occurrence frequencies are not taken into account in both representations. In other words, simply the fact of presence or absence of an n-gram or a word in a message is recorded in the document representation. Moreover, no stemming was applied for the word-based document representation. To test the performance of text classifiers with respect to the number of features, we selected the top-ranked n features as determined by the χ^2 feature selection metric, with $n \in \{100, 200, 300, 400, 500, 1000, 2000\}$. All experiments were performed with *10-fold cross validation*.

In order to evaluate the effectiveness of text classification algorithms applied to different document representations the *F–measure* as described in [15] is used. It combines the standard *Precision* P, cf. Equation (3), and *Recall* R, cf. Equation (4), measures with an equal weight as shown in Equation (5).

$$P = \frac{\text{number of relevant documents retrieved}}{\text{total number of documents retrieved}} \tag{3}$$

$$R = \frac{\text{number of relevant documents retrieved}}{\text{total number of relevant documents}} \tag{4}$$

$$F(P, R) = \frac{2 \cdot P \cdot R}{P + R} \tag{5}$$

The percentage of correctly classified instances is assessed by the *Accuracy* measure. It calculates the proportion of the number of correctly classified instances on the total number of instances in the collection, cf. Equation (6).

$$Accuracy = \frac{\text{number of correctly classified documents}}{\text{total number of documents}} \tag{6}$$

4.2 Experimental Results

Table 2 gives a comparison of the classification results for the two classifiers using the character n-gram representation and the word-based representation. In particular, the minimum, average and maximum F–measure values when applied to the *cleaned* and *complete* set are shown. Due to space limitations, we refrain from providing detailed class-based F–measure values. The results are based on 1000 features determined by the χ^2 feature selection metric. Note that the table's left part depicts the results for the supervised support vector machine (SVM) trained with sequential minimal optimization while the right part refers to the unsupervised self-organizing map (SOM). The results for the support vector machine are determined with the SMO implementation provided with the WEKA machine learning toolkit [14].

Table 2. The minimum, average and maximum F–measure values for the support vector machine (SVM) and the self-organizing map (SOM).

	Support vector machine (SVM)				**Self-organizing map (SOM)**			
	character n-grams		**word based**		**character n-grams**		**word based**	
	cleaned set	*complete* set	*cleaned* set	*complete* set	*cleaned* set	*complete* set	*cleaned* set	*complete* set
minimum	0.540	0.608	0.556	0.528	0.513	0.563	**0.615**	0.559
average	0.840	**0.902**	0.885	0.894	0.789	0.834	0.856	0.870
maximum	1	1	1	1	1	1	1	1

The support vector machine's F–measure values increase strongly when applied to the *complete* set of messages described by character n-grams. The average F–measure value is boosted by 6.2% in this particular case which is similar to the increase of the respective minimum F–measures. When applied to the word-based document representation, the average F-Measure value raises only marginally in case of the *complete* set. Interestingly, the minimum value is even smaller than the value obtained using the *cleaned* set. However, the largest average F-measure for the support vector machine is 90.2% obtained using the *complete* set with n-gram document representation.

The average F–measure values for the self-organizing map classifier show a comparable picture. The value increases by about 4.5% when the classifier is applied to the *complete* set described by character n-grams. In case of the word-based document representation the raise is only 1.4%. In contrast to the support vector machine, the largest average F-measure is obtained for the *complete* set based on the word-based document representation. Overall, the support vector machine outperformed the self-organizing map classifier. Especially with the n-gram document representation SVM is substantially better than SOM.

In Figure 1 the classifiers' accuracy values for different numbers of features, are shown. Figure 1(a) depicts the percentage of correctly classified instances using the support vector machine (SVM) and Figure 1(b) illustrates the results obtained for the self-organizing map (SOM) classifier. Each curve corresponds to a distinct combination of document representation and message set, e.g. the *cleaned* set described by means of character n-grams.

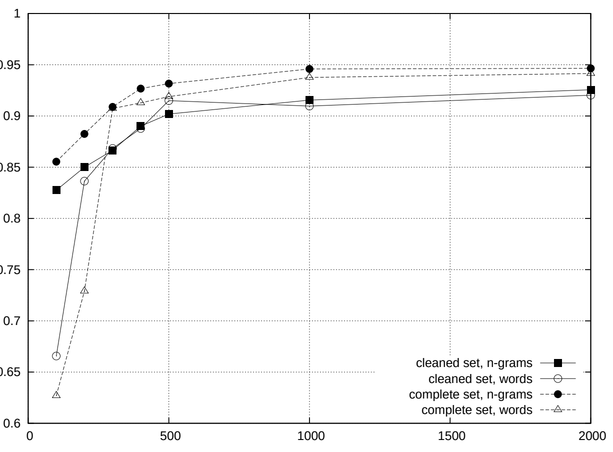
When we consider the support vector machine, cf. Figure 1(a), the accuracy values for the word-based document representation are remarkably low in case of a small number of features. Regardless of the message set, results are roughly 20% worse than those obtained when character n-grams are used. As soon as the number of features exceeds 300, the accuracy values for the word-based representation catch up with those of the n-grams. However, the character n-gram document representation outperforms the word-based approach almost throughout the complete range of features.

In case of the self-organizing map, cf. Figure 1(b), a similar trend is observed at the beginning. The n-gram document representation outperforms the word-based approach dramatically, as long as the number of features is below 300. Generally, once the number of features exceeds 300 the accuracy values for the word-based representation get ahead of those obtained for the character n-gram document representation.

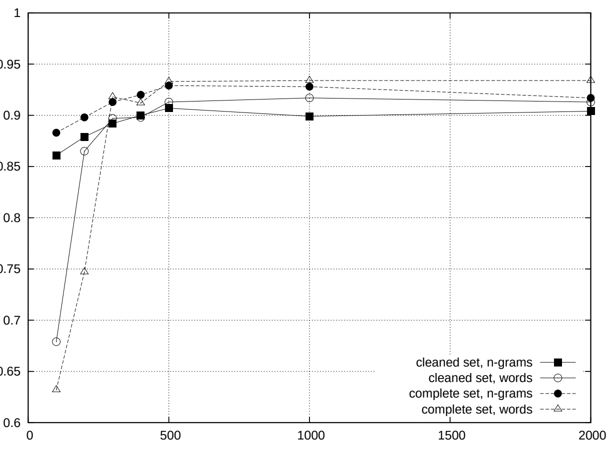
By using the self-organizing map for document space organization we gain as an additional benefit the concise visual representation of the document space as depicted in Figure 2. In this case the result of the self-organizing map for the *complete* data set based on n-gram document representation reduced to 500 features is shown. The available class information is exploited for coloring the map display. More precisely, each class is randomly assigned a particular color code and the color of a unit is determined as a mixture of the colors of documents colors assigned to that unit. Note that class information was not used during training, it is just used for superimposing color codes to the result of the training process. We are currently working on a more sophisticated coloring technique for self-organizing maps along the idea of *smoothed data histograms* [10].

It is obvious from the map display that the *spam* cluster is located on the top of the map with a remarkable purity, i.e. the number of misclassified legitimate messages is very small. On the lower left hand side of the map, an area related to messages from various mailinglists, such as *dbworld, seworld, isaus*, is found. It is remarkable how well the self-organizing map separates the various mailinglists when taking into account that some of the messages are highly similar. On the lower right hand side of the map, messages relating to university business are located, e.g. *teaching* and *department.* Again, the separation between these classes is achieved to a remarkably high degree.

For easier comparison, enlarged pictures of four regions of the self-organizing map are provided in Figures 3 to 6. Note that in these figures reference to the



(a) Support vector machine (SVM)



(b) Self-organizing map (SOM)

Fig. 1. Classification accuracy.

classes is given with the names originally chosen for the e-mail folders. Some of these names have German origin. So, "lehre" refers to *lectures*, and "insti" refers to *department.* The coordinates of the respective regions within the overall map are given in the caption of the figures. In particular, Figure 3 shows an area of the map on the left hand center featuring messages assigned to the *dilbert* and *spam* clusters. The *dilbert* messages are neatly arranged within the larger area of *spam* messages. Figure 4 depicts an enlarged view of the lower left hand corner of the map containing various messages from the mailinglists *dbworld* and *seworld.* Note that messages of the *position* class, i.e. messages related to academic job announcements are neatly embedded within this cluster. Moreover, messages from the *isaus* mailinglist are mapped to an adjacent area. This makes perfect sense, since these messages are primarily concerned with academic announce-

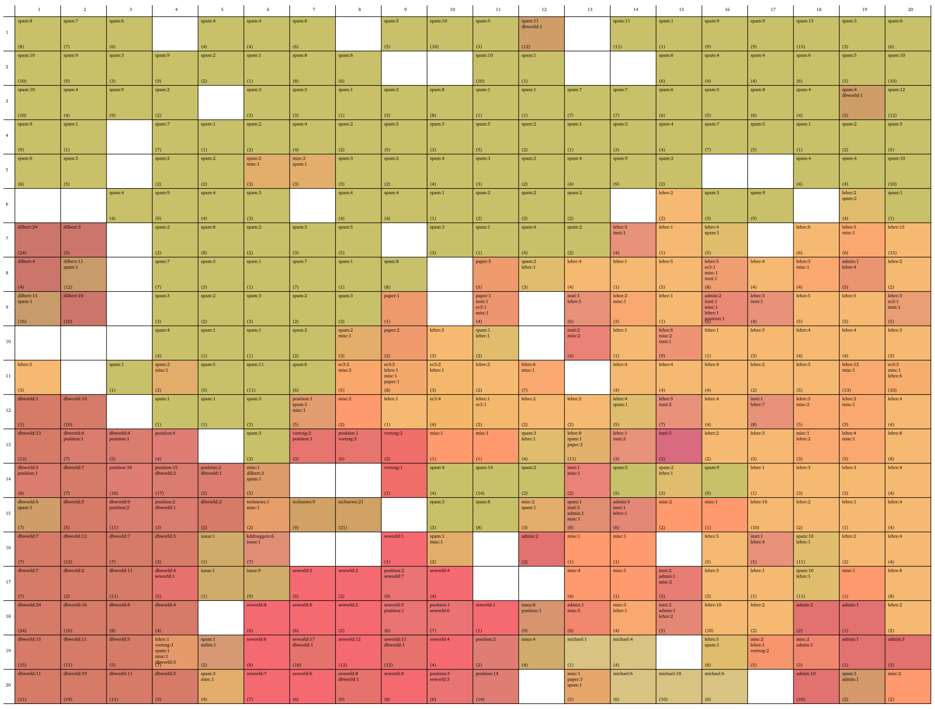


Fig. 2. A self-organizing map of the *complete* set, n-grams and 500 features.

ments related to Australia. Figure 5 enlarges the right center area of the map which primarily features messages related to university business. In particular, this cluster contains messages dealing with *teaching* and *department* issues. Finally, we show an enlargement of the area containing the messages of the *michael* cluster in Figure 6. This area is especially remarkable since no misclassification occurred during the unsupervised training process of the self-organizing map.

5 Conclusion

In this paper, a comparison of support vector machines and self-organizing maps in multi-class categorization is provided. Both learning algorithms were applied to a character n-gram as well as a word-based document representation. A corpus personal e-mail messages, manually split into multiple classes, was used. The impact of e-mail meta-information on classification performance was assessed.

In a nutshell, both classifiers showed impressive classification performance with accuracies above 90% in a number of experimental settings. In principle, both the n-gram-based and the word-based document representation yielded comparable results. However, the results for the n-gram-based document representation were definitely better in case of an aggressive feature selection strategy.

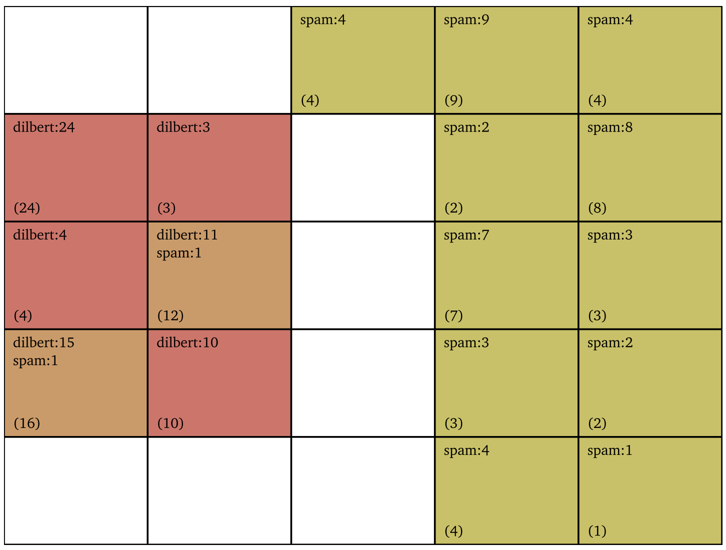


Fig. 3. Enlarged view of selected regions of the self-organizing map: *dilbert* and *spam*: col 1–5, row 6–10

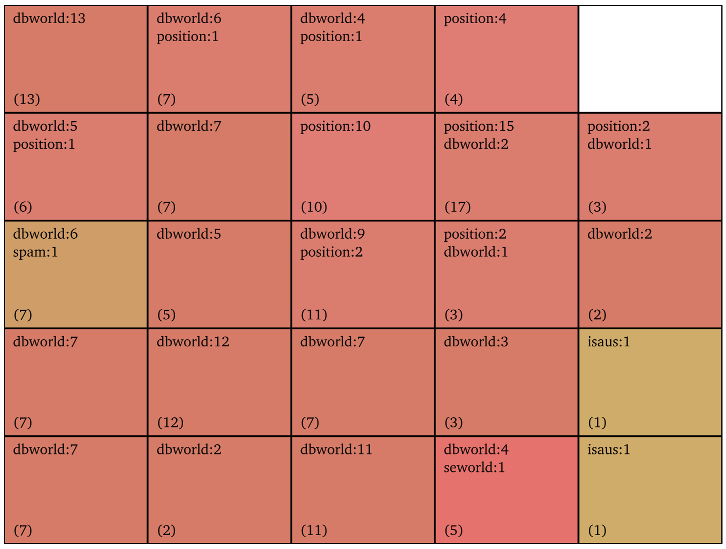
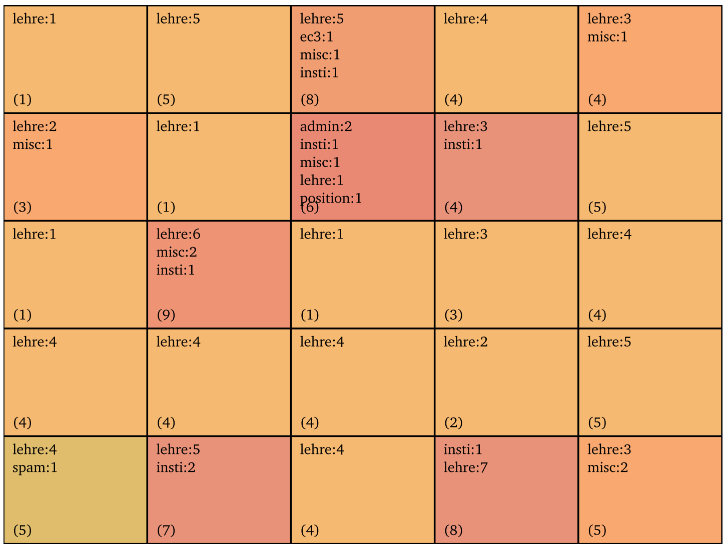
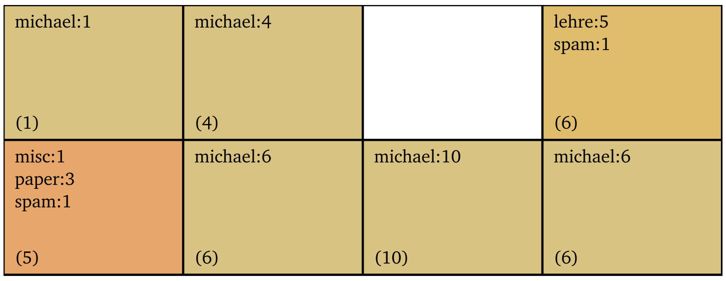


Fig. 4. Enlarged view of selected regions of the self-organizing map: mailinglist: col 1–5, row 13–17



lehre:1 (1)	lehre:5 (5)	lehre:5 ec3:1 misc:1 insti:1 (8)	lehre:4 (4)	lehre:3 misc:1 (4)
lehre:2 misc:1 (3)	lehre:1 (1)	admin:2 insti:1 misc:1 lehre:1 position:1 (6)	lehre:3 insti:1 (4)	lehre:5 (5)
lehre:1 (1)	lehre:6 misc:2 insti:1 (9)	lehre:1 (1)	lehre:3 (3)	lehre:4 (4)
lehre:4 (4)	lehre:4 (4)	lehre:4 (4)	lehre:2 (2)	lehre:5 (5)
lehre:4 spam:1 (5)	lehre:5 insti:2 (7)	lehre:4 (4)	insti:1 lehre:7 (8)	lehre:3 misc:2 (5)

Fig. 5. Enlarged view of selected regions of the self-organizing map: *teaching* and *department*: col 14–18, row 8–12



michael:1 (1)	michael:4 (4)		lehre:5 spam:1 (6)
misc:1 paper:3 spam:1 (5)	michael:6 (6)	michael:10 (10)	michael:6 (6)

Fig. 6. Enlarged view of selected regions of the self-organizing map: *michael*: col 13–16, row 19–20

The more features are selected, the more favorable are the results for the word-based document representation. The only exception to that was the result for the support vector machine which produced the best result based on the 2000 top-ranked n-grams selected according to the χ^2 metric.

The accuracies of the self-organizing map are just slightly worse than those of support vector machines. This is all the more remarkable because the self-organizing map is trained in an unsupervised fashion, i.e. the information on class membership is not used during training. Moreover, training of the self-organizing map results in a concise graphical representation of the similarities between the documents. In particular, documents with similar contents are grouped closely together within the two-dimensional map display.

Acknowledgments

Many thanks are due to the Machine Learning Group at The University of Waikato for their superb WEKA toolkit (http://www.cs.waikato.ac.nz/ml/).

References

1. I. Androutsopoulos, G. Paliouras, V. Karkaletsis, G. Sakkis, C. Spyropoulos, and P. Stamatopoulos. Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. In *Proc. PKDD-Workshop Machine Learning and Textual Information Access*, Lyon, France, 2000.
2. H. Berger and D. Merkl. A Comparison of Text-Categorization Methods applied to N-Gram Frequency Statistics. In *Proc. Australian Joint Conf. Artificial Intelligence*, Cairns, Australia, 2004.
3. C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 1998.
4. W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *Proc. Int'l Symp. Document Analysis and Information Retrieval*, Las Vegas, NV, 1994.
5. G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.
6. J. Golbeck and J. Hendler. Reputation network analysis for email filtering. In *Proc. Conf. Email and Anti-Spam*, Mountain View, CA, 2004.
7. T. Kohonen. *Self-organizing maps.* Spinger-Verlag, Berlin, Germany, 1995.
8. T. Mitchell. *Machine Learning.* McGraw-Hill, Boston, MA, 1997.
9. K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An Introduction to Kernel-Based Learning Algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, March 2001.
10. E. Pampalk, A. Rauber, and D. Merkl. Using smoothed data histograms for cluster visualization in self-organizing maps. In *Proc. Int'l Conf. Artificial Neural Networks*, Madrid, Spain, 2002.
11. F. Peng and D. Schuurmans. Combining naive Bayes and n-gram language models for text classification. In *Proc. European Conf. Information Retrieval Research*, pages 335–350, Pisa, Italy, 2003.
12. J. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, 1999.

13. F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
14. I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools with Java implementations.* Morgan Kaufmann, San Francisco, 2000.
15. Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proc. Int'l ACM SIGIR Conf. R&D in Information Retrieval*, Berkeley, CA, 1999.
16. Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proc. Int'l Conf. Machine Learning*, Nashville, TN, 1997.

Weighted Evidence Accumulation Clustering

F.Jorge F. Duarte[1], Ana L.N.Fred[2], André Lourenço[2] and M. Fátima C. Rodrigues[1]

[1]Departamento de Engenharia Informática, Instituto Superior de Engenharia do Porto, Instituto Superior Politécnico, Portugal
GECAD – Grupo de Investigação em Engenharia do Conhecimento e Apoio à Decisão
`{jduarte,fr}@dei.isep.ipp.pt`
[2]Instituto de Telecomunicações , Instituto Superior Técnico, Lisboa, Portugal
`{afred,arlourenco}@lx.it.pt`

Keywords. Clustering, Combining Multiple Partitions, Weighting Cluster Ensembles, Validity Indices

Abstract. We explore evidence accumulation (EAC) for combining clustering ensembles. According to EAC, a voting mechanism, where each partition has an identical weight in the combination process, is used to combine *N* partitions into a co-association matrix. This matrix is constructed based on co-occurrences of pairs of patterns in the same cluster. A final data partition is obtained by applying a clustering algorithm over this co-association matrix. In this paper we propose the idea of weighting the partitions differently (WEAC). Depending on the quality of the partitions, measured by internal and relative validity indices, each partition contributes differently in a weighted co-association matrix. We propose two ways of weighting each partition: SWEAC, using a single validation index, and JWEAC, using a committee of indices. The new approach is evaluated experimentally on synthetic and real data sets, in comparison with the EAC technique and the graph-based combination methods by Strehl and Gosh, leading in general to better results.

1 Introduction

The aim of clustering is to organize patterns into clusters so that patterns within a cluster are more similar to each other than are patterns belonging to different clusters. Even though there are hundred of clustering algorithms in the literature [1-3], no single algorithm can effectively find by itself all types of cluster shapes and structures. With the objective to solve this limitation, some combination clustering ensemble approaches have been proposed [4-8, 23-28] based on the idea of combining the results of a clustering ensemble into a final data partition.

The evidence accumulation clustering (EAC) method, by Fred and Jain, considers each clustering result as an independent evidence of data organization, and combines a clustering ensemble into a single combined data partition using a voting mechanism. This voting mechanism produces a mapping of N clusterings into a new similarity measure between n patterns, summarized in an $n \times n$ co-association matrix:

$$Co\_assoc(i, j) = votes_{ij} / N$$

where $votes_{ij}$ is the number of times the pattern pair (i,j) is assigned to the same cluster among the N clusterings. The final combined data partition (P^*) is obtained by applying a clustering algorithm to the co-association matrix. The final number of clusters can be fixed or automatically chosen using lifetime criteria [5-6].

Strehl and Ghosh see the cluster ensemble problem as an optimization problem based on the maximal average mutual information between the combined data partition and the clustering ensemble, exploring graph theoretical concepts. The clustering ensemble is mapped into a hypergraph, where vertices correspond to samples, and partitions are represented as hyperedges. They presented three heuristics to solve this problem: the hypergraph-partition algorithm (HGPA) cut a minimum number of hyperedges in the hypergraph using HMETIS algorithm with the objective of obtaining unconnected components of approximately the same size; the meta clustering algorithm (MCLA) applies a graph-based clustering to hyperedges in the hypergraph representation with the purpose of reducing the number of hyperedges; the cluster-based similarity partitioning algorithm (CSPA) is similar to the EAC approach, producing a similarity co-association matrix from the hyperedges representation of the partitions, and the final partition is obtained by applying the METIS algorithm to this similarity matrix.

In this paper we introduce a new approach (WEAC), based on the work by Fred et al. [4-6] on evidence accumulation clustering. WEAC consists of a weighted voting mechanism on the clustering ensemble, leading to a weighted co-association matrix (*w_co_assoc* matrix). We explore two different ways to weight each clustering to be incorporated in the *w_co_assoc* matrix. In the first method, the Single Weighted EAC (SWEAC), each clustering is evaluated by a relative or internal cluster validity index and the contribution of each clustering is weighted by the value obtained for this index. In the second method, the Joint Weighted EAC (JWEAC), each clustering is evaluated by a set of relative and internal cluster validity indices and the contribution of each clustering is weighted by all results obtained with each of these indices. For comparison, we used in experiments two internal indices and fourteen relative indices. The final combined partition is obtained by clustering the obtained *w_co_assoc* matrix.

The proposed WEAC approach is evaluated experimentally in this paper, on a comparative study with the EAC, HGPA, MCLA and CSPA methods.

Section 2 summarizes the cluster validity indices used in WEAC. Section 3 presents the proposed Weighted Evidence Accumulation Clustering (WEAC) and the experimental setup used. In section 4 a variety of synthetic and real data sets are used to evaluate the performance of WEAC. Finally, in section 5 we present the conclusions.

2 Cluster Validity Indices

How many clusters are present in the data and how good is the clustering itself are two important questions that have to be addressed in any clustering. Cluster validity indices provide the formal mechanisms to give an answer to these questions. For an overview of cluster validity measures and comparatives studies see for instance [9,10] and the references therein.

We can consider three approaches to assess cluster validity [11]: external validity indices, where the results of a clustering algorithm are evaluated based on a pre-specified structure that is assumed on the data set and reflects our intuition about the clustering structure of the data set (ground truth); internal validity indices, where we evaluate the clustering results in terms of quantities that involve the data representations themselves, and; relative validity indices, where a clustering structure is evaluated by comparing it to other clustering results, produced by the same algorithm but with different input parameters.

In this paper we make use of a set of internal and relative clustering validity indices, extensively used and referenced in the literature, to assess the quality of data partitions; external validity criteria is excluded, since it requires the use of a priori information about cluster structure. The two internal indices used are: the Hubert Statistic and Normalized Hubert Statistic (NormHub) [12]. The fourteen relative indices considered are: Dunn index [13], Davies-Bouldin index (DB) [14], Root-mean-square standard error (RMSSDT) [15], R-squared index (RS) [15], the SD validity index [10], the S_Dbw validity index [10], Caliski & Cooper cluster validity index (CH) [16], Silhouette statistic (S) [17], index I [18], XB cluster validity index, [19], Squared Error index (SE), Krzanowski & Lai (KL) cluster validity index [20], Hartigan cluster validity index (H) [21] and the Point Symmetry index (PS) [22].

3 Weighted Evidence Accumulation Clustering (WEAC)

WEAC is an extension of the EAC paradigm by weighting the influence of each data partition of the clustering ensemble in the combination process, based on the quality of these partitions, as assessed by cluster validity indices. In a simple voting mechanism a set of bad clusterings can overshadow another isolated good clustering, thus leading to poor clustering results. We expect to obtain better combination results by weighting the partitions in a weighted co-association matrix according to a measure of cluster validity, by giving higher relevance to better partitions in the clustering ensemble.

Given a clustering ensemble $P = \left\{P^1, P^2, \ldots, P^N\right\}$ with N partitions of n objects (patterns), and a corresponding set of normalized indices with values in the interval [0,1] measuring the quality of each of these partitions, the clustering ensemble is mapped into a weighted co-association matrix:

$$w\_co\_assoc(i,j) = \sum_{L=1}^{N} \frac{vote_{Lij} . VI^L}{N},$$

where N is the number of clusterings, $vote_{Lij}$ is a binary value, 1 or 0, depending if the object pair (i,j) has co-occurred in the same cluster (or not) in the Lth partition, and VI^L is the normalized cluster validity index value for the Lth partition. The combined data partition is obtained by applying a clustering algorithm to the weighted co-association matrix. The proposed WEAC method is schematically described in table 1. In WEAC we used two different ways of weighting each data partition:

1. Single Weighted EAC (SWEAC): in this method, the quality of each data partition is assessed by a single normalized relative or internal cluster validity index, and each vote in the *w_co_assoc* matrix is weighted by the value of this index:
$$VI^L = norm\_validity\left(P^L\right)$$
2. Joint Weighted EAC (JWEAC): in this method, the quality of each data partition is assessed by a set of relative and internal cluster validity indices, each vote in the *w_co_assoc* matrix being weighted by the overall contributions of these indices:
$$VI^L = \sum_{ind=1}^{NInd} \frac{norm\_validity_{ind}\left(P^L\right)}{NInd}$$
where $NInd$ is the number of cluster validity indices used, and $norm\_validity_{ind}\left(P^L\right)$ is the value of the indth validity index over the partition P^L.

In our experiments, we used sixteen cluster validity indices, as presented in section 2.

Table 1. WEAC approach

```
Input:
$P = \{P^1, P^2, \ldots, P^N\}$ - Clustering Ensemble with $N$ data partitions
$VI = \{VI^1, VI^2, \ldots, VI^N\}$ - Normalized Cluster Validity Index values of the corresponding data partitions
$n$ – number of data patterns
Output: Combined data partitioning.
Initialization: set $w\_co\_assoc$ to a null $n \times n$ matrix.
1. For $L$=1 to $N$
   Update the $w\_co\_assoc$: for each pattern pair $(i,j)$ in the same cluster, set
   $w\_co\_assoc(i,j) = w\_co\_assoc(i,j) + \frac{vote_{Lij} . VI^L}{N}$
$vote_{Lij}$ - binary value (1 or 0), depending if the object pair $(i,j)$ has co-occurred in the
        same cluster (or not) in the $L^{th}$ partition
2. Detect consistent clusters in the weighted co-association matrix using a clustering algorithm
```

3.2 Experimental Setup

3.2.1 Construction of Clustering Ensemble

We can use several different approaches to construct clustering ensembles, such as: applying different clustering algorithms; using the same clustering algorithm with different parameter values/initializations; clustering different views/features of the data; using different preprocessing and/or feature extraction mechanisms; perturbing the data set using techniques such as bootstrapping or boosting. In [5], clustering ensembles were generated by random initialization of the K-means algorithm. In this paper, besides the K-means algorithm (KM), we also explore other clustering methods to construct clustering ensembles: Single Link (SL), Complete-Link (CL), Average-Link (AL) and Clarans (CLR). We study the effect of combining clusterings produced by a single algorithm with different initializations and/or parameters values and the effect of combining clusterings produced by different clustering algorithms with different initializations and/or parameters values. Specifically, each clustering algorithm uses different values of k and K-means and Clarans additionally use different initializations of clusters centers. We explore also a clustering ensemble including all the partitions produced by all the clusterings algorithms (ALL). Considering k_{min} and k_{max} the minimum and maximum initial number of clusters, the procedure used to produce partitions is as follows:

For K-means and Clarans clustering algorithms:

```
1. Do $N$ times
  1.1. Randomly select $k$ in the interval $[k_{min},k_{max}]$ and $k$ clusters centers.
```

1.2. Run the algorithm with the above k and random initialization to produce a partition.

For SL, CL and AL clustering algorithms:

1. Do $k = k_{min}$ to k_{max}
   1.1. Run the algorithm with the above k to produce a partition.

3.2.2 Normalization of Cluster Validity Indices

Some indices are intrinsically normalized but others are not. For some of them the best result is the highest and for others the lowest value. For the indices of the first type, when the index only has values greater than zero, the normalization is made by dividing the value obtained for the index by the maximum value obtained over all partitions (*index_value=value_obtained/Maximum_value*). For indices of the second type, when the index only has values greater than zero, the normalization is made by dividing the minimum value obtained over all partitions by the partition value obtained for the index. (*index_value= Minimum_value/value_obtained*). The Normalized Hubert Statistic and Silhouette index are intrinsically normalized between [-1,1] but we only consider values between [0,1]. Some other indices increase (or decrease) as the number of clusters increase and it is not possible to find neither the maximum nor the minimum. In these cases, we search for the value of k at which a significant local change in the value of the index occurs. This change appears as a "knee" in the plot and is an indication of the number of clusters underlying the data set. Table 2 presents the criteria to obtain the best value with each validity index.

Table 2. Criteria to obtain the best value according to each validity index

Index	Criteria	Index	Criteria	Index	Criteria	Index	Criteria
Hubert	"Knee"	RMSSDT	"Knee"	CH	Max	SE	"Knee"
NormHub	Max	RS	"Knee"	S	Max	KL	Maximum
Dunn	Max	SD	Min	I	Max	H	Smallest k≥1: H(k)=10
DB	Min	S_Dbw	Min	XB	Min	PS	Minimum

Usually the highest (or lowest) value obtained in an index based on the "knee" is not the best value for that index. Therefore, this kind of indices can't be integrated directly in the *w_co_assoc* matrix. The best value of the index is where the "knee" is identified. The value 1 is assigned to the clustering associated to the "knee" in this index. The method we follow to incorporate the indices based on the "knee" in the co-association matrix was the following: running each of the clustering algorithms (SL, CL, AL, CLR and KM), varying the number of clusters to be obtained between [1, $k_{maximum}$] where $k_{maximum}$ is the maximum number of clusters we believe to exist in the data set; then, in each algorithm, we have to compare the clustering associated to the "knee" with each of the other clusterings produced by this algorithm. We used an external index, the Consistency index (C_i), proposed in [1] to compare these clusterings; $C_i(P, P_{knee})$ where P is the clustering we want to validate and P_{knee} the clustering associated to the knee. Consistency index is defined as the fraction of shared samples in matching clusters of two clusterings. The Consistency index is equal to the percentage of correct labelling when data partitions have the same number of clusters. Consider two clusterings with an arbitrary number of clusters and with the samples enumerated and referenced using the same labels in every clustering, s_i, $i=1,\ldots,n$. Each cluster has an equivalent binary valued vector representation, each position indicating the truth value of the proposition: *sample i belongs to the cluster*. The following notation is used:

$P_i \equiv$ clustering $i : (nc_i,\ C_1^i \ldots C_{nc_i}^i)$

$nc_i \equiv$ number of clusters in clustering i

$C_j^i = \{s_l : s_l \in$ cluster j of clustering $i\} \equiv$ list of samples in the j^{th} cluster of clustering i

$X_j^i : X_j^i(k) = \begin{cases} 1 & if\_s_k \in C_j^i \\ 0 & otherwise \end{cases}, k=1,\ldots,n \equiv$ binary valued vector representation of cluster C_j^i

The Consistency index (Ci) is defined in [1] as: $C_i = \frac{1}{n} \sum_{i=1}^{\min\{nc_1, nc_2\}} n\_shared_i$

where it is assumed that clusters occupy the same position in the ordered clusters lists of the clusterings, and n_shared_i is the number of samples shared for the i^{th} clusters.

We did this procedure to Hubert Statistic, RMSSDT index, RS index and Squared Error index. In Hartigan cluster validity index the estimated number of clusters is the smallest $k = 1$ such that $H(k)= 10$. Since Hartigan index is not calculated for values of k greater than the estimated number of clusters (usually obtained negative values) we have to apply to this index the same procedure applied to the indices based on the "knee" to obtain an index value for clusterings with k's greater than the estimated number of clusters.

3.2.3 Extraction of the Final Data Partition

The obtained co-association matrix represents a new similarity matrix between patterns to which a clustering algorithm must be applied in order to extract the combined data partition. We tested the SL, CL, AL and WR algorithms in the final extraction phase of P^*. In the results shown next, we assumed the final number of clusters known. To evaluate the performance of the combination methods, we compare the combined data partitions with ground truth information, obtained from known labeling of the data. We used the Consistency index described in [1] to compare these clusterings.

4 Experimental Results

4.1 Data sets

Synthetic data sets For simplicity of visualization we considered 2-dimensional patterns. These data sets were produced aiming the evaluation of the performance of WEAC in a multiplicity of conditions, like distinct data sparseness in the feature space, arbitrary shaped clusters, well separated and touching clusters. Figure 1 plots these data sets.

The Bars data set has 2 classes (200 and 200) and the density of the patterns increasing with increasing horizontal coordinate. The Cigar data set has 4 classes (100, 100, 25 and 25). The Half Rings data set is composed by 3 uniformly distributed classes (150, 150 and 200) within half-ring envelops. The Rings data set consists of 500 samples organized in 4 classes (25, 75, 150 and 250). The Spiral data set consists of 200 samples divided evenly in 2 classes.

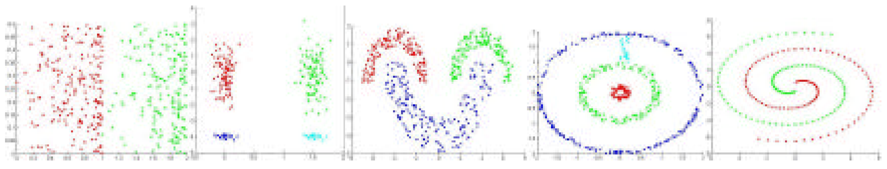

(a) Bars (b) Cigar (c) Half Rings (d) Rings (e) Spirall

Fig. 1. Synthetic Data Sets

Real Data Sets Four real-life data sets were considered to show the performance of the WEAC: Breast Cancer, Iris, DNA microarrays and Handwritten Digits. The Breast Cancer data set (http://www.ics.uci.edu/~mlearn/MLRepository.html) has 683 samples (9 features) spitted in two classes: Benign and Malignant. The Iris data set is divided in three types of Iris plants (50 samples per class), characterized by 4 features, and with one class well separated from the other two, which are intermingled. The Yeast Cell data set (DNA microarrays) consists of the fluctuations of the gene expression levels of over 6000 genes over two cell cycles. The available data set is restricted to the 384 genes with 17 features (http://staff.washington.edu/kayee/model/) whose expression level peak at different time points corresponding to the 5 phases of the cell cycle. It was used the logarithm of the expression level (Log Yeast) and a "standardized" version (Std Yeast) of the data (with mean 0 and variance 1). The Handwritten Digits, is available at the UCI repository (http://www.ics.uci.edu/~mlearn/MLRepository.html), and consists in 3823 samples, each with 64 features. A subset (Optical) composed by the first 100 samples of all the digits was used from a total of 3823 training samples (64 features).

4.2 Combination of Clustering Ensembles using WEAC

The quality of the combined data partition, $P*$, obtained with the WEAC method is evaluated by computing the consistency of $P*$ with ground truth information P^0, using Ci($P*$,P^0). We assume that the true number of clusters is known, being the number of clusters in $P*$.

Tables 3-12 show the values of $C_i(P*,P^0)$ over the experiments with both synthetic (Bars, Cigar, Half Rings, Rings and Spiral) and real data (Breast Cancer, Iris, Std Yeast, Log Yeast and Optical). In these tables, rows are grouped by the clustering ensembles construction method. Inside each clustering ensemble construction method appears the three clustering methods used to extract the final combined partition. K-means and Clarans based clustering ensembles have N=200 clusterings each, obtained with k randomly chosen in the set {10,…,30}. SL, CL and AL based clustering ensembles have N=21 data partitions, each corresponding to a different number of clusters, k, in the set {10,…,30}. ALL gather the partitions produced by all the methods, with N=463.

Analyzing the tables 3-12, we can conclude that we achieve in general better results with both versions of WEAC when comparing with EAC. In JWEAC we can find many situations where the results are the same as those of EAC, some other situations where the JWEAC results outperform EAC's and in fewer situations the JWEAC results are worse than EAC's. The SWEAC results of each cluster index are in many situations equal to the EAC results, in other situations the EAC results are improved with the SWEAC approach and in fewer situations the EAC results are better than those of SWEAC.

Concerning the clustering ensemble construction methods, we can see that in 7 out of the 10 data sets used, the partitions produced by the k-means clustering algorithm, provide the better results in the EAC. In the JWEAC approach the same happened in 6 data sets. So, we can conclude that k-means algorithm is a good option to produce cluster ensembles for these approaches.

Table 3. Breast Cancer

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
SL	SL	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15
	AL	65.15	66.33	66.33	65.15	65.15	66.33	66.33	65.45	66.33	65.15	66.33	66.33	66.33	66.33	66.33	66.33	66.33	65.15
	WR	**68.08**	**68.08**	**68.08**	**68.08**	**68.08**	**68.08**	**68.08**	**68.08**	**68.23**	**66.76**	**66.47**	**68.23**	**68.08**	**66.76**	**68.08**	**68.08**	**68.08**	**68.08**
AL	SL	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81
	AL	68.81	71.01	94.88	94.88	94.88	94.88	94.88	68.81	68.81	94.88	**94.88**	94.88	94.88	94.88	68.81	94.88	90.19	74.52
	WR	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**	**94.88**	**96.49**	**94.88**	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**	**96.49**
CL	SL	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81	68.81
	AL	68.81	66.76	**96.63**	66.76	73.94	**96.63**	**96.63**	**96.63**	**96.63**	**96.63**	68.37	**96.63**	**96.63**	76.72	**96.63**	**96.63**	**96.63**	**96.63**
	WR	**96.05**	**96.63**	**96.63**	**96.05**	**96.05**	**96.63**	**96.63**	96.05	**96.63**	96.05	**96.63**	96.05	**96.63**	**96.05**	96.05	**96.63**	**96.63**	96.05
KM	SL	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57	64.57
	AL	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**	**97.07**
	WR	61.20	61.20	68.67	61.20	61.20	68.67	68.67	59.74	68.67	61.20	68.67	61.20	68.67	61.20	61.20	68.67	61.20	61.20
CLR	SL	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15
	AL	**96.05**	**96.05**	**94.58**	**94.58**	**96.05**	**94.58**	**94.58**	**95.75**	**95.90**	**95.75**	**95.90**	**94.58**	**94.58**	**96.05**	**96.05**	**94.58**	**96.05**	**96.05**
	WR	48.32	47.00	47.00	47.00	48.32	47.00	47.00	48.32	47.00	46.27	45.83	48.32	47.00	48.32	48.32	47.00	47.00	48.32
ALL	SL	65.15	65.15	65.15	65.89	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15	65.15
	AL	93.85	65.89	65.89	94.88	65.89	65.89	65.89	65.89	94.00	94.44	94.14	65.74	65.89	65.15	66.03	65.89	65.89	65.74
	WR	**96.93**	**94.00**	**94.73**	**96.34**	**94.73**	**94.73**	**94.73**	**97.07**	**96.93**	**96.34**	**96.78**	**96.78**	**94.73**	**94.29**	**94.88**	**94.00**	**94.00**	**96.34**

Table 4. Iris

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
SL	SL	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33
	AL	67,33	67,33	67,33	34,67	67,33	67,33	67,33	34,67	34,67	64,67	67,33	34,67	67,33	68,00	68,00	67,33	34,67	64,67
	WR	**70,00**	**70,00**	**70,67**	**70,00**	**91,33**	**70,67**	**70,67**	**91,33**	**70,67**	**90,00**	**70,67**	**70,00**	**70,67**	**70,00**	**70,67**	**70,67**	**70,67**	**70,67**
AL	SL	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33	69,33
	AL	69.33	69.33	**78.00**	48.00	38.67	**78.00**	**78.00**	48.67	40.00	**79.33**	65.33	40.00	**78.00**	48.67	**78.00**	69.33	69.33	72.00
	WR	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**77,33**	**78,00**	78,00	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**
CL	SL	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00
	AL	50,00	50,00	47,33	63,33	45,33	47,33	47,33	52,00	50,00	50,00	59,33	47,33	47,33	40,00	47,33	50,00	47,33	36,67
	WR	**67,33**	**70,67**	**70,67**	**70,67**	**67,33**	**70,67**	**70,67**	**61,33**	**70,67**	**67,33**	**70,67**	**70,67**	**70,67**	**67,33**	**70,67**	**70,67**	**70,67**	**70,67**
KM	SL	74.67	74.67	74.67	74.67	74.67	74.67	74.67	69.33	74.67	74.67	74.67	69.33	74.67	69.33	74.67	74.67	74.67	69.33
	AL	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**
	WR	**90,67**	**90,67**	**90,67**	**90,67**	84,00	**90,67**	**90,67**	84,00	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**
CLR	SL	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00	68,00
	AL	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**	**90,67**
	WR	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00	46,00
ALL	SL	67,33	67,33	67,33	67,33	67,33	67,33	67,33	68,00	67,33	74,67	67,33	67,33	67,33	67,33	67,33	67,33	67,33	67,33
	AL	**90,67**	69.33	69.33	89.33	69.33	69.33	69.33	**96,00**	90.00	**90,67**	**90,67**	**90,67**	70.00	69.33	90.00	69.33	69.33	69.33
	WR	**90,67**	**90,67**	**90,67**	**90,67**	**96,67**	**90,67**	**90,67**	**96,00**	**97,33**	**90,67**	**90,67**	**90,67**	**96,67**	**96,00**	**90,67**	**96,00**	**90,67**	**94,00**

Table 5. Rings

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
SL	SL	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00	44,00
	AL	44,00	**81,80**	55,00	59,00	42,00	55,00	55,00	**61,00**	43,40	44,00	43,40	47,40	55,00	44,00	58,80	53,60	46,20	53,60
	WR	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**	**61,00**	**59,80**	**74,60**	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**	**59,80**
AL	SL	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20	36,20
	AL	36.20	36.20	38.40	35.60	41.80	38.40	38.40	38.00	35.20	43.40	38.00	36.80	38.40	39.00	38.40	41.80	38.00	36.80
	WR	**77,40**	**77,40**	**60,20**	**77,40**	**85,60**	**60,20**	**60,20**	**74,20**	**60,20**	**85,60**	**77,40**	**85,60**	**60,20**	**77,40**	**77,40**	**60,20**	**60,20**	**77,40**
CL	SL	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00	41,00
	AL	41,00	39,40	42,20	36,60	34,20	42,20	42,20	40,60	40,60	61,80	49,80	43,20	42,20	56,60	40,60	42,20	48,40	37,20
	WR	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**	**63,20**
KM	SL	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**78,00**	**79,80**	**78,00**	**78,00**	**78,00**	**85,40**	**78,00**	**78,00**	**85,40**	**78,00**	**78,00**	**79,80**
	AL	44,60	71,60	43,60	51,00	44,80	43,60	43,60	43,60	43,60	43,60	44,60	43,60	43,60	43,60	43,60	43,60	43,60	44,60
	WR	57,40	59,40	54,00	57,40	59,40	54,00	54,00	62,60	54,00	58,20	59,40	57,60	54,00	58,20	54,20	54,00	54,00	59,40
CLR	SL	48,00	52,60	52,60	48,00	45,00	52,60	52,60	48,00	52,60	44,80	50,60	50,60	52,60	52,60	52,60	52,60	52,60	50,60
	AL	**56,60**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,60**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**	**56,40**
	WR	51,60	51,40	51,40	51,40	51,40	51,40	51,40	51,60	51,40	51,00	49,80	51,40	51,40	51,40	51,40	51,40	51,40	51,40
ALL	SL	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**80,20**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**	**73,80**
	AL	54.60	43.40	44.80	44.80	43.40	61.00	61.00	47.60	47.60	50.00	45.20	44.40	61.00	47.60	48.40	43.40	43.40	48.40
	WR	56,20	71,80	71,80	71,80	61,80	71,80	71,80	61,80	71,80	58,40	71,00	66,20	71,80	61,80	71,00	71,00	71,80	71,00

Table 6. Bars

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
SL	SL	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50	51,50
	AL	51,50	**94,25**	**95,25**	54,25	**95,75**	**95,25**	**95,25**	**94,25**	50,50	51,50	54,00	51,50	**95,25**	50,50	54,25	51,50	**95,75**	**94,25**
	WR	**94,00**	94,00	94,00	**94,00**	94,00	94,00	94,00	**94,25**	**94,00**	**95,75**	**95,75**	**94,00**	94,00	**94,00**	**94,00**	**94,00**	94,00	94,00
AL	SL	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75	55,75
	AL	55.75	60.25	58.75	58.75	**66.25**	58.75	58.75	51.00	66.25	58.75	58.75	58.75	58.75	58.75	72.75	51.00	55.75	71.00
	WR	**76,00**	**76,00**	**76,00**	**76,00**	64,25	**76,00**	**76,00**	**64,25**	**76,00**	**76,00**	**76,00**	**76,00**	**76,00**	**76,00**	**76,00**	**76,00**	**76,00**	**76,00**
CL	SL	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50	56,50
	AL	56,50	56,50	56,25	56,50	54,75	56,25	56,25	56,50	62,00	56,50	**71,50**	**74,75**	56,25	61,25	61,25	56,50	51,50	63,00
	WR	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	64,00	64,00	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**	**64,00**
KM	SL	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	52.75	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**
	AL	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**
	WR	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**
CLR	SL	**51,75**	**51,75**	**51,75**	**51,75**	**51,75**	**51,75**	**51,75**	**51,75**	**51,75**	**50,25**	50,25	50,25	**51,75**	**51,75**	**51,75**	**51,75**	**51,75**	50,25
	AL	50,75	50,75	50,75	50,75	50,75	50,75	50,75	50,75	50,75	**50,25**	50,75	50,75	50,75	50,75	50,75	50,75	50,75	50,75
	WR	51,00	51,00	51,00	51,00	51,00	51,00	51,00	51,00	51,00	**50,25**	**51,00**	**51,00**	51,00	51,00	51,00	51,00	51,00	**51,00**
ALL	SL	51,50	**98,75**	54,75	50,75	50,75	54,75	54,75	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	54,75	50,75	**98,75**	**98,75**	**98,75**	**98,75**
	AL	**96,25**	**98,75**	98.75	**98,75**	**98,75**	98.75	98.75	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**	98.75	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**
	WR	80,00	**98,75**	**99,50**	**98,75**	94,00	**99,50**	**99,50**	**98,75**	**98,75**	64,25	**98,75**	64,25	**99,50**	**98,75**	**98,75**	**98,75**	**98,75**	**98,75**

Table 7. Cigar

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
SL	SL	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80	40,80
	AL	40,80	97,20	77,20	40,80	40,80	77,20	77,20	39,60	86,00	51,60	77,20	40,80	77,20	40,40	86,80	40,80	40,80	78,00
	WR	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**78,40**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**98,00**	**94,40**
AL	SL	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40	58,40
	AL	58.40	47.20	58.40	47.20	58.40	46.40	46.40	58.40	58.40	58.40	58.40	44.80	46.40	41.60	58.40	46.40	46.40	44.80
	WR	**62,00**	**62,00**	**62,00**	**62,00**	**62,40**	**62,00**	**62,00**	**72,40**	**62,00**	**62,00**	**62,00**	**62,00**	**62,00**	**62,00**	**62,00**	**62,00**	**62,00**	**62,00**
CL	SL	56,40	56,40	56,40	56,40	56,40	56,40	56,40	56,40	**56,40**	56,40	56,40	**56,40**	56,40	56,40	**56,40**	**56,40**	56,40	56,40
	AL	56,40	50,00	**66,40**	56,00	69,20	**66,40**	**66,40**	56,40	**56,40**	**66,40**	64,00	40,00	**66,40**	**66,40**	42,80	53,60	**66,40**	**72,80**
	WR	**76,40**	**66,40**	45,60	**66,40**	**72,00**	45,60	45,60	**72,00**	45,60	**66,40**	**66,40**	45,60	45,60	**66,40**	45,60	45,60	45,60	45,60
KM	SL	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**
	AL	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80	70,80
	WR	71,60	71,60	71,60	71,60	70,80	71,60	71,60	**100,00**	71,60	71,60	70,80	71,60	71,60	71,60	71,60	71,60	71,60	71,60
CLR	SL	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**
	AL	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**	**49,60**
	WR	43,20	43,20	43,20	43,20	43,20	43,20	43,20	43,20	43,20	43,20	43,20	46,80	43,20	43,20	43,20	43,20	43,20	43,20
ALL	SL	**56,00**	**88,40**	**88,40**	**88,40**	88,40	50,80	50,80	**100,00**	**88,40**	**100,00**	**100,00**	**100,00**	**88,40**	**100,00**	**88,40**	50,80	**88,40**	**100,00**
	AL	43.20	**88,40**	**88,40**	71.60	88.40	**87,60**	**87,60**	88.40	52.00	71.60	70.40	83.20	**88,40**	88.40	71.60	**54,80**	71.60	87.60
	WR	43,20	71,60	77,60	60,00	**95,20**	75,20	75,20	84,80	43,20	71,60	84,80	84,80	61,20	**100,00**	68,40	44,00	60,00	**100,00**

Table 8. Half Rings

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
	SL	65,80	65,80	65,80	65,80	65,80	65,80	65,80	65,80	65,80	**39,60**	65,80	65,80	65,80	65,80	65,80	65,80	65,80	65,80
SL	AL	65,80	**100,00**	**100,00**	65,80	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**39,60**	**100,00**	34,40	**100,00**	65,80	**100,00**	65,20	**100,00**	65,80
	WR	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**39,60**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**
	SL	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40	48,40
AL	AL	48.40	48.40	52.80	**73.60**	**49.60**	61.40	61.40	48.40	48.40	**60.20**	61.80	61.60	61.40	48.80	48.40	52.80	52.80	39.20
	WR	**51,80**	**64,60**	**64,60**	64,60	49,20	**64,60**	**64,60**	**56,60**	**64,60**	51,80	**64,60**	**64,60**	**64,60**	**51,80**	**64,60**	**64,60**	**64,60**	**64,60**
	SL	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**	**50,20**
CL	AL	**50,20**	47,60	37,40	45,40	44,80	37,40	37,40	45,80	37,40	37,40	47,60	37,40	37,40	37,40	45,80	44,40	44,40	**50,20**
	WR	**50,20**	**50,20**	48,80	48,80	**50,20**	48,80	48,80	**50,20**	48,80	**50,20**	**50,20**	**50,20**	48,80	**50,20**	48,80	48,80	48,80	48,80
	SL	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	69.80	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**	**99,80**
KM	AL	95,00	95,00	95,00	95,00	95,00	94,60	94,60	86,40	94,60	95,00	95,00	86,00	94,60	95,00	94,60	95,00	95,00	95,00
	WR	77,40	77,40	95,00	95,00	77,40	95,00	95,00	77,80	**95,00**	77,40	77,40	77,40	95,00	77,40	77,40	95,00	95,00	77,40
	SL	54,80	54,80	54,80	54,80	54,80	54,80	54,80	54,80	54,80	**39,60**	54,80	54,80	54,80	54,80	54,80	54,80	54,80	54,80
CLR	AL	**55,00**	55,40	55,40	55,40	55,40	55,40	55,40	**55,00**	55,40	**39,60**	**55,40**	**55,40**	55,40	55,40	55,40	55,40	55,40	55,40
	WR	53,00	**59,20**	**59,40**	**59,20**	**59,20**	**59,40**	**59,40**	51,20	**59,40**	**39,60**	54,00	54,00	**59,40**	**59,20**	**59,40**	**59,40**	**59,40**	**59,40**
	SL	64,60	**95,00**	**99,80**	**95,00**	95,00	68,60	68,60	95,00	**95,00**	**95,00**	**95,00**	**95,00**	68,60	**95,00**	**95,00**	**100,00**	**99,80**	**95,00**
ALL	AL	**99,80**	**95,00**	95.00	**95,00**	95.00	94.80	94.80	99.80	**95,00**	**95,00**	**95,00**	**95,00**	94.80	**95,00**	**95,00**	91.40	95.00	**95,00**
	WR	71,80	**95,00**	95,20	**95,00**	**100,00**	**100,00**	**100,00**	**100,00**	**95,00**	**95,00**	**95,00**	77,40	**100,00**	**95,00**	**95,00**	99,80	95,00	**95,00**

Table 9. Log Yeast

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
	SL	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,68**
SL	AL	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	**35,42**	35,42
	WR	35,16	35,16	35,16	35,16	35,16	35,16	35,16	**35,42**	35,16	**35,42**	34,90	35,16	35,16	**35,42**	35,16	35,16	35,16	34,90
	SL	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21	30,21
AL	AL	30.21	**32.29**	**30.47**	**36.46**	27.60	**30.47**	**30.47**	26.82	**35.68**	30.47	**30.47**	**32.03**	**30.47**	**32.81**	**32.29**	**32.29**	**30.47**	**32.29**
	WR	**30,99**	30,99	26,30	30,99	**30,99**	26,30	26,30	**31,25**	26,30	**30,99**	26,30	30,99	26,30	30,99	30,99	26,30	26,30	30,99
	SL	**39,58**	**39,58**	**39,58**	**39,58**	39,58	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**	**39,58**
CL	AL	**39,58**	31,51	32,29	34,38	**40,89**	32,29	32,29	31,51	31,51	28,91	26,56	27,60	32,29	31,51	32,03	33,85	32,29	35,42
	WR	31,25	32,29	31,51	32,29	32,55	31,51	31,51	32,55	31,51	31,25	31,51	32,55	31,51	31,25	31,25	31,51	31,51	31,25
	SL	33.85	34.37	34.37	34.37	33.85	34.37	34.37	33.85	34.90	34.37	34.90	33.85	34.37	33.85	33.85	34.37	34.37	33.85
KM	AL	**41,41**	**41,41**	**41,41**	**41,41**	**41,41**	**41,41**	**41,41**	**43,49**	**43,49**	**41,41**	**43,49**	**41,41**	**41,41**	**41,41**	**41,41**	**41,41**	**41,41**	**41,41**
	WR	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59	33,59
	SL	36,72	36,72	36,46	36,46	36,72	36,46	36,46	36,72	**36,46**	36,46	**36,46**	36,46	36,46	36,72	36,46	36,46	36,46	36,72
CLR	AL	**36,98**	**37,24**	**37,24**	**37,24**	**38,54**	**37,24**	**37,24**	**36,98**	35,68	**37,24**	35,68	**36,98**	**37,24**	**37,24**	**37,24**	**37,24**	**37,24**	**43,49**
	WR	34,64	34,64	35,68	35,16	34,64	35,68	35,68	34,37	35,16	35,16	35,42	35,68	35,68	35,16	35,68	35,68	35,68	35,16
	SL	38,54	36,72	35,68	38,54	36,98	35,68	35,68	35,68	36,46	38,54	36,46	36,20	35,68	**35,16**	35,68	36,72	35,42	35,42
ALL	AL	**40,63**	**40,63**	**40,63**	**40,63**	**40,36**	**40,63**	**40,63**	**40,36**	**40,89**	**40,63**	**40,89**	**40,63**	**40,63**	34.38	**40,63**	**40,63**	**40,63**	**39,84**
	WR	36,46	36,20	33,07	36,20	32,29	33,07	33,07	34,11	36,20	36,20	32,55	35,94	33,07	33,33	35,94	35,94	35,94	34,11

Table 10. Optical

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
	SL	10,60	10,60	10,60	10,60	10,60	10,60	10,60	10,60	10,60	**10,10**	10,60	10,60	10,60	10,60	10,60	10,60	10,60	10,60
SL	AL	10,60	10,60	10,60	10,60	10,60	10,60	10,60	10,60	10,60	**10,10**	10,60	10,60	10,60	10,60	10,60	10,60	10,60	10,60
	WR	**30,50**	**30,50**	**30,50**	**30,60**	**30,50**	**30,50**	**30,50**	**30,60**	**30,50**	**10,10**	**30,50**	**30,50**	**30,50**	**30,60**	**30,50**	**30,50**	**30,50**	**30,70**
	SL	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70	75,70
AL	AL	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70	75.70
	WR	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**	**84,80**
	SL	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80
CL	AL	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80	51,80
	WR	**56,50**	**57,00**	**57,00**	**57,00**	**56,50**	**57,00**	**57,00**	**57,30**	**57,00**	**57,30**	**53,70**	**57,00**	**57,00**	**57,30**	**56,50**	**53,70**	**53,70**	**57,00**
	SL	30.20	30.20	30.20	30.20	30.20	30.20	30.20	40.00	30.30	30.20	30.30	30.20	30.20	30.20	30.20	30.20	30.20	30.20
KM	AL	78,80	78,80	78,60	78,60	78,80	78,80	78,80	78,90	78,40	78,60	78,50	78,80	78,80	78,80	78,80	78,80	78,80	78,80
	WR	**80,40**	**80,20**	**80,40**	**80,20**	**80,20**	**80,40**	**80,40**	**80,20**	**80,00**	**80,40**	**80,70**	**80,20**	**80,20**	**80,20**	**80,20**	**80,20**	**80,20**	**80,20**
	SL	20,30	20,30	20,30	20,30	20,30	20,30	20,30	20,30	20,20	20,30	20,20	20,30	20,30	20,30	20,30	20,20	20,30	20,30
CLR	AL	79,00	79,00	79,10	79,00	79,00	78,80	78,80	81,50	78,70	78,90	81,10	79,20	78,70	79,00	81,60	**81,40**	78,90	81,50
	WR	**87,40**	**82,80**	**80,30**	**90,20**	**87,40**	**80,30**	**80,30**	**87,40**	**80,20**	**90,10**	**81,50**	**81,90**	**81,40**	**87,40**	**83,30**	80,20	**79,90**	**90,20**
	SL	40,00	30,30	30,30	40,00	40,10	30,30	30,30	30,30	30,20	30,00	30,20	30,30	30,30	30,30	30,30	30,30	30,30	30,30
ALL	AL	**79,40**	**79,20**	79.20	**79,40**	**79,00**	**81,40**	**81,40**	68.60	**79,40**	**79,60**	**81,20**	**79,10**	**79,20**	77.60	**80,90**	62.60	69.60	**80,90**
	WR	78,60	78,70	**79,30**	78,90	78,90	78,60	78,60	**78,60**	77,60	79,30	79,00	78,90	78,70	**78,70**	78,90	**79,50**	**77,60**	78,90

Table 11. Spiral

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
	SL	50,50	50,50	50,50	50,50	50,50	50,50	50,50	50,50	50,50	**50,50**	50,50	50,50	50,50	50,50	50,50	50,50	50,50	50,50
SL	AL	50,50	**98,00**	50,50	**98,00**	**98,00**	50,50	50,50	**98,00**	**98,00**	**50,50**	**98,00**	**98,00**	50,50	**98,00**	**98,00**	50,50	50,50	50,50
	WR	**96,50**	96,50	**98,00**	96,50	96,50	**98,00**	**98,00**	96,00	**98,00**	**50,50**	**98,00**	96,50	**98,00**	96,50	96,50	**98,00**	**98,00**	**98,00**
	SL	**52,00**	**52,00**	**52,00**	**52,00**	**52,00**	**52,00**	**52,00**	52,00	**52,00**	52,00	52,00	52,00	**52,00**	**52,00**	52,00	**52,00**	52,00	52,00
AL	AL	**52.00**	**52.00**	50.00	**52.00**	50.00	50.00	50.00	**58.00**	**52.00**	**58.00**	**58.00**	**58.00**	50.00	**52.00**	**58.00**	**52.00**	**58.00**	**58.00**
	WR	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00
	SL	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**	**54,00**
CL	AL	**54,00**	**54,00**	**54,00**	50,00	50,00	**54,00**	**54,00**	**54,00**	**54,00**	50,00	50,00	50,00	**54,00**	50,00	50,00	52,00	**54,00**	**54,00**
	WR	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00	50,00
	SL	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**
KM	AL	51,50	50,00	51,50	51,50	51,50	51,50	51,50	55,00	53,00	50,00	50,50	51,50	51,50	50,00	52,50	51,50	51,50	51,50
	WR	54,00	52,00	52,00	54,00	58,50	52,00	52,00	51,00	55,00	50,50	55,00	55,00	52,00	50,50	54,00	58,50	58,50	55,50
	SL	51,00	51,00	51,00	51,00	51,00	51,00	51,00	51,00	51,00	55,00	51,00	**55,00**	51,00	51,00	51,00	51,00	51,00	51,00
CLR	AL	**56,00**	**56,00**	**56,00**	**56,00**	**56,00**	**56,00**	**56,00**	**56,00**	**56,00**	**57,00**	**55,00**	**55,00**	**56,00**	**56,00**	**55,00**	**56,00**	**56,00**	**55,00**
	WR	55,00	55,00	55,00	55,00	55,00	55,00	55,00	55,00	55,00	54,50	53,50	53,50	55,00	55,00	**55,00**	55,00	55,00	53,50
	SL	**68,00**	**90,00**	60,00	**68,00**	**90,00**	**68,00**	**56,00**	**100,00**	**68,00**	**100,00**	**100,00**	**74,00**	**68,00**	**100,00**	**80,00**	60,00	68,00	**100,00**
ALL	AL	51.00	51.50	56.00	50.50	52.00	52.00	52.00	52.00	56.50	52.00	52.00	52.00	52.00	84.00	52.00	**100,00**	**100,00**	52.00
	WR	52,50	50,00	**75,50**	50,00	70,00	51,50	51,50	50,00	53,50	50,00	50,00	50,00	51,50	74,00	50,00	96,00	96,00	50,00

Table 12. Std Yeast

		EAC	JWEAC	Hubert	NormHub	Dunn	RMSSDT	RS	S_dbw	CH	S	I	XB	SE	DB	SD	H	KL	PS
SL	SL	35,68	35,68	35,68	35,68	35,68	35,68	35,68	35,68	35,68	**35,42**	35,68	35,68	35,68	35,68	35,68	35,68	35,68	35,68
	AL	35,68	36,46	36,46	35,68	35,68	36,46	36,46	36,46	**37,24**	**35,42**	36,46	35,68	36,46	36,46	35,68	35,68	35,68	35,68
	WR	**36,98**	**36,98**	**36,98**	**36,98**	**36,98**	**36,98**	**36,98**	**37,24**	36,98	**35,42**	**36,98**	**36,98**	**36,98**	**37,24**	**36,98**	**36,98**	**36,98**	**36,98**
AL	SL	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29	57,29
	AL	57.29	65.36	65.10	65.36	55.73	65.10	65.10	57.29	57.29	57.29	57.29	65.89	65.10	35.94	66.41	57.29	65.10	47.14
	WR	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**	**69,53**	**69,79**	**69,53**	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**	**69,79**
CL	SL	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64	34,64
	AL	34,64	**51,04**	40,63	34,64	45,83	40,63	40,63	33,33	45,05	47,40	44,79	**56,25**	40,63	45,57	44,53	34,38	40,63	42,45
	WR	**48.18**	48.18	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**	48.18	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**	**48.18**
KM	SL	35.94	35.94	35.94	35.94	35.94	35.94	35.94	36.98	48.70	35.94	48.96	35.94	35.94	35.94	35.94	35.94	35.94	35.94
	AL	**69,01**	**69,01**	**69,01**	**68,49**	**67,97**	**69,01**	**69,01**	**67,97**	**69,01**	**69,01**	**69,01**	**67,45**	**69,01**	**69,01**	**67,97**	**69,01**	**69,01**	**69,01**
	WR	57,29	58,07	58,33	58,07	56,77	58,33	58,33	57,03	52,60	58,07	52,08	56,51	56,51	57,29	56,51	56,77	58,33	56,51
CLR	SL	49,48	57,81	54,17	57,81	58,07	54,17	54,17	49,48	54,17	54,17	54,69	49,48	54,17	49,48	49,48	54,17	54,17	49,48
	AL	**61,72**	**61,72**	**61,98**	**61,72**	**61,46**	**61,72**	**61,72**	**61,46**	**61,98**	**61,72**	**62,24**	**61,72**	**61,72**	**61,72**	**61,72**	**61,72**	**61,72**	**61,72**
	WR	54,69	54,95	53,39	54,95	56,51	53,39	53,39	52,60	50,26	54,95	53,39	55,21	53,39	56,51	55,21	58,85	53,39	55,21
ALL	SL	36,46	36,20	36,20	36,46	36,20	36,20	36,20	36,20	49,48	36,46	49,22	36,20	36,20	36,20	36,20	36,20	36,20	36,20
	AL	**68,49**	**68,49**	**68,49**	**68,49**	**67,97**	**68,23**	**68,23**	45.57	**68,23**	**68,23**	**68,23**	**67,19**	**68,23**	36.72	**68,49**	45.05	**68,23**	36.72
	WR	55,99	59,38	57,03	55,99	58,59	57,29	57,29	**57,81**	54,69	58,33	54,69	59,11	57,81	**60,16**	61,98	**60,42**	57,81	**57,29**

Table 13 presents the results obtained by the three Strehl combination heuristics for all data sets. The results of MCLA heuristic for the ALL clustering ensemble are not presented due to computational problems related with the high number of data partitions present in this clustering ensemble. The best results are achieved almost in all situations with CSPA and MCLA methods.

Table 13. Values obtained by the three heuristics of Strehl combination method

		Spiral	Log Yeast	Std Yeast	Optical	Cigar	Breast	Iris	Halfrings	Bars	Rings
SL	CSPA	96	**36,72**	**34,11**	37,5	70,8	55,78	90	91,6	**99**	70
	HGPA	51	22,66	21,09	10,7	51,6	50,07	60	54	52	37,6
	MCLA	**98**	35,42	30,73	**31,6**	**86,8**	**67,94**	**68**	**93,4**	98	**75,6**
AL	CSPA	56	**29,95**	61,98	82	57,6	82,72	62,67	**65,8**	89,5	**53,4**
	HGPA	52	25,78	41,93	21,6	53,6	52,12	52	51,4	51	39,6
	MCLA	**62**	27,08	**70,57**	**84,1**	**69,6**	**96,49**	**64**	42	**97,75**	52,8
CL	CSPA	**52**	34,9	45,83	59,5	47,2	83,89	49,33	54,4	**62,5**	47
	HGPA	**52**	31.51	39.58	39.9	**59,2**	53.29	**71,33**	50.6	51.5	34.8
	MCLA	**52**	**37,76**	**53,13**	**61,6**	52,4	**96,63**	70,67	**57,8**	61	**44,2**
KM	CSPA	**54,5**	33,85	57,29	84,2	70,4	84,63	**98**	**93,4**	97,75	45,2
	HGPA	53,5	**38,28**	55,99	75,8	**72,8**	82,43	97,33	89,2	98	67,2
	MCLA	54	33,07	**59,11**	**89**	61,2	**84,77**	**98**	92,8	**98,75**	**70,4**
CLR	CSPA	**58,5**	32,03	57,55	**83,4**	**36,8**	**76,43**	**96,67**	**63,6**	**51,5**	**51,4**
	HGPA	57	32,29	55,73	72	**36,8**	81,41	96	63,4	51,75	48
	MCLA	52,5	**32,55**	**58,59**	75,1	36,4	81,41	**96,67**	63	51,75	47,8
ALL	CSPA	51,5	**34,37**	**55,99**	**84,5**	**56,4**	**83,31**	**98**	**93,2**	**98,25**	42
	HGPA	**52**	33,33	52,86	75,9	36,8	86,24	97,33	59,4	51,25	**49**

Table 14 presents best individual results produced by each clustering method (lines SL to KM) and best combined results per combination strategy (lines EAC to MCLA). As shown, almost in all data sets the SWEAC results outperform the single application of all the clustering algorithms and the SWEAC results are always better or equal of EAC results. In the Optical, Log Yeast and Rings data sets, the superiority of EAC and JWEAC and even more that of SWEAC is particularly evident. In Cigar and Half Rings data sets, both the EAC and WEAC approaches obtain 100%, which are much better results than the ones obtained by other algorithms.

To compare the influence of the different cluster validity indices in SWEAC results, we first calculated, for each data set, the improvement between all the values obtained in each index and the corresponding values obtained with the EAC approach. Next, the average of those improvements was calculated for each index in each data set.

Table 14. Best single and combined results

	Spiral	Log Yeast	Std Yeast	Optical	Cigar	Breast	Iris	Halfrings	Bars	Rings
SL	100	34.9	36.2	10.6	60.4	65.15	68	95	50.25	58.8
CL	52	28.91	66.67	51.8	55.6	92.83	84	72	98.75	36.8
AL	52	28.65	65.89	75.7	87.2	94.29	90.67	73.4	98.75	34
CLR	58	30.99	57.55	73.86	82.4	95.9	89.33	77.4	97	44.4
KM	52.5	29.43	64.06	67.71	63	96.49	89.33	75.6	98	38.8
EAC	100	41.41	69.79	87.4	100	97.07	90.67	100	98.75	78
SWEAC	100	43.49	69.79	90.2	100	97.07	97.33	100	99.5	85.6
JWEAC	100	41.41	69.79	84.8	100	97.07	90.67	100	98.75	81.8
CSPA	96	36.72	61.98	84.5	70.8	84.63	98	93.4	99	70
HGPA	57	38.28	55.99	75.9	72.8	86.24	97.33	89.2	98	67.2
MCLA	98	37.76	70.57	89	86.8	96.63	98	93.4	98.75	75.6

Table 15 presents the average of all the previous values of each index obtained for each data set. In WEAC (both the SWEAC and JWEAC approaches), we achieved better average results than with EAC, by weighting the clusterings in the *w_co_assoc* matrix with the obtained indices values. The average improvement obtained with the cluster validity indices in all data sets was of 3,25%. The same happened with the JWEAC approach where the average improvement was of 5,35%, a value much better than the average improvement obtained in the SWEAC approach. These results show that the SWEAC and JWEAC approaches increase the quality of the combined data partitions when compared with the EAC approach.

In the SWEAC approach, none of the cluster validation indices performed systematically better than the others. Different validation indices achieved the best Ci SWEAC results, depending of the data set, but overall they performed better than EAC in average. However, as we can see in table 16, in 9 of the 10 data sets used, the Normalized Hubert Statistic (NormHub), in average, improves the results of EAC. Only in Iris data set this doesn't happen. It should also be highlighted that in all used data sets the best Ci result using NormHub (SWEAC approach) is as good as the best EAC Ci result or even better than it. In fact, 1 result is better (Optical) and the other 9 are equal to Ci EAC results (table 17). Therefore, based on these two facts, we can conclude that choosing NormHub index in the SWEAC approach is a good choice to obtain good results.

Table 15. Average percentual increase in the performance of JWEAC and SWEAC as compared to EAC, over all data sets

JWEAC	Hubert	NormHub	Dunn	RMSSTD	RS	S_dbw	CH	S	I	VXB	SE	DB	SD	H	KL	PS
5,35	3,78	2,76	3,83	3,05	2,96	3,57	3,00	1,78	5,05	2,41	3,28	2,81	4,20	1,97	3,54	3,92

Table 16. Average percentual increase in the performance of SWEAC approach using NormHub index when compared to EAC

Spiral	Log Yeast	Std Yeast	Optical	Cigar	Breast	Iris	Halfrings	Bars	Rings
4,49	0,73	1,78	0,24	7,2	1,79	-2,37	9,68	1,96	2,51

Table 17. Ci results of the SWEAC approach using NormHub and of the EAC approach, in all data sets

	Spiral	Log Yeast	Std Yeast	Optical	Cigar	Breast	Iris	Halfrings	Bars	Rings
EAC	100	41,41	69,79	87,4	100	97,07	90,67	100	98,75	78
S_Dbw	100	41,41	69,79	90,2	100	97,07	90,67	100	98,75	78

Table 18 presents the number of times EAC, SWEAC and JWEAC approaches obtained better, worse and equal values than Strehl approach. Each of these values is relative to a clustering ensembles construction method (six in each approach). EAC, SWEAC and JWEAC achieved in more data sets better results than Strehl. We can also see that JWEAC and even more SWEAC obtain a greater number of cases of better results than Strehl. Considering the sum of all data sets, all above three approaches present also better results than Strehl.

Doing the same type of comparison between SWEAC and JWEAC (table 19) the results by data set are almost equivalent, however considering the sum of all data sets, JWEAC obtained a higher number of times better results than SWEAC. If in SWEAC, instead of considering the best value obtained, we consider the average of all the values, we can see (table 20) that JWEAC also gets a better performance than SWEAC. This shows that JWEAC is a robust approach to incorporate all the indices in the weighting of the data partitions in the co-association matrix. Table 21 shows the same type of comparison between the EAC, the SWEAC and the JWEAC approaches. Both SWEAC and JWEA C obtain in general better results than EAC.

Table 18. Number of better, worse and equal values obtained comparing EAC, SWEAC and JWEAC with Strehl

Strhel	EAC			SWEAC			JWEAC		
	Better	Worse	Equal	Better	Worse	Equal	Better	Worse	Equal
Spiral	3	3	0	3	2	1	3	2	1
Log Yeast	5	1	0	5	1	0	5	1	0
Std Yeast	4	2	0	5	1	0	4	2	0
Optical	2	4	0	2	4	0	1	5	0
Cigar	4	2	0	6	0	0	5	1	0
Breastcancer	4	1	1	4	0	2	4	0	2
Iris	1	5	0	2	4	0	1	5	0
Halfrings	3	3	0	4	2	0	3	3	0
Bars	1	3	2	2	2	2	2	2	2
Rings	5	1	0	5	1	0	6	0	0
	32	25	3	38	17	5	34	21	5

Table 19. Number of better, worse and equal values obtained comparing SWEAC with JWEAC

JWEAC	SWEAC		
	Better	Worse	Equal
Spiral	71	45	172
Log Yeast	47	87	154
Std Yeast	30	105	153
Optical	49	46	193
Cigar	42	60	186
Breastcancer	63	38	187
Iris	42	35	211
Halfrings	42	60	186
Bars	23	45	220
Rings	53	75	160
	462	596	1822

Table 20. Number of better, worse and equal values obtained comparing JWEAC with SWEAC (AVG)

JWEAC	SWEAC (AVG)		
	Better	Worse	Equal
Spiral	6	6	6
Log Yeast	6	8	4
Std Yeast	3	12	3
Optical	6	7	5
Cigar	6	5	7
Breastcancer	7	5	6
Iris	4	7	7
Halfrings	4	12	2
Bars	2	9	7
Rings	7	7	4
	51	78	51

Table 21. Number of better, worse and equal values obtained comparing SWEAC and JWEAC with EAC

EAC	SWEAC			JWEAC		
	Better	Worse	Equal	Better	Worse	Equal
Spiral	66	47	175	3	3	12
Log Yeast	73	76	139	4	3	11
Std Yeast	77	61	150	7	1	10
Optical	48	65	175	2	4	12
Cigar	67	38	183	4	3	11
Breastcancer	62	55	171	3	4	11
Iris	47	44	197	1	1	16
Halfrings	104	59	125	6	2	10
Bars	76	24	188	5	0	13
Rings	85	84	119	5	4	9
	705	553	1622	40	25	115

5 Conclusions

In this paper we present a new approach (WEAC) that explores and extends the idea of EAC, proposing the weighting of multiple clusterings by internal and relative validity indices. The K-means, Clarans, SL, CL and AL algorithms are used to produce clustering ensembles. We employ two different ways to combine the clustering ensembles: using only the clusterings produced by a single algorithm with different initializations and/or parameters values; and using clusterings produced by different clustering algorithms with different initializations and/or parameters values. Using a voting mechanism, the multiple clusterings are weighted in the SWEAC version by an internal or relative index to be integrated in a *w_co_assoc* matrix; in the JWEAC version all internal and relative indices contribute to weight each partition. The final partition is obtained by clustering the *w_co_assoc* matrix using the SL, CL, AL or WR algorithms. Experimental results with both synthetic and real data show that these approaches lead in general to better results than the EAC and Strehl methods. The evaluation of results is based on a consistency index between the combined partition and the ideal data partition taken as ground truth.

These preliminary results show that the association of weighting mechanisms with cluster combination techniques is a promising tool, worth of further investigation.

References

[1] A.k. Jain and R.C. Dubes, *Algorithms for Clustering Data,* Prentice Hall, 1988.

[2] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data clustering: A review", *ACM Computing Surveys*, 31(3):264-323, September 1999.

[3] J. Han, M. Kamber, *Data Mining- Concepts and Techniques,* Morgan Kaufmann Publishers, 2001.

[4] A. Fred, "Finding consistent clusters in data partitions", *Multiple Classifier Systems,* Vol. LNCS 2096, Josef Kittler and Fabio Roli, editors, pp. 309-318, Springer, 2001.

[5] A. Fred, A.K. Jain, "Evidence accumulation clustering based on the k-means algorithm", *S.S.S.P.R.,* Vol. LNCS 2396, T.Caelli et al., editor, Springer-Verlag, 2002, pp. 442–451.

[6] A. Fred and A.K. Jain, "Combining Multiple Clusterings using Evidence Accumulation", *IEEE Transactions on Pattern analysis and Machine Intelligence,* Vol.27, No.6, June 2005, pp. 835-850.

[7] A. Strehl and J. Ghosh. "Cluster ensembles - a knowledge reuse framework for combining multiple partitions", *Journal of Machine Learning Research 3*, 2002.

[8] A. Topchy, A.K. Jain, and W. Punch, "Combining Multiple Weak Clusterings", IEEE Intl. Conf. on Data Mining, 2003, Melbourne Florida, pp. 331-338.

[9] M. Meila and D. Heckerman, "An Experimental Comparison of Several Clustering and Initialization Methods", Proc. 14th Conf. Uncertainty in Artificial Intelligence, p.p. 386-395, 1998.

[10] M. Halkidi, Y. Batistakis, M. Vazirgiannis, "*Clustering algorithms and validity measures*", Tutorial paper in the proceedings of the SSDBM 2001 Conference.

[11] S. Theodorodis, K. Koutroubas, *Pattern Recognition Academic Press*, 1999.

[12] L.J. Hubert, J. Schultz, “Quadratic assignment as a general data-analysis strategy”, *British Journal of Mathematical and Statistical Psychology*, Vol. 29, 1975, pp. 190-241.

[13] J.C. Dunn, “Well separated clusters and optimal fuzzy partitions”, *J. Cybern*, Vol. 4, 1974, pp. 95-104.

[14] D.L. Davies, D.W. Bouldin, “A cluster separation measure”, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 1, No2, 1979.

[15] S.C. Sharma, *Applied Multivariate Techniques*, John Willwy & Sons, 1996.

[16] R.B.Calinski, J.Harabasz, “A dendrite method for cluster analysis”, *Communications in statistics 3*, 1974, pp. 1-27.

[17] L. Kaufman, P. Roussesseeuw, *Finding groups in data: an introduction to cluster analysis*, New York, Wiley, 1990.

[18] U. Maulik, Bandyopadhyay, “Genetic Algorithm Based Clustering Technique”, *Pattern Recognition*, Vol. 33, pages, 2000, pp. 1455-1465.

[19] X.L. Xie, G. Beni, “A Validity Measure for Fuzzy Clustering”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 13, pages 841-847, 1991.

[20] W. Krazanowski, Y. Lai, “A criterion for determining the number of groups in a dataset using sum of squares clustering”, Biometrics, 1985, pp. 23-34.

[21] J.A. Hartigan, “Statistical theory in clustering”, J. Classification, 1985, 63-76.

[22] C.H. Chou, M.C. Su, E. Lai, “A new cluster validity measure and its application to image compression”, Pattern Analysis and Applications, Vol. 7, 2004, pp. 205-220.

[23] S.T. Hadjitodorov, L. I. Kuncheva, L. P. Todorova, Moderate Diversity for Better Cluster Ensembles, Information Fusion, 2005, accepted

[24] X.Z. Fern, C.E. Broadley, “Random projection for high dimensional data clustering: a cluster ensemble approach”, 20th International Conference on Machine Learning, ICML;Washington, DC, 2003, pp. 186-193.

[25] S Monti; P. Tamayo; J. Mesirov; T. Golub, ”Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data”, Machine learning, 52, 2003, pp. 91-118.

[26] A. Topchy, B. Minaei-Bidgoli, A.K. Jain, W. Punch, “Adaptive Clustering Ensembles”, Proc. Intl. Conf on Pattern Recognition, ICPR’04, Cambridge, UK, 2004, pp. 272-275.

[27] B. Minaei-Bidgoli, A. Topchy, W. Punch, “Ensembles of Partitions via Data Resampling”, Proc. IEEE Intl. Conf. on Information Technology: Coding and Computing, ITCC04, vol. 2, April 2004, pp. 188-192.

[28] E. Dimitriadou, A. Weingessel, K. Hornik, “Voting-Merging: An Ensemble Method for Clustering”, Artificial Neural Networks – ICANN, August 2001.

Predicting Foreign Exchange Rate Return Directions with Support Vector Machines

Christian Ullrich
Institute AIFB
University of Karlsruhe
D-76128 Karlsruhe, Germany
BMW Group
D-80788 Munich, Germany
christian.ullrich@bmw.de

Detlef Seese
Institute AIFB
University of Karlsruhe
D-76128 Karlsruhe, Germany
seese@aifb.uni-karlsruhe.de

Stephan Chalup
School of Electrical Engineering & Computer Science
University of Newcastle
Callaghan, NSW 2308, Australia
chalup@cs.newcastle.edu.au

Abstract. Forecasting financial time series is an important and complex problem in machine learning and statistics. This paper examines and analyzes the general ability of Support Vector Machine (SVM) models to correctly predict and trade daily EUR/GBP, EUR/JPY and EUR/USD exchange rate return directions. For this purpose, six SVM models with varying standard kernels along with one exotic p-Gaussian SVM are compared to investigate the separability of Granger-caused input data in high dimensional feature space. To ascertain their potential value as out-of-sample forecasting and quantitative trading tool, all SVM models are benchmarked against traditional forecasting techniques. We find that hyperbolic SVMs consistently perform well in terms of forecasting accuracy and in terms of trading performance via a simulated strategy. Moreover, it is found that p-Gaussian SVMs perform reasonably well in predicting EUR/GBP and EUR/USD return directions.

Keywords. Financial time series, foreign exchange rate, support vector machine, kernels.

Predicting Foreign Exchange Rate Return Directions with Support Vector Machines

Abstract. Forecasting financial time series is an important and complex problem in machine learning and statistics. This paper examines and analyzes the general ability of Support Vector Machine (SVM) models to correctly predict and trade daily EUR/GBP, EUR/JPY and EUR/USD exchange rate return directions. For this purpose, six SVM models with varying standard kernels along with one exotic p-Gaussian SVM are compared to investigate the separability of Granger-caused input data in high dimensional feature space. To ascertain their potential value as out-of-sample forecasting and quantitative trading tool, all SVM models are benchmarked against traditional forecasting techniques. We find that hyperbolic SVMs consistently perform well in terms of forecasting accuracy and in terms of trading performance via a simulated strategy. Moreover, it is found that p-Gaussian SVMs perform reasonably well in predicting EUR/GBP and EUR/USD return directions but not EUR/JPY.

Keywords. Financial time series, foreign exchange rate, support vector machine, kernels.

1. Introduction

Over the past several decades, researchers have used various forecasting methods to study time series events. For example, the 1960s saw the development of a number of large macroeconometric models purporting to describe the economy using hundreds of macroeconomic variables and equations. Although complicated linear models can track the data very well over the historical period, they often perform poorly for out-of-sample forecasting ([37]). This has often been interpreted that the explanatory power of exchange rate models is extremely poor. Nelson ([40]) discovered that univariate ARMA models with small values for *p* and *q* produce more robust results than the big models. Box and Jenkins ([5]) developed the autoregressive integrated moving average (ARIMA) methodology for forecasting time series events. The basic idea of ARIMA modeling approaches is the assumption of linearity among the variables. However, there are many time series events for which the assumption of linearity may not hold. Clearly, ARIMA models cannot be effectively used to capture and explain nonlinear relationships. When ARIMA models are applied to processes that are nonlinear, forecasting errors often increase greatly as the forecasting horizon becomes longer. To improve forecasting nonlinear time series events, researchers have developed alternative modeling approaches. These include nonlinear regression models, the bilinear model ([17]), the threshold autoregressive model ([53]), and the autoregressive heteroscedastic model (ARCH) by Engle ([13]). Although these methods have shown improvement over linear models for some specific cases, they tend to be application specific, lack generality, and are often harder to implement ([58]).

An alternative strategy is for the computer to attempt to learn the input/output functionality from examples, which is generally referred to as supervised learning. During the last decade, the application of artificial neural networks (ANN) as supervised learning methods has exploded in a variety of areas. ANN is a general-purpose model that has been used as a universal function approximator ([22]). Researchers have used the ANN methodology to forecast many nonlinear time series events ([21], [51], [59]).

Apart from that, ANNs have been used to develop prediction algorithms for financial asset prices, such as technical trading rules for stocks and commodities ([14], [31], [32], [48], [50], [55], [56]). The effectiveness of ANNs and their performance in comparison to traditional forecasting methods has also been a subject of many studies ([10], [56]). ANNs have proven to be comprehensive and powerful for modeling nonlinear dependencies in financial markets ([46]), notably for exchange rates ([3], [12], [33], [39]). However, ANN models have been criticized because of their black-box nature, excessive training times, danger of overfitting, and the large number of parameters required for training. As a result deciding on the appropriate network involves much trial and error. These shortcomings paired with the logic that complex real-world problems require more sophisticated solutions than a single network led to idea of combining ANNs with other technologies to hybrid and modular solutions ([1]). For a survey of the application of ANN to forecasting problems in general see [57] and [58].

Support Vector Machines ([4], [54]) are a new kind of supervised learning system that map the input dataset via kernel into a high dimensional feature space in order to enable non-linear data classification and regression. SVM has proven to be a principled and very powerful method that in the few years since its introduction has already outperformed many other systems in a variety of applications, such as text categorisation ([26]), image processing ([43], [44]), hand-written digit recognition ([34]) and bioinformatic problems, for example protein homology detection ([25]) and gene expression ([7]). Subsequent applications in time series prediction ([38]) further indicated the potential that SVMs have with respect to the economic and financial audience. In the special case of predicting Australian foreign exchange rates, [28] showed that moving average-trained SVMs have advantages over an ANN based model which was shown to have advantages over ARIMA models ([27]). Kamruzzaman and Sarker [29] had a closer look at SVM regression and investigated how they perform with different standard kernel functions. It was found that Gaussian radial basis and polynomial kernels appeared to be a better choice in forecasting the Australian forex market than linear or spline kernels. However, although Gaussian kernels are adequate measures of similarity when the representation dimension of the space remains small, they fail to reach their goal in high dimensional spaces ([15]).

The task in this paper is twofold. We examine the general ability of SVMs to correctly classify daily EUR exchange rate returns. Indeed, it is more useful for traders and risk managers to forecast exchange rate fluctuations rather than their levels. To predict that the level of the EUR/USD, for instance, is close to the level today is trivial. On the contrary, to determine if the market will rise or fall is much more complex and interesting. Since SVM performance depends to the most extent on choosing the right kernel, we empirically verify the use of customized p-Gaussians by comparing them with a range of standard kernels.

The remainder is organized as follows: in the next section, we conduct statistical analyses of EUR/GBP, EUR/JPY and EUR/USD time series. Section 3 outlines the procedure for obtaining an explanatory input dataset. In section 4, we formulate the SVM as applied to exchange rate forecasting and present the kernels used. Section 5 describes the benchmarks and metrics used for model evaluation. Section 6 gives the results.

2. Exchange Rate Statistics

The purpose of this section is to examine the statistical properties of daily EUR/GBP, EUR/JPY and EUR/USD exchange rate data from 1 January 1997 to 31 August 2003. This is done for mainly two reasons. First, time series analysis gives an understanding on the degree of randomness inhibited in the chosen time interval. Non-randomness is an important indicator for the generation of meaningful forecasts and exists, if a time series does not consist of independent and identically distributed (i.i.d.) values. Second, statistical analysis provides a foundation for traditional ARIMA model building in order to identify benchmark models for the SVM methodology taken.

The investigation is based on London daily closing prices. The series for the period from 1997 to 1998 were constructed by using the fixed EUR/DEM conversion rate agreed in 1998, combined with the GBP/DEM, JPY/DEM and USD/DEM daily market rates. Note that we do not include year 2004 in our analysis since it will be needed for out-of-sample forecasting and is not known beforehand.

The results of the statistical inference procedure taken are depicted in Table 1. As a first step we ensured that the time series data we work with are stationary. Stationarity is a necessary property to apply for statistical standard concepts such as volatility and correlation. Informally, a series is said to be (weakly or covariance) stationary, if neither the mean nor the autocovariances depend on time ([20], p.45). The test results on the null of nonstationarity (ADF and PP) and stationarity (KPSS) are basically consistent. For level data we can assume nonstationarity despite the contradictory ADF result for the EUR/GBP series. Based on this finding, all series were transformed into stationary ones with regards to [5]. First differences of the price data were taken and the same tests as above were conducted subsequently. The test statistics suggest that now all three exchange rate series are strongly difference-stationary, i.e. integrated of order one (I(1)).

The Jarque-Bera test indicates that the hypothesis of normally distributed returns has to be rejected at a high level within our chosen time interval. The reason can be found in the excess kurtosis as compared to the normal distribution. Among the three series, EUR/JPY exhibits the most leptokurtic behaviour whereas EUR/USD shows weaker signs of fat tails.

A major objective when analysing stationary time series is to detect linear dependencies among the data through identifying an appropriate linear model. Univariate time series models can only be explained by their own lagged values, i.e. by autoregressive (AR) terms as explanatory variables in their representation. Furthermore, if the underlying process is stochastic and stationary, the errors can be linear combinations of white noise at different lags, so the moving average (MA) part of the model refers to the structure of the error term. The most general model for a stationary process is an integrated autoregressive moving average model ARIMA(*p*,*q*,*r*) with *p* autoregressive terms, *q* moving average terms and integration order *r*, with *r=1* in our case. ARIMA(*p*,*q*,*1*) models are also referred to as ARMA(*p*,*q*) models. Estimating *p* and *q* is commonly done by visual inspection of the autocorrelation function (ACF) and partial autocorrelation function (PACF) for MA models and low-order AR models ([20]). ACF and PACF functions characterize the pattern of temporal, linear dependence that is existent in the series. Since independent variables are always uncorrelated, testing for zero autocorrelation is equivalent to testing for linear independency. We calculate Ljung-Box (LB) Q-statistics ([36]) for the null hypothesis of linear independency among variables with up to 24 lags. We found that for all three series linear dependencies are not neglectable within reasonable bounds. To remove them, we specified linear models according to the following procedure: in order to account for stable regression coefficients that are significantly different from zero, tests on omitted and redundant variables were implemented. Once the possibly best model was found, its residuals were retested according to LB and the Breusch-Godfrey LM test, alternatively. We find that simple MA and ARMA models with low degrees of freedom provide the best results while preserving generalization ability for forecasting. Model selection is further confirmed by the Schwarz information criterion which imposes a larger penalty for additional AR(*p*) or MA(*q*) coefficients than the Akaike criterion ([19]).

Both, the LB Q-statistics and the Breusch-Godfrey LM statistics of the regression residuals indicate that serial dependencies have now disappeared at any lag. However, although linear independency can be inferred, non-linear dependencies might still exist.

We investigate the origin of non-normal behavior by focusing on the phenomenon of heteroskedastic processes. Heteroskedasticity is motivated by the observation that in many financial time series the magnitude of residuals appeared to be related to the magnitude of recent residuals ([13]). In order to detect these second-moment dependencies (conditional variances), we first calculated the autocorrelations and partial autocorrelations of the squared residuals and computed the LB Q-statistics for the corresponding lags. If squared residuals do not exhibit autoregressive conditional heteroskedasticity (ARCH), autocorrelations and partial autocorrelations should be zero at all lags and the Q-statistics should not be significant. The opposite holds at very high significance levels for EUR/GBP and EUR/JPY. The ARCH-LM testing result displayed in Table 1 confirms that ARMA residuals for EUR/GBP and EUR/JPY exhibit considerable amounts of heteroskedasticity: the null hypothesis of zero heteroskedasticity is clearly rejected for all selected lags at the 1% level. The result for EUR/USD is less clear: according to both, Q-statistics and ARCH-LM testing results, the hypothesis of a constant variance can only be rejected at higher lags and with slightly lower confidence. This brings us to an important result, which has also been reported in literature ([11], [49]): ARCH processes are leptokurtic, or “fat-tailed”, relative to the normal. The weaker test statistics for EUR/USD can be justified by a kurtosis that is not considerably higher than 3 and a skewness that is close to zero.

Lee, White and Granger ([35]) examine the performance of a range of tests on nonlinearity across a variety of data generating processes. They find that no single test dominates all the others. In the light of this finding, it is advisable to use more than one test. The tests for non-linearity that we apply are Ramsey's RESET-Test ([45]) and the BDS-Test ([6]), which has proven to be a particular successful instrument ([23], [24]). The Ramsey RESET-Test checks the null of a correctly specified linear model by adding a certain number n of higher order fitted terms. If the coefficients of these terms are significantly different from zero, it can be inferred that the linear model is not good enough due to existing additive nonlinearities. The BDS test for the null of an i.i.d. is suitable for proving the existence of nonlinearities in mean and nonlinearities in variance. This means that both the existence of additive and multiplicative nonlinearities in time series can be shown. The test statistic is asymptotically normally distributed, it was calculated by using the AR(1) and GARCH(1,1) residuals. For both tests, the results are corresponding. Whereas for EUR/GBP and EUR/JPY the null is rejected at noticeably high confidence levels, it cannot be rejected for EUR/USD. Note that up to this stage, the question whether non explainable nonlinearities have to be attributed to more refined nonlinear stochastic models or to chaotic ones remains open. For the EUR/USD series only few nonlinearities have been detected, indicating that linear model residuals are supposedly random. Still, it remains to be seen how well SVM models will be able to exploit nonlinearities and compare to linear benchmark models.

3. Data Selection

The procedure of obtaining an exploratory dataset can be divided into two phases ([42]): specifying and collecting a large amount of data at first, and then reducing the dimensionality of the dataset by selecting a subset of that data for efficient training (feature extraction). Since there is a trade-off between accuracy as represented by the entire dataset and the computational overheads of retaining all parameters without application of feature extraction/selection techniques, the data selection procedure is also referred to as the "curse of dimensionality" which was first noted by [2]. The merit of feature extraction is to avoid multicollinearity, a problem that is common to all sorts of regression models. If multicollinearity exists, explanatory variables have a high degree of correlation between themselves meaning that only a few important sources of information in the data are common to many variables. In this case it may not be possible to determine their individual effects.

3.1 Phase One

The obvious place to start selecting data, along with the EUR/GBP, EUR/JPY and EUR/USD is with the other leading traded exchange rates. In addition, we selected related financial market data, including stock market price indices, 3-month interest rates, 10-year government bond yields and spreads, the price of Brent Crude oil, and the prices of silver, gold and platinum. Due to the bullish commodity markets we also decided to include daily prices of assorted metals being traded on the London Metal Exchange, as well as agricultural commodities. Macroeconomic variables hardly play a role in daily FX movements and were disregarded. All data were obtained from Bloomberg.

All the series span a seven-year time period from 1 January 1997 to 31 December 2004, totaling 2349 trading days. The data is divided into two periods: the first period runs from 1 January 1997 to 31 August 2003 (1738 observations), is used for model estimation and is classified in-sample. The second period, from 1 September 2003 to 31 December 2004 (350 observations), is reserved for out-of-sample forecasting and evaluation. Missing observations on bank holidays were filled by linear interpolation.

3.2 Phase Two

Having collected an extensive list of candidate variables, the explanatory viability of each variable has been evaluated. The aim was to remove those input variables that do not contribute significantly to model performance. For this purpose, we took a two-step procedure.

First, pair-wise Granger Causality tests ([16]) with lagged values until k=20 were performed on stationary I(1) candidate variables. The Granger approach to the question of whether an independent variable x causes a dependent variable y is to see how much of the current y can be explained by past values of y and then to see whether adding lagged values of x can improve the explanation. Y is said to be Granger-caused by x if x helps in the prediction of y, or equivalently if the coefficients on the lagged x's are statistically significant. The major advantage of the Granger causality principle is that it is able to distinguish causation from correlation. Hence the known problem of spurious correlations can be avoided ([18]). We find that EUR/GBP is Granger-caused by 11 variables, namely

- EUR/USD, JPY/USD and EUR/CHF exchange rates,
- IBEX, MIB30, CAC and DJST stock market indices,
- the prices of platinum and nickel as well as
- 10-year Australian and Japanese government bond yields.

Further, we identify 10 variables that significantly Granger-cause EUR/JPY, namely

- EUR/CHF exchange rate
- IBEX stock market index
- the price of silver
- Australian 3-month interest rate
- Australian, German, Japanese, Swiss and US government bond yields along with
- UK bond spreads.

For EUR/USD, Granger causality tests yield 7 significant explanatory variables:

- AUD/USD exchange rate,
- SPX stock market index and
- the prices of copper, tin zinc, coffee and cocoa.

Second, we carried out linear principal component analysis (PCA) on Granger caused explanatory datasets in order to check for computational overheads. PCA is generally considered as a very efficient method for dealing with the problem of multicollinearity. It allows for reducing the dimensionality of the underlying dataset by excluding highly intercorrelated explanatory variables. This results in a meaningful input for the learning machine. Per cumulative R², which we required to be not lower than 0.99, significant multicollinearity could not be detected for any dependent variable. Consequently, the datasets were not reduced any further and all variables were kept.

4. SVM Classification Model and Kernels

4.1 SVM Classification Model

One of the major reasons for the rise to prominence of the SVM ([4], [54]) is its ability to cast nonlinear classification as a convex optimization problem. The basic idea is to project the input data via kernel into a more expressive, high dimensional feature space where the SVM algorithm finds the decision plane that has maximum distance from the nearest training patterns. Applying the so-called "kernel trick" ([47]) guarantees that linear classification in feature space is equal to nonlinear classification in input space.

In this paper, we focus on the task of predicting a rise (labeled "+1") or fall (labeled "-1") of daily EUR/GBP, EUR/JPY and EUR/USD exchange rate returns. To predict that the level of the EUR/USD, for instance, is close to the level today is trivial. On the contrary, to determine if the market will rise or fall is much more complex and interesting for a currency trader. We apply the C-Support Vector Classification (C-SVC) algorithm as described in ([9], [54]) and implemented in R packages "e1071" ([8]) and "kernlab" ([30]):

Given training vectors $x_i \in R^n$, $i = 1,...,l$, in two classes, and a vector $y \in R^l$ such that $y_i \in \{+1,-1\}$, C-SVC solves the following problem:

$$\min_{w,b,\xi} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \tag{1}$$

$$y_i \left(w^T \phi(x_i) + b \right) \geq 1 - \xi_i$$
$$\xi_i \geq 0, i = 1,...,l.$$

Its dual representation is

$$\min_{\alpha} \quad \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \tag{2}$$

$$0 \leq \alpha_i \leq C, i = 1,...,l,$$
$$y^T \alpha = 0$$

where *e* is the vector of all ones, *C* is the upper bound, *Q* is an *lxl* positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is the kernel, which maps training vectors x_i into a higher dimensional, inner product feature space by the function ϕ. The decision function is

$$f(x) = \operatorname{sign}\left(\sum_{i=1}^{l} y_i y_j K(x_i, x) + b\right). \qquad (3)$$

Training a SVM requires the solution of a very large quadratic programming optimization problem (QP) which is solved by using the Sequential Minimization Optimization (SMO) algorithm ([41]). SMO decomposes a large QP into a series of smallest possible QP problems which can be solved analytically. Hence time consuming numerical QP optimization as an inner loop can be avoided.

4.2 Kernel Functions

Ever, since the introduction of the SVM algorithm, the question of choosing the kernel has been considered as crucial. This is largely due to the effect that the performance highly depends on data preprocessing and much less on the linear classification algorithm to be used. However, how to efficiently find out which kernel is optimal for a given learning task is still a rather unexplored problem. Under this circumstance, the best we can do is to compare a range of kernels with regards to their effect on SVM performance. Standard kernels chosen in this paper include the following:

- Linear: $k(x, x') = \langle x, x' \rangle$
- Polynomial: $k(x, x') = \left(scale \cdot \langle x, x' \rangle + \text{offset}\right)^{\text{degree}}$
- Laplace: $k(x, x') = \exp\left(-\sigma \|x - x'\|\right)$
- Gaussian radial basis: $k(x, x') = \exp\left(-\sigma \|x - x'\|^2\right)$
- Hyperbolic: $k(x, x') = \tanh\left(scale \cdot \langle x, x' \rangle + \text{offset}\right)$
- Bessel: $k(x, x') = \dfrac{\text{Bessel}_{(v+1)}^{n}\left(\sigma \|x - x'\|\right)}{\left(\|x - x'\|^{-n(v+1)}\right)}$

In addition, we verify the use of customized p-Gaussian kernels $K(x_i, x_j) = \exp\left(-d(x_i, x)^p / \sigma^p\right)$, where p and σ are two parameters and $d(x_i, x) = \left(\sum_{i=1}^{n} |x_i - x|^2\right)^{1/2}$ defines the Euclidean distance between data points. Compared to the widely used RBF kernels, p-Gaussians include a supplementary degree of freedom in order to better adapt to the distribution of data in high-dimensional spaces ([15]). The two parameters p and σ depend on the specific input set for each exchange rate return time series. More specifically, we calculate p and σ as proposed in [15]:

$$p = \frac{\ln\left(\frac{\ln(0.05)}{\ln(0.95)}\right)}{\ln\left(\frac{dF}{dN}\right)}; \quad \sigma = \frac{d_F}{(-\ln(0.05))^{1/p}} = \frac{d_N}{(-\ln(0.95))^{1/p}} \tag{4}$$

In the case of EUR/USD, for instance, we are considering 1737 8-dimensional objects. Hence we calculate 1737x1737 distances and compute the 5% (d_N) and 95% (d_F) percentiles in that distribution.

In order to avoid the known problem of overfitting, we determine robust estimates for *C* and *scale* (σ) for every kernel through 20-fold cross validation.

5. Benchmarks and Evaluation Method

Letting y_t represent the exchange rate at time *t*, we forecast the variable

$$sign(\Delta y_{t+h}) = sign(y_{t+1} - y_t) \tag{5}$$

where $h = 1$ for a one-period forecast with daily data.

5.1 Naïve Model

The naive strategy assumes that the most recent period change is the best predictor of the future. The simplest model is defined by $sign(\hat{y}_{t+1}) = sign(y_t)$ where Δy_t is the actual rate of return at period t and Δy_{t+1} is the predicted rate of return for the next period.

5.2 ARMA(p,q) Model

An autoregressive moving average model with *p* autoregressive terms and *q* moving average terms ARMA(*p*, *q*) is a univariate time series model. Such a model can only be explained by its own lagged values, i.e. with autoregressive (AR) terms as explanatory variables in its representation. If the process is stochastic and stationary the errors can be linear combinations of white noise at different lags, so the moving average (MA) part of the model refers to the structure of the error term. ARMA(*p*, *q*) models are the most general family of models for representing stationary processes and are given by

$$y_t = c + \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + ... + \alpha_p y_{t-p} + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \beta_q \varepsilon_{t-q}\ , \tag{6}$$

where $\varepsilon_t \sim$ i.i.d. $(0, \sigma^2)$. For our analyses we use the model estimates from Section 2 as represented in Table 2, that is

- $y_t = c_t + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \beta_3 \varepsilon_{t-3}$ for the EUR/GBP series,
- $y_t = c_t + \varepsilon_t + \beta_1 \varepsilon_{t-1}$ for the EUR/JPY series, and

- $y_t = c + \alpha_1 y_{t-1} + \varepsilon_t + \beta_1 \varepsilon_{t-1}$ for the EUR/USD series.

MA(*q*) models are only useful for predictions up to *q* steps ahead. Since $\varepsilon_{t+1}, \varepsilon_{t+2}, ...$ are unknown they are set to zero and the s-step-ahead predictions for $s \le q$ are given by

- $\hat{y}_{t+s} = \hat{c} + \hat{\beta}_s \varepsilon_t + \hat{\beta}_{s+2} \varepsilon_{t-2}$ for the EUR/GBP series, and
- $\hat{y}_{t+s} = \hat{c} + \hat{\beta}_s \varepsilon_t$ for the EUR/JPY series.

For the EUR/USD-ARMA(*1,1*) model the s-step-ahead prediction is

- $\hat{y}_{t+s} - \hat{c} = \hat{\alpha}_1 (\hat{y}_{T+s-1} - \hat{c}) + \hat{\beta}_s \varepsilon_t$

for $s \le q$. For $s > q$ only the AR part determines the forecasts.

5.3 Evaluation

The evaluation procedure in this paper is twofold. Out-of-sample forecasts are evaluated both statistically via confusion matrices and practically via trading simulations. Generally, a predictive test is a single evaluation of the model performance based on comparison of actual data with the values predicted by the model. For this purpose, confusion matrices are used to illustrate the amount of correctly specified and misspecified forecasts in classification tasks. Since we are equally insterested in predicting ups and downs, the accuracy rate defined as the sum of true positives and true negatives divided by the total amount of observations is the right statistical performance measure to apply.

In addition, practical or operational evaluation methods focus on the context in which the prediction is used by imposing a metric on prediction results. More generally, when predictions are used for trading or hedging purposes, the performance of a trading or hedging metric provides a measure of the model's success. We set up a trading simulation where, first of all, return predictions $\hat{y}_t$ were translated into positions. Next a decision framework I_t was established that tells us when the underlying asset is bought or sold depending on the level of the price forecast. We define a single threshold τ, which in our model is set to τ = 0 and use the following mechanism:

$$I_t = \begin{cases} 1 & if\ \hat{y}_t < y_{t-1} - \tau \\ -1 & if\ \hat{y}_t > y_{t-1} + \tau \\ 0 & otherwise \end{cases} \text{ with } I_t = \begin{cases} 1 & if\ the\ position\ is\ long \\ -1 & if\ the\ position\ is\ short \\ 0 & if\ the\ position\ is\ neutral \end{cases} \tag{7}$$

For measuring prediction performance on the operational level, a profit and loss (P&L) metric is chosen. The gain or loss π_t on the position at time *t* is $\pi_t = I_{t-1}(y_t - y_{t-1})$. As depicted in Table 3, nine P&L related performance measures were defined: cumulated P&L, Sharpe ratio as the quotient of annulised P&L and annualised volatility, maximum daily profit, maximum daily loss, maximum drawdown, Value-at-Risk with 95% confidence, average gain/loss ratio and trader's advantage. Accounting for transaction costs (TC) is important in order to assess trading performance in a realistic way. Between market-makers an average cost of 3 pips (0.0003) per trade for a tradable amount of typically 10 to 20 million EUR is considered as a reasonable guess and thus incorporated in net cumulated profit.

6. Results and Discussion

In order to compare forecasts for the same series across different models, accuracy rates for the out-of-sample period are depicted by bar charts as shown in Figures 1 to 3 below. Note that foreign exchange markets are highly liquid and thus considered as very efficient. Consequently, if SVM accuracy rates outperform those of naïve or random strategies, the SVM technique can be generally justified to predict exchange rate return directions. In addition, Tables 4 through 6 give the results of the trading simulation. Dominant strategies are represented by the maximum value(s) in each row and are written in bold. The following conclusions can be drawn:

- In the case of statistical evaluation, both the naïve and the linear model are beaten by SVM with a suitable kernel choice. Statistically, the SVM approach is therefore justified.
- We find that hyperbolic SVMs deliver superior performance for out-of-sample prediction across all three currency pairs. In the case of EUR/GBP, the Laplace SVM performs equally well as the hyperbolic SVM. Other models are outperformed by the hyperbolic kernel SVM more clearly in the cases of EUR/JPY and EUR/USD. This observation makes hyperbolic kernels promising candidates to map all sorts of financial market return data into high dimensional feature spaces.
- Operational evaluation results confirm statistical ones in the case of EUR/GBP. Both the hyperbolic and the Laplace SVM give best results along with the RBF SVM. For EUR/JPY and EUR/USD the results differ. The statistical superiority of hyperbolic SVMs cannot be confirmed on an operational level which is contradictory to the EUR/JPY and EUR/USD operational results at first glance. The reason for this phenomenon stems from the fact that operational evaluation techniques do not only measure the number of correctly predicted exchange rate ups and downs. They also include the magnitude of returns. Consequently, if local extremes can be exploited, forecasting methods with less statistical performance may yield higher profits than methods with greater statistical performance. Thus, in the case of EUR/USD, the trader would have been better off by applying a p-Gaussian SVM in order to maximize profit. In regards to EUR/JPY, we find that no single model is able to outperform the naïve strategy. The hyperbolic SVM, however, still dominates two performance measures.
- P-Gaussian SVMs perform reasonably well in predicting EUR/GBP and EUR/USD return directions but not EUR/JPY. For the EUR/GBP and EUR/USD currency pairs, p-Gaussian data representations in high dimensional space lead to better generalization than Gaussians due to an additional degree of freedom *p*.
- Future research direction will focus on further improvements of SVM models, for instance, examination of other sophisticated kernels, proper adjustment of kernel parameters and the development of data mining and optimization techniques for selecting the appropriate kernel. In light of this research, it would also be interesting

to see if the dominance of hyperbolic SVMs can be confirmed in further empirical investigations on financial market return prediction.

7. References

[1] A.S. Andreou, E.F. Georgopoulos, and S.D. Likothanassis, "Exchange-Rates Forecasting: A Hybrid Algorithm based on genetically optimized adaptive neural networks", *Computational Economics*, vol. 20, no. 3, 2002, pp. 191-210.

[2] R. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, New Jersey, 1961.

[3] P.J. Bolland, J.T. Connor, and A.P. Refenes: "Application of neural networks to forecast high frequency data: foreign exchange", in: *Non-linear Modelling of High Frequency Financial Time Series*, C. Dunis and B. Zhou (eds.), Wiley, 1998.

[4] B.E. Boser, I.M. Guyon, and V.N. Vapnik, (1992), "A training algorithm for optimal margin classifiers", in: D. Haussler (ed.), *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, 1992, ACM Press, pp. 144-152.

[5] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, rev. ed., San Francisco, Holden-Day, 1976.

[6] W. Brock, D. Dechert, J. Scheinkmann, and B. LeBaron, "A test for independence based on the correlation dimension", *Econometric Reviews*, vol. 15, no. 3, 1996, pp. 197-235.

[7] M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. Furey, M. Ares, and D. Haussler, "Knowledge- based analysis of microarray gene expression data using support vector machines", *Technical Report*, University of California, Santa Cruz, 1999.

[8] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines (version 2.31), *Technical Report*, Department of Computer Sciences and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001.

[9] C. Cortes and V. Vapnik, "Support vector network", *Machine Learning*, vol. 20, 1995, pp. 273-297.

[10] C. De Groot and D. Wurtz, "Analysis of univariate time series with connectionist nets: a case study of two classical examples", *Neuro Computing*, vol. 3, pp. 177-192, 1991.

[11] F.X. Diebold, "Empirical modeling of exchange rate dynamics", Springer, New York, 1988.

[12] C.L. Dunis and M. Williams, "Modelling and trading the EUR/USD exchange rate: do neural network models perform better?", Working Paper, Center for International Banking, Economics and Finance, February 2002.

[13] R.F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of UK inflation", *Econometrica*, vol. 50, 1982, pp. 987-1008.

[14] M.B. Fishman, D.S. Barr and W.J. Loick, "Using neural nets in market analysis", *Technical Analysis of Stocks and Commodities*, vol. 9, no. 4, 1991, pp. 135-138.

[15] D. Francois, V. Wertz, and M. Verleysen, "About the locality of kernels in high-dimensional spaces", *ASMDA 2005 - International Symposium on Applied Stochastic Models and Data Analysis*, Brest, France, 2005, pp. 238-245.

[16] C.W.J. Granger: "Investigating causal relations by econometric models and cross-spectral methods", *Econometrica*, vol. 37, 1969, pp. 424-438.

[17] C.W.J. Granger and A.P. Anderson, *An introduction to bilinear time series models*, Gottingen: Vandenhock and Ruprecht, 1978.

[18] C.W.J. Granger and P. Newbold, "Spurious regressions in econometrics", *Journal of Economics*, vol. 2, 1979, pp. 111-120.

[19] A.A. Grasa, *Econometric model selection: a new approach*, Kluwer, 1989.

[20] J.D. Hamilton, *Time Series Analysis*, Princeton University Press, Princeton, New Jersey USA, 1994.

[21] T. Hill, M. O'Connor, and W. Remus, "Neural network models for time series forecasts", Management Science, vol. 42, no. 7, 1996, pp. 1082-1092.

[22] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks, vol. 2, 1989, pp. 359-366.

[23] D.A. Hsieh, "Testing for Nonlinear Dependence in Daily Foreign Exchange Rates", Journal of Business, vol. 62, 1989, pp. 339-368.

[24] D.A. Hsieh, "Chaos and Nonlinear Dynamics: Application to Financial Markets", *Journal of Finance*, vol. 46, no. 5, 1991, pp. 1839-1877.

[25] T.S. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers", in: *Advances in Neural Information Processing Systems*, M.S. Kearns, S.A. Solla, and D.A. Cohn (eds.), vol. 11, MIT Press, 1998.

[26] T. Joachims, "Text categorization with support vector machines", *Proceedings of European Conference on Machine Learning (ECML)*, 1998.

[27] J. Kamruzzaman and R.A. Sarker, "Forecasting of currency exchange rate: a case study", *Proceedings of the IEEE International Conference on Neural Networks & Signal Processing (ICNNSP 2003)*, Nanjing, 2003.

[28] J. Kamruzzaman and R.A. Sarker, "Application of support vector machine to Forex monitoring", *submitted to 3rd Int. Conf. On Hybrid Intelligent Systems HIS03*, Melbourne, 2003.

[29] J. Kamruzzaman, R.A. Sarker, and I. Ahmad, "SVM Based Models for Predicting Foreign Currency Exchange Rates", *Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003)*, 2004.

[30] A. Karatzoglou, K. Hornik, A. Smola, and A. Zeileis, „kernlab – An S4 package for kernel methods in R", *Journal of Statistical Software*, vol. 11, no. 9, 2004.

[31] J.O. Katz, "Developing neural network forecasters for trading", *Technical Analysis of Stocks and Commodities*, vol. 10, no. 4, 1992.

[32] J. Kean, "Using neural nets for intermarket analysis", *Technical Analysis of Stocks and Commodities*, vol. 10, no. 11, 1992.

[33] C.M. Kuan and T. Liu, "Forecasting exchange rates using feedforward and recurrent neural networks", *Journal of Applied Econometrics*, vol. 10, 1995, pp. 347-364.

[34] Y. LeCun, L.D. Jackel, L. Bottou, A. Brunot, C.Cortes, J.S. Denker, H. Drucker, I. Guyon, U..A. Müller, E.Säckinger, P. Simard, and V. Vapnik, "Comparison of learning algorithms for handwritten digit recognition," in: F. Fogelman-Soulié and P. Gallinari (eds.), *Proceedings ICANN'95 – International Conference on Artificial Neural Networks*, vol. 2, pp. 53-60, EC2, 1995.

[35] T. H. Lee, H. White and C.W.J. Granger, "Testing for neglected nonlinearity in time series models", Journal of Econometrics, vol. 56, 1993, pp. 269-290.

[36] G. Ljung and G. Box, "On a measure of lack of fit in time series models", Biometrika, 66, 1979, 265-270.

[37] R. Meese and K. Rogoff, "Empirical exchange rate models of the seventies: do they fit out-of-sample?", Journal of International Economics, 1983, pp. 3-24.

[38] K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen and V. Vapnik, "Using Support Vector Machines for Time Series Prediction", in: Advances in Kernel Methods, B. Schölkopf, C.J.C. Burges and A.J. Smola (eds.), MIT Press, 1999, pp. 242-253.

[39] I. Nabney, C. Dunis, R. Rallaway, S. Leong, and W. Redshaw, "Leading edge forecasting techniques for exchange rate prediction", in: Forecasting Financial Markets, C. Dunis (ed.), Wiley, 1996.

[40] C.R. Nelson, "The Prediction Performance of the F.R.B.-M.I.T.-PENN Model of the U.S. Economy", American Economic Review, vol. 62, 1972, pp. 902-917.

[41] J.C. Platt, “Sequential minimal optimization: A fast algorithm for training support vector machines”, Technical Report, Microsoft Research, 1998.

[42] M. Plutowski and H. White, ”Selecting concise training sets from clean data”, IEEE Transactions on Neural Networks, vol. 4, no. 2, 1993.

[43] M. Pontil and A. Verri, “Object recognition with support vector machines”, IEEE Trans. On PAMI, vol. 20, 1998, pp. 637-646.

[44] M.J. Quinlan, S.K. Chalup, and R.H. Middleton, “Application of SVMs for colour classification and collision detection with AIBO robots”, Advances in Neural Information Processing Systems, vol. 16, 2004, pp. 635-642.

[45] J.B. Ramsey, “Tests for specification errors in classical linear least squares regression analysis”, Journal of the Royal Statistical Society, Series B, vol. 31, 1969, pp. 350-371.

[46] A.-P. Refenes, Y. Abu-Mostafa, J. Moody, and A. Weigend (eds.), “Neural Networks in Financial Engineering”, Proceedings of the Third International Conference on Neural Networks in the Capital Markets, World Scientific, 1996.

[47] B. Schölkopf, “The kernel trick for distances”, TR MSR 2000-51, Microsoft Research, Redmond, WA, Advances in Neural Information Processing Systems, 2001.

[48] Y.L. Shih, “Neural nets in technical analysis”, Technical Analysis of Stocks and Commodities, vol. 9, no. 2, 1991, pp.62-68.

[49] E. Steurer, “Ökonometrische Methoden und maschinelle Lernverfahren zur Wechselkursprognose”, Physica-Verlag, Heidelberg, 1997.

[50] G.S. Swales and Y. Yoon, “Applying artificial neural networks to investment analysis”, Financial Analyst Journal, vol. 48, no. 5, 1992, pp.70-80.

[51] Z. Tang, C. Almedia, and P.A. Fishwick, “Time series forecasting using neural networks vs. Box-Jenkins methodology”, Simulation, vol. 57, 1991, pp. 303-310.

[52] Z. Tang and P.A. Fishwick, “Feedforward neural nets as models for time series forecasting”, ORSA Journal of Computing, vol. 5, 1993, pp. 374-385.

[53] H. Tong, Threshold models in non-linear time series analysis, New York, Springer-Verlag, 1983.

[54] V. Vapnik, Statistical Learning Theory, Wiley, 1998.

[55] H. White, “Economic prediction using neural networks: the case of IBM daily stock returns”, Proceedings of the IEEE International Conference on Neural Networks, July 1988.

[56] F.S. Wong, “Fuzzy neural systems for stock selection”, Financial Analyst Journal, vol. 48, pp. 47-52, 1992.

[57] G. Zhang, E.B. Patuwo, and M.Y. Hu, “Forecasting with artificial neural network: The state of the art”, International Journal of Forecasting, vol. 14, 1998, pp. 35-62.

[58] G. Zhang, “An investigation of neural networks for linear time-series forecasting”, Computers & Operations Research, vol. 28, 2001, pp. 183-202.

[59] G. Zhang, “Time Series forecasting using a hybrid ARIMA and neural network model”, Neurocomputing, vol. 50, 2003, pp. 159-175.

Table 1. Statistical Testing Procedure

Criterion	Null-Hypothesis	Testing Procedure	Time Series Input	Test Statistic Output		
				EURGBP	EURJPY	EURUSD
Stationarity	Nonstationarity	Dickey-Fuller Test (ADF)	y_t, Δy_t	-2.71**, -44.07***	-1.72, -39.58***	-2.27, -44.84***
		Philipps-Perron Test (PP)	y_t, Δy_t	-2.61*, -44.18***	-1.73, -39.53***	-2.30, -44.75***
	Stationarity	Kwiatkowski-Philipps-Schmidt-Shin Test (KPSS)	y_t, Δy_t	1.63***, 0.31	1.99***, 0.23	2.25***, 0.49**
Normal Distribution	Normal Distribution	Jarque-Bera Test (JB)	Δy_t	79.13***	53.36***	110.53***
Autocorrelation	No Autocorrelation	Ljung-Box (LB)	Δy_t, ARMA-Residuals of Δy_t	k=1: 5.58**, 0.00 k=2: 5.58*, 0.03 k=3: 11.23**, 0.03 k=4: 11.31**, 0.11 k=5: 11.32**, 0.11 k=6: 11.33*, 0.12 k=7: 11.33, 0.12 k=8: 13.22, 2.18 k=9: 13.78, 2.71 k=10: 14.26, 3.09 k=15: 17.24, 5.34 k=20: 23.10, 10.83 k=24: 33.03, 19.53	k=1: 4.47**, 0.00 k=2: 4.78*, 0.27 k=3: 5.39, 0.76 k=4: 6.37, 1.75 k=5: 6.81, 2.36 k=6: 8.30, 3.96 k=7: 8.31, 3.97 k=8: 10.85, 6.46 k=9: 10.89, 6.49 k=10: 12.37, 7.79 k=15: 17.23, 12.26 k=20: 22.95, 18.43 k=24: 23.62, 18.97	k=1: 9.60***, 0.03 k=2: 12.18***, 0.07 k=3: 13.41***, 0.07 k=4: 14.67***, 0.34 k=5: 14.68**, 0.54 k=6: 14.75**, 0.55 k=7: 15.65**, 1.66 k=8: 16.80**, 2.99 k=9: 16.83*, 3.06 k=10: 18.29*, 4.44 k=15: 21.26, 7.55 k=20: 24.93, 11.78 k=24: 27.44, 14.51
		Breusch-Godfrey Serial Correlation Lagrange-Multiplier Test (F)	ARMA-Residuals of Δy_t	0.0392	0.2119	0.1685
		Durbin-Watson Test (DW)	Δy_t, ARMA-Residuals of Δy_t	1.9915, 1.9965	1.9980, 2.0008	1.9905, 2.0037
Heteroskedasticity	No Heteroskedasticity	Ljung-Box (LB)	(ARMA-Residuals)² of Δy_t	k=1: 21.82 k=2: 27.81 k=3: 55.01*** k=4: 58.05*** k=5: 75.90*** k=6: 86.44*** k=7: 104.63*** k=8: 107.02*** k=9: 111.53*** k=10: 119.70*** k=15: 140.35*** k=20: 164.70*** k=24: 180.57***	k=1: 103.59 k=2: 125.07*** k=3: 125.60*** k=4: 142.69*** k=5: 157.72*** k=6: 171.01*** k=7: 173.01*** k=8: 183.71*** k=9: 191.36*** k=10: 192.58*** k=15: 215.38*** k=20: 328.67*** k=24: 342.69***	k=1: 0.08 k=2: 0.27 k=3: 1.45 k=4: 1.63 k=5: 9.24** k=6: 13.25** k=7: 13.99** k=8: 14.57** k=9: 19.23*** k=10: 20.97*** k=15: 29.75*** k=20: 35.85*** k=24: 42.04***
		ARCH LM Test (F)	ARMA-Residuals of Δy_t	k=1: 22.06*** k=4: 12.48*** k=8: 9.85*** k=12: 7.26***	k=1: 109.77*** k=4: 33.65*** k=8: 18.70*** k=12: 13,04***	k=1: 0.0768 k=4: 0.3950 k=8: 1.7377* k=12: 1.6951*
Nonlinearity	Linearity	Ramsey-RESET-Test (F)	ARMA-Residuals of Δy_t	n=1: 3.73* n=2: 6.30*** n=3: 4.23*** n=4: 3.60***	n=1: 3.49* n=2: 8.93*** n=3: 5.95*** n=4: 4.69***	n=1: 0.3910 n=2: 0.8979 n=3: 0.3283 n=4: 2.3085*
		Brock-Dechert-Scheinkmann Test (BDS)	ARMA-Residuals of Δy_t	m=2: 0.0107*** m=3: 0.0186*** m=4: 0.0251*** m=5: 0.0287***	m=2: 0.0088*** m=3: 0.0195*** m=4: 0.0262*** m=5: 0.0298***	m=2: -0.0005 m=3: 0.0008 m=4: 0.0021 m=5: 0.0031
			GARCH(1,1)-Residuals of Δy_t	m=2: 0.0109*** m=3: 0.0188*** m=4: 0.0254*** m=5: 0.0289***	m=2: 0.0086*** m=3: 0.0192*** m=4: 0.0259*** m=5: 0.0294***	m=2: -0.0005 m=3: 0.0008 m=4: 0.0021 m=5: 0.0032

*, **, *** indicate signficance at the 10%-, 5%-, 1% significance level
k:= number of lags
n:= number of fitted terms included in test regression
m:= number of correlation dimension for which test statistic is calculated

Table 2. ARMA model estimates

```
Dependent Variable: LN(EURGBP,1)
Method: Least Squares
Sample: 1/02/1997 9/01/2003
Included observations: 1738
Convergence achieved after 4 iterations
Newey-West HAC Standard Errors & Covariance (lag truncation=7)
Backcast: 12/30/1996 1/01/1997

Variable              Coefficient     Std. Error    t-Statistic       Prob.

C                       -3.58E-05      0.000116     -0.307893        0.7582
MA(1)                   -0.053492      0.025867     -2.068005        0.0388
MA(3)                   -0.055915       0.02709     -2.064017        0.0392

R-squared                0.005966    Mean dependent var            -3.53E-05
Adjusted R-squared        0.00482    S.D. dependent var             0.005447
S.E. of regression       0.005434    Akaike info criterion         -7.590528
Sum squared resid        0.051233    Schwarz criterion             -7.581103
Log likelihood           6599.169    F-statistic                     5.20668
Durbin-Watson stat       1.998712    Prob(F-statistic)              0.005566
```

```
Dependent Variable: LN(EURJPY,1)
Method: Least Squares
Sample: 1/02/1997 9/01/2003
Included observations: 1738
Convergence achieved after 4 iterations
Backcast: 1/01/1997

Variable              Coefficient     Std. Error    t-Statistic       Prob.

C                       -7.84E-05      0.000205     -0.382154        0.7024
MA(1)                     0.02883      0.023994      1.201542        0.2297

R-squared                0.000827    Mean dependent var            -7.85E-05
Adjusted R-squared       0.000252    S.D. dependent var             0.008313
S.E. of regression       0.008312    Akaike info criterion         -6.741048
Sum squared resid        0.119943    Schwarz criterion             -6.734764
Log likelihood            5859.97    F-statistic                    1.437179
Durbin-Watson stat       1.999829    Prob(F-statistic)               0.23076
```

```
Dependent Variable: LN(EURUSD,1)
Method: Least Squares
Sample(adjusted): 1/03/1997 9/01/2003
Included observations: 1737 after adjusting endpoints
Convergence achieved after 11 iterations
Newey-West HAC Standard Errors & Covariance (lag truncation=7)
Backcast: 1/02/1997

Variable              Coefficient     Std. Error    t-Statistic       Prob.

C                       -8.32E-05      0.000149     -0.558307        0.5767
AR(1)                   -0.583958      0.214689     -2.720017        0.0066
MA(1)                    0.519186       0.22621      2.295155        0.0218

R-squared                0.006446    Mean dependent var            -8.32E-05
Adjusted R-squared         0.0053    S.D. dependent var             0.006439
S.E. of regression       0.006422    Akaike info criterion         -7.256372
Sum squared resid        0.071519    Schwarz criterion             -7.246942
Log likelihood           6305.159    F-statistic                    5.624804
Durbin-Watson stat       2.001353    Prob(F-statistic)              0.003673
```

Table 3. Operational performance measures

Cumulated Profit and Loss	$PL_T^C = \sum_{t=1}^{T} \pi_t$
Sharpe Ratio	$SR = \frac{PL^A}{\sigma^A}$, with $PL_T^A = 252 * \frac{1}{T} \sum_{t=1}^{T} \pi_t$ and $\sigma_T^A = \sqrt{252} * \sqrt{\frac{1}{T-1} * \sum_{t=1}^{T} (\pi_t - \bar{\pi})^2}$
Maximum daily profit	$Max(\pi_1, \pi_2, ..., \pi_T)$
Maximum daily loss	$Min(\pi_1, \pi_2, ..., \pi_T)$
Maximum drawdown	$MD = Min\left(PL_t^C - \underset{i=1,...,t}{Max} \left(PL_i^C \right) \right)$
Value-at-Risk	$VaR = \mu - Q(\pi, 0.05)$, $\mu = 0$
Net cumulated profit and loss	$NPL_T^C = \sum_{t=1}^{T} (\pi_t - I_t * TC)$, where $I_t = 1$ if $\pi_{t-1} * \pi_t < 0$ else $I_t = 0$
Average gain/loss	$\frac{AG}{AL} = \frac{(Sum\ of\ all\ \pi_t > 0)/\#up}{(Sum\ of\ all\ \pi_t < 0)/\#down}$
Trader's Ad-vantage	$TA = 0.5 * \left(1 + \left(\frac{(WT * AG) + (LT * AL)}{\sqrt{(WT * AG^2) + (LT * AL^2)}} \right) \right)$ with WT := number of winning trades, LT:= number of losing trades, AG:= average gain in up periods, and AL:= average loss in down periods

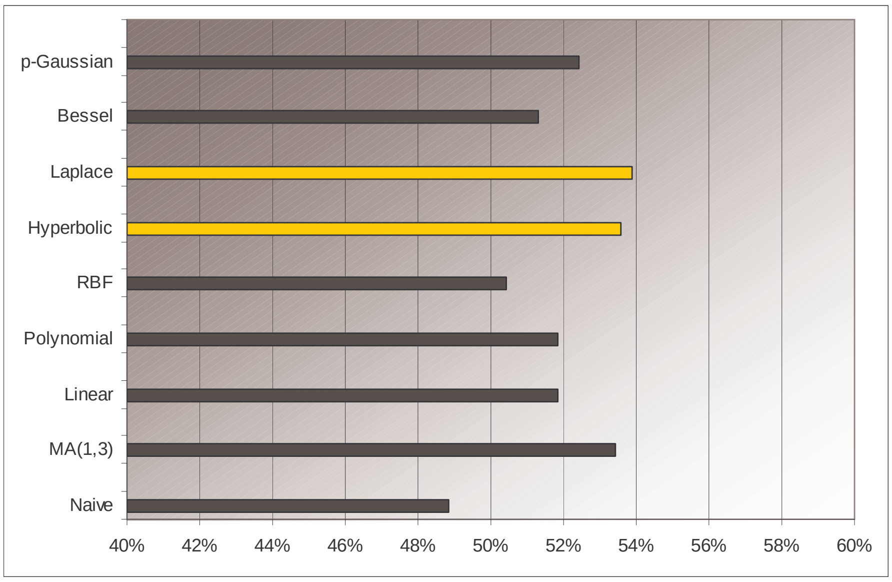


Fig. 1. Classification performance EUR/GBP

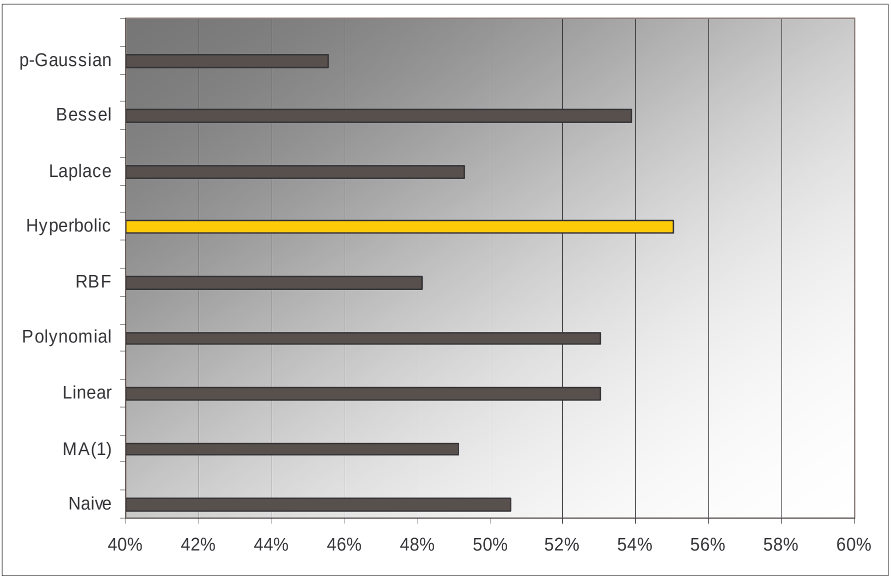


Fig. 2. Classification performance EUR/JPY

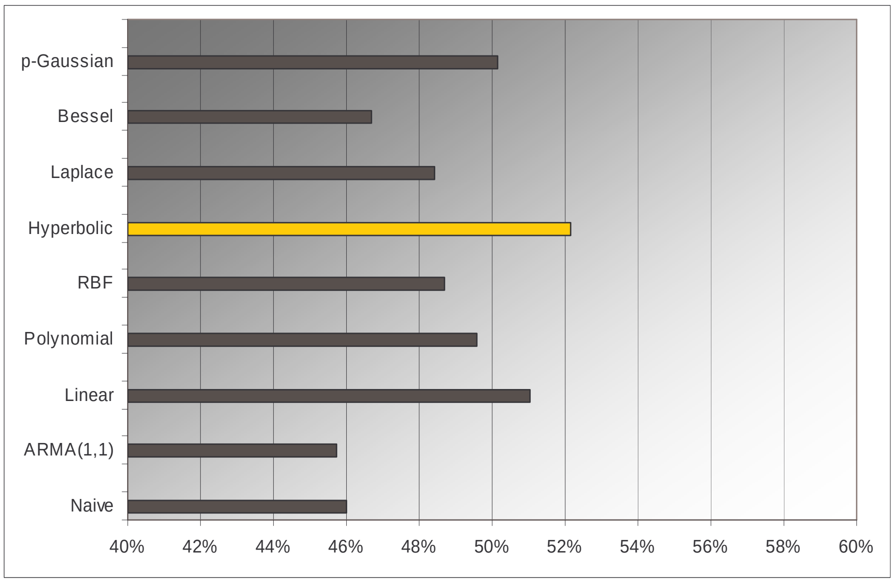


Fig. 3. Classification performance EUR/USD

Table 4. Operational performance EUR/GBP

EUR/GBP	Naive	MA(1,3)	Linear	Polynomial	RBF	Hyperbolic	Laplace	Bessel	p-Gaussian
Cumulative P&L	-0.00750	-0.00953	-0.09360	-0.09360	-0.03896	**0.10360**	0.01546	-0.04114	0.05958
Sharpe ratio	-0.07966	-0.10112	-0.99367	-0.99367	-0.41354	**1.09938**	0.16407	-0.43671	0.63235
Maximum daily profit	0.01492	0.01492	**0.01684**	**0.01684**	**0.01684**	0.01492	**0.01684**	0.01385	0.01232
Maximum daily loss	-0.01684	-0.01684	-0.01492	-0.01492	**-0.01385**	-0.01684	**-0.01385**	-0.01684	-0.01684
Maximum drawdown	-0.03811	-0.03811	-0.03619	-0.03619	**-0.03496**	-0.03811	-0.03512	-0.03564	-0.03811
VaR (alpha = 0.05)	-0.00695	-0.00734	-0.00752	-0.00752	-0.00728	-0.00698	**-0.00691**	-0.00744	-0.00694
Net Cumulative P&L	-0.06120	-0.01013	-0.12750	-0.12750	-0.09026	**0.05590**	-0.01964	-0.09214	0.01428
Avg gain/loss ratio	**1.05178**	0.85038	0.80370	0.80370	0.91714	1.03981	0.89932	0.88235	1.01891
Trader's Advantage	0.00000	**1.00000**	0.53003	0.53003	0.48716	0.48144	0.58986	0.39350	0.43507

Table 5. Operational performance EUR/JPY

EUR/JPY	Naive	MA(1)	Linear	Polynomial	RBF	Hyperbolic	Laplace	Bessel	p-Gaussian
Cumulative P&L	**0.05441**	-0.11333	-0.09477	-0.09477	-0.21907	-0.13867	-0.28671	-0.31145	-0.24980
Sharpe ratio	**0.38680**	-0.80435	-0.67432	-0.67432	-1.55679	-0.98622	-2.03603	-2.21115	-1.77460
Maximum daily profit	**0.02187**	0.02187	0.02068	0.02068	0.02068	0.02174	0.02068	0.02068	0.02050
Maximum daily loss	**-0.02050**	-0.02174	-0.02187	-0.02187	-0.02187	-0.02187	-0.02187	-0.02187	-0.02187
Maximum drawdown	-0.08535	-0.08659	-0.06479	-0.06479	-0.08672	**-0.06197**	-0.08672	-0.06479	-0.08672
VaR (alpha = 0.05)	**-0.01003**	-0.01144	-0.01092	-0.01092	-0.01111	-0.01081	-0.01127	-0.01145	-0.01130
Net cumulative P&L	**0.00281**	-0.11363	-0.15267	-0.15267	-0.27607	-0.19837	-0.34461	-0.36185	-0.30260
Avg gain/loss ratio	**1.04111**	0.92829	0.89996	0.89996	0.88278	0.86458	0.83323	0.83752	0.82177
Trader's advantage	0.00000	0.00000	0.43005	0.43005	0.43247	**0.43647**	0.41154	0.40350	0.40139

Table 6. Operational performance EUR/USD

EUR/USD	Naive	ARMA(1,1)	Linear	Polynomial	RBF	Hyperbolic	Laplace	Bessel	p-Gaussian
Cumulative P&L	-0.18070	-0.22255	-0.13259	-0.13259	-0.00927	0.04797	-0.10055	-0.16166	**0.10182**
Sharpe ratio	-1.23452	-1.52256	-0.90434	-0.90434	-0.06296	0.32520	-0.68505	-1.10372	**0.68905**
Maximum daily profit	0.01962	0.01962	0.01667	0.01667	0.01962	**0.01962**	0.01889	0.01869	0.01889
Maximum daily loss	-0.01889	-0.01889	-0.01962	-0.01962	**-0.01869**	-0.01889	-0.01962	-0.01962	-0.01962
Maximum drawdown	-0.04172	**-0.04112**	-0.04484	-0.04484	-0.04391	-0.04410	-0.04484	-0.04484	-0.04484
VaR (alpha = 0.05)	-0.01247	-0.01179	-0.01260	-0.01260	-0.01176	**-0.01085**	-0.01183	-0.01165	-0.01116
Net cumulative P&L	-0.23680	-0.22345	-0.17429	-0.17429	-0.05967	-0.00003	-0.14525	-0.21056	**0.05112**
Avg gain/loss ratio	0.94708	0.93486	0.88117	0.88117	1.03619	0.96269	0.94573	0.94569	**1.10874**
Trader's advantage	0.00000	0.31863	**0.62531**	**0.62531**	0.56826	0.55311	0.58379	0.42194	0.49915

Author Index