Comparison of SVM and Some Older Classification Algorithms in Text Classification Tasks

Fabrice Colas[1] and Pavel Brazdil[2]

[1] LIACS, Leiden University, The Netherlands, fcolas@liacs.nl
[2] LIACC-NIAAD, University of Porto, Portugal, pbrazdil@liacc.up.pt

Summary. Document classification has already been widely studied. In fact, some studies compared feature selection techniques or feature space transformation whereas some others compared the performance of different algorithms. Recently, following the rising interest towards the Support Vector Machine, various studies showed that SVM outperforms other classification algorithms. So should we just not bother about other classification algorithms and opt always for SVM ?

We have decided to investigate this issue and compared SVM to kNN and naive Bayes on binary classification tasks. An important issue is to compare optimized versions of these algorithms, which is what we have done. Our results show all the classifiers achieved comparable performance on most problems. One surprising result is that SVM was not a clear winner, despite quite good overall performance. If a suitable preprocessing is used with kNN, this algorithm continues to achieve very good results and scales up well with the number of documents, which is not the case for SVM. As for naive Bayes, it also achieved good performance.

1 Introduction

The aim of using artificial intelligence techniques in text categorization is to build systems which are able to automatically classify documents into categories. But as the feature space, based on the set of unique words in the documents, is typically of very high dimension, document classification is not trivial. Various feature space reduction techniques were suggested and compared in [13, 9]. A large number of adaptive learning techniques have also been applied to text categorization. Among them, the k nearest neighbors and the naive Bayes are two examples of commonly used algorithms (see for instance [7] for details). Joachims applied the Support Vector Machine to document classification [4]. Numerous classifier comparisons were done in the past [12, 14, 4, 2].

Some algorithms like the SVM are by default binary classifiers. Therefore, if we have a problem with more than two classes, we need to construct as

many classifiers as there are classes (*one versus all* strategy). However, it is not fair to compare a single multi-class naive Bayes (or kNN) classifier to n SVM classifiers (for n classes). This is why we have decided to focus on *one against one* classification tasks. Moreover, Fürnkranz [3] showed that a *round robin* approach using the set of *one against one* classifiers, performs at least as well as a *one versus all* approach. These binary problems involve also smaller amounts of data, which means that the classifiers are faster to train. The properties of the train set have much influence on the classifier learning abilities. Therefore, focusing on binary classification tasks allows one to carefully control the nature of train sets. Finally, directly studying multi-class classification tasks tends to obscure the particular behaviors of the classifiers on some classes which may be of interest.

We seek answers to the following questions. *Should we still consider old classification algorithms in text categorization or opt systematically for SVM classifiers ? What are the strength and weaknesses of the SVM, naive Bayes and kNN algorithms in text categorization on a set of simple binary problems ? Are there some parameter optimization results transferable from one problem to another ?* Before giving the answers, our experimental settings and evaluation methodology are described. Then, our results regarding the parameter optimization are presented. The optimized versions are then used in further comparative studies, which are used to answer the above questions.

2 Document Collection, Algorithms and Evaluation Methodology

2.1 Document Collection

For our experiments we used the well known `20newsgroups` dataset composed of 20000 newsgroup emails (removed email headers and no stemming). We chose to study the set of *one against one* binary classification tasks of this dataset. Thus, $\frac{20(20-1)}{2} = 190$ classification tasks were examined. Given the large dimensions of the problem, sub sampling techniques were applied to observe the classifier learning abilities for an increasing train set size. We also used the *Information Gain* to impose an ordering on a set of attributes. We chose this heuristic for its simplicity and its good performance [13, 9].

2.2 Algorithms

In this paper, two well known classifiers are compared to the Support Vector Machine namely the kNN and the naive Bayes. These two classifiers were chosen because of their simplicity and their generally good performance reported in document classification. With respect to the SVM, the SMO implementation of Platt, available in the `libbow` [6] library, has been used.

Let us consider the classification function Φ of the data points $\boldsymbol{x}_i$ $(i = 1...l)$ into a class $y_i \in \mathcal{C} = \{+1, -1\}$. Let d be the dimension of the feature space. The three classification algorithms are presented in the following subsections.

Support Vector Machine.

The SVM problem (*primal*) is to find the decision surface that maximizes the margin between the data points of the two classes. Following our results and previously published studies in document classification [12, 14], we limit our discussion to *linear* SVM. The *dual* form of the *linear* SVM optimisation problem is to maximize :

$$\begin{aligned} \boldsymbol{\alpha}^* = \text{maximise}_{\boldsymbol{\alpha}} & \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle, \\ \text{subject to } & \sum_{i=1}^{l} y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C, i = 1...l \end{aligned} \tag{1}$$

with α_i the weight of the examples and C the relative importance of the complexity of the model and the error. The class prediction $\hat{\Phi}(\boldsymbol{x}')$ of the point $\boldsymbol{x}'$ is given by :

$$\hat{\Phi}(\boldsymbol{x}') = sign(\sum_{i=1}^{l} \alpha_i y_i \langle \boldsymbol{x}_i, \boldsymbol{x}' \rangle + b) = sign(\langle \boldsymbol{w}^*, \boldsymbol{x}' \rangle + b) \tag{2}$$

where $\boldsymbol{w}^* = \sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x}_i$.

***k* Nearest Neighbors.**

Given a test point, a predefined similarity metric is used to find the k most similar points from the train set. For each class y_i, we sum the similarity of the neighbors of the same class. Then, the class y_i with the highest score is assigned to the data point $\boldsymbol{x}'$ by the k nearest neighbors algorithm.

$$\hat{\Phi}(\boldsymbol{x}') = argmax_{y_j \in \mathcal{C}} \sum_{i=1}^{k} \delta(y_j, \Phi(\boldsymbol{x}_i)) sim(\boldsymbol{x}_i, \boldsymbol{x}') \tag{3}$$

Naive Bayes

Let $P(y_i)$ be the prior probability of the class y_i and $P(a'_j|y_i)$ be the conditional probability to observe attribute value a'_j given the class y_i. Then, a naive Bayes classifier assign to a data point $\boldsymbol{x}'$ with attributes $(a'_1...a'_d)$ the class $\hat{\Phi}(\boldsymbol{x}')$ maximizing :

$$\hat{\Phi}(\boldsymbol{x}') = argmax_{y_i \in \mathcal{C}} P(y_i) \prod_{j=1}^{d} P(a'_j|y_i) \quad (4)$$

2.3 Evaluation Methodology

A classical 10-fold cross validation was used to estimate classifier performance. We chose the macro averaged F_1 measure $MF_1 = \frac{2 \times MPrecision \times MRecall}{MPecision + MRecall}$ [10], where the $MPrecision$ and the $MRecall$ measures are the averages of the precision and the recall computed on the basis of the two confusion matrices (in one, a class is considered positive and the other negative ; in the other the assignment is interchanged). Finally, we recorded the global processing time in seconds (the sum of the training and the testing time). As the size of the test set is nearly the same for each experiment, this processing time reflects mostly the train time of the classifiers.

3 Experimental Results

3.1 Parameter Optimization Results

We ran some preliminary experiments on `20newsgroups` to find the best parameter values. These experiments were restricted to three binary classification tasks[3]. Our results are presented in the following sections for SVM and kNN.

Support Vector Machines.

Various parameters of SVM were considered in the attempt to optimize the performance of this algorithm. The parameter C was varied and various kernel functions were tried as well. None of those lead to interesting improvements in terms of performance (MF_1) or processing time. So, the default value $C = 200$ and a *linear* kernel are used.

We have also varied the ϵ parameter controlling the accepted error. We have found that ϵ had no influence on MF_1 as long as its value was smaller or equal to 0.1. However, ϵ did affect the training time. Indeed the time could be reduced by a factor of 4 in the best case (see Fig. 1 (A) with 500 features), when the largest value of ϵ (0.1) was used. Our hypothesis is that the precision of the optimisation problem is simplified when an acceptable optimal hyper plane is bounded by a larger error ϵ. Therefore, it seems that no high precision is needed to train SVM on these binary classification tasks. Fig. 1 (A) and (B) portray the training time of SVM for various values of ϵ when the size of the feature space is varied and when the number of documents in the train set is increased.

[3] `alt.atheism` vs. `talk.religion.misc`, `comp.sys.ibm.pc.hardware` vs. `comp.sys.mac.hardware`, `talk.politics.guns` vs. `talk.politics.misc`

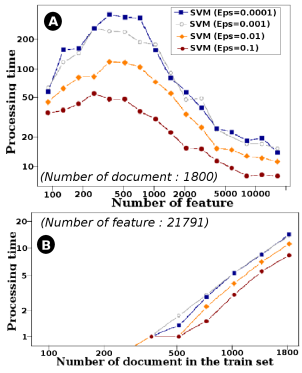


Fig. 1. *Processing time of the SVM classifier on* `alt.atheism` *vs.* `talk.religion.misc`*, for several values of* ϵ*, given an increasing number of features (**A**) and an increasing number of documents in the train set (**B**).*

k Nearest Neighbors.

Two parameters were considered to optimize the performance of the kNN, the *number k of nearest neighbor* and the *feature space transformation*. Indeed, to achieve good performance with kNN, the feature space should be transformed to a new one. Common transformation in text mining are based on the number of occurences of the i^{th} term tf_i, the inverse document frequency which is defined as the ratio between the total number of documents N and the number of documents containing the term df_i, a normalization constant κ.

$$\begin{aligned} \Phi_{\texttt{atc}}(x_i) &= \frac{(\frac{1}{2} + \frac{tf_i}{2tf_{max}})\ log(\frac{N}{df_i})}{\kappa} \\ \Phi_{\texttt{ntn}}(x_i) &= tf_i\ log(\frac{N}{df_i}) \\ \Phi_{\texttt{lnc}}(x_i) &= \frac{log(tf_i)}{\kappa} \end{aligned} \tag{5}$$

Thirty measures (3 problems, 10-fold cross validation) caracterized the experimental results for each parameter setting. To compare these results, a simple heuristic based on a pairwise t-test (95% confidence interval) was used. When a significative difference was observed regards the results of one parameter setting to one other, a victory point was attributed to the best setting. In case of tie, no point was given. Train sets with the maximum number of document[4] were used whereas the feature space was composed of *all* the possible attributes.

Number of Nearest Neighbors.

We observed that large k values lead to relatively good performance. Indeed, the contribution towards the class score of the neighbors is weighted by their

[4] A binary task involves 2×1000 documents. Considering that 10-fold cross validation is used, each training set includes 1800 documents and the test set 200.

similarity to the test point. Therefore, the farthest neighbors have little effect on the class score. However, the best number of nearest neighbors is $k = 49$. This optimal k value (49) is interestingly quite close to the one of Yang (45) in [12] with completely different experimental settings (`Reuters-21570`, classification task seen as a single multi-class problem). As a result, it seems that k values between 45 and 50 are well suited for text classification tasks.

We first ran all our experiments with $k = 5$. Therefore the kNN results could be slightly improved in the following comparative study if we used the optimized value for k.

Feature Space Transformation.

About 400 transformations were evaluated. Our observation is that any term frequency is suitable, but not the binary transformation (value 1 or 0), depending whether a particular word is (or is not) present. This is coherent to the previous study of McCallum et al. [5]. In the same way, the inverse document frequency should be systematically applied because, as it is well known, it decreases the importance of common words occurring in numerous documents. The normalization did not affect the performance. In the coming comparative study, the transformations[5] presented in formulas 5 are used.

3.2 Some Classifier Comparisons

The aim of our experiments was to examine the classifier learning abilities for an increasing number of documents in the train set (learning curves), and also, how the performance is affected by the number of attributes of the feature space. In the study involving learning curves, *all the features* were selected. Similarly, when the behaviors for an increasing number of features were studied, the train set was composed of its *maximum size*, containing as many documents of both classes.

First of all, we have observed that architectural variables (sample selection, algorithm parameters, feature subset selection, working feature space) had often a *larger* impact on the performance than the choice of individual classifiers. In fact, if suitable architectural variables are chosen and if the parameter settings of the classifiers get correctly optimized, then the differences between the algorithms are not very large.

Moreover, the behaviors of the classifiers are very similar across the classification tasks. This is illustrated in Fig. 2 (A) and (B) which shows the performance of the three algorithms on two *typical* binary classification tasks among the 190. The figure shows how the performance depends on the number of documents in the train set. Fig. 2 (A) shows that naive Bayes is slightly

[5] A weighting scheme is composed of two parts, for example `ntn.lnc` or `atc.atc` (SMART Information Retrieval System notations). The first group of three letters word describes the feature space transformation for the documents in the train set, whereas the second group describes the feature space for the test documents.

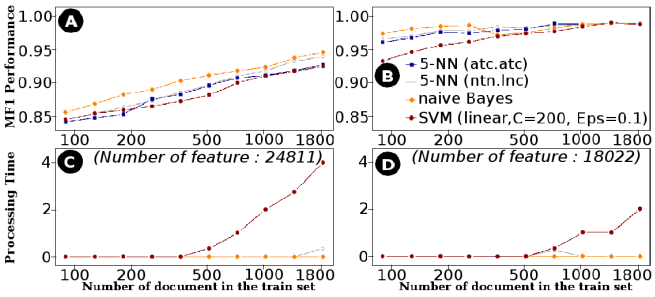


Fig. 2. *Performance (**A**), (**B**) and processing time (**C**), (**D**) of kNN, SVM and naive Bayes for two problems :* `comp.graphics` *vs.*`comp.sys.ibm.pc.hardware` *(**A**), (**C**) and* `comp.os.ms-windows.misc` *vs.* `talk.politics.misc` *(**B**), (**D**) for an increasing number of documents in the train set.*

better than the other algorithms for all train sizes. However, the difference from the worst performer is not very large (about 2 or 3%).

Fig. 2 (B) shows that naive Bayes starts with an advantage when a small number of documents are used in the train set, but then as the number of documents increases, the difference diminishes. When 1800 documents are used, the performance is virtually identical to the other classifiers. SVM is however in a disadvantage, when we consider the processing times. These are not only much higher than for the other algorithms, but also, the processing time tends to grow quadratically with the number of documents in the train set (see Fig. 2 (C), (D) and Fig. 4 (D)).

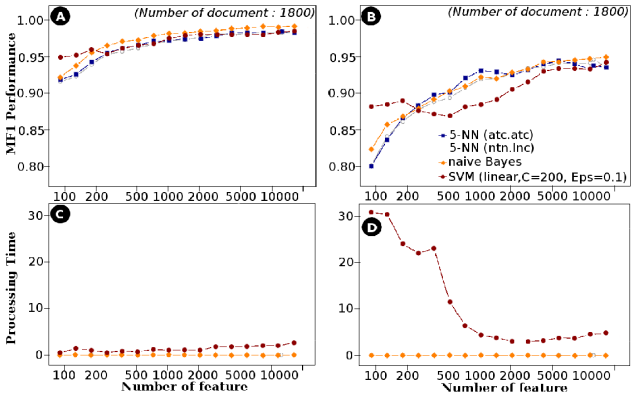


Fig. 3. *Performance (**A**), (**B**) and processing time (**C**), (**D**) of kNN, SVM and naive Bayes for two problems :* `rec.motorcycles` *vs.* `sci.med` *(**A**), (**C**) and* `comp.sys.ibm.pc.hardware` *vs.*`sci.electronics` *(**B**), (**D**) for an increasing number of attribute in the feature space.*

Regards the number of features, all three classifiers tend to achieve better performance on large feature set (see Fig. 3 (A) and (B)). However, the SVM processing time can be particularly high if the number of features is small (see Fig. 3 (D) and Fig. 4 (C)). Besides, regards performance of SVM an interesting pattern can be observed on some tasks (see Fig. 3 (B) and Fig. 4 (A)). First, a maximum is reached for a relatively small feature set. Then, the performance decreases until it reverses its tendency again.

On the problem involving `alt.atheism` and `talk.religion.misc` (Fig. 4), both three classifiers achieved relatively poor performance when compared to other classification tasks. In fact, as the two newsgroups are closely related, it is difficult to determine to which category the documents belong. In this task, SVM outperforms naive Bayes and kNN for small feature spaces (see Fig. 4 (A), 100-200) whereas it performs poorly on large feature spaces (500-20000). Although this behavior is specific to this task, it is still a surprising result. Indeed, it is often said that SVM deals well with large number of features. It appears that naive Bayes and kNN do this better here. However, it could be taken into consideration when constructing the learning curves. For instance, the learning curve of SVM shown in Fig. 4 (B) which uses *all* the possible features (21791), could be pushed up if a smaller feature set was used (200).

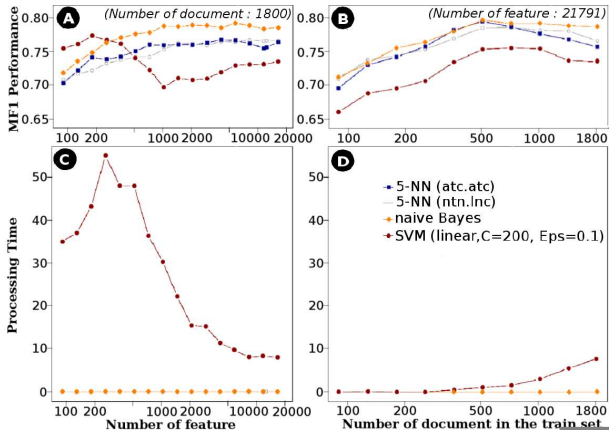


Fig. 4. *Performance (**A**), (**B**) and processing time (**C**), (**D**) of naive Bayes, kNN and SVM on* `alt.atheism` *versus* `talk.religion.misc`*, given an increasing number of features (**A**), (**C**) and an increasing number of documents in the train set (**B**), (**D**). right.*

On most of the classification tasks, the training time of SVM increases linearly with the number of features (see Fig. 3 (C)). However, the search for the optimal hyper plane of SVM may require very large training time. For

example, the largest training times among the 190 classification tasks occur on the problem presented in Fig. 4. Indeed, correlating the above-mentioned pattern, SVM training time is higher for small feature spaces than for large ones (Fig. 4 (C) and Fig. 3 (D)). Therefore, training SVM on a extended feature space tends to be faster on these particular tasks.

Discussion.

As explained earlier, comparing a single multi-class naive Bayes (or kNN) to n SVM classifiers (n the number of categories) is definitively not fair for naive Bayes (or kNN). However, this is the approach followed in some published comparative studies [2, 12, 14].

In [2], the SVM SMO version of Platt was claimed to outperform naive Bayes and other learning methods. However, the optimal number of features was not investigated for each classifier. Indeed, 300 features were selected for SVM which may not be far from the optimal setting. But only 50 were used for naive Bayes. According to our results naive Bayes performs much better with large number of features. Also, the Mutual Information (MI) was used to select features which may not be the best option according to [13]. Finally, they studied the set of *one-against-all* classifiers for each type of algorithm. However, this approach tends to obscur the particular behavior of the classifiers on the various classification tasks.

Recently, a study [1] showed that the architectural parameters often have a more significant impact on performance than the choice of individual learning technique. The work presented here also confirms this. This is why we have decided not to do simple classifier comparisons and present tables with performance results. We preferred to compare the general tendencies of different classifiers when certain parameters are varied.

4 Conclusion

Firstly, we showed that kNN and naive Bayes are still worth considering. Both classifiers achieved good overall performance and are much faster than SVM to use. Indeed, the cost to train SVM for large train set is a clear drawback.

Secondly, compared to SVM, both kNN and naive Bayes are very simple and well understood. SVM is however, more appealing theoretically and in practice, its strength is its power to adress non-linear classification taskw. Unfortunately, most of the tasks examined here were not like that. The simplest SVM based on a *linear* kernel and a large error ϵ were found to be sufficient.

We also observed that results highly depend of the adopted methodology. We have focused here on simple binary classification tasks. Regards kNN, the optimal number k of nearest neighbors is interestingly close to the ones used in other comparative studies carried out on different problems.

As our primary objective is to arrive at general conclusions, transferable from one domain to another, we need to validate our results on other document classification tasks. For this purpose, new experiments are actually being carried out. Moreover, if we are interested to recommend a classifier with suitable parameter settings, we should have a good way of characterizing the given documents and develop a good meta-learning strategy.

Acknowledgements. The first author wishes to thank P. Brazdil of the LIACC-NIAAD, and also A.-M. Kempf and F. Poulet of the ESIEA Recherche Institute, for having him introduced to his research. We wish to express our gratitude to K. R. Patil and C. Soares for their relecture and to all colleagues from LIACC-NIAAD, University of Porto for their encouragement and help. Finally, the Portuguese Pluri-annual support provided by FCT is gratefully acknowledged.

References

1. W. Daelemans, V. Hoste, F. D. Meulder, and B. Naudts. Combined optimization of feature selection and algorithm parameters in machine learning of language. In *Proceedings of the European Conference of Machine Learning*, pages 84–95, 2003.
2. S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management*, pages 148–155, 1998.
3. J. Fürnkranz. Pairwise classification as an ensemble technique. In *Proceedings of the 13th European Conference on Machine Learning*, pages 97–110, 2002.
4. T. Joachims. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods: Support Vector Machines*. 1998.
5. A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. *AAAI-98 Workshop on Learning for Text Categorization*, 1998.
6. A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow, 1996.
7. T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
8. J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report 98-14, Microsoft Research, 1998.
9. M. Rogati and Y. Yang. High-performing feature selection for text classification. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 659–661, 2002.
10. Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, pages 69–90, 1999.
11. Y. Yang. A scalability analysis of classifiers in text categorization. In *Proceedings 26th ACM International Conference on Research and Development in Information Retrieval*, 2003.
12. Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.
13. Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420, 1997.
14. T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, pages 5–31, 2001.

An automatic graph layout procedure to visualize correlated data

Mario Inostroza-Ponta, Regina Berretta,
Alexandre Mendes, and Pablo Moscato

Newcastle Bioinformatics Initiative
School of Electrical Engineering and Computer Science
Faculty of Engineering and Built Environment
The University of Newcastle, Callaghan, NSW, 2308, Australia

and
ARC Centre in Bioinformatics
Contact email:Pablo.Moscato@newcastle.edu.au

Abstract. This paper introduces an automatic procedure to assist on the interpretation of a large dataset when a similarity metric is available. We propose a visualization approach based on a graph layout methodology that uses a Quadratic Assignment Problem (QAP) formulation. The methodology is presented using as testbed a time series dataset of the Standard & Poor's 100, one the leading stock market indicators in the United States. A weighted graph is created with the stocks represented by the nodes and the edges' weights are related to the correlation between the stocks' time series. A heuristic for clustering is then proposed; it is based on the graph partition into disconnected subgraphs allowing the identification of clusters of highly-correlated stocks. The final layout corresponds well with the perceived market notion of the different *industrial sectors*. We compare the output of this procedure with a traditional dendogram approach of hierarchical clustering.

1 Introduction

The Standard & Poor's 100 index is one the leading stock market indicators in the United States. It measures the performance of the 100 largest U.S. companies, corresponding to over US$ 6 trillion in terms of market capitalization[1] and it is composed of stocks from different sectors.In the stock market, the changes of the value of a given company are highly correlated with the time series of its stock price. Two contributions to the study of market dynamics ([1];[2]) reported on the application of *Self-Organizing Maps* and *Chaotic Map Synchronization* to two different datasets composed of the price variation of the stocks in the Dow Jones index. A graph-based approach using 6,546 *financial instruments* (stocks, indexes, etc.) traded in the US markets has also been recently introduced [3].

[1] http://www2.standardandpoors.com/spf/pdf/index/factsheet_sp100.pdf

In this paper we propose a new graph layout visualization method and use it to uncover interesting relationships between the stocks of the S&P100 index used as a case study. We will consider that each stock corresponds to a node of a graph; the edges' weights will be related to the correlation between stocks. The method recursively divides the graph in disconnected subgraphs. Once the subgraphs (clusters) are defined, we solve a sequence of Quadratic Assignment Problems (QAP) using a memetic algorithm (MA) which will determine their relative position in the layout. Finally, another instance of the QAP is solved to find how the clusters are distributed, now considering each cluster as a single element.

The Quadratic Assignment Problem (QAP) belongs to the *NP-hard* [4] class and is a well-studied combinatorial optimization problem [5, 6, 7]. Informally, we are given a set of n facilities and m locations ($m \geq n$), and the task is to assign each facility to a location taking into account the flow between each facility and the distance between the locations. The objective is to minimize the overall transportation cost between all the facilities. For our case study, we will use the correlation between stocks to determine the flow between facilities. The locations will be points in a grid, with the distances between them given by the Euclidean metric. We have as input a flow matrix between the stocks, and from this matrix we create a weighted graph. We can understand this graph as a proximity graph; its edges will also have a strong influence in the layout process as will be described later. The result is a graph layout where clusters of stocks with similar dynamical behavior are promptly identified, and no user-intervention is required during the process.

The use of MAs to address the QAP can be dated back to Carrizo et al.(1992) [8] and Merz and Freisleben (1999) [9]. In this paper, we employ a similar MA to those successfully used before for other combinatorial optimization problems, including Number Partitioning [10] and the Asymmetric Travelling Salesman [11] problems among others. Two local search methods are used; one of them has an embedded Tabu Search [12].

This paper is organized as follows. In Sec. 2 we describe the graph layout procedure. Section 3 describes the memetic algorithm for the QAP. The result of applying this method on the S&P100 dataset is presented in Sec. 4, followed by the conclusions in Sec. 5.

2 Graph Layout Procedure

The graph layout procedure proposed in this paper is composed of 3 steps: *creation of a distance matrix*, *proximity graph clustering algorithm* and *creation of QAP instances that will be solved using the MA*. We explain each step using the S&P100 dataset.

2.1 Distance Matrix

To create the distance matrix D of the S&P100 dataset, we took the second derivative of the weekly closing price variation of the stocks that compose the index, between the years 1999 and 2004. The work of Ausloos and Ivanova (2002) [13] advocates the use of the second derivative (which represents the acceleration of the stock price), arguing in their studies of "*pressure, acceleration and force indicators*" that it contains more information than the first derivative. The expression of the three-point rule for the second derivative of the stock price at time t is given by

$$y_i(t) = \frac{P_i(t-h) - 2.P_i(t) + P_i(t+h)}{h^2}, \tag{1}$$

where $P_i(t)$ represents the closing price of the stock i in the week t and h represents the interval used to calculate the derivative; in this case, $h = 1$ week. At the end, we normalize the result by dividing it by $P_i(t-h)$, so to eliminate any bias introduced by the actual price of the stock. The distance matrix $D = \{d_{ij}\}$ is defined as $d_{ij} = 1 - \rho_{ij}$, where ρ_{ij} is the Pearson correlation between stocks i and j using the values calculated with function 1. The two most correlated stocks ($\rho = 0.802$) are Schlumberger Ltd. and Baker Hughes Inc., while the two most anti-correlated ($\rho = -0.38$) are Alcoa Inc. and Anheuser-Busch Co. There are only 459 pairs of stocks with $\rho < 0$.

2.2 Proximity graph clustering algorithm

We use the matrix D to build our *ad-hoc* proximity graph using the *minimum spanning tree* and the *k-nearest neighbors* graphs, which we will refer to as G_{MST} and G_{kNN} respectively, as follows: Initially, we create a complete undirected weighted graph $G(V, E, w)$, using the matrix D, where the weight $w_{ij} = d_{ij}$. The minimum spanning tree $G_{MST}(V, E_{MST})$ is defined as a connected, acyclic subgraph containing all the nodes of G and whose edges sum has minimum total weight. The graph G_{kNN} is represented by $G_{kNN}(V, E_{kNN})$, where $e_{ij} \in E_{kNN}$ iff j is one of the k nearest neighbors of i. Our proximity graph, namely $G_{cluster}(V, E_{cluster})$, is constructed such that $E_{cluster} = E_{MST} \cap E_{kNN}$. This type of proximity graphs was used also in González-Barrios and Quiroz (2003) [14]. In this work we decided to set k as the minimal value such that G_{kNN} is still connected while in Ref. [14] they have a different approach.

2.3 Creating and solving QAP instances

We consider the QAP with n elements and $m > n$ positions. The QAP has as input a matrix $F = \{f_{ij}\}$ of flows between the n elements and a matrix $L = \{l_{ij}\}$ of distances between m grid locations. The objective is to assign the n elements to the m locations such that the function $Cost(S) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} l_{S(i)S(j)}$ is

minimized, where the notation $S(i)$ represents the assigned location of element i in solution S. The flow matrix F is created using distance matrix D according to:

$$f_{ij} = \begin{cases} \frac{1000}{d_{ij}} & \text{if } e_{ij} \in E_{cluster}; \\ \frac{1}{d_{ij}} & \text{otherwise.} \end{cases} \quad (2)$$

Clearly, higher (respectively lower) flows will correspond to elements that are similar (respectively dissimilar). A good solution for the QAP will thus put the elements with a high flow closer in the layout, which is exactly our goal. Additionally, two elements with an edge in $G_{cluster}$ have their flow multiplied by a factor of 1,000, thus enforcing their proximity in the final layout. The matrix L is generated from the distances of points in a square grid of $m = g^2$ positions, with $m \gg n$. In this work, we set $\lceil g = 2\sqrt{n} \rceil$ and l_{qp} is the Euclidean distance between each locations p and q for all $1 \leq q, p \leq m$).

Assume that the graph $G_{cluster}$ contains c disconnected subgraphs ($G^1_{cluster}$, $G^2_{cluster}, ..., G^c_{cluster}$). Then each subgraph $G^i_{cluster}$ becomes a QAP instance and is solved separately. Finally, we solve one last QAP, where each element is a subgraph $G^i_{cluster}$. The instance for this problem is created by building a fully connected graph $G_C(V_C, E_C, w_C)$ where $|V_C| = c$ and the weight $w_{C_{ij}}$ corresponds to the flow between subgraphs $G^i_{cluster}$ and $G^j_{cluster}$, calculated as:

$$w_{C_{ij}} = \frac{\sum_{p \in G_i} \sum_{q \in G_j} f_{pq}}{|V^i_{cluster}| * |V^j_{cluster}|}. \quad (3)$$

In the next section, we will describe the main characteristics of the memetic algorithm used to tackle the QAP problem.

3 Memetic Algorithm

Memetic Algorithms (MAs) is a name that designates a class of powerful population-based metaheuristics with many successful practical applications ([15, 16, 17]). In our MA implementation (see the pseudo-code in Figure 1), we have a population of *agents* composed of two solutions (namely *pocket* and *current*). The idea behind this is that while the *current* solutions are constantly being modified by recombination and mutation, the *pockets* maintain a memory of the best solutions found. The population is organized with a hierarchical ternary tree structure, divided in four overlapped subpopulations of four agents each (one *leader* and three *supporters*). The *supporters* of the first subpopulation are the *leaders* of the others. This population structure has been used before [10] and in Ref. [11] this was the best structure in a comprehensive test of alternative topologies. The method **updatePop**() is responsible for making the best solutions climb the tree towards the upper agents. The method initially verifies the *pocket* solution of each agent, checking whether it is worse than the *current* one. Whenever that happens, the *pocket* is replaced by the

```
memeticAlgorithm()
    pop = initializePop(); updatePop(pop)
    repeat
        for i=0 to 12
            offspring =recombination(selectSol(parentA,parentB))
            localSearchTS(offspring)
        updatePop(pop); 8-neighborLS(agent^0_pocket)
    until max_number_of_generations
```

Fig. 1. Pseudo-code of the memetic algorithm implemented for the QAP.

current. Then, for each subpopulation, the method replaces the leader's *pocket* solution with the best supporter's *pocket* whenever the latter has better cost. A solution is represented as an integer array S of size n, where $S(i) = k$ means that the element i is assigned to location k. The agents are initialized with random solutions, where the elements are spread uniformly at random across all the available locations. Also during the initialization step, we optimize the *pocket* solutions by applying a local search that incorporate a Tabu Search (see Section 3.2).

3.1 Recombination

Concerning the selection of the parent solutions, the method **selectSol**() uses two strategies, depending on whether the population has lost diversity or not. We consider that a population is diverse if its *pocket* solutions differ at least in one value from a set of 20% of randomly chosen positions. If *diversity has not been lost*, one of the parents is the *pocket* solution of a leader agent. The second parent is the *pocket* solution from a supporter agent *within the same subpopulation*. The new solution created replaces the *current* solution of the supporter agent selected. On the contrary, if *diversity has been lost*, both parents are *pocket* solutions from supporter agents. However, in this case *the agents belong to different subpopulations*. The offspring replaces the *current* solution in one of the supporter agent. Once the parents were selected, a recombination algorithm is used to create a new solution. Our memetic algorithm uses a similar recombination to that introduced by Merz [9] and it is explained with the help of a step-by-step example described in Figure 2. Initially, all the elements assigned to the same location in both parents are copied to the offspring (elements A and E). Afterwards, we select at random an unassigned element from the offspring, say D, and look at its location in one of the parents, say parent 2. Thus, the method assigns location #3 to element D. Next, we look at the location of D in parent 1 (i.e. location #1) and check which element is in location #1 in parent 2 (i.e. element G), assigning its location to the offspring (i.e. element G goes to location #1). The process is repeated, now checking the location of element G in parent 1 (location #4). However, as location #4 is not present in parent 2, the process stops. We repeat the process starting with element H in parent

1. After processing all the elements in the offspring, element B still does not have a location because both locations #3 and #12 have already been taken. This does not happen when $n = m$. In Ref. [9] the authors do not envision this possibility because they considered only the case $n = m$. In this case, we consider a straight path between those locations and choose a random location over it, in this case location #6. If all the locations along the line have already been taken, a random one from any of the parents is chosen. Complementary to

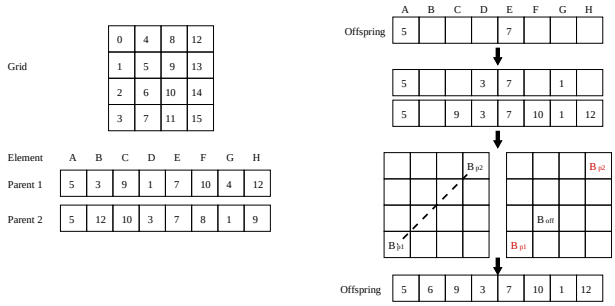


Fig. 2. A step-by-step description of the crossover procedure for the QAP problem.

the recombination operator, the mutation swaps the locations of three randomly selected elements in the solution. We use a 3-element swap scheme because in the **localSearchTS**() method (explained next in Section 3.2) all the 2-element swap movements are already considered. Mutation is always applied over the offspring after recombination.

3.2 Local Search algorithms

We implemented two local search methods (see Figure 1). (**localSearchTS**()) includes a Tabu Search implementation [12] and it is described next. The neighborhood of a solution S is defined by the swap of all pairs of elements of S. The algorithm chooses the swap that causes the best improvement in the QAP objective function. If such swap does not exist, we perform the swap that least worsens the solution. After a swap is done, any swap that brings the elements back to their previous positions become *tabu* for a number of iterations. However, a *tabu* swap shall be accepted if the objective function value of the new solution is better than the incumbent – i.e. an *aspiration criterion*. This local search is applied on each *pocket* solution of the population at the beginning of the MA and on each *current* solution after the recombination phase.

The second local search method (**8-neighborLS**()) iteratively selects an element at random and tries to move it to the eight surrounding locations in the grid, using a best-improvement strategy. Every time an element is moved to a new position, we test again its eight adjacent locations, until no improvement is possible anymore. The process iterates for all elements of the solution. As

this algorithm performs just a fine-tuning of the solution, it is applied only to the *pocket* solution of the leader agent of the population.

4 Computational Results

The memetic algorithm was coded using Java SDK 1.5.1 and generated the graph layout for the S&P100 dataset in less than 20 seconds of CPU time in a 3.0 GHz Pentium IV machine with 512Mb of RAM. The resulting graph contains 10 clusters and is shown in Figure 3. In this instance all the elements are labelled according to the industrial sector that they belong to; this allows us to better analyze the quality of the layout. Initially, this analysis takes into consideration each cluster defined by the proximity graph, as we expect those clusters to reflect the classification by industrial sector. Then, within each cluster, we will analyze any relevant structure uncovered by the QAP. Because of the space restrictions, we only give the analysis of one cluster.

Cluster #8 could be easily classified as a *services* cluster because 10 of its 17 elements belong to that sector. However, a better classification of the elements in this cluster could be obtained using the information from the layout produced by our method. In the left side of the layout there are four companies related with the packaging industry (**Alcoa**, **Du Pont**, **Allegheny Technologies** and **3M**) and two related with paper products (**OfficeMax** and **Weyerhaeuser**). These companies have been joined together with **International Paper**, which has a participation in both industries. Next to them, we can find the two railroad companies, **Norfolk Southern** and **Burlington Northern Santa Fe**. Finally there is a group of seven companies (**Black & Decker**, **Limited Brands**, **May Department Stores**, **Wal-Mart**, **Radioshack**, **Home Depot** and **Sears**) mainly related with the stores industry. The last company of this cluster is **Rockwell Automation**. It has no clear relation with the other companies, but as a *conglomerate* we cannot consider it an outlier. To compare our layout we use the classical dendogram (Figure 4) obtained with a hierarchical clustering method provided by the European Bioinformatics Institute (EBI)[2], using average linkage (UPGMA) clustering based on "correlation measure based distance" (uncentered). Even though the clustering methods developed at EBI are aimed to analyze biological datasets, their hierarchical clustering is a general approach which can be used in datasets from any source. The input is also the second derivatives of the weekly stock prices. While some technological sectors seems present, the dendogram analysis has its problems. Clusters are only defined when we "cut" the tree. Our methodology managed to automatically separate most of the sectors into distinct natural clusters uncovering similarities in the dynamics of groups of stocks. In addition, for the clusters without a sound sector majority, the QAP created a layout where the elements from different sectors were organized into smaller groups (e.g. clusters #7 and #8). The

[2] http://ep.ebi.ac.uk/EP/EPCLUST/

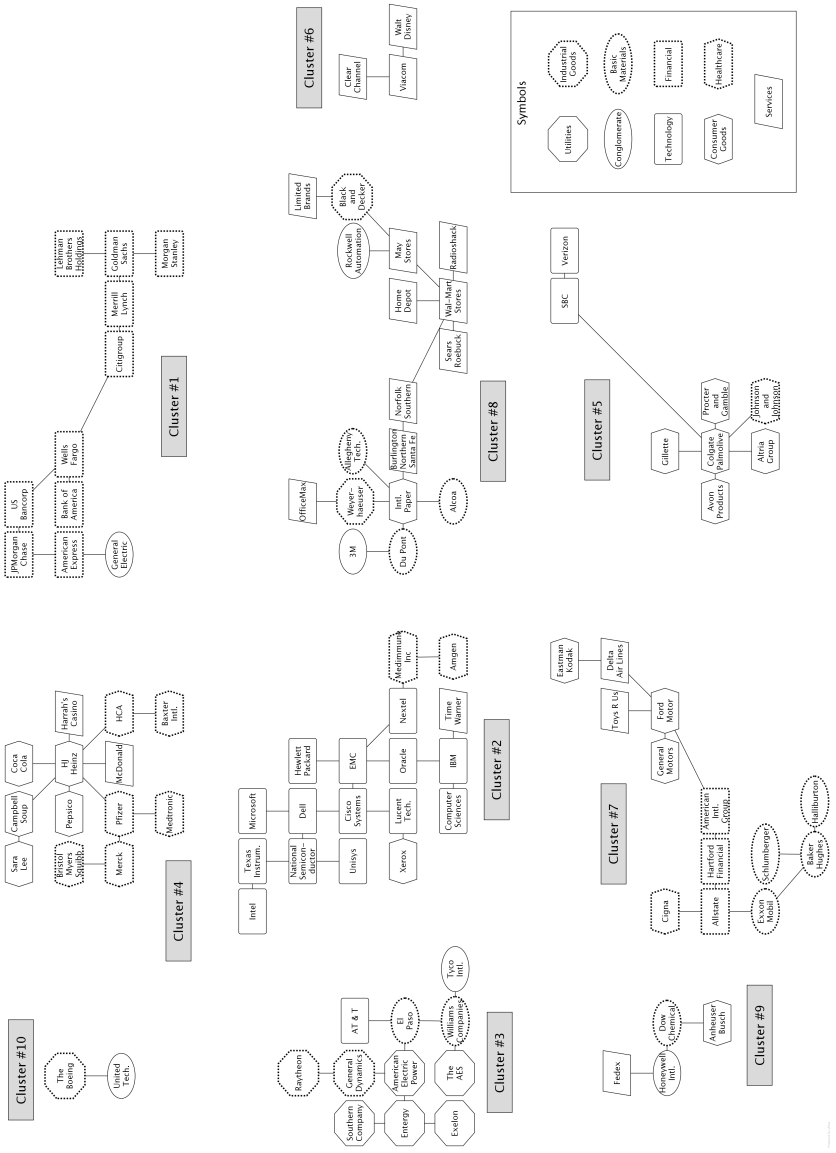


Fig. 3. Graph layout for the S&P100 dataset. The memetic algorithm solved one QAP for each cluster and an extra QAP considering each cluster as a single element, obtaining the final layout. Each shape indicates a different industrial sector represented in the dataset.

quality of the results for the S&P100 dataset supports the use of this method as a new clustering/visualization tool for other time-series data analysis problems. Our visualization methodology is not restricted to the clustering method used

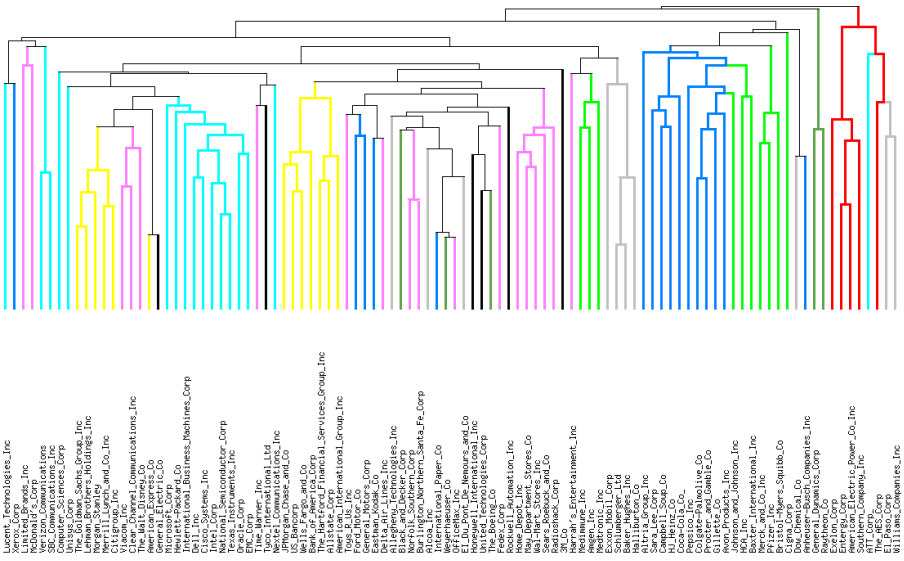


Fig. 4. Dendogram representing the average linkage hierarchical clustering from EBI.

here and the use of the QAP model and a memetic algorithm can be a good alternative for another approach to the automatic layout of weighted graphs.

5 Conclusions

A methodology for correlated data visualization based on the Quadratic Assignment Problem (QAP) was introduced in this paper. It uses a recursive partition approach that divides the dataset into small clusters. A memetic algorithm solves a separate QAP for each subgraph. The final solution obtained by our methodology correctly identifies the majority of the sectors present in a stock market dataset. In some cases, elements from distinct sectors but similar dynamical behaviors are located within the same component of the proximity graph. The success in obtaining a high-quality layout for a financial market instance – which is a challenge due to their inherent near-chaotic behaviour – makes us believe that this technique can be very useful for other problem domains.

References

1. V.V. Gafiychuk, B. Yo. Datsko, and J. Izmaylova. Analysis of data clusters obtained by self-organizing methods. *Physica A: Statistical Mechanics and its Applications*, 341:547–555, 2004.

2. N. Basalto, R. Bellotti, F. De Carlo, P. Facchi, and S. Pascazio. Clustering stock market companies via chaotic map synchronization. *Physica A: Statistical Mechanics and its Applications*, 345(1-2):196–206, 2005.
3. V. Boginski, S. Butenko, and P.M. Pardalos. On structural properties of the market graph. In A. Nagurney, editor, *Innovations in Financial and Economic Networks*, pages 28–45. Edward Elgar Publishing Inc, 2003.
4. S. Sahni and T. González. P-complete approximation problems. *Journal of the Association for Computing Machinery*, 23(3):555–565, 1976.
5. R. Burkard, E. Çela, P. Pardalos, and L. Pitsoulis. The quadratic assignment problem. In P. Pardalos and D. Du, editors, *Handbook of Combinatorial Optimization*, pages 241–338. Kluwer Academic Publishers, 1998.
6. E. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17(4-5):443–455, 1991.
7. C.A. Oliveira, P.M. Pardalos, and M.G.C. Resende. Grasp with path-relinking for the quadratic assignment problem. In C.C. Ribeiro and S.L. Martins, editors, *Lecture Notes in Computer Science*, volume 3059, pages 356–368. Springer-Verlag, 2004.
8. J. Carrizo, F.G. Tinetti, and P. Moscato. A computational ecology for the quadratic assignment problem. In *Proceedings of the 21st Meeting on Informatics and Operations Research*, Buenos Aires, Argentina, August, 1992.
9. P. Merz and B. Freisleben. A comparison of memetic algorithms, tabu search and ant colonies for the quadratic assigment problem. In *Proceedings of the 1999 International Congress of Evolutionary Computation (CEC'99)*, Washington DC, USA, 6-9 July, 1999.
10. R. Berretta and P. Moscato. The number partitioning problem: An open challenge for evolutionary computation? In D. Corne and M. Dorigo, editors, *New Ideas in Optimization*, pages 261–278. McGraw-Hill, 1999.
11. L.S. Buriol, P.M. Franca, and P. Moscato. A new memetic algorithm for the asymmetric traveling salesman problem. *Journal of Heuristics*, 10(3):483–506, 2004.
12. F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, Massachusetts, 1997.
13. M. Ausloos and K. Ivanova. Mechanistic approach to generalized technical analysis of share prices and stock market indices. *The European Physical Journal B*, 27:177–187, 2002.
14. J.M. González-Barrios and A.J. Quiroz. A clustering procedure based on the comparison between the k nearest neighbors graph and the minimal spanning tree. *Statistics & Probability Letters*, 62(1):23–34, 2003.
15. P. Moscato, C. Cotta, and A. Mendes. Memetic algorithms. In G. Onwubolu and B. Babu, editors, *New Optimization Techniques in Engineering*, pages 53–86. Springer-Verlag, 2004.
16. P. Moscato. Memetic algorithms. In P. Pardalos and M. Resende, editors, *Handbook of Applied Optimization*. Oxford University Press, New York, NY, USA, 2002.
17. P. Moscato and C. Cotta. Memetic algorithms. In T.F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*. Chapman & Hall/CRC, 2006. to appear.

Knowledge Perspectives in Data Grids

Luis Eliécer Cadenas, Emilio Hernández

Universidad Simón Bolívar, Departamento de Computación y T. I.,
Apartado 89000, Caracas 1080-A, Venezuela

Abstract. In this paper a methodology for accesing scientific data repositories on data grids is proposed. This methodology is based on ontology specification and knowledge representation. The concept of *Knowledge Perspective* is introduced, as the action of applying particular scientific conjectures or theories to the interpretation of experimental data and information. Data grid environments provide high levels of security and virtualization, which allow the users to create new data services on the data server side. These new services are based on the user's knowledge perspective. An implementation of this concept is presented, on a Globus-enabled Java execution platform.

1 Introduction

Computationally intensive technologies are very important in many areas of scientific research. These technologies are currently used to process, either locally or in distributed environments, considerable amounts of data and information. A new term has been coined to reference scientific research strongly dependent on computational and net-based collaboration: e-science [1]. Distributed platforms for data processing, increasingly known as grids, provide basic technologies for integrating multi-institutional sets of computational resources to support data processing. However, available tools are far from offering the levels of flexibility and capability required to transit the long way between data processing and knowledge generation. In this paper we propose and evaluate the concept of *knowledge perspective*, a tool for managing scientific data and experimental information in Data Grids environments. We define a knowledge perspective, or simply a *perspective*, as the consequence of applying a formalization of a theory to scientific data in order to help in the interpretation of experimental data and information.

In principle, scientific theories can be formalized as sets of universal quantified sentences, using First Order Logic (FOL). By selecting a set of such sentences we can define a theoretical framework (i.e an interpretation or viewpoint) for a specific experimental dataset. This selection may define relevant facts for the contrastation process of a particular theory. We can define, using FOL, concepts, properties, relations and sentences (i.e. closed formulas) that represent subsets of a particular scientific theory. In the context of processing a data source (or a combination of several data sources) for knowledge generation, there could be a first processing level in which the "raw" data is processed in

order to generate annotations and/or indexes. These indexes and annotations could highlight the relevant facts of the data according to the theory. In further processing levels the annotations can be semantically correlated in order to corroborate theories or conjectures.

The main contribution of this work is a computational model that allows the users to process data, in the context of Data Grids, which is epistemologically consistent with the nature of the scientific research activity. The users can safely create their own knowledge perpectives on the server or grid side, without the intervention of grid or system administrators. The operational base helps us manipulate and process efficiently very big distributed data sources in Data Grids. We implemented this model using SUMA/G [2], a distributed architecture for execution of Java programs which is implemented on top of Globus.

The rest of this paper is organized as follows. Section 2 formalizes the knowledge perspective concept and its relationship with the scientific and research activity. Section 3 introduces a general architecture to implement a knowledge perspective service in grids environments. Section 4 shows a practical example of the usage of this system to a bibliographic data source. Section 5 revises related work and section 6 offers our conclusions and future work.

2 Knowledge Perspectives

We define the concept of Knowledge Perspective from the definition of three sets. Lets Γ be the set that represents the objects x_i in the data source:

$$\Gamma = \{x_1, x_2, x_3, \ldots x_{n_2}\}$$

Given a set of predicates $P = \{p_1, p_2, p_3, \ldots p_{n_1}\}$, where each p_i represent attributes or relationships among elements of Γ, we can define Ω, which is a set of sets Φ_i:

$$\Omega = \{\Phi_1, \Phi_2, \Phi_3, \ldots \Phi_{n_1}\}$$

where the elements in each set Φ_i are tuples with elements in Γ satisfying the predicate p_i. Each p_i stands for a property or relationship in the ontology used to process the data source and could be organized in a taxonomical hierarchy. This hierarchy is described using description logic formalisms. This process is a first step to produce the knowledge perspective. Normally, elements of Ω (i.e sets Φ_i) are the product of annotating the data source using the concepts or properties p_i.

Λ is a possibly empty set of closed formulas (i.e. sentences) of predicate logic $A_i = W_i(x_1, x_2 \ldots x_{h_i})$. Each A_i represents conjectures or definitions about objects, properties and relationships in the data source, based on atoms p_i in P.

A Knowledge Perspective is then defined as an ordered tuple of sets:

$$\Pi = (\Gamma, \Omega, \Lambda)$$

In order to process a Knowledge Perspective we define at least two steps.

First, the annotation process over the data source, which consists in checking which objects are related through the predicate p_i. In order to do so, the user should provide the methods to verify each predicate over the data source. These methods are used to annotate the data source, probably producing indexes to objects having the property or standing in the relationship represented by p_i.

We define then the second stage of a knowledge perspective computation as the process of producing a set $\Omega^{'}$ using Ω and Λ. We can say that tuples $(\Gamma, \Omega, \Lambda)$ and $(\Gamma, \Omega^{'}, \Lambda)$ represent the same Knowledge Perspective. However, the validation of the conjectures A_i can be considered as the production of new knowledge, restricted to the data sources analyzed and using the vocabulary contained in P.

As an example to illustrate the previous definitions we can think of Γ as a data repository with astronomical images, Ω as a collection of sets of stars where each set has all the stars with the same apparent magnitude. Λ could be a set of predicate logic formulas (i.e. sentences or assertions in the theory) explaining the formation of supernovas, as a consequence of changes in the apparent magnitude within particular time frames. The computation of the apparent magnitude (i.e. the process to produce Ω) is done through an ontological annotation of the elements in Γ, and could be the product of processing the images or the result of using some existing catalogue.

3 Knowledge Perspectives implementation in Data Grids

We implement perspectives as new services, installed directly on the data source by the users. This is possible in data grids because of the security levels they provide. This approach has several advantages. Firstly, the user could send a short specification in a high level language (i.e. FOL) and the process is done at the data source. In this way it is possible to reduce the cost of data transfers. Secondly, it would facilitate data processing in places with legal restrictions for data transfers. In third place, it permits multiple views about the same data set. In this way different researchers or members of a comunity can share different points of view for the same data. Finally, new data services can evolve with the data source through updating mechanisms of the defined knowledge perspectives. Any data provider should offer, in addition to a normal data access service, a mechanism to process data *in-situ* and hosting services associated with data models installed by authorized users.

3.1 Architecture

The proposed architecture provides services to install new data queries and access services. These new services are built by processing the original data sets, providing in this way an additional perspective.

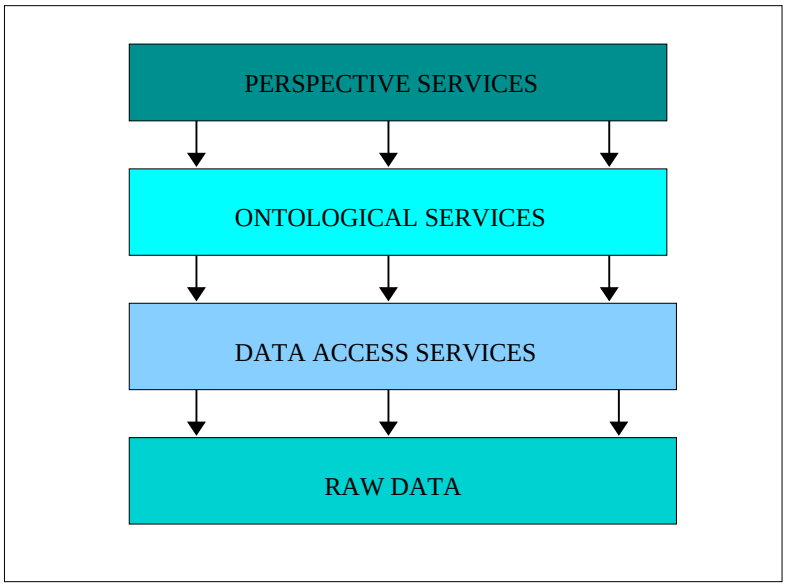


Fig. 1. Service Levels

Figure 1 shows the proposed architecture from the point of view of the services required in the Grid to offer perspective services. We defined the interfaces (API) required at each level and the related operational semantic. Through these interfaces we can virtualize the knowledge perspective service and integrate the same concept across many architectures, facilitating the deployment of distributed perspectives.

We propose a three layer architecture:

- The *Data Access Services* layer defines basic interfaces and the required services to access data sources. This service level would be typically installed by the data provider and offers abstractions to manipulate data sources, regardless the data format.
- The *Ontological Services* layer defines the interfaces and services required to create, store, manipulate and reason over ontologies. (i.e. UploadOntology, CheckOntology, etc). Using services at this level, the users can design an ontology which is adequate to their perspectives, with the required description of objects, properties and relationships. The users must then develop methods to produce the first level annotations. After this process the users obtain what we call the (*perspective 0*) level. Finally, the users develop the set of assertions (conjectures set and logical inferences) to be applied to *perspective 0* data in order to produce the *perspective 1* output.
- Finally, the *Perspective Manipulation Service* layer defines basic interfaces required to create, store and manipulate perspectives as objects, at both *perspective 0* and *perspective 1* levels (i.e. MakePerspective, QueryPerspective).

The execution of these services materialize both perspective levels by creating indexes using the data source and the user-provided ontology and theory. The first processing level establishes a match among objects in the data source and the satisfied predicates. The second one uses logical assertions in the Λ set to produce new satisfied inferences.

Currently, perspective, data and ontology services are defined and installed in SUMA/G[3], a grid infrastructure to execute Java bytecode in distributed environments, based on Globus services. In this software architecture we implemented a metaservice to install new services (SIMG). A *service* in this context is defined using a *service name*, a list containing all the services required to execute the new service (*requirements*), an *API*, a *documentation* and the set of packages that implements the service. The infrastructure offers great flexibility to install new services represented by java objects. This java object could have a constructor to annotate the data source and offers methods to access the annotated data source. For a future version we are defining a specialized *proxy* to query distributed databases processed using the perspective service. We provide facilities to query the data source, through the perspective service, using an option called *submit*. This option executes the queries asynchronously, and the results are stored temporarily in the execution agent. The user can ask at any time for these results using a mediator.

The service for installing new services directly by the users (SIMG) is crucial for the developing and installation of knowledge perspectives as defined in this work. The main reason is flexibility, because the users can process remote data transparently, i.e. in the same way they would process local data, in a secure way.

4 An example using Wordnet

As a proof of concept we implemented an example that allows us to improve data recovery from a Mysql database that contains information about scientific papers. We used Wordnet, a lexicographic reference system, available online [4]. The database was installed in an execution agent of SUMA/G, together with database access services, ontologies and perspectives as described in section 3.1. We used a Prolog version of Wordnet and developed a metainterpreter. The metainterpreter receives as input an english word and produces recursively as output an RDF file representing a taxonomical subtree with all the hiponyms of the word. This ontology is computed automatically by the metainterpreter and can be manipulated using primitives and methods provided through perspective and ontologies services. Using our perspective service we produce an index that points to papers which mention in the title any of the words contained in the hyponim tree.

In this example the data source is a relational database that contains information about scientific papers. Each table in this database represents a type of

object in the universe. We use an ontology (*science* [5]) to describe the objects represented in the data source. We defined an RDF Schema to describe in a generic way any relational database. This schema is shown in figure 2.

```
<rdf_:Tabla rdf:about="&rdf_;kb_db_00055"
 rdf_:Nombre_de_tabla="Profesores"
 rdf_:Representa="Science:Academic−Staff"
 rdf_:tamano="7912"
 rdfs:label="kb_db_00055">
<rdf_:tiene_atributos rdf:resource="&rdf_;kb_db_00056"/>
<rdf_:tiene_atributos rdf:resource="&rdf_;kb_db_00058"/>
<rdf_:tiene_atributos rdf:resource="&rdf_;kb_db_00059"/>
<rdf_:tiene_atributos rdf:resource="&rdf_;kb_db_00060"/>
</rdf_:Tabla>
<rdf_:Atributo rdf:about="&rdf_;kb_db_00056"
 rdf_:Longitud_atributo="50"
 rdf_:Nombre_Atributo="Science:First−Name"
```

Fig. 2. RDF description of the database

Through this schema we describe the objects in our data source and the meanings they stand for, using an ontology as reference (*science*). For example, the relation *Talkabout* could be defined in such a way that express the user's perspective. *Talkabout(paper,biology)* would mean that *paper* is a scientific paper about biology. In the predicate *Talkabout* the second argument is taken from a controlled vocabulary (i.e. the subtree of hyponim relationships produced through the metainterpreter). We want to process the data source to identify all the objects in the relationship *Talkabout*.

In this example, when we process a perspective, an index over the data source is generated. This index is an interpretation in the framework of a particular theory. Each sentence in the theory used to produce the perspective (each sentente A_i in Λ) generates a table with as many columns as the arity of A_i plus one column identifying the predicate. In our example, the only relation is hyponymy. The following sentences show a part of the subtree produced by the word *biology*:

$$\forall(X)Embriology(X) \rightarrow Biology(X)$$
$$\forall(X)Botany(X) \rightarrow Biology(X)$$
$$\forall(X)Phytology(X) \rightarrow Biology(X)$$

For this example our perspective $\Pi = (\Gamma, \Omega, \Lambda)$ is defined as follows:

– Γ has scientific papers. In order to identify properties, predicates and relevant objects in the table we have used a description based on a RDF Schema and the *Science* Ontology.

– Ω are all the papers X_i satisfying the predicate TalkAbout(X_i,Biology).
– $\Omega^{'}$ has all the papers X_i added beacuse it satisfies the sentences in Λ.
– Λ has the transitive clausure of the hyponim relation in the Biology subtree.

In this way we produce an index for each word in the subtree using an RDF Schema and the science ontology to clarify the meaning of table names in the original database. This is the annotation process at the *perspective 0* level. Then we use the hyponym relations to add relations between words in the index, corresponding to our second level of annotation *perspective 1*.

Once the perspective is represented by an index (or by any other data structure implemented by the user) later queries take a considerably shorter time. In other words, from the point of view of performance, the *perspective 0* creation could take a long time, depending on the size of the database and the kind of processing performed on the raw data. However, once created, the annotations and indexes will speed up further processing, such as *perspective 1* creation and later queries and conjecture validations. In our example, such queries to the paper database take a time in the order of a few milliseconds, when executed from a remote computer located in the same local area network.

5 Related Work

Semantic techniques on grid environments can be roughly classified into two groups: those that provide knowledge about the grid resources and those that provide knowledge about the data grid contents [6]. The first one is used to describe, discover, manipulate and compose services while the second one is used to produce more knowledge through ontological resources in order to describe and discover new data relationships. In [7] a general architecture is proposed, in which there is a clear separation between the semantic grid level and the knowledge grid level. The semantic grid level uses ontologies to describe services in the grid while the knowledge grid level uses semantic techniques to process data and produce knowledge. Some of these proposals are based on computer agents [8] which can offer autonomy and negotiation capabilities to grid environments [9]

The Semantic Grid research community is mainly working on developing techniques using ontologies in order to improve knowledge access and recovery in the grid [10][11][12][13][14]. Ontology languages and reasoning techniques are fundamental to describe resources and services in this framework [15][16]. Most of the languages being considered use description logic to provide an automatic classification of resources and services with a model theoretic semantic. Recently, some proposals account for the lack of nonmonotonic reasoning techniques and rule languages usage in order to implement some of the requirements of the semantic web and semantic grid communities (for example negotiation of services) [17]. A main concern is to provide the adequate level of expressivity without loosing decidibility or tractability. The capability to describe resources

and services in a declarative language helps us to create automatic discovering and composition techniques which could improve the current capability of the grid to produce new knowledge.

The Virtual Data System [18] is an architecture for data virtualization. Using the virtual data language *VDL* users can describe workflows over datasets. Data transformation processes could be discovered and composed. Metadata about transformations, derivations, and datasets are registred in the distributed virtual data catalog. The *Knowledge Grid* [19] is an architecture for distributed data mining. The system uses ontologies [20] to describe data mining services and help users to elaborate data mining workflows. Comb-e-chem [21] is creating the infrastructure to analyze correlations and predict properties in chemical structures using techniques known as publication at source. Comb-e-chem provides services to create workflows, aggregate experimental data, select datasets and also annotate and edit data sources. Using the concept of *publication at source* all these data can be reused many times. MyGrid [22] offers an infrastructure to support research in bioinformatic. MyGrid provides data and resource integration services using semantic technologies to improve service discovery, data flow and distributed processing. Comparatively our proposal offers:

- A technique to link logical theories, described using FOL and Description Logics with data sources. This link explicitly shows relations among theories and data subsets producing indexes. These indexes improve data access in large datasets.
- Facilities to use a high level language (FOL) to describe data processing in data grids. Our data modeling process is completely defined with reference to FOL sentences. Annotation methods required to make *Perspective 0* annotations could be provided as libraries. In this way a researcher needs only to define the process by using FOL.
- A processing technique which leaves the data source unchanged.
- A flexible way to create views over data. Each user could have her own perspective over each data set.
- A process to identify objects, properties and relations in the framework of an arbitrary, user defined, theory. In this way the researcher could identify data objects confirming the theory used to process it.
- A technique for processing data at the source, avoiding issues related to the transfer of large amounts of data.
- An architecture of ontology services to implement the knowledge perspective concept.
- A technique to provide many points of view over data, increasing opportunities of knowledge discovery and scientific advance.

This is achieved through the combination of (1) a methodology based on ontology specification and knowledge representation and (2) appropriate data grid services that allow users to define their own ontological services.

6 Conclusions and Future Work

In this work we propose a methodology that establishes a bridge between data manipulation techniques based on ontological criteria and secure data access in grids. We base this methodology in a concept we call *Knowledge Perspective* which allows researchers to manipulate scientific data according to a theoretical framework.

From the viewpoint of knowledge representation and management, we propose the use of a high level language (First Order Logic) and a specification about how to compute a knowledge perspective. Using the grid environment each user could have the authorization level and enough computational and data resources to create indexes in the data source. We present a Globus-enabled Java platform that allows the users to define their own data services based on ontological description of the data. Both contributions allow the grid users to define new services and data access interfaces, consistent with their own knowledge perspectives.

Our initial results, reported in this paper, show the feasibility of using this concept when applied to frameworks where the information has low complexity levels. We need further research and tests for larger and more complex datasets. We describe distributed data sources using ontologies, facilitating data mediation and integrated access to heterogeneus data sources. We plan to implement further mediation techniques in the future. Ongoing research is oriented to applying and evaluating this technology in databases where the data objects are more complex, such as images. In this case the predicates associated to the objects can be satisfied using image processing algorithms.

References

1. Hey, T., Trefethen, A.: "e-science and its implications". Philosophical Transactions of the Royal Society **361**(1809) (2003) 1809–1825
2. Blanco, E., Cardinale, Y., Figueira, C., Hernndez, E., Rivas, R., Rukoz, M.: Remote data service installation on a grid-enabled java platform. In: 17th International Symposium on Computer Architecture and High Performance Computing SBAC-2005. (2005) 85–91
3. Cardinale, Y., Hernández, E.: Parallel Checkpointing on a Grid-enabled Java Platform. Lecture Notes in Computer Science (European Grid Conference EGC2005) (2005) To appear.
4. Fellbaum, C.: Wordnet: An Electronic Lexical Database. MIT Press (1999)
5. Freitas, F.: Ontology of science. Technical report, Universidade Federal de Santa Catarina (2001)
6. Goble, C., De Roure, D., Shadbolt, N., Fernandes, A.: Enhancing services and applications with knowledge and semantics. In Foster, I., Kesselman, C., eds.: The Grid 2: Blueprint for a New Computing Infrastructure. Morgan-Kaufmann (2004)

7. Goble, C., De Roure, D.: The semantic grid: Myth busting and bridge building. In: 16th European Conference on Artificial Intelligence (ECAI-2004), Valencia, Spain (2004) 1129–1135
8. Rana, O.F., Pouchard, L.: Agent based semantic grids: Research issues and challenges. Journal of Parallel and Distributed Computing Practices (2003)
9. Roure, D.D., Shadbolt, N., Jennings, N.: The semantic grid: Past, present and futur. In: Proceedings of The IEEE. (2005)
10. Goble, C., De Roure, D.: The semantic web and grid computing. In Kashyap, V., Shklar, L., eds.: Real World Semantic Web Applications. Volume 92 of Frontiers in Artificial Intelligence and Applications. IOS Press (2002)
11. Cannataro, M., Talia, D.: Semantics and knowledge grids: Building the next-generation grid. IEEE Intelligent Systems **19**(1) (2004) 56–63
12. Chen, L., Shadbolt, N., Tao, F., Puleston, C., Goble, C., Cox, S.: Exploiting semantics for e-science on the semantic grid. In: Web Intelligence (WI2003) workshop on Knowledge Grid and Grid Intelligence. (2003) 122–132
13. De Roure, D., Hendler, J.: E-science: the grid and the semantic web. IEEE Intelligent Systems **19**(1) (2004) 65–71
14. Newhouse, S., Mayer, S., Furmento, S., McGough, S., Stanton, J., Darlington, J.: Laying the foundations for the semantic grid. In: AISB Workshop on AI and Grid Computing. (2002)
15. Horrocks, I.: Daml-oil: A reason-able web ontology language. In: Proceedings of EDBT. Number 2287 in Lecture Notes in Computer Science, Springer (2002) 2–13
16. et al, M.D.: Owl: Web ontology language 1.0 reference. Technical report, World Wide Web Consortium (2002)
17. Kifer, M., Bruijn, J.d., Boley, H., Fensel, D.: A realistic architecture for the semantic web. In: International Conference on Rules and Rule Markup Languages for the Semantic Web. (2005)
18. Foster, I., Voeckler, J., Wilde, M., Zhao, Y.: The virtual data grid: A new model and architecture for data-intensive collaboration. In: CIDR 2003 Conference on Innovative Data System Research. (2003)
19. Cannataro, M., Talia, D.: The knowledge grid. CACM **46**(1) (2003) 89–93
20. Cannataro, M., Comito, C.: A data mining ontology for grid programming. In: 1st International Workshop on Semantics in Peer-to-Peer and Grid Computing (SemPGrid2003). (2003)
21. Frey, J.G., Bradley, M., Essex, J., Hursthouse, M., Lewis, S., Luck, M., Moreau, L., De Roure, D., Surridge, M., Welsh, A.: Combinatorial chemistry and the grid. In Berman, F., Hey, A.J., Fox, G.C., eds.: Grid computing: making the global infrastructure a reality. Wiley Series in Communications Networking and Distributed Systems. John Wiley & Sons Ltd., Chichester, UK (2003) 945–962
22. Goble, C., Pettifer, S., Stevens, R., Greenhalgh, C.: Knowledge integration: In silico experiments in bioinformatics. In Foster, I., Kesselman, C., eds.: The Grid: Blueprint for a New Computing Infrastructure Second Edition. Morgan Kaufman (2004)

On the Class Distribution Labelling Step Sensitivity of CO-TRAINING

Edson T. Matsubara, Maria C. Monard, and Ronaldo C. Prati

Department of Computer Science
ICMC/USP - São Carlos
Laboratory of Computational Intelligence - LABIC
P.O. Box 668
13560-970 São Carlos,SP, Brazil.
{edsontm,mcmonard,prati}@icmc.usp.br

Abstract. CO-TRAINING can learn from datasets having a small number of labelled examples and a large number of unlabelled ones. It is an iterative algorithm where examples labelled in previous iterations are used to improve the classification of examples from the unlabelled set. However, as the number of initial labelled examples is often small we do not have reliable estimates regarding the underlying population which generated the data. In this work we make the claim that the proportion in which examples are labelled is a key parameter to CO-TRAINING. Furthermore, we have done a series of experiments to investigate how the proportion in which we label examples in each step influences CO-TRAINING performance. Results show that CO-TRAINING should be used with care in challenging domains.

1 Introduction

Semi-supervised learning uses a set of examples where only a few examples are labelled, and the goal is to predict the labels of the remaining unlabelled examples. The main idea of semi-supervised learning is to investigate ways whereby using the unlabelled data it is possible to effectively improve classification performance, compared with a classifier build only using the labelled data, *i.e.* without considering the unlabelled data. For these reasons, semi-supervised learning is considered as the middle road between supervised and unsupervised learning.

Methods that have been proposed under this paradigm include the multi-view semi-supervised CO-TRAINING method (1), dealt with in this work. CO-TRAINING applies to datasets that have a natural separation of their attributes into at least two disjoint sets, so that there is a partitioned description of each example into each distinct view. For each view, the set of few labelled examples is given to learning algorithms to induce independent classifiers. Each classifier is used to classify the unlabelled data in its respective view. Afterwards, examples which have been classified with a higher degree of confidence for all views are included in the set of labelled examples and the process is repeated

using the augmented labelled set until a stop criterion is met. However, due to the limited number of initial training examples available in semi-supervised learning, it is not possible to estimate the class distribution of the dataset in advance. Furthermore, when examples are labelled, as there is no information concerning class distribution, we do not know in which class proportion the higher confidence labelled examples should be included in the set of labelled examples in each iteration. This is a question of practical importance, and in this work we analyse the effect of class distribution in CO-TRAINING. Experimental results of CO-TRAINING performance with respect to accuracy, number of incorrectly labelled examples and AUC show that, although the best results are obtained if the true class distribution of the examples is known, for some domains where there is a great separability among classes the performance of CO-TRAINING can also be competitive when this information is not available. However, CO-TRAINING should be used with caution in challenging domains.

The rest of this work is organised as follows: Section 2 presents related work on semi-supervised learning. Section 3 describes CO-TRAINING. Section 4 discusses the class distribution sensitivity problem. Section 5 reports the experimental results, and Section 6 concludes the work.

2 Related Work

Semi-supervised learning algorithms can be divided into single-view and multi-view (2; 3). In a single-view scenario the algorithms have access to the entire set of domain attributes. Single-view algorithms can be split up into transductives (4), Expectation Maximization (EM) variations (5), background knowledge based algorithms (6) and graph-based methods (3). In a multi-view setting, the attributes are presented in subsets (views) which are sufficient to learn the target concept. Multi-view algorithms are based on the assumption that the views are both *compatible* (all examples are labelled identically by the target concepts in each view), and *uncorrelated* (given the label of any example, its descriptions in each view are independent)

The CO-TRAINING algorithm provides the basis for multi-view learning. Following CO-TRAINING some multi-view learning algorithms have been proposed, such as: CO-EM (7) which combines EM and CO-TRAINING; CO-TESTING (2) which combines active and semi-supervised learning, and CO-EMT (2) an extension of CO-TESTING with CO-EM. The use of Support Vector Machines (SVM) instead of *Naive Bayes* (NB) as the base-learning learner is proposed in (8). An improved version of CO-EM using SVM is proposed in (9) showing experimental results that outperform other algorithms. CO-TRAINING requires the instance space to be described with sufficient and redundant views. On the other hand, the TRI-TRAINING algorithm (10) neither requires this nor imposes any constraints on the supervised learning algorithm; its applicability is broader than previous CO-TRAINING style algorithms. The majority of these applications and related work barely consider the class distribution.

3 The CO-TRAINING Algorithm

Given a set of N examples $E = \{E_1, ..., E_N\}$ defined by a set of M attributes $\mathbf{X} = \{X_1, X_2, ..., X_M\}$ and the class attribute Y, where we only know the class attribute for a few examples, CO-TRAINING needs at least two disjoint and compatible views D_1 and D_2 of the set of examples E to work with. In other words, for each example $j = 1, 2...N$ in D_1 we should have its j-th counterpart (compatible example) in D_2. We shall refer to these two views as $\mathbf{X}_{D_1}$ and $\mathbf{X}_{D_2}$ such that $\mathbf{X} = \mathbf{X}_{D_1} \cup \mathbf{X}_{D_2}$ and $\mathbf{X}_{D_1} \cap \mathbf{X}_{D_2} = \emptyset$. Furthermore, the set of labelled examples in each view should be adequate for learning.

Set E can be divided into two disjoint subsets L (Labeled) and U (Unlabelled) of examples. Both subsets L and U are further divided into two disjoint views respectively called, L_{D_1}, L_{D_2} and U_{D_1}, U_{D_2}. These four subsets L_{D_1}, L_{D_2}, U_{D_1} and U_{D_2}, illustrated in Figure 1, as well as the maximum number of iterations k, constitute the input of CO-TRAINING described by Algorithm 1.

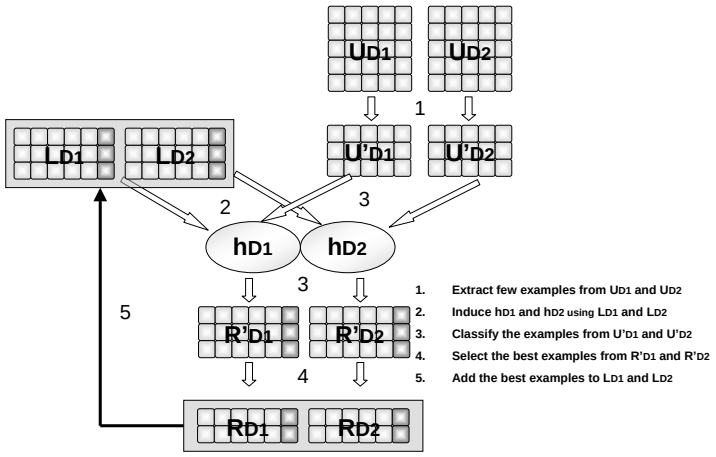


Fig. 1. CO-TRAINING

Initially, two small pools U'_{D_1} and U'_{D_2} of compatible unlabelled examples, withdrawn from U_{D_1} and U_{D_2} respectively, are created, and the main loop of Algorithm 1 starts. First, the sets of training examples L_{D_1} and L_{D_2} are used to induce two classifiers h_{D_1} and h_{D_2}, respectively. Next, the set of examples U'_{D_1} is labelled using h_{D_1} and inserted in R'_{D_1}, and the set of examples from U'_{D_2} is labelled using h_{D_2} and inserted in R'_{D_2}. Both sets of labelled examples are given to the function *bestExamples* which is responsible for ranking compatible examples from R'_{D_1} and R'_{D_2} that have the same class label prediction, and

Algorithm 1: CO-TRAINING

Input: $L_{D_1}, L_{D_2}, U_{D_2}, k$
Output: L_{D_1}, L_{D_2}
Build U'_{D_1} and U'_{D_2} as described;
$U_{D_1} = U_{D_1} - U'_{D_1}$;
$U_{D_2} = U_{D_2} - U'_{D_2}$;
for $i = 0$ *to* k **do**
  Induce h_{D_1} from L_{D_1};
  Induce h_{D_2} from L_{D_2};
  $R'_{D_1} = h_{D_1}(U'_{D_1})$ set of classified examples from U'_{D_1};
  $R'_{D_2} = h_{D_2}(U'_{D_2})$ set of classified examples from U'_{D_2};
  $(R_{D_1}, R_{D_2}) = bestExamples(R'_{D_1}, R'_{D_2})$;
  $L_{D_1} = L_{D_1} \cup R_{D_1}$;
  $L_{D_2} = L_{D_2} \cup R_{D_2}$;
  if $U_{D_1} = \emptyset$ **then return**(L_{D_1}, L_{D_2}) **else**
    Randomly select compatible examples from U_{D_1} and U_{D_2} to replenish U'_{D_1} and U'_{D_2} respectively;
  end
end
return(L_{D_1}, L_{D_2});

selecting from them the "best" pairs of compatible examples to be inserted in L_{D_1} and L_{D_2} respectively. After that the process is repeated until a stop criterion is met — either the maximum number of iterations defined by the user or the set U_{D_1} (or its counterpart U_{D_2}) is empty.

Algorithm 1 describes the general idea of CO-TRAINING using the same base-learning learning algorithm (*Naive Bayes* in the original proposal) which makes it possible to construct a third classifier from h_{D_1} and h_{D_2} called combined classifier (1). Furthermore, Algorithm 1 only uses two visions and binary class datasets. However, as suggested by its authors, there are several features that can be included in the original version. Our implementation of CO-TRAINING includes several such features which enable us to test its behavior under different situations. These features include: more than two visions; more than two classes; variable number of examples and proportion of examples by class in the initial labelled sets L_{D_i} as well as sets U'_{D_i}; different base-learning algorithms; maximum number of "best" classified examples in each class that can be inserted in L_{D_i} during each iteration, and others.

4 Class proportion labelling sensitivity of CO-TRAINING

A common assumption in the design of standard learning algorithms is that training examples are drawn from the same underlying distributions the model is expected to make predictions. In CO-TRAINING, though, this assumption does not hold because the training set of examples is growth while the algorithm is

running, and the amount of labelled examples, as well as the proportion in which examples are labelled, is generally a parameter of the algorithm set by the user.

For example, suppose we are using CO-TRAINING to label data for web page classification. In a typical application, we construct a robot crawler that visits some web sites and downloads all pages of interest. We then ask a human expert to hand label some web pages with the classes we are interested in. As we generally do not know how many examples should be labelled for each class, a fair option is to ask the expert to label an even number of examples for each class. Another option is to draw a small sample of examples and ask the expert to label this sample. Although one may argue that the latter option would produce a more reliable estimate of the class distribution than the former, this is not necessarily true as the crawler might have some bias when retrieving web pages. Thus, in both cases we do not have a good estimate of which proportion we should label examples in each CO-TRAINING iteration.

As CO-TRAINING is an iterative process, where examples labelled in previous iterations are used to build models to label new data, in this work we argue that the proportion in which examples are labelled is a key parameter of the CO-TRAINING algorithm. The main point is that we may not know beforehand the true underlying distribution we should use as a parameter for CO-TRAINING beforehand. As the base-classifier might be sensitive to class skews, feeding the algorithm with a class distribution different from the true one would bias the base-classifier used by CO-TRAINING towards an inaccurate classifier. As a consequence, the number of examples incorrectly labelled would increase, degrading the performance of CO-TRAINING.

Although it is very difficult to characterize the effect that changing class distribution would have in learning algorithms, several studies evaluate its behaviour for a number of well-known algorithms. (11) conducts an extensive experimentation using the decision tree algorithm C4.5 with datasets sampled under several different class distributions. The authors conclude that, on average, the natural class distribution produces the most accurate classifiers. (12) claims that when the independence assumption of attributes is violated, the Naive Bayes algorithm is affected by changing class distributions. The author shows that this sensitivity also holds for other algorithms, such as logistic regression and hard margin SVMs. (13) further extends these results claiming that the sensitivity could not only be attributed to the learning system but also to the dataset at hand. As CO-TRAINING uses learning algorithms as base-classifiers, this sensitivity is automatically inherited from the learning system. The next section shows how this sensitivity affects the results for the datasets used in our experiments.

5 Experimental Evaluation

We carried out an experimental evaluation using three different text datasets: a subset of the UseNet news articles (20-NewsGroups) (14); abstracts of academic papers, titles and references collected from *Lecture Notes in Artificial Intelligence* (LNAI) (15) and links and web pages from the COURSE dataset (1).

For the first dataset we created a subset of the 20-newsgroups selecting 100 texts from `sci.crypt`, `sci.electronics`, `sci.med`, `sci.space`, `talk.politics.guns`, `talk.politics.mideast`, `talk.politics.misc` and `talk.religion.misc`. All texts from the first 4 newsgroups were labelled as `sci` (400 - %50) and texts from the remaining newsgroups were labelled as `talk` (400 - %50). The LNAI dataset contains 396 papers from *Case Based Reason* (277 - 70%) and *Inductive Logic Programming* (119 - 30%). The COURSE dataset[1] consists of 1051 web pages collected from various Computer Science department web sites, and divided into several categories. This dataset already provides the two views for each web page example. One view consists of words appearing on the page, and the other view consists of the underlined words from other pages which point to the web page. However, analysing the examples in the original dataset, we found 13 examples which are either empty (no text) or its compatible example in the counterpart view is missing. Thus, the original dataset was reduced to 1038 examples. Similar to (1), web pages were labelled as course (221 - 20%), and the remaining categories as non-course (817 - 80%).

Using PRETEXT [2], a text pre-processing tool we have implemented (16), all text datasets were decomposed into the attribute value representation using the bag-of-words approach. Stemming and Luhn cut-offs were also carried out. For datasets NEWS and LNAI the two views were constructed following the approach we proposed in (17), using *1-gram* representation as one view and *2-gram* as the second view of the datasets. For the *2-gram* view in the NEWS dataset, the minimum Luhn cut-off was set to 3. For the remaining views, the minimum Luhn cut-off was set to 2. The maximum Luhn cut-offs were left unbounded. For dataset COURSE *1-gram* was used in both views, named TEXT and LINKS. Table 1 summarises the datasets used in this work. It shows the dataset name (Dataset); number of documents in the dataset (#Doc); number of generated stems (#Stem); number of stems left after performing Luhn cut-offs in each view (#Attributes), and class distribution (%Class).

As all datasets are completely labelled, we can compare the labels assigned by CO-TRAINING in each iteration with the true labels of the datasets. In other words, we use CO-TRAINING in a simulated mode, in which the true labels are hidden from the algorithm and are only used to measure the number of examples wrongly labelled by CO-TRAINING. In our experiments we used *Naive Bayes* (NB) as a CO-TRAINING base-classifier. In order to obtain a lower bound of the error that CO-TRAINING can reach on these datasets, we measured the error

[1] `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-51/www/co-training/data/`

[2] `http://www.icmc.usp.br/~edsontm/pretext/pretext.html`

rate of NB using all labelled examples using 10-fold cross-validation. Results (mean error and respective standard deviation) are shown in the last column (NB Error) of Table 1.

Dataset	#Doc	View	#Stem	#Attr.	Class	%Class	NB Error	Overall Error
NEWS	800	1-gram	15711	8668	`sci`	50%	2.5 (1.7)	1.6 (1.0)
					`talk`	50%	0.8 (1.2)	
		2-gram	71039	4521	`sci`	50%	2.0 (2.0)	1.3 (1.2)
					`talk`	50%	0.5 (1.1)	
LNAI	396	1-gram	5627	2914	ILP	30%	1.7 (3.7)	1.5 (1.8)
					CBR	70%	1.4 (1.9)	
		2-gram	21969	3245	ILP	30%	1.8 (1.7)	1.8 (1.7)
					CBR	70%	1.5 (1.9)	
COURSE	1038	TEXT	13198	6870	`course`	20%	16.3 (5.4)	6.5 (2.3)
					`non-course`	80%	3.8 (2.0)	
		LINKS	1604	1067	`course`	20%	9.6 (7.6)	14.6 (3.5)
					`non-course`	80%	16.0 (4.7)	

Table 1. Datasets description and *Naive Bayes* error

To assess the behaviour of CO-TRAINING using cross-validation, we adapted the sampling method as follows: first, the examples in both views are paired and marked with an ID. Then, we sample the folds so that both training and test samples are compatible, *i.e.*, an example marked with a given ID appears only in the training or test sample in both views.

All experiments were carried out using the same number of initial labelled examples (30 examples) evenly distributed by class (50% - 50%). In each iteration, up to 10 "best" examples were allowed to be labelled. Furthermore, to analyse the impact of the class distribution we varied the number of examples in each class. We used 0.6 as a threshold to select the best examples, *i.e.* compatible candidates must have been labelled by NB with a probability greater than 0.6.

Table 2 shows the mean value and standard deviation of results obtained using 10-fold cross validation. The first line indicates the maximum number of examples by class that can be labelled in each iteration: `sci`/`talk` for NEWS, ILP/CBR for LNAI and `course`/`non-course` for COURSE dataset. For each dataset the first four lines show the number of examples in each class that have been wrongly (W) or rightly (R) labelled; LSize is the number of examples labelled by CO-TRAINING, including the 30 initial examples; USize is the number of unlabelled examples left; Error and AUC are respectively the error rate and the area under the ROC curve of the combined classifier, and Wrong is the total number of examples wrongly labelled. The best mean results for these last three measures are in bold.

For all datasets CO-TRAINING ended due to reaching the condition of an empty set of unlabelled examples in iterations 64, 28 and 86 for datasets NEWS, LNAI and COURSE respectively. As can be observed, best results for NEWS and

COURSE datasets are obtained whenever examples are labelled considering the dataset distribution (5/5 for NEWS and 2/8 for COURSE). For LNAI dataset, although the best result is not obtained for its exact proportion 3/7, it is obtained by its similar proportion 2/8. For this dataset, labelling examples using a slight biased proportion towards the minority and most error-prone class (see Table 1) seems to improve classification. In both cases the total number of labelled examples is the same (LSize $\simeq$ 300). The main difference is in the error of each class: while 3/7 proportion labels all CBR examples correctly, 2/8 proportion labels all ILP examples correctly.

Moreover, for the best results the mean error rate of the combined classifiers are compatible with the once obtained using the labelled examples (Table 1), although the COURSE dataset presents a far greater variance.

	2/8	3/7	5/5	7/3	8/2
NEWS dataset					
sci(W)	18.00 (26.45)	10.60 (15.47)	1.10 (1.85)	0.40 (0.52)	0.80 (0.42)
sci(R)	344.50 (2.72)	339.40 (2.50)	325.70 (11.51)	203.60 (0.52)	139.50 (1.51)
talk(W)	1.60 (1.17)	2.20 (0.63)	5.70 (10.03)	42.50 (30.34)	131.00 (18.89)
talk(R)	139.40 (1.17)	201.80 (0.63)	324.30 (10.03)	345.70 (1.89)	347.80 (3.08)
LSize	503.50 (26.53)	554.00 (15.30)	656.80 (9.77)	592.20 (30.07)	619.10 (17.00)
U'Size	206.50 (26.53)	156.00 (15.30)	53.20 (9.77)	117.80 (30.07)	90.90 (17.00)
Error	3.00 (3.24)	2.38 (3.70)	**1.88 (2.14)**	6.25 (5.14)	19.00 (3.53)
AUC	0.98 (0.02)	0.98 (0.03)	**0.99 (0.02)**	0.97 (0.04)	0.92 (0.05)
Wrong	19.80 (26.96)	12.80 (15.80)	**6.80 (11.77)**	43.70 (30.29)	133.50 (19.31)
LNAI dataset					
ilp(W)	0.00 (0.00)	1.30 (1.25)	5.40 (1.71)	9.30 (3.23)	12.30 (5.10)
ilp(R)	69.00 (0.00)	94.20 (2.20)	101.00 (1.49)	100.80 (1.14)	101.70 (1.57)
cbr(W)	0.70 (0.95)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
cbr(R)	230.30 (0.95)	204.00 (0.00)	150.00 (0.00)	96.00 (0.00)	69.00 (0.00)
LSize	300.00 (0.00)	299.50 (1.08)	256.40 (2.41)	206.10 (3.54)	183.00 (5.10)
U'Size	50.00 (0.00)	50.50 (1.08)	93.60 (2.41)	143.90 (3.54)	167.00 (5.10)
Error	**1.26 (1.33)**	2.02 (2.00)	2.03 (1.07)	3.28 (1.69)	4.80 (3.03)
AUC	**1.00 (0.00)**	1.00 (0.01)	0.99 (0.01)	0.99 (0.01)	0.99 (0.01)
Wrong	**0.70 (0.95)**	1.30 (1.25)	5.60 (1.90)	9.30 (3.23)	12.50 (5.04)
COURSE dataset					
course(W)	34.40 (29.73)	103.90 (66.05)	252.30 (72.89)	423.40 (27.35)	434.80 (112.58)
course(R)	146.00 (26.82)	132.80 (27.26)	155.50 (13.34)	175.40 (6.00)	179.30 (10.89)
ncourse(W)	5.30 (3.13)	7.20 (8.00)	4.20 (4.59)	1.50 (2.92)	2.40 (3.34)
ncourse(R)	505.20 (154.07)	307.10 (227.37)	146.80 (110.20)	81.60 (31.65)	81.30 (56.98)
LSize	690.90 (150.92)	551.00 (186.16)	558.80 (49.82)	681.90 (23.39)	697.80 (66.62)
U'Size	239.10 (150.92)	379.00 (186.16)	371.20 (49.82)	248.10 (23.39)	232.20 (66.62)
Error	**14.11 (13.26)**	32.65 (20.15)	49.43 (15.95)	61.91 (8.07)	60.29 (17.28)
AUC	**0.92 (0.08)**	0.82 (0.11)	0.71 (0.09)	0.68 (0.07)	0.67 (0.07)
Wrong	**40.20 (31.71)**	112.80 (67.28)	258.70 (72.08)	429.80 (25.59)	442.60 (111.98)

Table 2. CO-TRAINING results for NEWS, LNAI and COURSE datasets

Analysing the behaviour of CO-TRAINING when changing the class distribution of labelled examples shows an interesting pattern. For the balanced dataset NEWS, skewing the proportion of labelled examples towards the `talk` class (*i.e*, labelling more examples from the `talk` class: 7/2 and 8/2) does not diminish the performance significantly. The other way dramatically increases the error rate (from 1.88 in 5/5 labelling to 19.00 in 8/2 labelling) as well as in the

number of examples incorrectly labelled (6.8% to 133.50%). For the imbalanced datasets the picture is clearer. Both the error rate and the number of incorrectly labelled examples increase as we go towards the opposite direction in terms of proportion of labelled examples.

Another interesting result is related to the AUC. For the datasets with high AUC values — NEWS and LNAI —(near 1), the degradation in performance is weaker than for the COURSE dataset. This is because AUC values near 1 are a strong indication of a domain with a great separability, *i.e.*, domains in which the classes could be more easily separated from the others, and it is easy for the algorithm to construct accurate classifiers even if the proportion of examples in the training set is different from the natural one.

6 Conclusions and Future Work

In this work we analyse, for a fixed set of few labelled examples, the relationship between the unknown class distribution of domains and CO-TRAINING performance with respect to which proportion we should label examples in each iteration. Experimental results evaluated using the labelling accuracy, combined classifier error rate and AUC show that the best performance is achieved whenever we label examples in a proportion equal or close to the natural class distribution present in the datasets. Furthermore, labelling examples in proportions very different from the natural class distribution seems to decrease CO-TRAINING performance, especially in challenging domains. These results should be interpreted as a warning to anyone who is using CO-TRAINING for data labelling.

As future work, we are investigating ways to neutralise or overcome the class proportion labelling dependency of CO-TRAINING. (12) presents some methods aimed at correcting the class proportion when this proportion is not known in a classification context. It would be interesting to adapt this method to CO-TRAINING learning. A possible adaptation would be to label examples in the same proportion as the best examples appear in the L' set. This approach leads to labelling a flexible proportion of examples in each iteration and could bias the class distribution in the L set towards the natural one. However, experimental research should be carried out to analyse the feasibility of this approach.

Acknowledgements: The authors would like to thank FAPESP (Process 2005/03792-9) and CAPES, Brazil, for financial support.

References

[1] Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proc. 11th Annu. Conf. on Comput. Learning Theory, ACM Press, New York, NY (1998) 92–100

[2] Muslea, I.: Active Learning with Multiple Views (2002) PhD Thesis, University Southern California.

[3] Zhu, X.: Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison (2005) http://www.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf.
[4] Vapnik, V.: Statistical learning theory. John Wiley & Sons (1998)
[5] Nigam, K., Ghani, R.: Analyzing the effectiveness and applicability of co-training. In: Conference on Information and Knowledge Management. (2000) 86–93
[6] Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: Proc. of the 18th Int. Conf. on Machine Learning. (2001) 577–584
[7] Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.M.: Text classification from labeled and unlabeled documents using EM. Machine Learning **39** (2000) 103–134
[8] Kiritchenko, S., Matwin, S.: Email classification with co-training. Technical report, University of Otawa (2002)
[9] Brefeld, U., Scheffer, T.: Co-EM Support Vector Learning. In: Proc. of the Int. Conf. on Machine Learning, Morgan Kaufmann (2004) 16
[10] Zhou, Z.H., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. In: IEEE Transactions on Knowledge and Data Engineering. Volume 17. (2005) 1529–1541
[11] Weiss, G.M., Provost, F.J.: Learning when training data are costly: The effect of class distribution on tree induction. J. Artif. Intell. Res. (JAIR) **19** (2003) 315–354
[12] Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In Brodley, C.E., ed.: Proc of the 21st Int. Conf. on Machine Learning (ICML 2004), ACM (2004) 114–121
[13] Fan, W., Davidson, I., Zadrozny, B., Yu, P.S.: An improved categorization of classifier's sensitivity on sample selection bias. In: Proc of the 5th IEEE Int. Conf. on Data Mining (ICDM 2005), IEEE Computer Society (2005) 605–608
[14] Blake, C., Merz, C.: UCI Repository of Machine Learning Databases (1998) http://www.ics.uci.edu/~mlearn/MLRepository.html.
[15] Melo, V., Secato, M., Lopes, A.A.: Automatic extraction and identification of bibliographical information from scientific articles (in Portuguese). In: IV Workshop on Advances and Trend in AI, Chile (2003) 1–10
[16] Matsubara, E.T., Martins, C.A., Monard, M.C.: Pretext: A pre-processing text tool using the bag-of-words approach. Technical Report 209, ICMC-USP (2003) (in portuguese) ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/RT_209.zip.
[17] Matsubara, E.T., Monard, M.C., Batista, G.E.A.P.A.: Multi-view semi-supervised learning: An approach to obtain different views from text datasets. In: Advances in Logic Based Intelligent Systems. Volume 132., IOS Press (2005) 97–104

Two new feature selection algorithms with Rough Sets Theory

Yailé Caballero[(1)], Rafael Bello[(2)], Delia Alvarez[(1)], Maria M. Garcia[(2)]

[(1)]Department of Computer Science, University of Camagüey, Cuba.

{yaile, dalvarez}@inf.reduc.edu.cu

[(2)]Department of Computer Science, Universidad Central de Las Villas, Cuba.

{rbellop, mmgarcia}@uclv.edu.cu

Abstract: Rough Sets Theory has opened new trends for the development of the Incomplete Information Theory. Inside this one, the notion of reduct is a very significant one, but to obtain a reduct in a decision system is an expensive computing process although very important in data analysis and knowledge discovery. Because of this, it has been necessary the development of different variants to calculate reducts. The present work look into the utility that offers Rough Sets Model and Information Theory in feature selection and a new method is presented with the purpose of calculate a good reduct. This new method consists of a greedy algorithm that uses heuristics to work out a good reduct in acceptable times. In this paper we propose other method to find good reducts, this method combines elements of Genetic Algorithm with Estimation of Distribution Algorithms. The new methods are compared with others which are implemented inside Pattern Recognition and Ant Colony Optimization Algorithms and the results of the statistical tests are shown.

1. Introduction

Feature selection is an important task inside Machine Learning. It consists of focusing on the most relevant features for use in representing data in order to delete those features considered as irrelevant and that make more difficult a knowledge discovery process inside a database. Feature subset selection represents the problem of finding an optimal subset of features (attributes) of a database according to some criterion, so that a classifier with the highest possible accuracy can be generated by an inductive learning algorithm that is run on data containing only the subset of features [Zho01].

Rough Sets Theory was proposed by Z. Pawlak in 1982 [Paw82] and had received many extensions from his author that can be reviewed in [Paw91], [Paw94] and [Paw95]. The Rough Set philosophy is founded on the assumption that some information is associated with every object of the universe of discourse [Kom99a] and [Pol02]. Rough Set Model has several advantages to data analysis. It is based only on the original data and does not need any external information; no assumptions about data are necessary; it is suitable for analyzing both quantitative and qualitative features, and results of Rough Set Model are easy to understand [Tay02]. Several toolkits based on rough sets to data analysis have been implemented, such as Rosetta [Ohr97], and ROSE [Pre98]. An important issue in the RST is about feature selection.

An important issue in the RST is about feature reduction based on reduct concept. A reduct is a minimal set of attributes $B \subseteq A$ such that $IND(B)=IND(A)$, where $IND(X)$ is called the X-indiscernibility relation. In other words, a reduct is a minimal set of attributes from A that preserves the partitioning of universe (and hence the ability to perform classifications) [Kom99b].

The employment of reducts in the selection and reduction of attributes has been studied by various authors, among them are [Koh94], [Car98], [Pal99], [Kom99b], [Ahn00] and [Zho01].

However, this beneficial alternative is limited because of the computational complexity of calculating reducts. [Bel98] shows that the computational cost of finding a reduct in the information system that is limited by l^2m^2, where l is the length of the attributes and m is the amount of objects in the universe of the information system; while the complexity in time of finding all the reducts of information system is $O(2^lJ)$, where l is the amount of attributes and J is the computational cost required to find a reduct. However, good methods of calculating reducts have been developed, among them are those based on genetic algorithms, which allow you to calculate reducts with an acceptable

cost [Wro95], [Wro96], and [Wro98]; and others based on heuristic methods [Deo95], [Cho96], [Bel98], and [Deo98].

In this paper, two new methods for feature selection and its experimental results are presented: one of them using an evolutionary approach (epigraph 2) and the other by a greedy algorithm with heuristic functions (epigraph 3), which uses Rough Sets Theory.

2. Feature selection by using an evolutionary approach

The evolutionary approach had been used to develop methods for calculating reducts. Genetic Algorithms (GA) are search methods based on populations: Firstly, a population of random individuals is generated, the best individuals are selected, and lastly, the new individuals that make up the population will be generated using the mutation and crossover operators. In [Wro95], three methods for finding short reducts are presented. These use genetic algorithms and a greedy method and have defined the adaptability functions *f1*, *f2* and *f3*.

An adaptation of the Genetic Algorithm plan is the Estimation of Distribution Algorithms (EDA) [Muh99] but most of them don't use crossover or mutation because the new population is generated from the distribution of the probability estimated from the selected set. The principal problem of the EDA is the estimation of *ps(x, t)* and the generation of new points according to this distribution in a way that the computational effort is reasonable. For this reason, different approaches have been introduced to obtain the estimation of *ps(x, t)*.

One of the members of this family is the Univariate Marginal Distribution Algorithm (UMDA) for discrete domain [Muh98], which shows taking into account only univariate probabilities. This algorithm is capable of optimizing non-lineal functions, always and when the additive variance (lineal) of the problem has a reasonable weight in the total variance. The UMDA for continuous domain was introduced in 1999. In every generation and for every variable, the UMDA carries out statistic tests to find the density function that best adjusts to the variable. UMDA for continuous domain is an algorithm of structure identification in the sense that the density components are identified through hypothesis tests.

We have defined a method for calculating reducts starting from the integration of the adaptability functions (*f1*, *f2*, y *f3*) of the methods reported by Wróblewski in [Wro95] and the UMDA method, obtaining satisfactory results which are shown in Table 1. The values of the parameters that were used were: $N = 100$; $g = 3000$; $e = 50$; $T = 0.5$; where N is the number of individuals, g is the maximum number of evaluations that will be done, e is elitism, which means that the best 50 pass directly to the next generation; T is the percentage of the best that were selected to do all the calculations.

Table 1. Results obtained with the proposed Estimation Distribution Algorithms (EDA)

Name of data base	Algorithms with the different functions of Wróblewskii								
	f1			f2			f3		
(CaseCount, FeatureCount)	AT	LR	NR	AT	LR	NR	AT	LR	NR
Ballons-a (20,4)	0.167	2	1	1.860	2	1	0.260	2	1
Iris (150,4)	82.390	3	4	3.540	3	4	17.250	3	4
Hayes-Roth (133,4)	40.830	4	1	30.100	4	1	22.450	4	1
Bupa (345,6)	436	3	6.85	995.300	3	8	466	3	8
E-Coli (336,7)	64.150	3	6.85	1514	3	7	169.200	3	7
Heart (270,13)	337	3	8	2782	3	18	1109	3	17
Pima (768,8)	2686	3	17	6460	3	18.4	4387	3	18.6
Breast- Cancer (683,9)	1568	4	6.55	8250	4	7.83	2586	4	8
Yeast (1484,8)	1772	4	2	12964	4	2	2709	4	2
Dermatology (358,34)	1017	6.05	10.15	15553	6	14.90	30658	6	47
Lung-Cancer (27,56)	7.780	4.2	9.55	0.0956	4	15.95	264.200	4	38.6

AT: Average time required to calculate reducts (in seconds) *LR*: Average length of reducts found

NR: Average number of reducts found

The use of functions described by Wróbleskii [Wro95] in the Estimation of Distribution Algorithms resulted successful. EDA did the calculation of short reducts in little time when the set of examples was not very large (<600 cases), even when the number of attributes that describe the problem was large. The best combination resulted with Wróblewskii's function f1 with respect to the execution time; however f3 found a larger number of reducts in acceptable times.

3 Feature selection using Rough Sets Theory

Rough Sets Theory is a mathematical tool that had been used successfully to discover data dependencies and reduce the number of attributes contained in a dataset by purely structural methods [Jen03].

Reducts that are obtained by using Rough Sets are very informative and all the other attributes can be removed with a minimal information loss due to the use of the degree of dependency measure suggested by Ziarko in [Zia01] and very used by many others authors [Mod93], [Zho01], [Jen03].

Algorithms that calculate reducts are usually designed by using heuristics or random search strategies in order to reduce complexity. Heuristic search is very fast because this is not necessary to wait until the search ends but it doesn't' guarantee the best solution although a better one is known when it is founded in the process.

Now we are able to present RSReduct, a new method for finding reducts with Rough Sets. This is a greedy algorithm that starts with an empty set of attributes and builds good reducts in acceptable times by means of heuristic searches and it works adding the best measurement features by the heuristic function.

The idea of this algorithm is based on criteria of the ID3 method with respect to the normalized entropy and the gain of the attributes [Mit97] and dependency between attributes by means of Rough Sets.

In this algorithm we use the terms *R(A)* and *H(A)* proposed in [Piñ03].

The expression for R(A) which is a relevant measure of the attributes ($0 \leq R(A) \leq 1$) is:

$$R(A) = \sum_{i=1}^{k} \frac{|S_i|}{|S|} e^{(1 - C_i)} \quad (1)$$

Where *k* is the number of different values of feature *A*. *Ci* is the number of different classes present in the objects that have the value *i* for the feature *A*. /*Si*/ the amount of objects with the value *i* in the feature *A*, and /*S*/ is the amount of objects of the training set. This measure maximizes the heterogeneity among objects of different classes and minimizes the homogeneity among objects of the same one.

H(A) is obtained by the following algorithm:

1. For all the attributes of the problem, calculate their *R(A)* and form a vector. Determine the *n* best attributes for the calculations of the previous step. The value of n can be selected by the user. As a result of this step the vector *RM=(R(Ai), R(Aj),..)* with *n* = |*RM*| is obtained.
2. Determine the combinations of *n* in *p* (the value selected by the user) from the selected attributes in step II. The combination vector is obtained.

$$Comb = (\{Ai, Aj, Ak\},...\{Ai, At, Ap\}. \quad (2)$$

3. Calculate the dependency grade of the classes with respect to each one of the combinations obtained in the previous step. As a result of this step, the dependency vectors are obtained.

$$DEP(d) = (k(Comb1, d), k(Comb2, d),...k(Combr, d)) \quad (3)$$

where $$k = \frac{|POS_B(D)|}{|U|} \quad (4)$$ and

$$POS_B(D) = \cup_{B_*(X)} \quad (5)$$

If *k*=1 then d totally depends on *B*, while if *k<1*then *d* partially depends on *B*.

4. For each attribute "*A*" the value of *H(A)* is calculated by the following formula :

$$H(A) = \sum_{\forall i \,/\, A \in Combi} k(Combi, d) \quad (6)$$

Another alternative measure that has been used successfully is the gain ratio [Mit97]:

$$SplitInformation(S, A) = -\sum_{i=1}^{C} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \quad (7)$$

where *C* are the values of attribute *A*. This measure is the entropy of S with respect to attribute A.

The Gain Ratio measure (*G(A)*) is defined in terms of the earlier Gain measure [Mit97] and it means how much information gain produce attribute A or how important is this one to the database, as well a this SplitInformation, as follows:

$$G(A) = \frac{Gain(S, A)}{SplitInformation(S, A)} \quad (8)$$

$$Gain\ (S, A) = Entropy\ (S) - \sum_{v \in values\ (A)} \frac{|S_v|}{|S|} Entropy\ (S_v) \quad (9)$$

where, $values\ (A)$ is the set of possible values by attribute *A* and S_v is the subset of S for which *A* has the value v, that is, $S_v = \{s \in S | A(S) = v\}$.

$$Entropy\ (S) = \sum_{i=1}^{c} -P_i \log_2 P_i \quad (10)$$

where, P_i is the proportion of S belonging to class i.

Schlimmer and Tan in 1993 demonstrate that more efficient recognition strategies are learned, without sacrificing classification accuracy, by replacing the information gain attribute selection measure by the following measure [Mit97]:

$$C(A) = \frac{Gain^2(S, A)}{Cost(A)} \quad (11)$$

where *Cost(A)* is a parameter entered by the user which represents the cost of attribute *A*, a value between 0 and 1.

Nuñez in 1988 describes other measure [Mit97]:

$$C(A) = \frac{2^{Gain(S,A)} - 1}{(Cost(A) + 1)^w} \quad (12)$$

where *Cost(A)* is a parameter entered by the user which represents the cost of attribute *A*, a value between 0 and 1 and *W* is a constant value between 0 and 1 that determines the relative importance of the cost versus information gain.

Considering the measures *R(A)*, *H(A)*, *G(A)* and *C(A)* the new algorithm RSReduct, was written as follows:

Step1. Form the distinction table with a binary matrix *B* $(m^2-m)/2 \times (N+1)$. Each row corresponds to a pair of different objects. Each column of this matrix corresponds to an attribute; the last column corresponds to the decision (treated as an attribute).

Let *b((k,n), i)* an element of *B* corresponding to the pair (*Ok*,*On*) and the attribute *i*, for *i* that belongs to {*1*,....,*N*}

$$b((k,n),i) = \begin{Bmatrix} 1, if a_i(O_k) \neg \Re a_i(O_n) \\ 0, if a_i(O_k) \Re a_i(O_n) \end{Bmatrix} i \in \{1, ...N\} \quad (13)$$

$$b((k,n), N+1) = \begin{Bmatrix} 0, if d_i(O_k) \neq d_i(O_n) \\ 1, if d_i(O_k) = d_i(O_n) \end{Bmatrix} \quad (14)$$

where $\Re$ is similarity relation depending on the type of attribute a_i.

Step2. For each attribute "*A*", calculate the value of *RG(A)* for any of the following three heuristics and then form an ordered list of attributes starting from the most relevant attribute (which maximizes *RG(A)*).

Heuristic1: RG(A)=R(A)+H(A) (15)

Heuristic2: RG(A)=H(A)+G(A) (16)

Heuristic3: RG(A)=H(A)+C(A) (17)

Step3. With *i*=1, *R* = an empty set and (*A1, A2,...An*) an ordered list of attributes according to step 2, consider if *i<=n* then *R=RUAi, i=i+1*.

Step4. If *R* satisfies the Condition I then Reduct = minimal subset $R' \subseteq R$ does meet Condition I, stop (which means end).

$$\forall k, n \quad \forall a_i \in R \quad a_i(o_k) \Re a_i(o_n) \Rightarrow d(o_k) = d(o_n) \text{ (Condition I)}$$

Step5. In other case, repeat from step 3.

The Condition I, in step P4, uses the following relation between the objects x and q for the feature a:

$$q_a \Re x_a \Leftrightarrow sim(x_a, q_a) \geq \varepsilon \text{, where } 0 \leq \varepsilon \leq 1$$

RSReduct algorithm was tested with several datasets from the UCI machine learning repository that is available in the ftp site of the University of California. Some of the databases belong to real world data such as Vote, Iris, Breast Cancer, Iris, Heart and Credit, the other ones represent results obtained in labs such as Balloons-a, Hayes-Roth, LED, M-of-N, Lung Cancer and Mushroom.

The following results were obtained after using RSReduct with the three heuristic functions defined, also the execution time of the algorithm is compiled in each case:

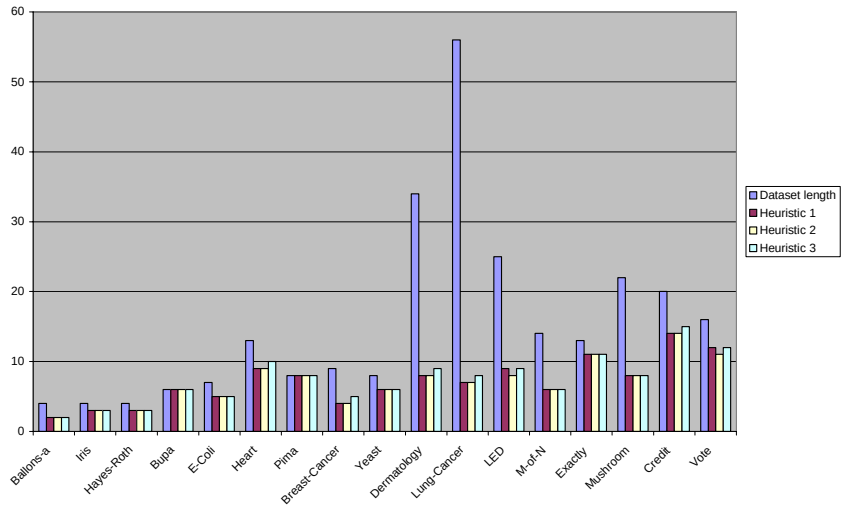
Table 2. Results obtained with the proposed Algorithm according to the different heuristics.

Name of Data Base (CaseCount, FeatureCount)	**Heuristic 1**		**Heuristic 2**		**Heuristic 3**	
	Time (second)	**Length of reduct**	**Time (second)**	**Length of reduct**	**Time (second)**	**Length of reduct**
Ballons-a (20,4)	5.31	2	3.12	2	16.34	2
Iris (150,4)	40.15	3	30.79	3	34.73	3
Hayes-Roth (133,4)	36.00	3	32.30	3	39.00	3
Bupa (345,6)	74.20	6	89.00	6	89.00	6
E-Coli (336,7)	57.00	5	41.15	5	46.60	5
Heart (270,13)	30.89	9	16.75	9	54.78	10
Pima (768,8)	110.00	8	110.00	8	110.00	8
Breast- Cancer (683,9)	39.62	4	31.15	4	32.56	5
Yeast (1484,8)	82.00	6	78.00	6	85.70	6
Dermatology (358,34)	148.70	8	125.9	8	190.00	9
Lung-Cancer (27,56)	25.46	7	18.59	7	31.5	8
LED (226,25)	78.10	9	185.00	8	185	9
M-of-N (1000,14)	230.26	6	162.50	6	79.4	6
Exactly (780,13)	230.00	11	215.00	11	230	11
Mushroom (3954,22)	86.20	8	64.10	8	67.2	8
Credit (876,20)	91.20	14	86.01	14	90.2	15
Vote (435,16)	37.93	12	21.25	11	26.9	12

C(A)→ Nuñez's measure, *Cost (A)*→ aleatories values, *W*=0.1

To illustrate how much was the reduction, the following graphic illustrates the initial length (colored with dark blue) of each dataset and the size of the reduct obtained with the three heuristic functions (colored with red, yellow and light blue respectively):

Figure 1 Reduction of the dataset length by RSReduct



Attending to the size of the reduct obtained, we can conclude that the algorithm is very efficiently. To support this affirmation, experimental results obtained with RSReduct were compared statistically with other feature selection methods implemented with Pattern Recognition (PR) [Alv05], Estimation of Distribution Algorithms (EDA) (epigraph 2) and Ant Colony Optimization Algorithms (ACO) [Cab05]. The tables of the results of the comparison among these methods are omitted, only we will give the results of the statistical tests. In this chance, we used Kruskal-Wallis test, this is a non parametrical test based on rank sums that compares more than two related groups at time in order to discover differences among them. Table 2 shows the *P* Values obtained for Kruskal-Wallis test with respect to execution time of the algorithms, as can be seen; for all the cases the results were lower than 0.05 with a 95% of statistical significance, in other words, there are significant difference among those methods.

Table 3. *P Values* for Kruskal-Wallis test among the three heuristic functions of RSReduct and other feature selection methods.

Representative datasets	P Value Heuristic 1 vs PR, EDA and ACO	P Value Heuristic 2 vs PR, EDA and ACO	P Value Heuristic 3 vs PR, EDA and ACO
Breast Cancer	0.0039	0.0039	0.0039
Lung Cancer	0.002	0.002	0.002
Mushroom	0.0034	0.0034	0.0034
Heart	0.0039	0.0039	0.0039
Dermatology	0.0265	0.0265	0.0265

The conclusion for this analysis is that if a sufficiently good reduct related to length and class differentiation can be obtained in a lower time, then the new method RSReduct decreases the computational cost in classification problems.

3. Conclusions

In this paper, the problem of selecting features by using the reduct concept was studied by presenting two new methods for the selection of attributes one of them combines EDA algorithms with Wroblewski functions and experimental results show that they are very efficiently taking into account that they calculate exhaustively all the reduct for the dataset. The other method is based on heuristics that don't guarantee to find better solution but an optimal one, a good reduct in this case. It was tested on several examples of training sets and experimental results show that this algorithm can build shorter reducts than others and also the computational time is decreased.

References

[Alv05] Álvarez, D. Feature selection for data analysis using Rough Sets Theory. Thesis of Computer Science Engineering. Thesis Director: Yailé Caballero, M.Sc. University of Camagüey, Cuba. 2005.

[Ahn00] Ahn, B.S. et al.. The integrated methodology of rough set theory and artificial neural networks for business failure predictions. Expert Systems with Applications 18, 65-74. 2000.

[Bel98] Bell, D. and Guan, J. Computational methods for rough classification and discovery. Journal of ASIS 49, 5, pp. 403-414. 1998.

[Cab05] Caballero, Y. Using Rough Sets Theory to treatment of the data. Thesis of Master in Computer Science. Thesis Director: Rafael Bello, PhD. Universidad Central de Las Villas, Cuba. 2005.

[Car98] Carlin, U.S. et al.. Rough set analysis of medical datasets and A case of patient with suspected acute appendicitis. In ECAI 98 Workshop on Intelligent data analysis in medicine and pharmacology.

[Cho96] Choubey, S.K. et al. A comparison of feature selection algorithms in the context of rough classifiers. In Proceedings of Fifth IEEE International Conference on Fuzzy Systems, vol. 2, pp. 1122-1128. 1996.

[Cho99] Chouchoulas, A. and Shen, Q. A rough set-based approach to text classification. Lectures Notes in Artificial Intelligence no. 1711, pp. 118-127. 1999.

[Deo95] Deogun, J.S. et al. Exploiting upper approximations in the rough set methodology. In Proceedings of First International Conference on Knowledge Discovery and Data Mining, Fayyad, U. Y Uthurusamy, (Eds.), Canada, pp. 69-74. 1995.

[Deo98] Deogun, J.S. et al. Feature selection and effective classifiers. Journal of ASIS 49, 5, pp. 423-434. 1998.

[Dim66] Dimitriev, A. N.; Zhuravlev, J. I.; Krendeleiev, F. P. . About mathematical principles of objects and phenomenon classification. Diskretnyi Analiz No. 7, pp. 3-15, 1966.

[Gre01] Greco, S. Et al. Rough sets theory for multicriteria decision analysis. European Journal of Operational Research 129, pp. 1-47, 2001.

[Jen03] Jensen R. and Qiang, S. "Finding rough sets reducts with Ant colony optimization". http://www.inf.ed.ac.uk/publications/online/0201.pdf 2003.

[Koc98] Koczkodaj, W.W. et al.. Myths about Rough Set Theory. Comm. of the ACM, vol. 41, no. 11, nov. 1998.

[Koh94] Kohavi, R. and Frasca, B. Useful feature subsets and Rough set Reducts. Proceedings of the Third International Workshop on Rough Sets and Soft Computing. 1994.

[Kom99a] Komorowski, J. Pawlak, Z. et al.. Rough Sets: A tutorial. In Pal, S.K. and Skowron, A. (Eds) Rough Fuzzy Hybridization: A new trend in decision-making. Springer, pp. 3-98. 1999.

[Kom99b] Komorowski, J. et al.. A Rough set perspective on Data and Knowledge. In The Handbook of Data mining and Knowledge discovery, Klosgen, W. and Zytkow, J. (Eds). Oxford University Press, 1999.

[Mau96] Maudal, O. Preprocessing data for neural network based classifiers: Rough sets vs Principal Component Analysis. Project report, Dept. of Artificial Intelligence, University of Edinburgh. 1996.

[Muh98] Mühlenbein H. The equation for the response to selection and its use for prediction. Evolutionary Computation 5(3), pp. 303-346, 1998.

[Muh99] Mühlenbein, H; Mahnig, T.; Ochoa, A. Schemata, distributions and graphical models on evolutionary optimization. Journal of Heuristics, 5(2), pp. 215-247. 1999.

[Ohr97] Ohrn, A. and Komorowski, J.. Rosetta: A rough set toolkit for analysis of data. In Proc. Third Int. Join Conference on Information Science, Durham, NC, USA, march 1-5, vol. 3, pp. 403-407. 1997.

[Pal99] Pal, S.K. and Skowron, A. (Eds).. Rough Fuzzy Hybridization: a new trend in decision-making. Springer-Verlag, 1999.

[Pal02] Pal, S.K. et al. Web mining in Soft Computing framework: Relevance, State of the art and Future Directions. IEEE Transactions on Neural Networks, 2002.

[Paw82] Pawlak, Z. Rough sets. International Journal of Information & Computer Sciences 11, 341-356, 1982.

[Paw91] Pawlak, Z. Rough SetsTheoretical Aspects of Reasoning About Data. Kluwer Academic Publishing, Dordrecht, 1991. En: http://citeseer.ist.psu.edu/context/36378.html

[Paw94] Pawlak, Z. and Skowron, A. “Rough sets rudiments”. Bulletin of International Rough Set Society. Volume 3, Number 3. http://www.kuenstliche-intelligenz.de/archiv/2001_3/pawlak.pdf

[Paw95] Pawlak, Z. “Rough Sets, Rough Relations and Rough functions”. R. Yager, M.Fedrizzi, J. Keprzyk (eds.): Advances in the Dempster – Shafer Theory of Evidence, Wiley, New Cork, pp 251 – 271. 1995 http://citeseer.ist.psu.edu/105864.html

[Piñ03] Piñero, P; Arco, L; García, M. and Caballero, Y. Two New Metrics for Feature Selection in Pattern Recognition. Lectures Notes in computer Science (LNCS 2905), pp. 488-497. Springer, Verlag, Berlin Heidelberg. New York. ISSN 0302-9743. ISBN 3-540-20590-X.

[Pol02] Polkowski, L.. Rough sets: Mathematical foundations. Physica-Verlag, p. 574. Berlin, Germany. 2002.

[Pre98] Predki, B. et al.. ROSE- Software implementation of the Rough Set Theory. In Polkowski, L. and Skowron, A. (Eds) Rough Sets and Current Trends in Computing, Proceedings of the RSCTC98 Conference. Lectures Notes in Artificial Intelligence vol. 1424, Berlin pp. 605-608.

[Tay02] Tay, F.E. and Shen, L.. Economic and financial prediction using rough set model. European Journal of Operational Research 141, pp. 641-659. 2002.

[Wil98] Wilson, Randall. Martinez, Tony R. Reduction Techniques for Exemplar-Based Learning Algorithms. Machine Learning. Computer Science Department, Brigham Young University. USA 1998.

[Wro95] Wroblewski, J. Finding minimal reducts using genetic algorithms. In Wang, P.P. (Ed). Proceedings of the International Workshop on Rough Sets Soft Computing at Second Annual Joint Conference on Information Sciences, North Carolina, USA, p. 679, pp. 186-189. 1995.

[Wro96] Wroblewski, J. Theoretical foundations of order-based genetic algorithms. Fundamenta Informaticae, vol. 28 (3,4), pp. 423-430. IOS Press. 1996.

[Wro98] Wroblewski, J. Genetic algorithms in decomposition and classification problems. In Polkowski, L. and Skowron, A. (Eds.). Rough sets in Knowledge Discovery 1: Applications, Case Studies and Software Systems. Physica-Verlag, pp. 472-492. 1998.

[Zho01] Zhong, N. et al.. Using Rough sets with heuristics for feature selection. Journal of Intelligent Information Systems, 16, 199-214. 2001.