Combined Unsupervised-Supervised Classification Method

Urszula Markowska-Kaczmar and Tomasz Switek

Wroclaw University of Technology, Wroclaw, Poland
`urszula.markowska-kaczmar@pwr.wroc.pl`
`http://www.iis.pwr.wroc.pl/~kaczmar/`

Abstract. In the paper a novel method of classification is presented. It is a combination of unsupervised and supervised techniques. First, the method divides the set of learning patterns into smaller ones in the clustering process. At the end of this phase a hierarchical structure of Self Organizing Map is obtained. Then for the leaves the classification rules are searched. To this end Bee Algorithm is used. The accuracy of the method was evaluated in an experimental way with the use of benchmark data sets and compared with the result of other methods.

1 Introduction

Classification techniques are key elements in solving problems in a number of disciplines. That is why with the rapid development of computer technology, many classification methods have been developed and used: SVMs [7], decision trees [6], neural networks [3], k-Nearest Neighbours (k-NN), Naive Bayes (NB) [6], rule based classification methods (for instance FOIL)[6] and others. They differ in the accuracy and ability of explanation of the classification decision. For numerical attributes some of them are strongly dependent on the discretisation method applied in preprocessing step. Generally, SVMs and neural networks tend to perform much better when dealing with multidimensions and continuous features. For these models a relatively large sample size is required in order to achieve its maximum prediction accuracy whereas NB may need a relatively small data set. Naive Bayes and the k-NN can be easily used as incremental learners whereas rule algorithms cannot. Some algorithms, for instances Naive Bayes is naturally robust to missing values since these are simply ignored in computing probabilities. On the contrary, k-NN and neural networks require complete patterns (without missing values in vectors). Moreover, k-NN is generally considered intolerant of noise because its similarity measure strongly depends on errors in attribute values, thus leading it to misclassify a new instance on the basis of the wrong nearest neighbours. Contrary to k-NN, rule based methods and most decision trees are considered resistant to noise because their pruning strategies avoid overfitting the data in general and noisy data in particular. More discussion about virtues and shortcomings of the methods can be found in [5].

In this paper, we have proposed a novel classification method which is a combination of unsupervised and supervised approaches. Its origin comes from obvious assumption that to solve smaller problem is much easier. That is why in the first phase the clustering is performed giving as the result subsets of patterns for which supervised classification is performed. The next section gives the general view of the method. Its details are presented in the section 3. Experimental study which aim was to evaluate the method is shown in section 4. The paper ends with conclusion summarizing the paper and describing the future plans.

2 CUSC Method – General Overview

CUSC is a novel inductive method of building a classifier. It uses both supervised and unsupervised learning paradigms. The main feature of CUSC is that it divides set of learning patterns into smaller ones in the clustering process using approach similar to Self Organizing Map but the structure of the network has a hierarchical form (a tree). This makes the supervised part of building CUSC easier, as the process is handling less complex sets at the time step.

A creation of the classifier has two phases. In the first phase a clustering method is applied. It is performed with the use of hierarchical neural network which adapts itself to the presented patterns. During training its structure is hierarchical and dynamic assuming form a tree (Fig.1) which is composed of both – structural elements that do not directly influence on classification decisions and leaves that a role is to classify patterns. These classifying elements of CUSC can be divided into two distinct types called *simple form* and *complex form*. The elements type is defined as a *simple form* when after building the structure the training patterns, assigned to it, are belonging to one class only.

The *complex form* elements after clustering contain the set of patterns from more than one class. In other words, the simple form classifying elements contain patterns with the label of a single class. For the *complex form* elements

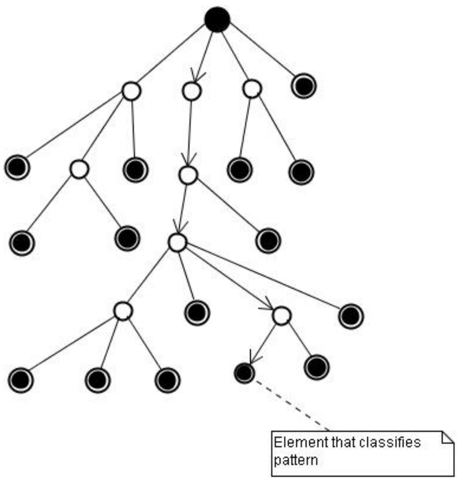

Fig. 1. The tree structure used in CUSC

classification proceeds according to the set of conjunctive rules that are searched in the second phase of the method. This set is used for determining appropriate label for the pattern that is to be classified after building classifier. The method of rules extraction for *complex form* elements in the structure is described in the subsection 3.2.

When the classifier is trained the process of decision making in CUSC consists of two stages. The first one needs to find appropriate classifying element in the tree and in the second one the classification decision is made by the classifying element chosen in the previous stage. The whole classification process is performed in four steps:

```
1) Initialization: set $E_p$ as the root node of a tree and as x the pattern to be classified.
2) If $E_p$ has any descendent nodes go to step 3) otherwise go to step 4).
3) Select a descendent node that presents the highest resemblance to the pattern x according to the similarity measure (for example the smallest Euclidean distance). Set as the $E_p$ the selected descendent node and go to step 2).
4) Classify the pattern x and return a label of a class as a result.
```

The last step, which performs the classification process is dependent on the type of classifying element that it is applied to. In case of *simple form* classifying elements classification is made by returning the label that the element holds. When the classifying element is of *complex form*, the classification is performed by making a decision based on the set of rules in the form of the IF...THEN (eq. 1).

$$IF\ prem_1\ AND\ prem_2\ AND\ ...\ AND\ prem_m\ THEN\ c_i \tag{1}$$

where premise $prem_i$ expresses a condition imposed on the value of attribute x_i. This condition must be satisfied by the pattern to classify it to class c_i.

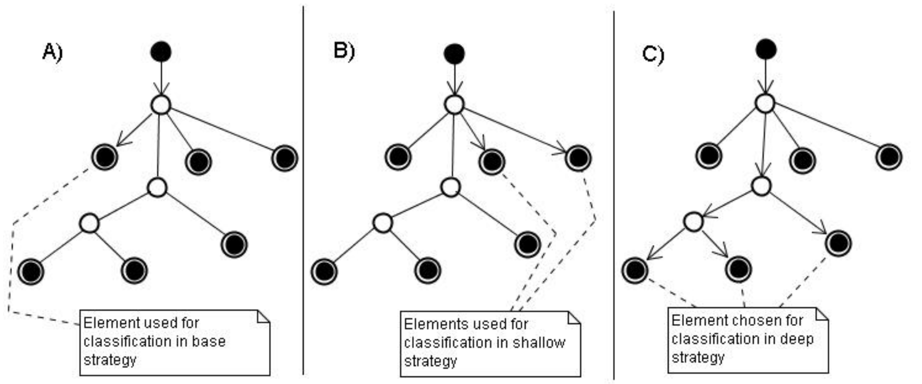
A)
Element used for classification in base strategy
B)
Elements used for classification in shallow strategy
C)
Element chosen for classification in deep strategy

Fig. 2. Strategies of surveying the : A) basic strategy; B) *shallow* strategy (when A) fails); C) *deep* strategy (when A) and B) fail)

In the case that the set of rules cannot be applied to the pattern (no rule is fired for it), the CUSC is trying to find the solution in the neighbourhood of the chosen classifying element. The method applies two strategies of surveying the neighbourhood: *shallow* strategy and *deep* strategy (Fig. 2). The shallow strategy consists in trying to classify the pattern by the classifying elements that are also descendent nodes of the parent of classifying element chosen in the sequence. If the shallow strategy fails in giving the result (lack of classification), the deep strategy is used. The deep strategy consists in determining the most common answer that is given by structural elements of the parent of classifying element chosen in the sequence. The answers given by structural elements are cumulated (the most common answer is chosen) from the sub-trees of these elements. In case that both strategies fail, the CUSC returns an empty label meaning that the classifier doesn't know the answer.

3 The Method in Detail

The process of creating our hybrid classifier can be divided into two phases: building a classifier structure, and a searching for a set of classification rules for complex form elements.

3.1 The First Phase – Building the Structure

This process is unsupervised and proceeds accordingly to the following sequence of steps:
1. Initialization: set the root of a tree as E_p and the initial learning set as X_p .
2. Check whether E_p is fulfilling the STOP condition. If it is TRUE then go to second phase, otherwise go to step 3.
3. Perform clustering of the X_p set. Set the acquired clusters as the children nodes of E_p and assign to them the sub-sets of X_p that they cover. If the clustering results give one cluster go to second phase.
4. For each of the children nodes of the E_p go to step 2. with treating the child node as E_p, and assign to X_p the set of training patterns that belong to E_p.

The STOP condition from the step 2. of this algorithm is preventing the over-growth of the tree structure. To fulfil the STOP condition, the number of patterns in the learning sub-set assigned to E_p must be smaller than the preset threshold, which dependent on the number of patterns in the initial learning set.

The clustering in CUSC is performed with the use of *Neural Clustering* (NC) method. NC is strongly related to SOM network. However, in contrary to SOM networks, NC is dynamically building the clusters structure allowing for their fast growth, and then gradual degradation. In this method the temperature and the mass of neurons are introduced. They have an influence on similarity and weight modification functions. With the timestep k the temperature decreases according to the eq.2.

$$T(k+1) = T(k) - \frac{1}{2} \cdot \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2}$$

where

$$x = \frac{1}{E(k/2)+1}; E - Entierfunction.$$

The role of the temperature is to affect an influence between neurons, which is realized by the weight modification (eq. 6). It enables NC method to create expanded sets of clusters in the early phase of clustering while the temperature is high, and allows gradual degradation providing stabilization of clusters in the final steps of the method.

Each of the structure neurons is characterized by its mass M_n. It assigns the relative number of patterns clustered by a neuron (a ratio of Z_n to Z – which is the number of all patterns used in the clustering process), eq. 3.

$$M_n = \frac{card(Z_n)}{card(Z)} \tag{3}$$

The neurons mass determines its behaviour in the learning process. The greater the mass is, the stronger the neuron affects others and it is less affected by them. Because the structure of clusters is dynamic and the patterns set assigned to each neuron can change over the time, there is a possibility that neuron mass will reach 0. In this case the neuron is removed from the structure. This results in degeneration of the tree. During training process, after each pattern presentation the distance D between weights $\mathbf{w_n}$ of n-th neuron and the given pattern $\mathbf{x}$ is calculated according to the eq.4.

$$D\big(\mathbf{w_n}(t), \mathbf{x}(t)\big) = F\big(\mathbf{w_n}(t), \mathbf{x}(\mathbf{t})\big) - M_n(k)F\big(\mathbf{w_n}(t), \mathbf{x}(\mathbf{t})\big) \tag{4}$$

where F is an Euclidian distance between the weights $\mathbf{w_n}$ and the given pattern $\mathbf{x}$, M_n is a mass of n-th neuron. In case the greatest distance is greater than an average distance between weights of existing neurons a new neuron is added to the structure. Its weights are set as equal to the given pattern. Otherwise the weights of existing neurons are updated. For the winning neuron (the neuron for which the distance D is the smallest one) the weights $\mathbf{w}_w$ are changed as follows:

$$\mathbf{w_w}(t+1) = \mathbf{w_w}(t) + T(k)\big(\mathbf{x}(t) - \mathbf{w_w}(t)\big)\big(1 - M_w\big) \tag{5}$$

The weights of neurons in the neighbourhood are changed according to the eq. 6.

$$\mathbf{w_n}(t+1) = \mathbf{w_n}(t) + \frac{T(k)\big(\mathbf{x}(t) - \mathbf{w_w}(t)\big)\big(1 - M_w\big)}{D\big(\mathbf{w_n}(t), \mathbf{w_w}(t+1)\big)} \tag{6}$$

In the next step the mass of the winning neuron is updated. In each epoch during training the temperature is updated, as well. The end of the first phase results in the clustered training patterns that create the patterns sets mutually exclusive.

3.2 The Second Phase – Searching for Classification Rules in the Complex Form Elements

As it was mentioned, the complex form elements cluster patterns that belong to more than one class. For such cases a set of conjunctive rules in the attributes

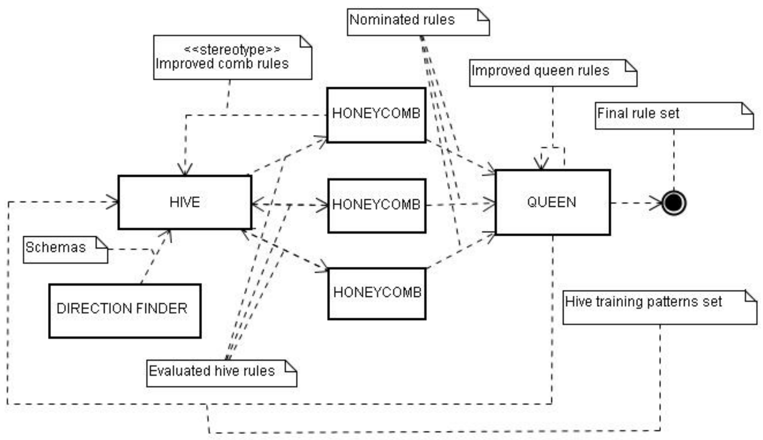

Fig. 3. Bee Miner structure and flow of rules between key elements

logic is searched by a rule extractor (Bee Miner). It is composed of four key elements (Fig. 3): *direction selector*, *hive*, *honeycombs*, and *queen*.

The *direction finder* is used only at the moment of creating new rules. Its task is to pick an appropriate set of attributes (schema) on which the new rules will be created. It is realized on the basis of evolutionary algorithm, where rank selection, mutation and homogenous crossover are applied. A chromosome consists of the list of attributes. The role of this algorithm is to search for subspaces where patterns from different classes can be easily separated. Individuals are evaluated on the basis of the density metrics. To calculate this value the gravity center for each class must be found and the average distance between patterns of this class and the center is computed next. Then these values are compared for pairs of classes in order to determine whether the classes are separated. The fitness value of an individual is based on the relative number of separated pairs of classes.

The purpose of a *hive* is creation and a rough evaluation to which honeycomb the premise part of a rule fits the best. The premise part is based on the schema selected in the direction finder mechanism. As the result of this step a list of input features is obtained in the form of a vector: $\mathbf{x}$=[x_1,x_2,..., x_k] where $k <= n$ (n is the dimension size for the classification problem). Then the schema is converted to the premise part of a rule by applying random operator and the value from the attribute domains. Generally, we can say the i premise has a form x_i R $value_k$, where x_i is a variable representing i-th attribute, R stands for an operator ($R \in \{<, >, =\}$) and $value_k$ is one of the possible values for this attribute. After the process of a rule evaluation the hive transfers newly created rules to appropriate honeycombs.

Each *honeycomb* represents one class, for which it tries to find the best set of rules. The mechanism in honeycomb is realized by a limited queue that gathers and orders the best rules found by hive. Each rule in the queue has a nomination counter that informs how many times the rule was in the queue in the honeycomb. If the counter reaches the appropriate value, the rule migrates to the set of nominated rules. These rules are collected by a queen.

The queen is re-evaluating and selecting rules in order to create the set of rules with the best accuracy. The rules taken to re-evaluation and selection are from the gathered nominated rules sets and from earlier created set of rules. The selection process of a new temporal solution relies on the choice of rules that are characterized by the highest accuracy.

4 Experimental Study

The classifier evaluation is based on the prediction accuracy (the percentage of correct predictions divided by the total number of predictions). The experiments were performed for four benchmark data sets taken from [1]. The results are compared with SVM and C4.5 results published at [2].

The characteristics of the applied data sets are presented at the bottom part of Table 1. As it can be noticed they contain a various number of attributes and a number of classes. At the top part of this table the results are shown. They represent an average from ten runs performed with the use k cross validation method with k=10. Throughout testing of the CUSC method two sets of parameters were used. The first one supported rules extraction (LGT set) by limiting the growth of CUSC structure. It increased the number of patterns for each single extraction. The second set of parameters (ETS set) limited rules extraction by building expanded tree structure. The research have shown that the quality of CUSC method results is strongly dependent on the parameter set. The differences between the results with the use GTS and ETS sets were reaching up to 8%. The best presented example are the results acquired for classification of Iris set. In that case the usage of ETS leads to creating structures without any classification elements in complex form. It effected in giving 97 % accuracy, while when the ETS set of parameters were used the accuracy reached only 90 %. On the other hand the situation of performing classification for the Wine data was opposite. In this case GTS set of parameters has given better results.

Table 1. The classification accuracy for CUSC and other methods for benchmark data sets; (WBC) is the abbreviation from Wisconsin Breast Cancer

Comparison with other methods in terms of accuracy				
	(WBC)	Sonar	Wine	Iris
CUSC	0.95 (ETS)	0.82(ETS)	0.94(LGT)	0.97(ETS)
SVM	0.97	0.82	0.97	0.96
C4.5	0.95	0.76	lack of data	0.95
data sets characteristics				
Pattern number	683	208	178	150
Class number	2	2	3	3
Attribute number	9	60	13	4

5 Conclusion and Future Work

For the tested data sets the results included in Table 1 show that CUSC has accuracy similar to SVM, and better than C4.5 classifier. But the key question when dealing with machine learning classification is not whether a learning algorithm is superior to others, but under which conditions a particular method can significantly outperform others on a given application problem. For this reason there is a need for more experimental study with the method. Nonetheless the results acknowledge relevance of the further works on CUSC.

However the current version of our classifier does not give uniform representation of all decision nodes the original goal was to obtain classification rules for all leaves. This assumption will be realized in the future. The further development of the method will be focused on decreasing a complexity of the method and on adjusting mechanisms of rules improvement in the Bee Miner method. Additionally, mechanism of local improvement of sub-trees is under development. Its purpose is to change or re-assemble those parts of tree structure that have negative influence on the classification accuracy.

References

1. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
2. Duch, W.: Datasets Used for Classification: Comparison of Results (2009), `http://www.is.umk.pl/projects/datasets.html`
3. Haykin, S.: Self-organizing maps. Neural networks - a Comprehensive Foundation, 2nd edn. Prentice-Hall, Englewood Cliffs (1999)
4. Karaboga, D.: An Idea Based On Honey Bee Swarm for Numerical Optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
5. Kotsiantis, S.B.: Supervised Machine Learning: A Review of Classification Techniques. Informatica 31, 249–268 (2007)
6. Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)
7. Sullivan, K., Luke, S.: Evolving Kernels for Support Vector Machine Classification. In: Genetic And Evolutionary Computation Conference Archive Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 1702–1707 (2007)