International Journal of Knowledge-based and Intelligent Engineering Systems 11 (2007) 227–235
IOS Press

Evolutionary approach to combined multiple models tuning

Gregor Stiglic* and Peter Kokol
Laboratory for System Design, Faculty of Electrical Engineering and Computer Science, Smetanova 17, 2000 Maribor, Slovenia

Abstract. This paper presents a combination of Combined Multiple Models (CMM) technique and evolutionary approach that is used for tuning of multiple parameters. Proposed hybrid classifier was tested in microarray gene expression analysis domain. This domain was chosen intentionally, because of the nature of Combined Multiple Models classifiers that are specialized in solving problems with high dimensionality and contain low number of samples. Evolutionary tuning of parameters in combination with validation dataset enables fine tuning of parameters that are usually set to pre-defined values. Using this technique another step in leveling the accuracy of comprehensible classifiers to those represented by ensembles of classifiers was made.

1. Introduction

Recently developed microarray technology allows measurement of expression levels for thousands of genes simultaneously. Classification or clustering techniques can be used to search for significant genes that can help us identifying different clinical states of the patient. To improve the accuracy of classification in microarray data many new methods have been developed. Unfortunately the majority of these approaches were so called 'black-box' methods, so the next logical step was development of techniques that would improve comprehensibility of almost incomprehensible classification algorithms like neural networks or ensembles of classifiers.

The most common approaches in rule extraction usually use 'black-box' models. An example of such system was presented by Domingos in [1] where he generalized the concept of oracle queries. His idea was that any 'black-box' model can be captured in a comprehensible classification model by using additional 'artificial' examples that are labeled according to the original model.

*Corresponding author. Tel.: +386 2 220 7465; Fax: +386 2 220 7272; E-mail: gregor.stiglic@uni-mb.si.

So why would this idea work? To answer this question we should return to the initial problem in the classification of microarray data – i.e. small number of samples on one side and enormous number of attributes on the other side. The Combined Multiple Models (CMM) method proposed by Domingos represents a solution to the small number of samples problem by multiplying them in form of artificial data points. The only drawback was that CMM method was not optimized for continuous values, but was meant to be used on typical datasets in machine learning domain consisting of discrete and continuous features. Microarray analysis datasets always consist of continuous values that represent gene expressions. To solve this problem CMM has to be effectively optimized for continuous values and because of the complexity of this optimization if performed manually, the proposed method uses evolutionary algorithms for the tuning of CMM parameters.

Evolutionary algorithms are proven powerful tools for solving optimization problems using mimicking mechanisms of natural evolution. The most popular type of evolutionary algorithms are genetic algorithms where approximate solutions to optimization and search problems are found using techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover).

ISSN 1327-2314/07/$17.00

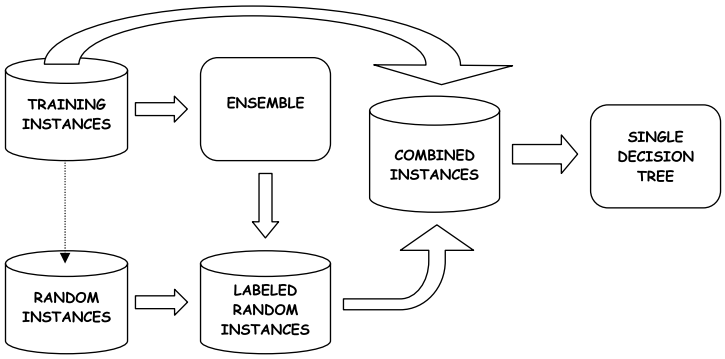

Fig. 1. Combined Multiple Models.

This paper presents a novel hybrid classification technique that uses artificial data points generation optimized for continuous values combined with evolutionary parameter tuning.

The rest of this paper is organized as follows. Section 2 presents methods for Combined Multiple Models and Evolutionary Tuning of Parameters. Next Section includes the experimental settings description and methods of pre-processing high dimensional microarray data. It is followed by Section 4 where the results and comparison to other classification methods are presented. In the last section, the main contribution of this paper is summarized and several issues for future works are indicated.

2. Evolutionary Combined Multiple Models

Combined Multiple Models (CMM) is one of the names used for methods that are able to build single comprehensible models from queries to 'black-box' models and was presented in [2]. CMM was later studied and further analyzed by Estruch et al. in [3]. Estruch used a new term for CMM – "mimetic classifiers", because of their ability to imitate more complex and more accurate classifiers.

2.1. Combined Multiple Models

Basic idea of CMM is to build a single classifier that would retain most of the knowledge that can be acquired by combination of multiple classifiers into ensembles. This is done by adding additional artificial data points to the learning dataset. Those additional data points are then classified (i.e. labeled) by applying the ensemble of classifiers that was trained on the learning dataset. In the final dataset the original training dataset examples are joined by new artificial dataset examples. The final dataset is then used to build a single comprehensible classifier. The whole process is shown in Fig. 1.

Concept of using artificial data points to build better classifiers was already used in several papers. One of the first such methods is active learning method proposed by Cohn et al. [4]. Another application of artificial examples was presented by Craven and Shavlik in [5] where they describe the learning of decision trees from neural networks. This approach was later used in several papers on neural networks knowledge extraction.

2.2. Proposed modifications

This paper presents the CMM based method that is specialized for microarray dataset classification problems. Optimization of the original method was done on artificial data points creation due to specific structure of the microarray datasets. Opposite to the original research [2], where most of the best results were achieved on the datasets containing nominal values, microarray analysis presents continuous-valued datasets. Therefore a new method called Combined Multiple Models for Continuous values (CMMC) is proposed. In this paper two new artificial data points creation techniques are examined which will be referred to as CMMC-1 and CMMC-2.

Both methods are based on multiplication of the original training set instances. Data points are generated from original training set by creating copies of original training set instances by minimal modifications of attribute values.

First variant is based on the variance of the gene expression values where each attribute value can be changed by adding the random value from the interval $\pm[\sigma, (1/3)\sigma]$ to the original gene expression value. Be-

1		2				3				4		5		6			
CMMC Type		CMMC Lower Boundary				CMMC Upper Boundary								C4.5 Pruning Confidence Threshold			
0	1	0.0	0.1	0.2	0.3	0.4	0.6	0.8	1.0	0.0	0.1	0.2	0.3				

Fig. 2. Genotype of CMMC parameter tuning.

cause of the large number of attributes only 50% of randomly selected attributes are changed. Result of such data point multiplication is a wide dispersion of the points around their base data point, while preserving the original training set distribution of the samples.

Second variant maintains the original distribution on even tighter area than the first one, especially when data points lie tightly together. This is done by generating random points in the interval x $\pm$ d, where x is the value of the attribute and d is distance to the nearest neighbor value of this attribute. Again only 50% of attributes are randomly selected for modification. From here on our proposed methods follow the steps proposed in [3] – i.e. labeling of the artificial data points and building of the final classifier which differs only in the final step where a decision tree instead of a set of rules is built.

2.3. Evolutionary tuning of parameters

Evolutionary computation (EC) mimics the processes of biological evolution with its ideas of natural selection and survival of the fittest to provide solutions for global optimization problems. It uses algorithms based on natural evolution principles explained by Darwinian Natural Selection that can provide much better results than conventional optimization techniques in realistic time frame. In its primary forms EC included Evolutionary Strategies proposed by Schwefel et al. [6], followed by Genetic Algorithms proposed by Holland in [7] and refined by Goldberg in [8], and Genetic Programming by Koza [9].

In our research genetic algorithms are used for tuning of parameters that can contribute to optimal performance of CMMC method. Our proposed modification to original CMM method requires even more "fine-tuning" of additional parameters when using CMM for continuous values.

Therefore a genetic algorithm is proposed to search for the optimal balance between the boundary values of CMMC artificial data points creation process. Additional to boundary values the proposed method also tries to optimize the pruning level of final CMMC decision trees. Figure 2 represents a graphical representation of the genotype that was used during evolutionary parameter tuning. Initial bit represents one of the two possible CMMC subtypes described in Section 2.1. Other three values are coded as two-bit genes and represent lower and upper CMMC boundary values and pruning level of the resulting C4.5 decision tree. The lowest row of Fig. 2 presents gene values are presented – for example lower boundary can represent 0, 10, 20 or 30 percent of CMMC's maximal boundary value.

2.3.1. Selection

Earlier versions of genetic algorithms are based on binary coding of genotype and are therefore using selection, mutation and cross-over operators.

The first step in selection process includes calculation of probability for individuals that will be copied to the candidates for next generation. The most basic probability estimation was proposed in [7,8] and is also known as *proportional selection*. For each chromosome C_i in P, where P is a population including chromosomes $C_1, \ldots, C_n$, the probability of inclusion of chromosome in P' that represents a copy of $P, p_s(C_i), i = 1, \ldots, N$ is calculated as

$$p_s(C_i) = \frac{f(C_i)}{\sum_j^N f(C_j)}.$$

Another possible approach is based on *ranking* of chromosomes [10] where fitness value is used for ranking, and $p_s(C_i), i = 1, \ldots, N$, is computed using the rank of C_i and therefore selection probability of each individual is assigned linearly according to their rank.

Second step in selection mechanism consists of sampling, based on the selection probabilities computed in the first step. The most basic one is *stochastic sampling with replacement* [7,8]. Inclusion of chromosome in this sampling technique is proportional to $p_s(C_i)$ with replacement. One of the more efficient sampling techniques called *stochastic universal sampling* was proposed by Baker in [11] and is very similar to the *stochastic sampling*, with an important difference that the number of copies of any chromosome is bounded by floor and ceiling of its expected number of copies based on $p_s(C_i)$.

2.3.2. Crossover

Selection is usually followed by crossover and mutation operations that are more dependent on the chosen representation. Crossover is a technique that tries to combine the best features from two parent chromosomes by combining the values of their genes in two new chromosomes called offspring. The simplest form of crossover operator that was proposed by Holland [7] and also used by Goldberg [8] is based on a simple rule that selects the first n genes from first chromosome and the next $m - n$ genes from the second chromosome to form the first offspring chromosome and vice-versa for the second chromosome, where m is number of genes in a chromosome and n is a random number between 0 and m. To effectively use genetic algorithms in problems involving continuous search space one should consider using specialized real coded genetic algorithms (RCGA) that differ from the binary coded genetic algorithms (BCGA) in implementation of crossover and mutation operators. The main difference from BCGA is in representation of chromosome that is represented a vector of real values.

Therefore a number of RCGA based crossover operators were proposed. Let us assume that $C_1 = (c_1^1 \ \ldots \ c_n^1)$ and $C_2 = (c_1^2 \ \ldots \ c_n^2)$ are two chromosomes selected for application of crossover operator. A few of the most commonly used crossover operators are described below:

Flat crossover

Proposed by Radcliffe in [12] where a simple crossover operator is used that randomly (uniformly) chooses h_i from interval $[c_i^1, c_i^2]$ to construct an offspring $H = (h_1, \ldots, h_i, \ldots, h_n)$.

Simple crossover

Is a RCGA version of the simplest BCGA crossover operator described above and was presented by Wright and Michalewicz [13,14]. Two new chromosomes

$$H_1 = (c_1^1, c_2^1, \ldots, c_i^1, c_{i+1}^2, \ldots, c_n^2)$$
$$H_2 = (c_1^2, c_2^2, \ldots, c_i^2, c_{i+1}^1, \ldots, c_n^1)$$

are built by randomly choosing a position of $i \in \{1, 2, \ldots, n-1\}$.

BLX-α crossover

A single offspring $H = (h_1, \ldots, h_i, \ldots, h_n)$ is created, where h_i is randomly (uniformly) chosen value of the interval $[c_{\min} - I \cdot \alpha, c_{\max} + I \cdot \alpha]$, $c_{\max} = \max(c_i^1, c_i^2)$, $c_{\min} = \min(c_i^1, c_i^2)$, $I = c_{\max} - c_{\min}$. This crossover operator was proposed by Eshelman et al. [15] and is considered as one of the most useful and still simple crossover operators. Variation of parameter α can cause very diverse behavior of the operator. Using $\alpha < 0$ will cause the population to converge toward average values of their intervals and consequently generate very low diversity and quick convergence toward local optimums. The most widely used α value is 0.5 that assures a balanced relationship between the convergence (exploitation) and divergence (exploration) is reached. This also means that the probability of the offspring lying between the values of its parent becomes equal to probability of offspring lying outside its parents.

2.3.3. Mutation

Mutation or the final operator in its most simple version as it was presented in [7,8] randomly chooses a gene from a chromosome and swaps its value from "0" to "1" or vice-versa. This is only applicable to BCGAs, while there are some mutation operators specialized for RCGAs.

Random mutation

Was proposed by Michalewitz in [14] and chooses c_i' randomly from an interval $[a_i, b_i]$, where c_i' is a gene from chromosome C that is to be mutated, $a_i = \min(c_i^1, c_i^2)$ and $b_i = \max(c_i^1, c_i^2)$.

Non-uniform mutation

Is a mutation operator that was also proposed by Michalewitz [14] and is much stronger than Random mutation mechanism. A gene is mutated according to the following formula

$$c_i' = \left\{ \begin{array}{l} c_i + \Delta(t, b_i - c_i) \text{ if } \tau = 0 \\ c_i - \Delta(t, c_i - a_i) \text{ if } \tau = 1 \end{array} \right\},$$

where t is a number of current generation, τ represents a random number that can have a value of zero or one and

$$\Delta(t, y) = y(1 - r^{(1 - \frac{t}{g_{\max}})^b}),$$

where g_{max} is a maximal number of generations, r is a random number from interval [0,1] and b is a parameter chosen by user, which determines the degree of dependency on the number of iterations. The returned value of this function lies on the interval $[0, y]$ and returns values closer to zero as the new generations are created. This functionality of the non-uniform mutation causes the operator to search in the initial space at the beginning and in local area towards the end of the algorithm execution.

The interested reader is advised to consult [16] to obtain extensive overview of genetic algorithms for real coded problems.

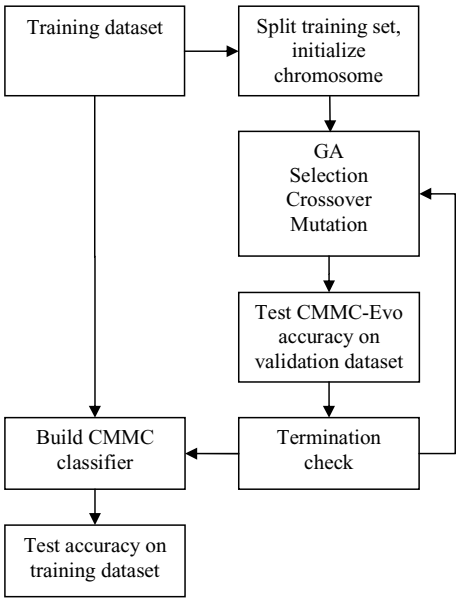

Fig. 3. Genetic parameter tuning model.

3. Experimental Settings

To set the parameters on-line during the classification process a model presented in Fig. 3 which uses a part of training dataset as a parameter tuning validation dataset is used. Termination conditions for genetic algorithm can be time or iteration based. The proposed model uses 10 iterations for improvement of the initial population. Because of computational limitations our model uses only 20 chromosomes, while selection parameter was set to 5 chromosomes per generation to breed a new generation. Proportional selection using stochastic sampling with replacement, simple crossover and random mutation were used for BCGA as suggested in [7,8].

As in the first experiment the whole genotype was binary coded, another experiment was carried out where all parameters, except type of CMMC algorithm, were replaced by real values. The only remaining mechanism from BCGA was selection, while crossover and mutation were replaced by BLX-α selection and non-uniform mutation operators. The results of the evaluations can be found in Section 5.

4. Datasets

To estimate the accuracy of proposed CMMC-Evo classifier and compare it to other classifiers, five microarray analysis datasets from Kent Ridge Biomedical Data Set Repository [17] were used. Five widely used publicly available gene expression datasets that were used in evaluation of the proposed method are presented in this section.

Leukemia dataset (**amlall**) comes from the research on acute leukemia by Golub et al. [18]. Dataset consists of 38 bone marrow samples from which 27 belong to acute lymphoblastic leukemia (ALL) and 11 to acute myeloid leukemia (AML). Each sample consists of probes for 6817 human genes. Golub used this dataset for training. Another 34 samples of testing data were used consisting of 20 ALL and 14 AML samples. Both datasets were joined and all 72 samples used for leave-one-out cross-validation testing.

Breast cancer dataset (**breast**) was published in [19] and consists of extremely large number of scanned gene expressions. It includes data on 24481 genes for 78 patients, 34 of which are from patients who had developed distance metastases within 5 years, the rest 44 samples are from patients who remained healthy from the disease after their initial diagnosis for interval of at least 5 years.

Lung cancer dataset (**lung**) includes the largest number of samples in our experiment. It includes 12533 gene expression measurements for each of 181 tissue samples. The initial research was done by Gordon et al. [20] where they try to classify malignant pleural mesothelioma (MPM) and adenocarcinoma (ADCA) of the lung.

Leukemia2 dataset (**mll**) tries to discern between 3 types of leukemia (ALL, MLL, AML). Dataset contains 72 patient samples, each of them containing 12582 gene expression measurements. Data was collected by Armstrong et al. and results published in [21].

Prostate Tumor dataset (**prostate**) contains 52 prostate tumor samples and 50 non-tumor (labeled as "Normal") prostate samples with around 12600 genes. The original study was conducted by Singh et al. in [22].

5. Results

One of the important issues in microarray classification is extremely high dimensionality of the original datasets. Therefore it is common to perform reduction of the initial feature set before the experiments. This usually contributes to lower time complexity and even increases the accuracy of classifiers. To include this important pre-filtering step, the whole experiment was divided in two steps. First step includes selection of the most appropriate feature selection method and tries to define the optimal number of features to select for the best performance of the classifiers.

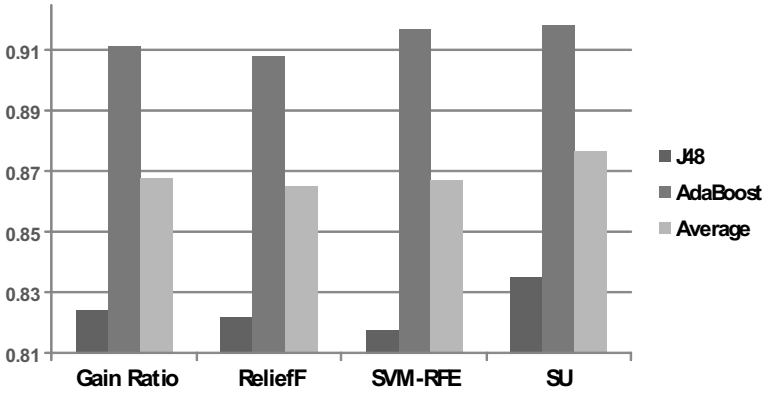

Fig. 4. Comparison of classification accuracy using four feature selection methods.

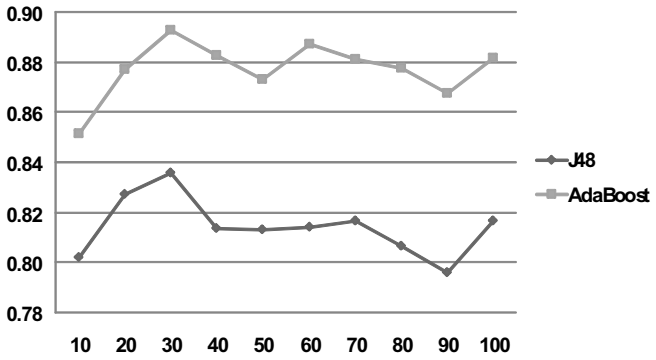

Fig. 5. Comparison of classification accuracy for different number of selected features.

5.1. Pre-filtering

Pre-filtering of features from datasets is a common pre-processing step in data mining of multi-dimensional datasets. It is used to reduce dimensionality, improve mining efficiency, increase mining accuracy, and enhance result comprehensibility [23,24]. This preprocessing step is especially important in microarray analysis where datasets usually contain extremely large number of features that represent gene expression levels. The first step of our experiments compared four feature selection methods based on feature ranking to improve the overall accuracy in classification testing. The following feature selection methods were used:

- Gain Ratio [25]
- Symmetrical Uncertainty (SU) [25]
- ReliefF [26]
- Support Vector Machine Recursive Feature Elimination (SVM-RFE) [27,28]

In our experiments all four feature selection methods were tested on five publicly available datasets described in Section 3. All tests were done using 2x10-fold cross-validation measuring the accuracy of J48 [29] and AdaBoost [30] classifiers at different numbers of retained features after pre-filtering step. The number of features used for classification ranged from 10 to 100. All experiments were conducted in WEKA machine learning environment [29]. Figure 4 presents the accuracy results for all four feature selection methods averaged over five datasets. It can be seen that Symmetrical Uncertainty based filter performed the best among all tested methods. Comparing classical decision tree (J48) to ensemble (AdaBoost) it can be noticed that SVM-RFE and SU return the best results with SU performing negligibly better.

Another important parameter to determine was optimal number of features that should be selected by feature selection method. Figure 5 presents the results of feature selection methods accuracy at different numbers of pre-selected features. The results show that the best accuracy rates can be achieved at around 30 pre-selected features. This is true for both tested methods – i.e. classical decision tree and ensemble of classifiers.

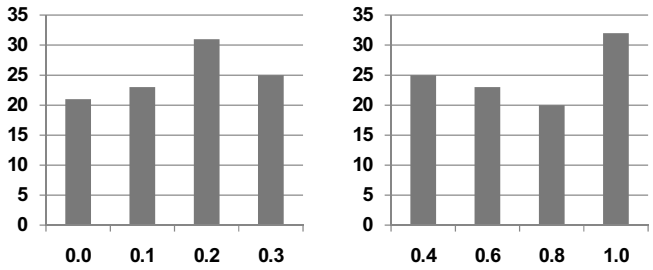

Fig. 6. Distribution of lower (left) and upper (right) boundary CMMC parameter after optimizations

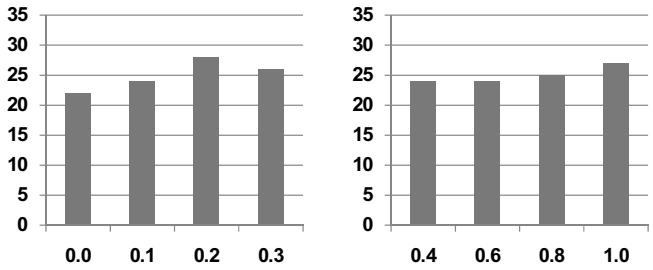

Fig. 7. Distribution of lower (left) and upper (right) boundary after optimizations using real-coded evolutionary approach.

Based on the above mentioned pre-processing experiments Symmetrical Uncertainty with 30 best ranked features was selected for the remaining experiments.

5.2. Classification

In order to evaluate our proposed method all experiments consisted of 10-fold cross-validation tests. Binary and real-coded evolutionary approaches of parameter tuning described in Section 2.3 were repeated 100 times to get the most common values of tuned parameters.

The accuracy of the proposed method was compared to J48 decision trees, default CMMC algorithms and AdaBoost using 100 boosted J48 decision trees.

Results of average accuracy for four different classifiers are presented in Table 1. It can be seen that CMMC-Evo classifier clearly outperformed classical decision tree, but still lacks some accuracy to perform at the level of ensemble based classifiers. This fact only shows that combining multiple models together still gains the most in terms of accuracy.

5.3. Evolutionary approach

There are two parameters that have direct influence on performance of CMMC classifier and were therefore the most interesting for tuning – i.e. lower and upper boundary values. The results shown (Fig. 6) are based on final selected values after evolutionary tuning was done. It can be seen that best results coincide with the lower boundary set to 0.2 and upper boundary to 1.

Table 1
Comparison of accuracy

Dataset	J48	CMMC (Average)	CMMC-Evo	Ada-Boost (J48 trees)
amlall	83.58	89.73	90.17	91.20
breast	64.81	67.45	74.11	86.97
lung	97.06	97.27	98.33	97.51
mll	85.11	88.79	92.86	89.55
prostate	86.83	87.13	87.36	94.25
Average	83.47	86.07	88.61	91.89

It was interesting to observe how real-coded genetic algorithms can tackle this problem.

To compare the results with the binary coded version of parameter tuning the final parameter values were discretized to the same intervals as in first experiment with binary coded genotype. Frequencies of tuned values (Fig. 7) show very similar results compared to the binary coded genotype, especially when comparing the tuning values for lower boundary value. In the upper boundary values comparison it can be noticed that the highest value was chosen again, but this time with lower frequency.

The other two parameters that were used in tuning process – i.e. type of CMMC and pruning threshold – did not have significant influence to the final results and

did not change any of the previously known facts. As our fitness function included only the accuracy and not the complexity of the decision tree, CMMC-2 clearly outperformed CMMC-1 which returns more complex but also more accurate trees. The reason why CMMC type was included in parameter tuning process was that there was a possibility that using specific values of tuned parameters, CMMC-1 could outperform CMMC-2.

Another important parameter that could have the same effect on accuracy or other tuned parameters was pruning level of decision tree. It turns out that this parameter can not significantly influence the accuracy as there were no considerable alterations in accuracy of the classifier when different values for pruning threshold were used.

6. Discussion

This paper presents a hybrid classification method based on modified CMM classification algorithm that was combined with evolutionary parameter tuning using genetic algorithm. The main advantage of the proposed CMMC-Evo over similar decision tree ensemble classification methods is combination of improved accuracy and good comprehensibility of the classifier. The accuracy of the proposed classifier building technique still cannot be compared to ensembles of similar classifiers, but it can achieve higher accuracy in comparison to classical decision trees and most importantly – the final classifier is still comprehensible. The experiments using binary and real coded parameter tuning showed that the results using different parameter values did not significantly change the overall accuracy of the proposed classification method. The values with the highest frequency were very similar to the values that were already used in our previous research and were set intuitively.

An interesting question for the future research is whether the size of genome that was used for parameter optimization can be extended by adding more parameters that have direct impact on accuracy of the CMM classification method. That kind of parameter could be the number of artificial data points generated that is usually set to some pre-defined value.

References

[1] P. Domingos, *Knowledge acquisition from examples via multiple models*, in Proc. of the 14th International Conference on Machine Learning, Morgan Kauffman, 1997, 98–106.

[2] P. Domingos, Knowledge Discovery Via Multiple Models, *Intelligent Data Analysis* **2**(1–4) (1998), 187–202.

[3] V. Estruch, C. Ferri, J. Hernández-Orallo and M.J. Ramírez-Quintana, *Simple Mimetic Classifiers*, in Proc. IAPR International Conference on Machine Learning and Data Mining (MLDM2003), 2003, 156–171.

[4] D. Cohn, L. Atlas and R. Ladner, Improving generalization with active learning, *Machine Learning* **15** (1994), 201–221.

[5] M.W. Craven and J.W. Shavlik, *Extracting comprehensible concept representations from trained neural networks*, in Working Notes on the IJCAI'95 Workshop on Comprehensibility in Machine Learning, Montreal, Canada, 1995, 61–75.

[6] H.P. Schwefel, Kybernetische Evolution als Strategie der exprimentellen Forschung in der Strömungstechnik, Master's thesis, Technical University of Berlin, 1965.

[7] J.H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor: University of Michigan Press, 1975.

[8] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesly, New York, 1989.

[9] J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

[10] J.E. Baker, *Adaptive Selection Methods for Genetic Algorithms*, in Proc. of International Conference on Genetic Algorithms, 1985, 101–111.

[11] J.E. Baker, *Reducing Bias and Inefficiency in the Selection Algorithm*, in Proc. of International Conference on Genetic Algorithms, 1987, 101–111.

[12] N.J. Radcliffe, Equivalence Class Analysis of Genetic Algorithms, *Complex Systems* **5**(2) (1991), 183–205.

[13] A. Wright, Genetic Algorithms for Real Parameter Optimization, in: *Foundations of Genetic Algorithms 1*, G.J.E. Rawlin, ed., Morgan Kaufmann, San Mateo, 1991, pp. 205–218.

[14] Z. Michalewitz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, New York, 1992.

[15] L.J. Eshelman, Real-Coded Genetic Algorithms and Interval Schemata, in: *Foundations of Genetic Algorithms 2*, L. Darell Whitley, ed., Morgan Kaufmann, San Mateo, 1993, pp. 187–202.

[16] F. Herrera, M. Lozano and J.L. Verdegay, Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis, *Artificial Intelligence Review* **4**(12) (1998), 265–319.

[17] J. Li and H. Liu, *Ensembles of cascading trees*, in Proc. IEEE International Conference on Data Mining, IEEE Computer Society, Melbourne, Florida, 2003, 585.

[18] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield and E.S. Lander, Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring, *Science* **286**(5439) (1999), 531–537.

[19] L.J. van't Veer, H. Dai, M.J. van De Vijver, Y.D. He, A.A. Hart, M. Mao, H.L. Peterse, K. Der Kooy, M.J. Marton, A.T. Witteveen, G.J. Schreiber, R.M. Kerkhoven, C. Roberts, P.S. Linsley, R. Bernards and S.H. Friend, Gene expression profiling predicts clinical outcome of breast cancer, *Nature* **415** (2002), 530–536.

[20] G.J. Gordon, R.V. Jensen, L.-L. Hsiao, S.R. Gullans, J.E. Blumenstock, S. Ramaswami, W.G. Richards, D.J. Sugarbaker and R. Bueno, Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma, *Cancer Research* **62** (2002), 4963–4967.

[21] S.A. Armstrong, J.E. Staunton, L.B. Silverman, R. Pieters, M.L. den Boer, M.D. Minden, S.E. Sallan, E.S. Lander, T.R. Golub and S.J. Korsmeyer, MLL translocations specify a distinct gene expression profile that distinguishes a unique leukaemia, *Nat Genet* **30**(1) (2002), 41–47.
[22] D. Singh et al., Gene expression correlates of clinical prostate cancer behavior, *Cancer Cell* **1** (2002), 203–209.
[23] A. Blum and P. Langley, Selection of relevant features and examples in machine learning, *Artifcial Intelligence* **97** (1997), 245–271.
[24] R. Kohavi and G. John, Wrappers for feature subset selection, *Artificial Intelligence* **97**(1–2) (1997), 273–324.
[25] M.A. Hall and L.A. Smith, *Practical feature subset selection for machine learning,* in Proceedings of the 21st Australian Computer Science Conference, 1998, 181–191.
[26] I. Kononenko, *Estimating attributes: analysis and extension of relief,* in Proceedings of European Conference on Machine Learning, 1994, 171–182.
[27] B.E. Boser, I.M. Guyon and V.N. Vapnik, *A training algorithm for optimal margin classifiers,* in Proceedings of the Fifth Annual Workshop on Computational Learning theory, COLT '92, ACM Press, New York, NY, 1992, 144–152.
[28] I. Guyon, J. Weston, S. Barnhill and V. Vapnik, Gene selection for cancer classification using support vector machines, *Machine Learning* **46** (2002), 389–422.
[29] I.H. Witten and E. Frank, *Data Mining: Practical machine learning tools with Java implementations,* Morgan Kaufmann, San Francisco, 2005.
[30] Y. Freund and R.E. Schapire, *Experiments with a New Boosting Algorithm,* in Proc. of the Thirteenth International Conference on Machine Learning, Morgan Kaufmann, 1996, 148–156.