Ensembles of Similarity-Based Models.

Włodzisław Duch and Karol Grudziński

Department of Computer Methods, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland.
E-mails: {duch,kagru}@phys.uni.torun.pl

Abstract. Ensembles of independent classifiers are usually more accurate and show smaller variance than individual classifiers. Methods of selection of Similarity Based Models (SBM) that should be included in an ensemble are discussed. Standard *k*-NN, weighted *k*-NN, ensembles of weighted models and ensembles of averaged weighted models are considered. Ensembles of competent models are introduced. Results of numerical experiments on benchmark and real-world datasets are presented.

1 Introduction.

A framework for Similarity-Based Methods (SBM) covers all methods based on computing similarity between the new case and cases in the training library [1]. The SBM framework is very rich, it includes such well-known methods as the *k*–Nearest Neighbor (*k*-NN) algorithm and it's extensions, originating mainly from machine learning and pattern recognition fields, as well as neural methods such as the popular multilayer perceptron networks (MLP) and networks based on radial–basis functions (RBF). Methods of the SBM type are based on specific parameterization of the $p(C_i|\mathbf{X};M)$ posterior classification probability, where the model M involves various procedures, parameters and optimization methods. Instead of focusing on improving a single method a search for the best method belonging to the SBM framework should select optimal combination of parameters and procedures for a given problem.

Similarity Based Learner (SBL) is a software system developed in our laboratory that systematically implements methods belonging to the SBM framework [2]. Methods implemented so far provide many similarity functions with different parameters, include several methods of feature selection, methods that weight attributes (based on minimization of the cost function or based on searching in the quantized weight space), methods of selection of interesting prototypes in batch and on-line versions, and methods implementing partial-memory of the evolving system. Currently our research focuses on implementation of network-type realizations of various SBM methods, weighting influence of reference vectors and speeding up the calculations.

A single model that may be developed by combining methods and adding parameters within the SBM framework may be improved further by combining or mixing many models. Mixture of models may not only improve the accuracy but also decrease variance of the model, stabilizing and improving its generalization

[3]. Except for SBL several other programs have been developed in our laboratory. All these programs are at present being integrated into a data-mining software that should allow to create ensembles of quite different methods.

In the next section methods for model combination are briefly discussed and an algorithm for selection of subsets of models that are included in an ensemble described. In the third section modifications of these algorithms suitable for similarity based learning are discussed. Results comparing stand-alone methods to ensembles of similarity based methods are presented for a few benchmark and real-world datsets. Finally some conclusions and plans for further work are given.

2 Ensembles of SBM Models

A few definitions are needed first. **An algorithm**, or **a method**, is a certain well-defined computational procedure. For example standard k-NN, a variant of the k-NN with optimization of weights scaling individual input features performed by search in the weight space, and by minimization of the cost function, are three distinct methods. Methods may have the same or different input and output parameters. **A model** is an instance of a method with specific values of parameters. For example the k–NN for $k = 1$ and $k = 3$ are two distinct models derived from the same method. **Combination, or an ensemble of models**, includes a procedure of selection of a set of models and a decision procedure to compute the final classification probability. Each of the models contributing to the combination should be trained on cases drawn from the same training set. Ensembles may involve models obtained from the same method (in this case adaptive parameters are optimized independently in every model) or from several different methods. Bagged models [3] are created by optimizing models of the same type on different subsets of the training set. Finally **a task or a scheme** is a sequence of one or more models which are added in succession to the ensemble.

In this paper two types of the base classifiers are used: the standard k-NN models and weighted k-NN models. Weighted k-NN uses distance functions:

$$D(\mathbf{X},\mathbf{Y})^{\alpha} = \sum_{i=1}^{n} s_i |X_i - Y_i|^{\alpha} \qquad (1)$$

parameterized by the α exponent (for Euclidean distances $\alpha = 2$ is taken) and by the scaling factors s_i that weight the importance of each attribute. These scaling factors are determined by minimization of the number of errors the classifier makes [4]. A multisimplex method or an adaptive simulated annealing minimization [5,6] has been used for this purpose. To reduce the number of numerical experiments only $\alpha = 1$ or 2 is considered here and only the results obtained with the simplex method are reported. Since classification results do not change if all attributes are rescaled by the same factor only $n-1$ attributes should be scaled. Fixing the most important attribute should lead to optimized values of the scaling factors $s_i \leq 1$.

2.1 Combining models.

Individual models are frequently unstable [3], i.e. quite different models are created as a result of repeated training (if learning algorithms are stochastic) or if the training set is slightly perturbed. The mixture of models allows to approximate complicated probability distributions quite accurately.

With $l = 1..K$ models providing estimation of probabilities $P(C_i|\mathbf{X};M_l)$ an additive combination

$$p(C_i|\mathbf{X};M) = \sum_{l=1}^{K} W_l P(C_i|\mathbf{X};M_l) \tag{2}$$

provides additional K linear parameters for model combination, determined using the standard Least Mean Squares (LMS) procedure. Majority voting is quite frequently used as a decision rule to combine results of multiple models. An empirical comparison of voting algorithms, including bagging and boosting, has been published by Bauer and Kohavi [7]. Tests were made using decision trees and naive Bayes method. The bagging algorithm uses classifiers trained on bootstrap samples, created by randomly drawing a fixed number of training data vectors from the pool which always contains all training vectors (i.e. drawing does not remove them from the pool). Results are aggregated by voting. AdaBoost (Adaptive Boosting) creates a sequence of training sets and determines weights of the training instances, with higher weights for those that are incorrectly classified. The arcing method uses a simplified procedure for weighting of the training vectors. Bauer and Kohavi [7] provided an interesting decomposition of bias and variance components of errors for these algorithms.

Renormalized product of different predictors has been advocated recently by Hinton [8] in context of unsupervised probability density estimation. Each individual model may have rather high variance but the product leads to a sharper probability distribution. Feature weighting or feature selection for different models may constrain different dimensions of the feature space and the product distribution will constrain all features. Each model may specialize in different aspect of the problem and thus instead of trying to generate and make an ensemble of the best models one should diversify them. Unfortunately the best diversification strongly depends on the type of problem analyzed. Renormalized product

$$p(C_i|\mathbf{X};M) = \frac{\prod_{l=1}^{K} P(C_i|\mathbf{X};M_l)}{\sum_{i=1}^{C}\prod_{l=1}^{K} P(C_i|\mathbf{X};M_l)} \tag{3}$$

may be optimized over all parameters of individual models by minimizing the log likelihood on the training data. Hinton's contrastive divergence approach requires minimization of the Kullback-Leibler (K-L) divergence between the data distribution and the Gibbs sampling equilibrium distribution minus the K-L divergence between the “one-step reconstruction" of Gibbs sampling and its equilibrium distribution. Computationally this procedure is very demanding. The product formula may be used for model aggregation without optimization if minimal probability is set to a small, non-zero value.

A novel approach to combination of models is presented below.

2.2 Ensembles of Competent Models (ECM)

So far all models selected to the ensemble were allowed to vote on the final result. Krogh and Vedelsby [9] showed that ensemble generalization error is small if highly accurate classifiers disagreeing with each other are used. The SBM models use prototypes and it is relatively easy to determine the areas of the input space in which a given model is highly competent (makes a few errors) and in which it fails. A simple algorithm that includes some information on the competence of different models is presented below.

Parameters of a given model $M_l, l = 1 \dots K$ are optimized on a train set using leave-one-out or cross-validation procedure. Then for each reference case $\mathbf{R}_i$ used in one of the SBM models – the case belonging to the true class $C(\mathbf{R}_i)$ – a list of predicted classes $C_j(\mathbf{R}_i)$ for all $j = 1 \dots m$ models is made. A competence vector $\mathbf{K}_j(\mathbf{R}_i) = \delta(C(\mathbf{R}_i), C_j(\mathbf{R}_i))$ for the area around $\mathbf{R}_i$ is created. In the decision phase nearest neighbor reference vectors are determined and only those classifiers that have been competent for all vectors are included in the voting procedure. If no competent models are found the vector given for classification is probably an outlier and should be left as 'rejected' or 'impossible to classify'.

A more sophisticated way of creating competent ensembles may be introduced if linear combinations are used instead of majority voting. Coefficients of linear combination should depend on the distance between the vector $\mathbf{X}$ and those reference vectors $\mathbf{R}_{l,k}$ of the feature space where model M_l is competent.

$$P(C_i|\mathbf{X};M) = \sum_{l=1}^{K} \sum_{m} W_l D(\mathbf{X}, \mathbf{R}_{l,m}) P(C_i|\mathbf{X};M_l) \tag{4}$$

should be a good choice, where $D(\mathbf{X}, \mathbf{R}_{l,m})$ functions estimate the competence of model M_l around the reference vectors $\mathbf{R}_{l,m}$. After renormalization $p(C_i|\mathbf{X};M) = P(C_i|\mathbf{X};M)/\sum_j P(C_j|\mathbf{X};M)$ gives final probability of classification. Since the problem is linear in W_l least mean squares optimization is sufficient to find the best parameters in this case. In contrast to AdaBoost and similar procedures explicit information about competence, or quality of classifier performance in different feature space areas, is used here.

2.3 Selection of *k*-NN models.

In the case of the standard *k*-NN, the classifier is used with different values of k on a training partition using leave-one-out algorithm and applied to the test partition. The predicted class is computed on the majority basis. To increase the classification accuracy one may first optimize $k, (k_1 \le k \le k_2)$ and select $m \le k_2 - k_1$ best classifiers for an ensemble model. In the case of weighted *k*-NN either k is optimized first and then best models created optimizing all weights, or best models are selected after optimization for a number of k values (a more accurate, but costly procedure).

Selecting a subset of best models that should be included in an ensemble is not an easy task since the number of possibilities grows combinatorially and obviously not all subsets may be checked. A variant of the best-first search (BFS) algorithm has been used for this selection. We have already used the BFS technique for the optimization of weights and for selection of the best attributes [10,11]. BFS algorithm can be used for majority voting of models derived from weighted-NN method based on minimization, or based on standard k-NN with different k, or for selection of an optimal sequence of any models.

The evaluation function $C(M_l)$ returns the classification accuracy on a validation set; this accuracy refers to a single model or to an ensemble of models selected so far. Let N denote the initial number of models from which selection is made and K the number of models that should be selected. The selection algorithm proceeds as follows:

1. Initialize:
   (a) Create a pool of N models, $\mathcal{M} = \{M_l\}, l = 1 \ldots N$.
   (b) Create an empty set for selected models $\mathcal{F} = \emptyset$.
   (c) Evaluate all models on the validation set, arrange them in a decreasing order $C(M_i) \geq C(M_j)$ for $i > j$.
   (d) Select the best model from the $\mathcal{M}$ pool and move it to the $\mathcal{F}$ pool.
2. Repeat for $L = 2 \ldots K$:
   (a) For $l = 1 \ldots N - L$ models M_l remaining in the pool $\mathcal{M}$ evaluate ensemble $C(\mathcal{F} \cap M_l)$ using majority voting.
   (b) Select the M_l model with highest performance and move it to the $\mathcal{F}$ pool.

At each step $N - L$ sequences consisting of L models are evaluated. If $K = N$ all models are incorporated into a sequence and this algorithm does not differ from the standard ‘majority voting’ procedure. Frequently the gain in performance may not justify additional complexity of adding a new model to the final pool and new models will be created and evaluated. This algorithm finds a pool of models corresponding to the highest classification accuracy on validation partition. In case of k-NN calculations may be done on the training partition in the leave-one-out mode instead of the validation partition.

Although the selection algorithm described above is more computationally expensive than the standard ‘majority voter’ it has a better chance to work better if a smaller subset of models is selected from a larger pool. Since the SBM scheme allows to add many parameters and procedures new models may also be created on demand if adding models created so far does not improve results. The model optimization (here minimization of the k-NN weights) is performed N times at the initialization stage on validation data. Re-optimization of models in the pool may be desirable but it would increase the computational costs significantly, therefore all model parameters are fixed after the initialization step.

2.4 Stablization of the weighted models

Maximization of the performance $C(M)$ using the simplex or simulated annealing methods may give quite different sets $\{s_i\}$ of the attribute weighting parameters leading to similar accuracy. In effect the variance of results obtained with weighted *k*-NN methods may be rather high. The stability of standard *k*-NN model [3] is lost.

Perhaps the simplest way to decrease the variance is based on averaging the weights for a few *k*-NN models created on the same data. Weights s_i scaling the components of distance function are obtained by maximization of the number of correct answers using the simplex or multisimplex methods [5]. These methods are stochastic, finding different sets of suboptimal parameters. In principle we could use simulated annealing or search for the best simplex solution obtained from multiple runs, reducing the variance to zero. In such case we would use bootstrap methods to generate different models for the ensemble. Since our goal in these numerical experiments is to investigate the influence of averaging and compare it to majority voted ensembles we have not used yet the sampling techniques to introduce more variability into our models. We have also not used yet the ensembles of competent methods.

Numerical experiments described in the next section include results of 4 methods: single weighted *k*-NN models (designated **WM**), models obtained by averaging weights for several WM models (designated **A-WM**), ensembles of weighted models (**E-WM**), and ensembles of weighted models with averaged weights (**AE-WM**). Performance of these methods depends on the number of models included in the ensemble and the number of models used for weight averaging.

3 Numerical experiments

To test some of the ideas presented above we have made a series of calculations using the well-known benchmark datasets taken from UCI repository [12] and real-world medical datasets. The WM symbol designates the single simplex minimization routine model. The A5-WM symbol designates a model that has been obtained from 5 weighted *k*-NN models (WM) independently optimized (differences come only from the stochastic properties of the simplex optimization procedure). The final weights are obtained by averaging over all 5 models. An ensemble of 5 weighted models is called E5-WM, and an ensemble of 10 weighted models, each obtained by averaging weights over 5 models, is called A5E10-WM. In all cases simplexes have been initialized randomly with weights ranging from 0 to 10 except for the ionosphere calculations where weights have been taken from the (0,1) range. A single weight corresponding to a highly-ranked feature is fixed at 1 to establish an absolute scale for distances.

First the ensemble selection method has been used with two artificial datasets, Monk-1 and Monk-3 [13]. These problems are designed for rule-based symbolic machine learning algorithms and the nearest neighbor algorithms usually do not work well in such cases. 6 symbolic features are given as input, 432 cases for testing. Previously we have obtained significant improvements for these datasets using feature selection and weighting [11]; here we are interested in improvements and stabilization of the weighted results due to the ensemble averaging. Calculations were repeated 5-10 times to estimate expected variance of the results.

Table 1. Results for the Monk-1 problem

Method	Accuracy %	Variance %
k-NN	89.5	–
WM	94.7	$\pm$ 5.3
A5-WM	99.6	$\pm$ 0.8
A10-WM	99.7	$\pm$ 0.8
E5-WM	98.4	$\pm$ 2.2
E10-WM	99.7	$\pm$ 0.6
A5E5-WM	99.7	$\pm$ 0.4
A5E10-WM	99.98	$\pm$ 0.07

For the Monk-1 problem 124 cases are given for training. Euclidean distance with *k*=1 was used; all calculations were repeated 10 times. Weighted *k*-NN does improves the result but the variance is quite high; an ensemble of 5 weighted models (E5-WM) still has quite high variance. Averaging combined with ensemble of 10 models achieves almost always 100% accuracy.

For Monk-3 problem 122 cases are given for training. Features 1 and 3 have been turned off by the initial feature selection procedure, increasing accuracy from 87.3% with all 6 features to 98.6% with 4 features. Weighted models slightly decrease the average accuracy (this decrease shows rather poor performance of the simplex optimization). Averaging over 5 models decreases the variance almost to zero. Results do not improve further because in this case there is some noise in the data and *k*-NN with feature selection achieves the optimal result.

Table 2. Results for the Monk-3 problem

Method	Accuracy %	Variance %
k-NN	98.6	–
WM	98.3	$\pm$ 0.6
A2-WM	98.5	$\pm$ 0.4
A5-WM	98.6	$\pm$ 0.0
A5E5-WM	98.6	$\pm$ 0.0

The *vowel dataset* is composed of 528 training and 462 test vectors, each with 10 continuous attributes derived from speech samples. The task to identify one of the 11 vowels is rather difficult and the standard *k*-NN with Euclidean function and $k = 1$ obtains only 56.3% correct answers. The best single model with $k = 9$ has only 56.5% accuracy on the test set. An ensemble of 10 *k*-NN models with $k = 1 \dots 10$ improves this result to 59.1%

The *ionosphere data* has 200 training and 150 test vectors, 2 classes and 34 continuous attributes. In this case significant improvement (4.6%) is obtained by using Manhattan distance function and optimizing *k*. Averaging over 5 weighted models did not improve results. Variance results for this dataset are missing because we performed each test only once (this is due to the high computational cost of the weighted method (34 parameters to optimize).

Poor result of the weighted model is probably due to the overfitting, since the number of parameters (34) is rather high for such small dataset (200 vectors). One of the reasons why averaging of weights may work is that it allows to avoid overfitting. Currently we have no validation test implemented in our software and the test is performed with the best weights found after the convergence of the minimization routine. Difficulty of finding good solutions in this large space may also be important. The simplex method reached accuracy on the training set that is about 5% higher than the multisimplex method (usually finding better solutions) but gave about 2% worse results on the test set.

Unfortunately differences on the training set were not reflected in improvements on the test set and there was no reduction of error using ensembles of 5 or 10 models. Other experiments performed with this dataset also indicate that accuracy on the training and the test set is not correlated. For comparison results of Shang and Breiman [14] obtained with boosted CART decision tree (DB-CART) and the C4.5 results are also provided.

Table 3. Results for the ionosphere data

Method	Accuracy %
DB-CART	91.3
1-NN, Euclidean	92.1
k-NN E10, Euclidean, $k = 1 \ldots 10$	92.7
C4.5	94.9
3-NN, Manhattan,	96.7
WM, Manhattan, k=3	95.3
WM, Manhattan, k=3 (Multisimplex)	97.3
A5-WM, Manhattan k=3	96.7
E5-WM,Manhattan k=3	96.7
A5E10-WM,Manhattan k=3	96.7

The *hepatobiliary disorders* dataset has been obtained from the Tokyo Dental and Medical University. It has 536 cases of which 163 are used as test cases, 9 features (values of biochemical tests) and 4 classes. This data has been used previously by Mitra, De and Pal [15] using a knowledge-based fuzzy MLP system with results on the test set in the range from 33% to 66.3%, depending on the actual fuzzy model used.

k-NN with Manhattan distance function reaches 77.9% accuracy for this dataset. This is already much better than many other methods give [16]. For example MLP neural network trained with RPROP gives accuracy that is below 70%. After applying feature selection 4 features were removed (features 2, 5, 6 and 9), increasing accuracy to 80.4%. With ASA optimization the best weighted k-NN model achieved

Table 4. Results for the hepatobiliary disease data

Method	Accuracy %	Variance %
LDA	65.0	–
C 4.5	75.5	–
1-NN, Manhattan	77.9	–
WNN, Manhattan, k=1	80.0	± 1.3
A5-WM	81.2	± 1.4
E5-WM	80.3	± 1.1
A5E5-WM	80.5	± 0.3
A5E10-WM	80.6	± 0.6

82.8% accuracy on the test set (83.4% on the training set). 5 averaged weighted k-NN methods gave better accuracy than ensembles or averaged ensembles, probably indicating that a better minimization should be used. These results are significantly better than those of all other classifiers applied to this data, including IB2-IB4, FOIL, LDA, DLVQ, C4.5, FSM and K* methods [16]. In particular poor results of the linear discrimination analysis should be noted.

Table 5. Results for the appendicitis data, using 10-fold stratified crossvalidation

Method	Accuracy %	Variance %
1-NN, Euclidean	85.8	± 1.4
8-NN, Manhattan	87.4	± 0.4
A5E5-WM, k=8	86.2	± 1.1
A5E10-WM, k=8	87.0	± 1.1

Appendicitis data, obtained from S. Weiss contains only 106 cases [18]. Since the data is too small to create separate test partition results given below were obtained from the 10-fold stratified crossvalidation tests. Since parameters of classification models may differ for each partition one could first optimize k and than perform weight optimization. We are aiming at a stable model for the whole data, therefore we have averaged first over the number of neighbors and selected optimal features (f_2 and f_4 features were removed). Using the value of k which was optimal for the largest number of partitions optimization of weights was performed for each partition and majority voting applied to the ensemble results.

The results of averaging and creating ensembles relatively to the single k=8 model have not been improved at all. They are already rather good: for comparison, accuracy of the C4.5 and CART in the leave-one-out tests is only 84.9% [18].

4 Conclusions and further work

Ensembles of models offer in some cases significant improvements over single classifiers but in other cases the results on the test set may get worse. The decrease of variance with the number of base classifiers may be slow. Averaging over model parameters - in this paper over feature weights – may help to decrease variance and sometimes gives better results than majority voting ensemble. Although sophisticated genetic-based models have been proposed to evolve the best subsets of models that should be included in ensembles [17] it is doubtful whether the added advantages justify the additional computational costs. Even the simplest majority voting method of combining *k*-NN models may significantly improve accuracy and stabilize results decreasing the variance of the final model.

For several datasets (except for the data described here we have tried sonar and hepatitis datasets from UCI [12]) the improvements have been insignificant. This shows that an ensemble of models of similar types may sometimes fail to improve the results. One reason for this may come from overfitting of the data (Ionosphere), another from poor correlation of results on the training and test sets. Full optimization of all model parameters in crossvalidation tests has not been done here (*k* was selected first, independently of weights), and that could have contributed to poor results on the appendicitis dataset. On the other hand results on the noisy hepatobiliary disorders data were significantly improved. For all data presented here results of the similarity-based algorithms were much better than results of popular C4.5 and CART decision trees.

Different types of procedures/parameters should be included in the search space for the best classification model [1]. The results could be improved further by employing better global minimization routines such as the multisimplex [5] or ASA (Adaptive Simulated Annealing) [6] but this leads to much higher computational cost of weighted methods which are already very expensive.

A number of other improvements to the ensemble creation algorithm presented here are planned. First, the majority voting scheme should be replaced by a linear combination of models. Combining probabilities of classification for each model M_l and using least square minimization procedure to determine coefficients of the combination may have some advantages. Second, the competence of each model should be included in the voting procedure, as described in the subsection on ensembles of competent models. Third, models of different type should be used in one ensemble. Fourth, boosting, bagging and other procedures [3,19] should help to create better pool of base models. We are currently investigating all these issues.

Acknowledgments: Support by the Polish Committee for Scientific Research, grant no. 8 T11E 042 19, is gratefully acknowledged.

References

1. Duch, W. (1998): A framework for similarity-based classification methods. In: Intelligent Information Systems VII, Malbork, Poland, 288-291

2. Grudziński, K., Duch, W. (2000): SBL-PM: A Simple Algorithm for Selection of Reference Instances for Similarity-Based Methods. In: Intelligent Information Systems IIS'2000. Physica Verlag, Springer, Berlin, Heidelberg, New York, 99-108
3. Breiman, L. (1998): Bias-Variance, regularization, instability and stabilization. In: Bishop, C. (Ed.) Neural Networks and Machine Learning. Springer, Berlin, Heidelberg, New York
4. Wettschereck D., Aha, W., Mohri, T. (1997): A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. Artificial Intelligence Review 11, 273-314
5. Gupta, H.V., Hsu, K., Sorooshian, S. (1997): Superior training of artificial neural networks using weight-space partitioning. In: Proc. of the Intern. Conf. Neural Networks (ICNN'97), Houston, 1919-1923
6. Ingberg, L. (1996): Adaptive simulated annealing (ASA): Lessons learned. J. Control and Cybernetics 25, 33-54
7. Bauer E., Kohavi, R. (1999): An empirical comparison of voting classfication algorithms: Bagging, Boosting and variants. Machine Learning 36, 105-139
8. Hinton, G. (2000): *Training products of experts by minimizing contrastive divergence.* Gatsby Computational Neuroscience Unit Technical Report 2000-004
9. Krogh, A., Vedelsby, J. (1995): Neural Network Ensembles, Cross Validation, and Active Learning. In: Advances in Neural Information Processing Systems 7, 231–238
10. Duch W., Grudziński K. (1999): Weighting and selection of features in Similarity-Based Methods. In: Intelligent Information Systems VIII, Ustroń, Poland, 32-36
11. Duch W., Grudziński K. (1999): Search and global minimization in similarity-based methods. In: Int. Joint Conference on Neural Networks (IJCNN), Washington, July 1999, paper no. 742
12. Mertz, C.J., Murphy, P.M., UCI repository of machine learning datasets, http://www.ics.uci.edu/AI/ML/MLDBRepository.html
13. Thrun S.B. *et al.* (1991): The MONK's problems: a performance comparison of different learning algorithms. Carnegie Mellon University, Technical Report CMU-CS-91-197
14. Shang N., Breiman, L. (1996): Distribution based trees are more accurate. Int. Conf. on Neural Information Processing, Hong Kong, Vol. I, 133-138
15. Mitra S., De R., Pal S. (1997): Knowledge based fuzzy MLP for classification and rule generation. IEEE Transactions on Neural Networks 8, 1338-1350
16. Duch, W., Adamczak, R., Grąbczewski, K., Żal, G., Hayashi, Y. (2000): Fuzzy and crisp logical rule extraction methods in application to medical data. In: P.S. Szczepaniak, P.J.G. Lisboa, J. Kacprzyk (eds.), Fuzzy systems in medicine. Physica - Verlag, Springer, Berlin, Heidelberg, New York, 593-616
17. Yao, X., Liu, Y. (1997): A New Evolutionary System for Evolving Artificial Neural Networks. IEEE Transaction on Neural Networks 8, 694–713
18. Weiss, S.M., Kapouleas, I. (1990): An empirical comparison of pattern recognition, neural nets and machine learning classification methods. In: Shavlik J.W., Dietterich, T.G., Readings in Machine Learning, Morgan Kauffman Publications, California
19. Opitz, D.W., Maclin, R. (1998): Popular ensemble methods: an empirical study, Journal of Artificial Intelligence Research 11, 169-198