An Experimental Comparison of Classifier Combining Methods Using Artificial Data

Fuad M. Alkoot

Higher Institute of Telecommunication & Navigation, PAAET
P.O.Box 313, Alshuhadaa, 48554, Kuwait

ABSTRACT

We experimentally compare the performance of four widely used combiner methods with the aim to show when our previously proposed method outperforms existing methods. Recently we have proposed a novel combiner method to detect autism. The proposed method outperformed existing combiner methods. However, we did not identify the reason behind its outstanding performance. We aim at finding when and why this method outperformed existing methods of bagging, boosting and random subspace methods. We repeat the experiments for varying number of classes, training set sizes and number of features using carefully designed artificial data.

Keywords: Classifier combining, bagging, feature based combiner, random subspace method, neural network.

Introduction

Machine learning tools are increasingly being used in many application areas to automate decisions. Researchers faced with the task of classification use classifiers or mathematical models that are able to perform the task of classification or decision making, based on a previously provided data [20]. These classifier models or experts have an ability to spot trends and relationships in large data sets, which makes them well suited for many applications.

In order to improve the accuracy of classifiers researchers have found that combining (fusing) the decisions of more than one classifier would yield superior results over the best single classifier. The publications [8,13,11,12, 14,17,18,19] are examples of early attempts at proving and using combiners successfully. The surprising benefits that can be accrued from the combination of multiple classifier designs over the best single expert has motivated a substantial research community to develop and investigate diverse approaches to multiple expert fusion [9,6,15,7, [21]. These range from simple combination rules which do not require any training [1,2,5,8,10], to sophisticated fusion mechanisms which can cope with unequal expert strengths, can even dynamically adapt to data [14,16].

In [3,4] we proposed a novel design philosophy for classifier combination by taking the view that the design of individual experts and fusion cannot be solved in isolation. Each expert is constructed as part of the global design of a final multiple expert system. The design process involves jointly adding new experts to the multiple expert architecture and adding new features to each of the experts in the architecture. The initial experiments showed a minor benefit from the new strategy [3,4]. Further investigation in [22] showed a major advantage over bagging and random subspace method or decision tree forests.

The reason behind the great performance was not identified. Therefore, we aim at finding when and why our method outperforms other methods. We can achieve that using theoretical or analytical methods of proof. In our experiments we aim at experimentally proving the advantage of our previously proposed combiner method over existing methods. We achieve that, at this initial phase, using different types of synthetic data.

In the next section we describe the combiner methods and the classifiers used in our experiments. We also describe how the artificial data sets were created to simulate the various conditions under investigation. In section 3 results are presented followed by the conclusion in section 4.

Experimental Methodology

- **Combiner methods**

We aim at comparing combiner methods at various conditions. Experiments are designed for different data types using k-NN and neural network classifiers. The compared combiners are bagging [6], random subspace, RSM [8], ArcX4 boosting [23] and our previously proposed Feature Selection based combiner, FSC [3, 4, 22]. Bagging predictors proposed by Breiman, is a method of generating multiple versions of a predictor or classifier, via bootstraping and then using those to get an aggregated classifier. The total number of samples in each bootstrap set is equal to those of the original training set. The second combiner 'RSM' [8] aims at creating diverse classifiers by assigning different features to each classifier. The number of features is set at a fixed value, m, less than the total number of features. Each classifier is assigned a subset of features that are randomly selected without replacement from the full feature set. This results in classifiers having different views of the data space. We set m to equal 67 percent of the total number of available features. In comparison to 50% recommended by [8] we found better rates are achieved at 67%. The third combiner is the feature selection based combiner, FSC, proposed by Alkoot & Kittler [3], [4], [22] and it is based on the principal that the feature selection and the combiner performance are linked. The best feature subset is selected for each classifier based on the combiner system performance instead of the individual classifier performance. The maximum possible number of classifiers that can be fused in the system is limited by the number of available features. We have set it to

5 maximum number of classifiers. Any feature selection method can be used to add the best feature subset such that the combiner system error rate is minimized. For each classifier under construction one feature is inserted at a time and the system performance is checked. After checking all features, the feature yielding the best system performance is permanently inserted to the classifier under construction. When the feature insertion process is completed for the maximum number of classifiers in the system the process is repeated from the first classifier until all features are used up or the system error rate is not improved by the insertion. The process continues if the addition of a new feature does not degrade the system performance, and there are an unused number of features. However, on the first run across the classifiers we add a feature even if it does not improve the system. That is, we force the insertion of the best feature to the classifiers, even if that does not improve the system. The feature selection method used is the 2-forward-1-backward method.

- **Classifier types**

We focus our experiments on two commonly known classifiers, k-nearest neighbor and neural network classifiers, [20]. For the nearest neighbor classifier k is set at $\sqrt{N}$, where N is the square root of the number of training samples. The distance metric used is the mahalanobis metric. The neural network classifier used here consists of three layers. The transfer function or output of the first two layers is log-sigmoid, while that of the output or third layer is purelin. The network training function used is backpropagation. The number of neurons in the first layer is equal to the number of features, while that for the hidden (second) layer is set at 5. The number of neurons at the output layer is equal to the number of classes.

- **Data creation method**

Combiner performance depends on many factors such as classifier type, fusion methods, number of combined classifiers and the data set. Some of the characteristics of the data set that may affect the combiner performance are number of features, number of samples, the number of classes and degree of overlap between the samples from the various classes. In this paper, we aim at finding the effect of data characteristics on combiner performance. Therefore, we create synthetic data with varying values of the aforementioned data characteristics.

We create synthetic data by adding two random numbers. One is generated from a normal distribution with zero mean and standard deviation S. The second is generated from a uniform random number generator between 0 and 1 which is multiplied by a factor n. Next, we add these two random outcomes to create a feature value for each sample. For each feature the values of S and n are changed, where S ranges between 0.1 and 1 for each feature, and n ranges between 0 and 0.9 for each S. This yields 100 combinations that constitute different distributions for 100 features. Samples of each feature are shifted so that the minimum sample value is zero.

For each of the 25 classes 500 samples at each feature are generated. However, values of each additional class above class 1 are shifted by an amount h to avoid overlap with the lower class. The value of h determines the amount of overlap with the previous class, and hence the degree of data difficulty.

Therefore, h =mean of lower class + f x S; where S is the standard deviation of the normal distribution from which the samples were drawn. We set f = 2 to generate easy data. Two additional data are generated by setting f to a random number generated from a normal distribution with mean = -3 and standard deviation =1 and mean = --4 and standard deviation =1. These two versions represent more difficult data, where the data values of the second class are shifted by less amount making the overlap with the previous classes higher. Figures show the data sample distributions for various classes and features. Table 1 presents the various parameter values of the created data sets.

Table 1. Data sets parameters

Samples	100	200	500	--	--	--
Features	10	20	80	150	500	--
Classes	2	3	4	5	15	25

The combination of these parameters yields 3 x 5 x 6 = 90 possible data sets. For each data set we repeat the experiments for two classifier types; namely k-NN and neural network classifiers. Also, the experiments are repeated for various number of combined classifiers of 3, 5, 7, 9 and 11.

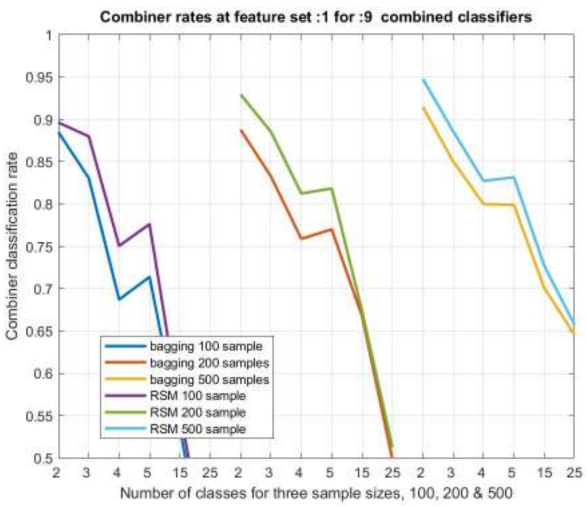
To compare the combiner methods, we need to repeat all the experiments for each of the four combiner methods. For each combiner the number of run experiments are 900, that yield a total of 3600 classification rates for all four combiners.

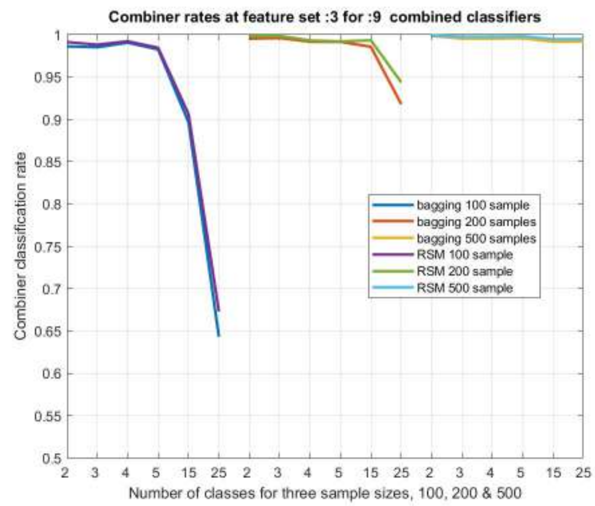
Currently, we have created 54 data sets. In the near future we will create the data sets for feature values 150 and 400. For the created data sets, we finished experiments on the bagging and RSM combiners. In the remaining time we will run the experiments for the remaining two combiner methods.

Results

In each of the figures below we present results for bagging and RSM combiners using the k-NN classifier, Fig. 1, and the neural network classifier, Fig. 2, at feature set sizes 10, 20, and 80, and using 9 combined classifiers. Each figure presents classification rates for both combiners at three sample sizes and for different number of classes. Experiments are ongoing because neural networks and the feature-based combiners require more time to finish. The runs for larger feature set sizes and boosting combiner will be conducted next.

Using a small number of features equal to 10 and for 3 combined classifiers we note that bagging outperforms RSM. For both combiners the classification rate decreases as the number of classes in the data set increases. As the feature set size increases to 20 the performance of both combiners increases with RSM achieving higher improvement as it reduces the gap with bagging. This trend of improvement continues for the largest feature set size of 80 features. We find that the behavior of combiners did not change with the change in the choice of combined classifiers, or number of combined classifiers.





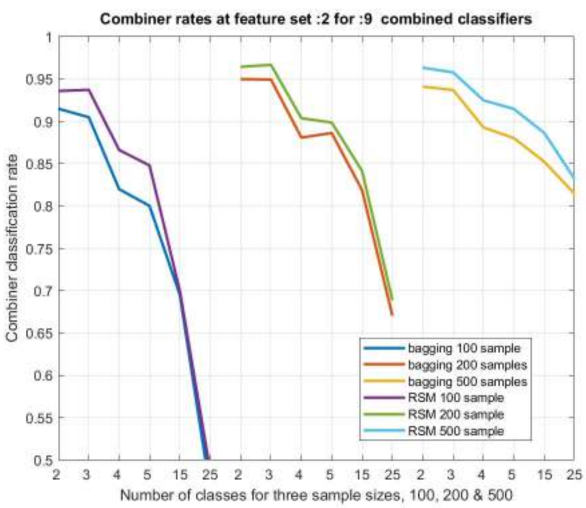
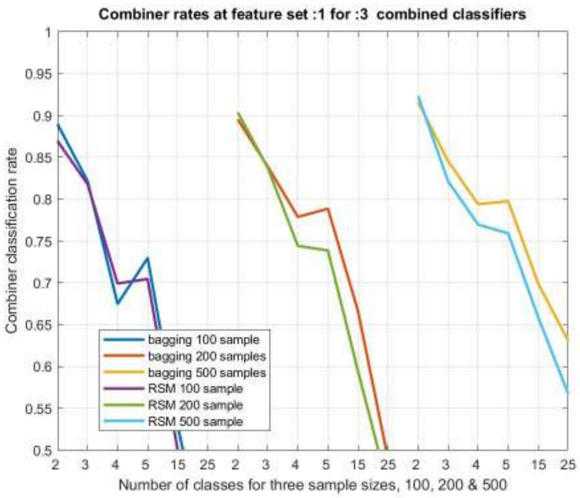
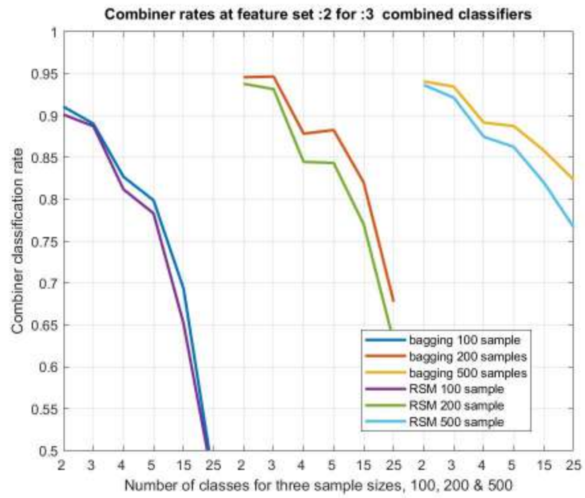


Figure 1: Comparing bagging and RSM methods when using neural networks classifiers at 10 , 20 and 80 features, using 9 combined classifiers





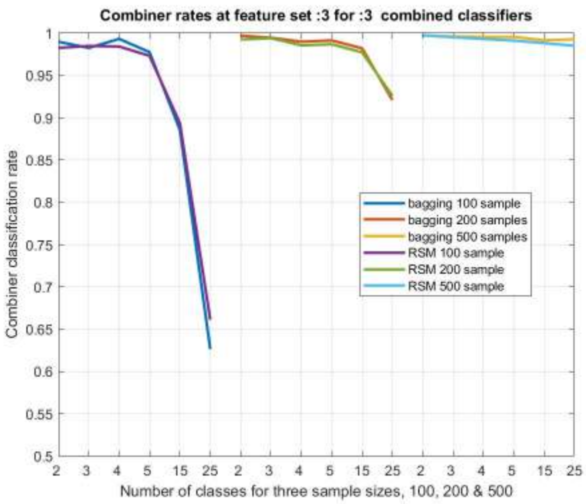


Figure 2: Comparing bagging and RSM methods when using neural networks classifiers at 10 , 20 and 80 features, using 3 combined classifiers

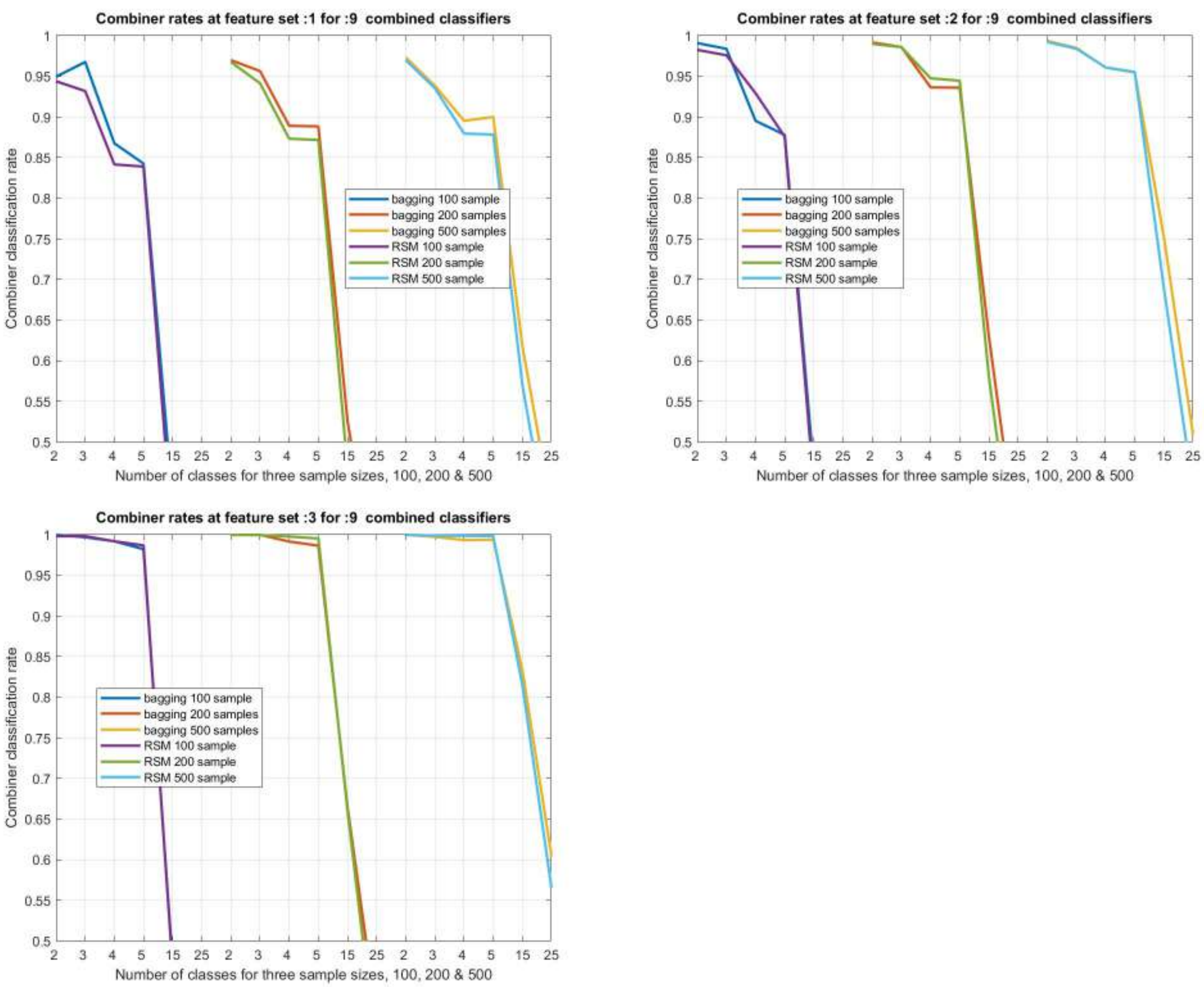


Figure 3: Comparing bagging and RSM methods when using neural networks classifiers at 10 , 20 and 80 features, using 9 combined classifiers

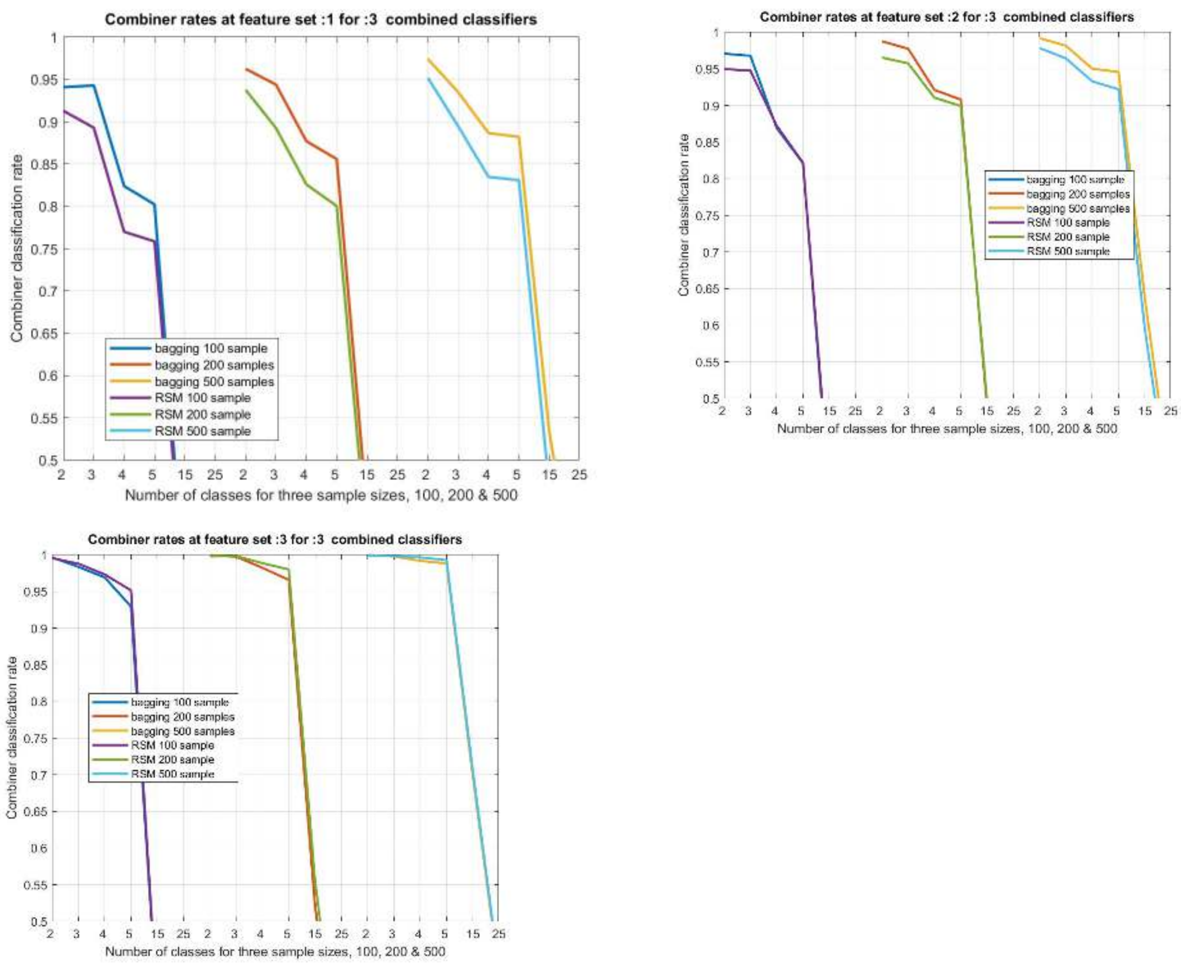


Figure 4: Comparing bagging and RSM methods when using neural networks classifiers at 10 , 20 and 80 features using 3 combined classifiers

Conclusion

Classifiers play a pivotal rule in the design of machine learning systems. Most research is directed towards improving the performance of classifiers. Classifier combination is one of the most effective methods to improve the performance rate of classifiers. Its use has been widely adapted by most researchers. However, different combiners with different performance rates exist. Scientists aim to select the most suitable combiner method for their application. The outcome of our project will help scientists and engineers identify which combiner method is most suitable, given some information on the application data. Currently we are running experiments on many different conditions or parameters. Current results show that as the number of classes increases both bagging and RSM underperform. Misclassification rate increases at higher number of classes as the number of features decreases. Future work will be on the feature-based combiner to see when it outperforms the existing methods.

Acknowledgement

The author thanks the public Authority for Applied Education & Training for supporting the paper and the research project under grant number TR-18-01.

References

[1] F. M. Alkoot and J. Kittler. Moderating knn classifiers. Pattern Analysis and Applications, 5(3):326-332, 2002

[2] F. M. Alkoot and J. Kittler. Modified product fusion. Pattern Recognition Letters, 23(8):957-965,2002

[3] F. M. Alkoot and J. Kittler. Feature selection for an ensemble of classifiers. In Proceedings of the SCI2000 conference, pages 622--627, Orlando, Florida, 2000.

[4] F. M. Alkoot and J. Kittler. Multiple expert system design by combined feature selection and probability level fusion. In Proceedings of the Fusion 2000 conference, volume II, pages ThC5(9--16), Paris, France, 2000.

[5] F. M. Alkoot and J. Kittler, "Experimental Evaluation of Expert Fusion Strategies", Pattern Recognition Letters, vol.20, Issue 11-13, Nov. (1999)

[6] L. Breiman. Bagging predictors. Machine Learning, 24:123–140, 1996.

[7] L.K. Hansen and P. Salamon. Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(10):993–1001, 10 1990.

[8] T. Ho. The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(8):832{844, 1998.

[9] M I Jordan and R A Jacobs. Hierarchical mixture of experts and the em algorithm. Neural Computation, 6:181–214, 1994.

[10] J. Kittler and F. M. Alkoot. Sum versus vote fusion in multiple classifier systems. IEEE Trans on PAMI, 25(1):110--115, 2003.

[11] J. Kittler, M. Hatef, R. Duin, and J. Matas. On combining classifiers. IEEE Trans Pattern Analysis and Machine Intelligence, 20(3):226–239, 1998.

[12] J. Kittler. Combining classifiers: A theoretical framework. Pattern Analysis and Applications, 1:18–27, 1998.

[13] D. Lee and S.Srihari. Dynamic classifier combination using neural networks. SPIE, 2422:26–37, 6 1995.

[14] G. Rogova. Combining the results of several neural network classifiers. Neural Networks, 7(5):777–781, 1994.

[15] R. E. Schapire. A brief introduction to boosting. In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1999.

[16] K Woods, W P Kegelmeyer, and K Bowyer. Combination of multiple experts using local accuracy estimates. IEEE Trans Pattern Analysis and Machine Intelligence, 19:405–410, 1997.

[17] Yuehui Chen, Author Vitae, Yaou Zhao, Yuehui Chen, Yaou Zhao. A novel ensemble of classifiers for microarray data classification. Applied Soft Computing Volume 8, Issue 4, Pages 1664–1669. 2008

[18] Kyung-joong Kim, Sung-Bae Cho, An evolutionary Algorithm Approach to optimal ensemble classifiers for DNA microarray data analysis. IEEE trans. On Evolutionary computation, vol 12, no 3, 2008.

[19] M. Dettling, "Bagboosting for tumor classification with gene expression data," Bioinformatics, vol. 20, no. 18, pp. 3583–3593, 2004.

[20] A.R. Webb. Statistical pattern recognition, 2nd ed., John Wiley and sons. 2002.

[21] Fuad M. Alkoot and Josef Kittler, "Population Bias Control for Bagging k-NN Experts", In Proceedings Sensor Fusion: Architectures, Algorithms, and Applications V, Orlando, Florida, 17-20/April. SPIE Volume 4385, pp36. 2001.

[22] Fuad Alkoot, A Alqallaf, Investigating machine learning techniques for the detection of autism. International Journal of Data Mining and Bioinformatics 16 (2), 141-169.

[23] Leo Breiman, Arching classifier, Ann. Statist. Vol. 26, number 3(1998), 801-849.