Experiments on Combining Classifiers

Hüseyin Gökhan Akçay

Bilkent University, Department of Computer Engineering,
06800 Ankara, Turkey
akcay@cs.bilkent.edu.tr

Abstract. In this paper, experiments on various classifiers and combining these classifiers are done, reported and analyzed. Combining the classifiers means having the single classifiers support each other in making a decision, instead of having only a single classifier's decision as the final decision. The base experiment involves, both, applying different single classifiers on a dataset and applying the combination of those classifiers on the same dataset. Also, the same experiment is conducted after applying feature reduction, Bagging and Boosting methods separately. The dataset used in our experiments has not been used in any publication yet. The performance of the classifiers in terms of accuracies is compared in all experiments. In the experiments, classifier combinations perform slightly better than the single classifiers in terms of overall accuracy. It is also seen that bad classifiers and features may contain valuable information for performance improvement in terms of accuracy by combining rules.

1 Introduction

The main objective of designing many machine learning systems is to achieve the best classification. This aim led to the development of different classification schemes for any machine learning problem. It has been observed that in such design studies, usually, none of the single classifiers, such as Bayes-normal, Parzen Windows, Decision Trees, Neural Networks and Support Vector Classifiers, is enough to distinguish the pattern classes optimally. The reason is that although one of the classifiers would give the best performance, the sets of patterns classified by the different classifiers would not necessarily overlap. Hence, different classifiers may be desired for different features.

These observations motivated the interests in combining classifiers. The idea is not to rely on a single decision making scheme. Instead, many single classifiers are used for decision making by combining their individual opinions to derive a consensus decision. The aim is to determine an effective combination method that makes use of the benefits of each classifier but avoids the weaknesses. Various classifier combination schemes have been devised and it as been experimentally demonstrated that some of them consistently outperform a single best classifier.

There are different ways of how the single classifiers are combined[3]. In this paper, three most popular classifier combining rules are tested and compared to single classifiers in terms of accuracy: Maximum combining classifier, Voting combining classifier and Product combining classifier.

The features of the dataset used in this paper are very similar to ones used in remote sensing image pixel classification. So, the results obtained in this paper can guide to researchers on that area about which classifiers may perform better on remote sensing images and whether combination methods would work for remote sensing data.

The paper is organized as follows. In section 2, related research is discussed. In Section 3, the dataset used in the experiments are explained. In Section 4, the single classifiers and the classifier combination rules are presented. In Section 5, the experimental results are given. In Section 6, the

results are analyzed. In Section 7, future work is explained. Finally, conclusions are drawn in Section 8.

2 Related Work

Kittler et al.[2] makes an experimental comparison of various classifier combination schemes and concludes that sum rule-outperforms other classifier combinations. Similar studies have been done by Duin et al.[1],[11]. In [1], he performs similar experiments by using a different digit data set. The results of his research are discussed in section 6. In [11], he presents an intuitive discussion on the use of trained combiners.

3 The Dataset

the dataset used in this paper is an image segmentation dataset. The dataset comes from the UCI repository of machine learning databases. This dataset has not yet been used in any other publication. The task is to classify the center pixel of a 3x3 patch from an image as belonging to one of 7 categories (brickface, sky, foliage, cement, window, path, grass). The inputs are typical image processing features of the patch. The features are very similar to the ones used in remote sensing image pixel classification. Hence, one can generalize the results of this experiment to remote sensing classification problems. Relevant information about the dataset is as follows:
- Number of Instances: Training data: 210 Test data: 2100
- Number of Attributes: 19 continuous features
- Feature Information:

1. region-centroid-col: the column of the center pixel of the region.
2. region-centroid-row: the row of the center pixel of the region.
3. region-pixel-count: the number of pixels in a region = 9.
4. short-line-density-5: the results of a line extraction algorithm that counts how many lines of length 5 (any orientation) with low contrast, less than or equal to 5, go through the region.
5. short-line-density-2: same as short-line-density-5 but counts lines of high contrast, greater than 5.
6. vedge-mean: measure the contrast of horizontally adjacent pixels in the region. There are 6, the mean and standard deviation are given. This attribute is used as a vertical edge detector.
7. vegde-sd: (see 6)
8. hedge-mean: measures the contrast of vertically adjacent pixels. Used for horizontal line detection.
9. hedge-sd: (see 8).
10. intensity-mean: the average over the region of (R + G + B)/3
11. rawred-mean: the average over the region of the R value.
12. rawblue-mean: the average over the region of the B value.
13. rawgreen-mean: the average over the region of the G value.
14. exred-mean: measure the excess red: (2R - (G + B))
15. exblue-mean: measure the excess blue: (2B - (G + R))
16. exgreen-mean: measure the excess green: (2G - (R + B))
17. 17. value-mean: 3-d nonlinear transformation of RGB. (Algorithm can be found in Foley and VanDam, Fundamentals of Interactive Computer Graphics)
18. saturation-mean: (see 17)
19. hue-mean: (see 17)

- Class Distribution:

- Classes: brickface, sky, foliage, cement, window, path, grass.
- 30 instances per class for training data.
- 300 instances per class for test data.

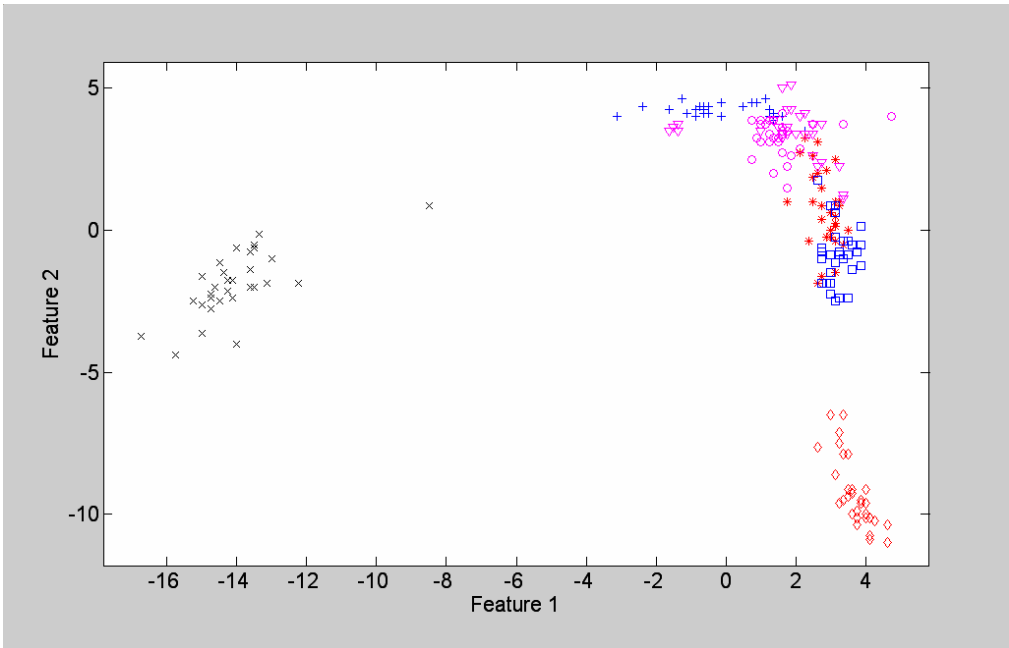

Fig. 1: Scatter plots of all classes, mapped on their first 2 principal components

In Fig. 1, scatter plots of the first two principal components (PCA) of the dataset is shown. The PCA plots are focused on the data distributions as a whole. Although it is not possible to extract quantitative features from these plots, they show that the data sets have nearly distinct class distributions.

4 The Classifiers[1]

For this experiment, classifiers are taken from the Matlab toolbox PRTools[7]. However, it is important to make the outputs of the classifiers comparable. For this purpose, normalized posterior probabilities are used. A posterior probability is a number $pij(\boldsymbol{x})$, in range between 0 and 1, computed for test sample $\boldsymbol{x}$ for each of the c classes, on which each of the m classifiers are trained. Each $pij(\boldsymbol{x})$ is normalized so that

$$\sum_{j}^{c} pij(\boldsymbol{x}) = 1, \quad 1 \le i \le m \tag{1}$$

4.1 Single Classifiers

A single classifier selects the class which maximizes the normalized probability. The classifiers, used to find each normalized probability $pij(\boldsymbol{x})$ for a given test sample $\boldsymbol{x}$, are summarized below:

Quadratic Bayes Normal Classifier (Bayes-Normal-2) This is the Bayes rule assuming Gaussian distribution, with a separate covariance matrix, for each class. Since the number of features is large for covariance matrix estimation, backward feature selection is applied before training the classifier.

Linear Bayes Normal Classifier (BayesNormal-1) This is, again, the Bayes rule assuming Gaussian distribution, with the same covariance matrix, for each class.

Nearest Mean Classifier This classifier assigns a sample to the class having the nearest mean.

Nearest Neighbor Classifier This rule assigns a sample to the class associated with the nearest neighbor in the training set.

[1] The classifiers' implementations are not explained in detail, since our aim is only to experiment their performances in terms of accuracy.

K-Nearest Neighbor Classifier In this rule, a sample is assigned to the class such that the majority of the k nearest neighbors in the training set belongs to.

Parzen Classifier Class densities are estimated using Gaussian kernels for each training sample. The kernel width is optimized for the training set using a leave-one-out error estimation.

Decision Tree Classifier Our algorithm computes a binary decision tree on the multi-class dataset. Thresholds are set such that the impurity is minimized in each step [8]. Early pruning is used in order avoid overtraining [9].

Neural Network Classifier This is a feed-forward neural network classifier with 1 hidden layer with 20 units.

Support Vector Classifier This is the standard SVC using a linear inner product kernel [10].

4.2 Combining Rules

Once the posterior probabilities $\{pij(\boldsymbol{x}), i = 1,m; j = 1,c\}$ for m classifiers and c classes is computed for test object $\boldsymbol{x}$, they have to be combined into a new set $qj(\boldsymbol{x})$ which can be used for the final classification. $qj(\boldsymbol{x})$ is computed by:

$$q\,j'(\boldsymbol{x}) = \text{rule}i(pij(\boldsymbol{x})) \tag{2}$$

$$q\,j(\boldsymbol{x}) = \frac{q\,j'(\boldsymbol{x})}{\sum_j pj(\boldsymbol{x}) = 1} \tag{3}$$

The final classification is made by:

$$\varphi(\boldsymbol{x}) = \text{argmax}j(q\,j(\boldsymbol{x})) \tag{4}$$

Three combining rules are used for rule in (2):

Voting combining classifier The classiffication is done by counting the votes for each class over the input classifiers and selecting the majority class. If this rule is substituted in 2:

$$q'_j(\boldsymbol{x}) = \sum_i \text{l}(\text{argmax}_i(p_{ij}(\boldsymbol{x}))=i), \qquad \text{I(y)=1 if y is true and I(y)=0 o.w.} \tag{5}$$

Maximum combining classifier The maximum combining classifier selects the selection of the single classifier giving the highest normalized probability. If this rule is substituted in 2:

$$q'_j(\boldsymbol{x}) = p_{\text{argmax}i(pij(\boldsymbol{x})),j}(x) \tag{6}$$

Product combining classifier In this rule, each single classifier gives a normalized probability value for each class. Then all normalized probabilities are multiplied per class. The class with the highest probability product wins. If this rule is substituted in 2:

$$q'_j(\boldsymbol{x}) = \prod_i p_{ij}(\boldsymbol{x}) \tag{7}$$

Note that each combining rule gives the same importance to each of the single classifiers in order to derive the final classification.

5 The Experiments

The experiments are aimed to compare the performance of single classifiers and their combination (including all single classifiers in the combination) in terms of accuracy. The single classifiers and their combination schemes used in the experiment are discussed in section 3. In the experiments, the typical design cycle of a pattern recognition system is followed: Data collection, feature choice, classifier choice, classifier training, and classification. The experiments can be split into four parts:

- Experiment over single classifiers and their combination
- Experiment over single classifiers using feature reduction and their combination
- Experiment over single classifiers using boosting and their combination
- Experiment over single classifiers using bagging and their combination

Each experiment is discussed below:

5.1 Experiment over single classifiers and their combination

To obtain baseline results, the single classifiers and their combinations are applied to the dataset. The results for each of the single classifiers and each of the three combination schemes are shown in Table 1. The accuracies for each class and the overall accuracy are demonstrated in the table. The results are given in percentages of correctly classified samples.

	Cl1	Cl2	Cl3	Cl4	Cl5	Cl6	Cl7	Overall
BayesNormal_1	0.9867	0.8400	0.8567	0.9900	0.9967	1.0000	0.6467	0.7600
BayesNormal_2	0.9633	0.9167	0.6267	0.9933	0.9900	0.9967	0.8567	0.9062
decisionTree	0.8967	0.8500	0.8100	0.9867	0.9700	1.0000	0.7833	0.8995
FLD	0.9733	0.4967	0.8933	0.9900	1.0000	0.9633	0.6900	0.8580
KNN	0.8833	0.8200	0.7100	0.9833	0.9400	1.0000	0.8000	0.8766
NearestMean	0.7100	0.7100	0.0867	0.8433	0.8967	0.9967	0.6767	0.7028
NearestNeighbor	0.8833	0.8200	0.7100	0.9833	0.9400	1.0000	0.8000	0.8766
NeuralNetwork	0.9567	0.9067	0.8400	0.9900	1.0000	1.0000	0.7767	0.9243
Parzen	0.8933	0.7867	0.3500	0.9900	1.0000	1.0000	0.8333	0.8361
SVC	0.9833	0.7867	0.9267	0.9900	0.9967	1.0000	0.7133	0.9138
Voting	0.9967	0.9033	0.8600	0.9967	1.0000	1.0000	0.8233	0.9400
Maximum	0.9867	0.8367	0.8733	0.9900	0.9967	1.0000	0.6467	0.9043
Product	0.9333	0.8567	0.6633	0.9900	1.0000	1.0000	0.8567	0.9000

Table 1: Accuracy rates for the experiment over single classifiers and their combination.

5.2 Experiment over single classifiers using feature reduction and their combination

It is generally observed that each feature in the dataset may not be useful for at least some of the discriminations. This is called the curse of dimensionality and some feature selection methods reduce dimensionality by selecting subsets of existing features in order to remain only the meaningful features. In this experiment, Sequential backward selection algorithm[12] is used in order to reduce the feature space. The single classifiers are trained in this reduced feature space and their combinations are taken accordingly. However, this experiment is not complete since we have results only for 6 out of 10 single classifiers yet. Also, no combination results is ready. Since we have not enough results for this experiment, it will no be analysed in section 6. In Table 2, the results of 6 single classifiers trained on reduced feature space are shown for each class and in overall. The results are given in percentages of correctly classified samples.

	Cl1	Cl2	Cl3	Cl4	Cl5	Cl6	Cl7	Overall
BayesNormal_1	0.9833	0.8100	0.8933	0.9900	0.9967	1.0000	0.6500	0.9033
BayesNormal_2	0.9633	0.9167	0.6267	0.9933	0.9900	0.9967	0.8567	0.9067
decisionTree	0.8967	0.8500	0.8100	0.9867	0.9700	1.0000	0.7833	0.8995
FLD	0.9733	0.4967	0.8933	0.9900	1.0000	0.9633	0.6900	0.8580
KNN	0.8833	0.8200	0.7100	0.9833	0.9400	1.0000	0.8000	0.8766
NearestMean	0.7100	0.7100	0.0867	0.8433	0.8967	0.9967	0.6767	0.7028

Table 2: Accuracy rates for the experiment over single classifiers using feature reduction and their combination.

5.3 Experiment over single classifiers using boosting and their combination

Boosting[5] is a general supervised method used to increase the accuracy of any classifier. It generates a classifier with a smaller error on the training data as it combines multiple hypotheses which individually have a large error. In this experiment, the single classifiers are used for boosting and the boosting results of each are used in combinations. In Table 4, the results of boosting of each classifier and their combinations are shown for each class and in overall. The results are given in percentages of correctly classified samples.

	Cl1	Cl2	Cl3	Cl4	Cl5	Cl6	Cl7	Overall
BayesNormal_1	0.9700	0.8233	0.7633	0.9900	1.0000	1.0000	0.6467	0.8847
BayesNormal_2	0.9567	0.9033	0.6233	0.9900	0.9900	0.9967	0.8667	0.9038
decisionTree	0.7200	0.3500	0.6400	0.7767	0.8567	0.7900	0.5700	0.6719
FLD	0.9467	0.4667	0.8633	0.9900	1.0000	0.9033	0.6800	0.8357
KNN	0.7600	0.8167	0.7100	0.9833	0.9000	1.0000	0.7300	0.8428
NearestMean	0.7167	0.6600	0.0800	0.8433	0.8900	0.9967	0.6500	0.6909
NeuralNetwork	0.9900	0.8167	0.8167	0.9900	0.9933	1.0000	0.7433	0.9071
Parzen	0.7900	0.8500	0.5967	0.9800	0.9367	0.9933	0.7633	0.8442
SVC	0.9933	0.7000	0.8400	0.9767	0.9967	1.0000	0.7233	0.8900
Voting	1.0000	0.8600	0.8800	0.9900	1.0000	1.0000	0.7467	0.9252
Maximum	0.9700	0.8200	0.7867	0.9900	1.0000	1.0000	0.6467	0.9009
Product	0.8000	0.9100	0.8267	0.9900	0.9800	0.9967	0.7233	0.8895

Table 3: Accuracy rates for the experiment over single classifiers using boosting and their combination.

5.4 Experiment over single classifiers using bagging and their combination

The bagging algorithm[6] tries to increase the accuracy of any classifier by performing bootstrapping iteratively. In this experiment, the single classifiers are used for bagging and the bagging results of each are used in combinations. In Table 4, the results of bagging of each classifier and their combinations are shown for each class and in overall. The results are given in percentages of correctly classified samples.

	Cl1	Cl2	Cl3	Cl4	Cl5	Cl6	Cl7	Overall
BayesNormal_1	0.9867	0.8000	0.8267	0.9900	1.0000	1.0000	0.6900	0.8990
BayesNormal_2	0.9733	0.9533	0.6433	0.9933	0.9900	0.9967	0.8333	0.9188
decisionTree	0.7133	0.4300	0.7567	0.9033	0.9400	0.9700	0.6300	0.7633
FLD	0.9633	0.4533	0.8300	0.9900	1.0000	0.9433	0.6300	0.7027
KNN	0.8600	0.7700	0.7267	0.9233	0.8900	1.0000	0.6300	0.8285
NearestMean	0.6633	0.6467	0.1867	0.8833	0.9033	0.9967	0.6900	0.7100
NeuralNetwork	0.9767	0.7467	0.9100	0.9900	1.0000	0.9967	0.8300	0.9214
Parzen	0.8067	0.8900	0.7133	0.9700	0.9933	0.9867	0.7600	0.8742
SVC	0.9333	0.6900	0.8067	0.9900	0.9933	0.9267	0.7467	0.8695
Voting	0.9933	0.8567	0.8933	0.9933	1.0000	0.9967	0.7767	0.9300
Maximum	0.9867	0.7967	0.8300	0.9900	1.0000	1.0000	0.6900	0.8990
Product	0.8567	0.8967	0.8467	0.9900	1.0000	0.9933	0.7867	0.9100

Table 4: Accuracy rates for the experiment over single classifiers using bagging and their combination.

6 Analysis

The main goal of these experiments is to compare the performances of single classifiers and their combinations (including all single classifiers in the combination) in terms of accuracy. The same comparison is done on bagged and bootstrapped single classifiers and their combinations, too. In addition, the single classifiers are trained on the data set whose feature space is reduced and these trained classifiers are combined. As well, it is analyzed which classifier combination method would perform best[2].

[2] Since both time and space complexity of classifiers is out of the scope of this paper, they are not analyzed.

The experiments demonstrate that single classifiers perform well on the dataset. Also, the results show that, in overall, combination rules, especially the Voting rule, give good results in all experiments. Duin et al. [1], also concludes that combining classifiers are very useful but, however, he observes that there is no overall winning combining rule.

From the experiments, it is interesting to note that even if the use of the combination rules yield better results in overall than by using each of the classifiers individually, single classifiers give better results considering accuracies for each class separately.

The Maximum and the Product rules seem to be noise sensitive as stated also in [1]. In all of the experiments, it can be investigated that a sudden decrease in the accuracy of some individual classifiers is most likely to decrease the accuracy of the Maximum and the Product rules. For example, if the results of class 3 are considered in all experiments, it is seen that a sudden decrease seen in the results for the NearestMean classifier more rapidly decrease the accuracies for class 3 in these two rules.

If the experiment over single classifiers and their combination is analyzed, Voting rule outperforms the single classifiers and other combinations, in overall. The reason why the Maximum and Product rules even perform worse than some single classifiers is this property their sensitivity to noise property such that a low performance even in a single classifier decreases these rules' performances rapidly. So, the Maximum or Product rules may not improve the performance compared to single classifiers, if the accuracy rates between the single classifiers differ a lot. In our experiments, since the number of single classifiers combined is 10, which is relatively large, the noise is very likely to be high. So, it is natural that the Maximum and Product rules do not improve the performance of some single classifiers.

If the experiment over single classifiers using bagging and their combination is analyzed, it is seen that Voting rule still outperforms the single classifiers and other combinations, in overall. Also, the bagging and boosting strategies does not seem to work well in this dataset for both the single classifiers and their combinations. Duin et al.'s experiments also indicate that combining different classifiers trained on the same classifier (e.g. boosting and bagging) may improve the accuracy rate, but is generally far less useful.

When the experimental results are analyzed, it can be realized that if the accuracies for all classes seem satisfactory, using combinations does not seem to improve the accuracy performance very much. But, in most cases using combinations may decrease the accuracy performance. However, if the percentages in some classes need to be improved, trying combinations may increase the individual classes' performance rapidly. But, in this case, combinations may degrade some other classes' performances giving sufficient results without any combination. Our experiments have many such other examples. For example, in the experiment over single classifiers and their combination, if one uses FLD, class 3 performance gives a performance of 0.4967, which is relatively bad. So, applying combination rules increase the accuracy rate for class 3 rapidly, while decreasing some classes' accuracy rates. But, in all such cases in all experiments, the decrease rates in such classes' accuracy performances seem to be very small compared to the increase rates in individual classes' accuracy performances.

7 Future Work

The main aim for future work is to use the decision of each single classifier in combination. Because, a classifier, which gives bad results in overall, may classify a sample correctly even if most it cannot be classified correctly by most of the other single classifiers. However, each of the combination rules in this paper gives equal weights into each single classifier's decision in order to derive the final decision, though one single classifier may be better in classifying the dataset than another. So, instead of behaving each individual classifier equally, an alternative approach that gives different weights into different classifiers' decisions may be followed. For example, in the experiment over single classifiers and their combination, although the NeuralNetwork classifier gives the best individual overall result, its decision is given the same weight with the decision of the NearestMean classifier, which gives the

worst individual overall result. So, it may be wiser to give the NeuralNetwork classifier more weight than the NearestMean classifier in the final decision.

In summary, we aim to use as many classifiers as possible. Then, the weight of each classifier in the final decision is assigned by its reliability and the reliability of each classifier is determined by its individual classification result.

8 Conclusion

In this paper, three classifier combination rules were investigated over a dataset which has not been used in any publication, yet. The features of the dataset are very similar to remote sensing datasets. Therefore, the experimental results presented here may serve alternatives to classification methods for classification of remote sensing data.

The experiments demonstrated that classifier combination methods can be considered desirable alternatives to single methods. All three combination rules outperformed several single classifiers in terms of accuracies. Especially, the Voting Rule is shown to be the best choice if the number of single classifiers to be combined is big.

In particular, combinations over single classifiers without performing any bagging, boosting and feature reduction gave the best results.

As future work, a combination schema that tries to make use of as many classifiers as possible in order to arrive at a final decision is to be tried. In this schema, each classifier is given a different weight in the final decision according to its reliability.

References

[1] R.P.W. DUIN, D.M.J. TAX, "Experiments with Classifier Combining Rules", MCS 2000, LNCS 1857, Springer-Verlag.

[2] J. KITTLER,M.HATEF, R.P.W. DUIN,J.MATAS, "On Combining Classifiers", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20 (3) (1998), 226--239.

[3] M. van Erp, L. Vuurpijl, and L. Schomaker, *An overview and comparison of voting methods for pattern recognition*, in Proc. of the 8th IWFHR. 2002, pp. 195-- 200, IEEE.

[4] G. J. Briem, J. A. Benediktsson, J. R. Sveinsson, "Multiple Classifiers Applied to Multisource Remote Sensing Data", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, no. 10, 2002.

[5] R. E. Schapire, "A brief introduction to boosting," in *Proc. 16th Int. Joint Conf. Artificial Intelligence*, 1999.

[6] L. Breiman, "Bagging predictors," Univ. California, Dept. Stat., Berkeley, Tech. Rep. 421, 1994.

[7] R.P.W. Duin, PRTools 3.0, "A Matlab Toolbox for Pattern Recognition", Delft University of Technology, 2000.

[8] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, "Classification and regression trees", Wadsworth, California, 1984.

[9] J. R. Quinlan, "Simplifying decision trees", *International Journal of Man-Machine Studies*, vol. 27, 1987, 221--234.

[10] V.N. Vapnik, "Statistical Learning Theory", John Wiley & Sons, New York, 1998.

[11] R. P. W. Duin. "The combining classifier: to train or not to train", *In Proc. of Int. Conf. on Pattern Recognition*, Quebec, Canada, August 2002.

[12] P Pudil, J Novovicova, and J Kittler. "Floating search methods in feature selection". *Pattern Recognition Letters*, 15:1119–1125, 1994.