A Comparison of Soft Fusion Methods Under Different Bagging Scenarios

FUAD ALKOOT, HUSSAIN QASEM

Kuwait

Abstract: We experiment with different fusion methods when bagging k-NN classifiers under various conditions. Experiments with four types of bagging are made at four training set sizes, using two metrics. The aim is to find the conditions for an optimum bagging performance. Additionally we aim to find the best rule under the specified conditions. We compare the performance of the different fusion strategies under each condition. Fusion methods used are Sum, Modified Product (MProduct) [2], Vote and Moderation [1]. Results show that the performance depends on the data used, number of nearest neighbors (k), number of fused classifiers and size of training set. Over all the three rules derived from Product show a close performance, while Vote shows an opposite performance. Among the three rules Moderation either follows Sum or MProduct. Results indicate MProduct outperforms Sum at many instances. At some of these instances Sum did not outperform the single classifier while MProduct did. Moderation is the second best, while Vote is inferior especially at even numbers of k. This is an inherited weakness of Vote, where at even number of close samples ties are randomly resolved. At k=1 all rules yield similar results. There are few instances where Moderation outperforms all. In general MProduct is the best choice.

Keywords: Bagging, KNN, Fusion strategies

1. Introduction

In order to improve the accuracy of classifiers researchers have found that fusing (combining) the decisions of more than one classifier would yield superior results over the best single classifier. In the past decade, the use of classifier fusion to improve classification accuracy has become increasingly popular [1,2,12,13,14,15,17,20,21,22] Bagging [6], proposed by Breiman, is a method of generating multiple versions of a predictor or classifier, via **b**ootstrapping and then using these to get an **agg**regated decision. The multiple versions of classifiers are formed by making bootstrap [14] replicas of the training set, and these are then used to train additional classifiers.

Bagging has been successfully applied to practical cases to improve the performance of unstable classifiers. A sample of such papers includes [10, 8]. Many authors have investigated its merits and compared bagging to boosting or other methods [4,7,9,16,18]. In [6] Breiman argued that bagging would not improve a nearest neighbor (NN) classifier, because it is stable. He confirmed this experimentally by showing that, when bagging NN classifiers, good results were achieved on 3 data sets out of 6. But, when bagging decision trees, better error rate on all 6 data sets was observed. Most research to date has been directed towards applying bagging to unstable classifiers, such as decision trees and neural networks.

We aim to determine methods and conditions that improve k-NN classifiers under bagging. In [3] we attempted to improve the performance of bagged k-NN classifiers, by proposing "modified bagging" which takes advantage of prior knowledge. More investigation is required to find the reasons behind its inconsistent performance. We aim to continue our investigations along several broad lines in an attempt to further improve the performance of bagged k-NN classifiers, beyond what we achieved through modified bagging, and find the methods, conditions and data properties that lead to the successful performance. Experimental investigation of some novel methods that lead to the destabilization of k-NN component classifiers, under varying data types, will be conducted. One such method would involve manipulation of the distance metric..

Our previous work [3] on bagging k-NN under small sample size showed that although regular bagging could not outperform the single classifier, modified bagging which is implemented with some modifications to the bagging method, may outperform the single classifier. The improvements were mostly at medium training set sizes, while at

large training set sizes it had a similar performance to regular bagging and the single classifier. Even at the medium set size, for few data types, we did not always achieve a significant improvement. More investigation is required to find the reasons behind this variable performance. We also aim to further improve the performance of bagged k-NN classifiers, beyond what we achieved through modified bagging, and find the methods, conditions and data properties that lead to the successful performance. We initially used the Euclidean distance metric to find the closest k samples. However, the limitation of the Euclidean metric is that it does ensure that the k nearest neighbors are drawn from a region with a constant a posteriori class probability distribution. This may lead to suboptimal performances. Therefore, we experimented with a more robust distance metric proposed by Short and Fukunaga [19]. The experiments were run for several soft fusion methods to find which leads to the largest improvement of bagging. And to gain more insight on the conditions that lead to a methods superiority. Overall MProduct [2] outperforms the rest most often.

In the next section we will overview the new metric. In section 3 we present the different bagging methods. Section 4 contains the experimental methodology. The results are presented in section 5. The paper is brought to conclusion in section 6.

2. The Optimal distance metric

Short and Fukunaga [15] showed that the variance of finite sample size estimates may be minimized by a proper choice of the metric. The use of such a metric should lead to performance improvement. The optimal distance metric is defined as follows. For a given test sample x_0_ the distance to a point X_e is defined as ;

$$d(x_0, X_e) = \left| V_{x0}^T (x_0 - X_e) \right| \quad (1)$$

Where V is the average gradient direction of the a posteriori probability of class ω_i at point x_0. Its estimate V x_0 is

$$\hat{V}_{x0} = \sum_{i=1}^{m} \left(\frac{n_i}{n}\right)^2 (\hat{M}_i(x_0) - \hat{M}_0(x_0)) \quad (2)$$

where n_i is the number of samples out of n that belong to class i. m is the number of classes. The local sample class i mean $\hat{M}_i(x_0)$ and local sample mixture mean $\hat{M}_0(x_0)$ are given as

$$\hat{M}_i(x_0) = \frac{1}{n_i} \sum_{x_j \in w_i} (X_1 - x_0) \quad (3)$$

$$\hat{M}_0(x_0) = \sum_{j=1}^{n} (X_j - x_0) \quad (4)$$

For k-Nearest Neighbors we find the k closest samples to x_0, then using their labels we decide which class x_0 belongs to by the majority voting. The necessary precondition is that this distance measure is applied only to the samples close to x_0 Hence, initially the closest n samples are found using the Euclidean distance, and then from among these n samples the closest k are found according to equation 1. We automatically select n to be $\frac{3k}{2}$ rounded to the closest integer.

3 Bagging and Modified Bagging Procedures

When a data set is small, the proportions of training patterns from the different classes may be unrepresentative. The probability of drawing a training set with samples from some class completely missing becomes non negligible. When this occurs, bagging may even become counterproductive.

For this reason we set out to investigate the effect of bootstrap control as suggested in [6]. Three modifications of the standard bagging method are considered. We name the standard procedure as method 1 and its modified versions as methods 2-4. The methods which exploit increasing amounts of prior knowledge can be summarized as follows.

Method 1 Given a learning set, each bootstrap set is created by sampling from the learning set randomly with replacement. The cardinality of each boot set is the same as the size of the training set.

Method 2 When bootstrap sets are created from the learning set we check the ratio of the number of samples per class in the bootstrap set. This ratio is compared to the ratio of samples per class in the learning set. If the difference between the compared ratios is larger than a certain class population bias

tolerance threshold we reject the bootstrap set. The bias tolerance threshold used is set at 10%.

Method 3 This method is similar to method 2 except that the bootstrap set ratio is compared to the ratio in the full set. By full set we mean the set containing all samples, learning and test samples. This full set ratio simulates a prior knowledge of the class distribution in the sample space.

Method 4 Here we only require that all classes be represented in the bootstrap set, without enforcing a certain ratio of samples per class. This is done by rejecting any bootstrap set that does not represent all classes.

4 Experimental Methodology:

In this project we experiment with bagging [6] k-NN classifiers. The experiments are simulated using Matlab on several Pentium based personal computers.

Four real data sets from the UCI repository [5] are used. Training and test sets are generated from the original data set. For each of the data sets a single training set is randomly taken from the original sample space, i.e. the full data set. The K-NN classifier built using this original learning set is referred to as the single classifier. The remaining samples are used as a test set. N Bootsets are generated using the training set. The decision of the N boot sets are aggregated to classify the test set. As suggested by Breiman we may start with N = 25 as the number of aggregated k-NN classifiers. The experiments are then repeated for varying N, number of component k-NN classifiers, to have an overview of the effect of the number of classifiers on system performance. These results are referred to as the bagged classifier results. We compare these results to those obtained from the single classifier, and to those obtained from other bagging methods of creating bootstrap sets, as outlined in section 3, for two types of learning sets. In the first case the learning set is created by randomly taking samples from the full data set. This results in a set that may contain some but not all classes. The second type of learning set is referred to as a modified learning set. It is constructed using Method 3 which was mentioned as a technique to create unbiased bootstrap sets. This results in a set that is representative of all the classes, with class population ratios similar to those of the full set. The modified learning set simulates an unbiased sample space.

To obtain unstable k-NN classifiers, we propose using an optimal distance metric proposed by Short and Fukunaga [19], as explained in section 2. We repeat the experiments using the Euclidean metric, in order to make a direct comparison and asses the merits of bagging in conjunction with the optimal metric.

In our previous experiments, for each set size, the number of nearest neighbors, k, is found automatically as the square root of the number of training sets rounded to the closest integer. In this paper 10 values of k are used for each training set size, to see the effect of k on the bagging performance under the new metric.

The above is repeated for the four training set sizes including 10, 20, 40 and 80 samples. The experiments are repeated for different data sets to find which leads to improved performance. The data sets used are obtained from the UCI repository, and the characteristics are shown in table 1. All experiments are repeated 100 times then averaged to achieve statistically reliable results.

5 Results:

We experiments with bagging kNN classifiers using four fusion methods. Our results, on 4 data sets, indicate that the three methods derived from Product yield similar performance, while Vote is different.

Data name	No. of samples	No. of features	No of classes
Wisconsin Breast Cancer, BCW	699	9	2
Ionosphere	351	34	2
Iris	150	4	3
Wine	178	13	3

At the largest training set size bagging with an optimal metric is able to outperform the single classifier contrary to when the Euclidean metric is used. At the largest set size sampling problems do not exist and the new metric yields a good performance. Generally the optimal metric yields superior results over the Euclidean metric. We will focus on comparing the performance of the fusion methods using the optimal metric.

Figures 1 to 12 show results for the four data sets using four bagging methods. Each figure shows the classification rate of a bagging method fused using four different fusion methods, in comparison to the single classifier using the Euclidean and optimal metrics. For each value of k we can compare the performances at different number of fused classifiers, ranging from 3 to 25 bootstrap sets. The values of the x-axis represent the number of fused bootstrap sets, when multiplied by 2 and added to 1. Therefore, 12 represents 25 bootstrap sets, while 5 represents 11.

Results for the **BCW** data, using regular bagging method 1, show that at the smallest set size MProduct is the best rule or equal to the best. At k=7 it is the only rule that can outperform the single classifier. This is true for the four set sizes. While Sum using the optimal metric bagging is almost never superior. It only outperforms MProduct insignificantly at the largest set size. At k=7, for all set sizes, MProduct shows the best performance.

Using bagging method 2, at the smallest set size, Vote is the best for k=5 and 7. We note that Vote in conjunction with the optimal metric is the best rule while Vote using Euclidean metric shows the weakest performance. At set size 2, Vote is the best at k=7. As the set size increases MProduct improves and all fusion methods become close. At the largest set size MProduct is either best or close to it.
We note that Vote deteriorates at even values of k. however at odd k, we found that the optimal metric improves Vote to reach the highest performance.

Using bagging method 3, Sum and Moderation are close to the leading MProduct rule, while Vote is worst than all. At the fourth bagging method, mostly MPrduct is best or equal to Sum and Moderation. We found that MProduct is consistently best at k=7 in addition to other values of k at different set sizes.

Using bagging method 4 we find that MProduct is best at k=3, 4 5, 7, and 8. As the set size increases the values of k where MProduct is best increases to k=7, 8, 9 and 10. We find that it is the only rule that outperforms the single classifier at k=7 for the largest set size when a modified training set is used. For the rest of the values of k and set sizes the rules show a close performance.

For the **Iris** at bagging method 1 MProduct is best at set sizes 1 and 2 when k = 4, 5 or 8. Interestingly at set size 1 only MProduct outperforms the single classifier. As the set size increases Moderation and Sum become best when k=4, while MProduct is still best at k=8. When bagging method 2 is used fusion methods are mostly close especially at large set sizes. For bagging method 3 we get mixed performances and it is uncertain which rule is best. Using bagging method 4 at set size 1, MProduct is the only rule that outperforms the single classifier. It is by far the best. As the set size increases MProduct remains to be the best but the difference in performances is reduced until all three rules become equal at set size 4. Vote is mostly worse than other rules.

For the **Ionosphere** data, we found that MProduct outperforms all at all training set sizes, except the smallest, and for all bagging methods when k=3 and 4. Also, for bagging methods 1 and 2 MProduct is best at k=7 and 8. Otherwise results are mixed.

For the **Wine** data, we found that for the smallest set size, MProduct is best at k=3 to 9. This advantage decreases as the set size increases. When modified learning set is used the rules perform equally. An exception is at the smallest set size where MProduct outperforms the rest.

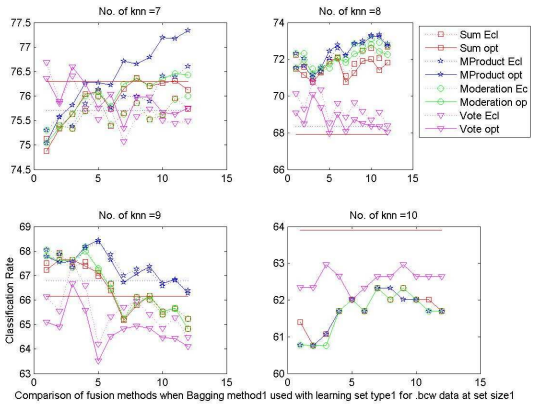

Figure 1 BCW Classification rate for the fusion rules using bagging method 1 at set size 1.

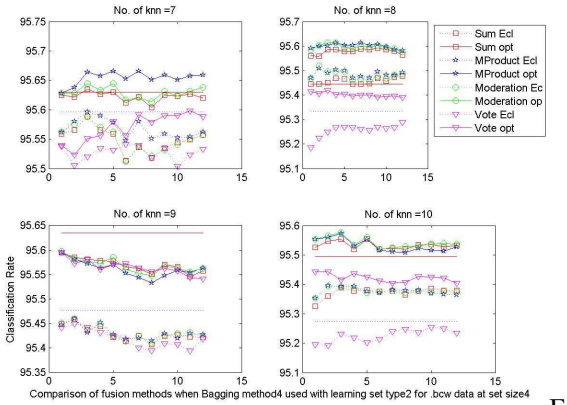

Figure 2 BCW Classification rate for the fusion rules using bagging method 4 at set size 4

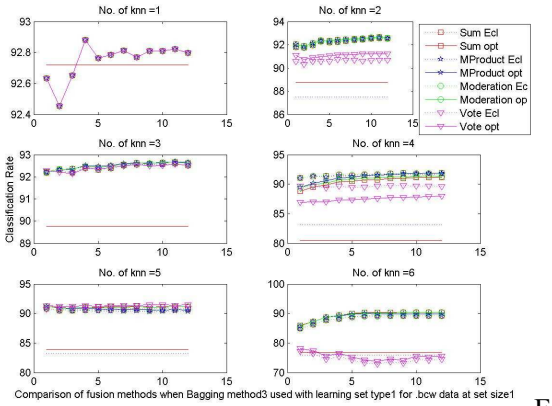

Figure 3 BCW Classification rate for the fusion rules using bagging method 3 at set size 1

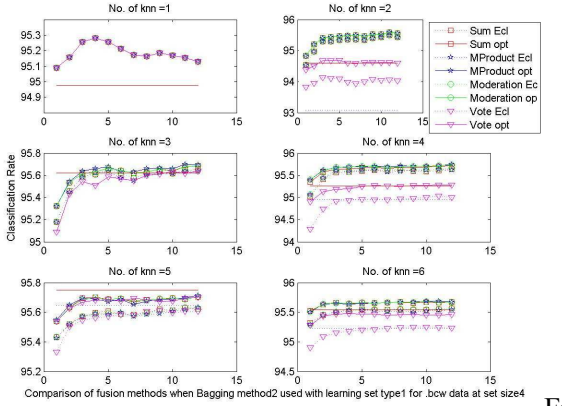

Figure 4 BCW Classification rate for the fusion rules using bagging method 2 at set size 4

6 Discussions:

Results generally do not show a single rule outperforming the rest. However, we note that the three rules derived from the product rule show similar results. Vote is sometimes close to them but mostly falls behind. There are some instances where MProduct outperforms the rest. Occasionally it is the only rule that outperforms the single classifier. A close study of these instances will yield useful information on how to improve the bagging performance.

Varying Number of combined bootstraps:

We experimented with combining an odd number of bootstrap sets ranging from 3 to 25. To verify the effect of the number of combined classifiers on the rules performance under the compared metrics. We find that increasing the number of fused classifiers improves the rules. Mostly we achieve the best results early in the curve, at 5 or 9. However to consistently obtain best results more than 20 classifiers must be combined.

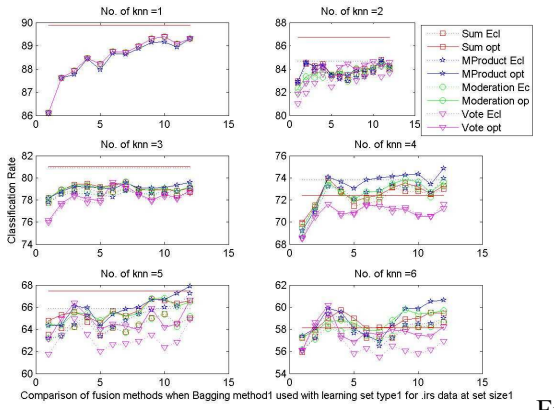

Figure 5 Iris Classification rate for the fusion rules using bagging method 1 at set size 1.

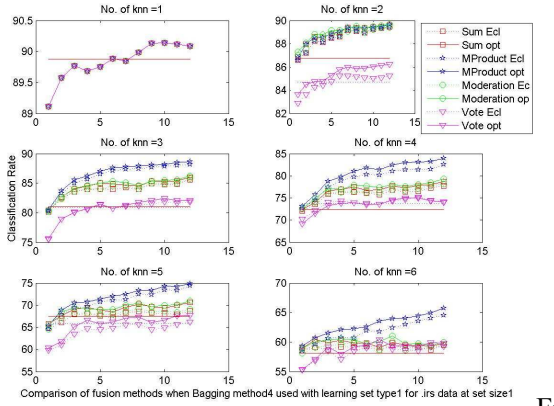

Figure 6 Iris Classification rate for the fusion rules using bagging method 4 at set size 1

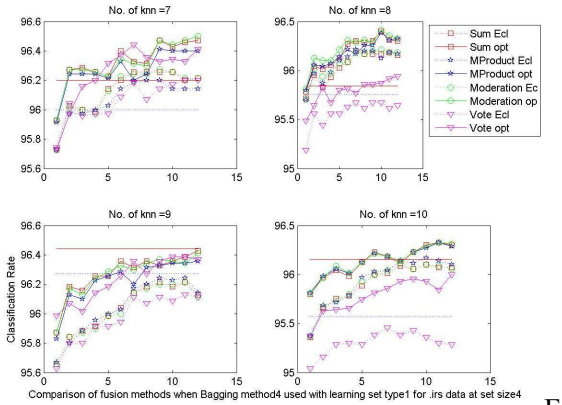

Figure 7 Iris Classification rate for the fusion rules using bagging method 4 at set size 4.

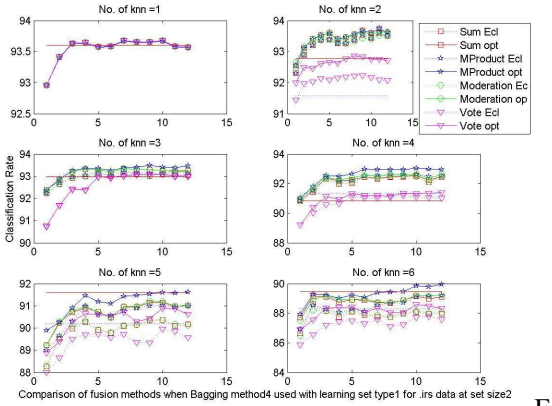

Figure 8 Iris Classification rate for the fusion rules using bagging method 4 at set size 2.

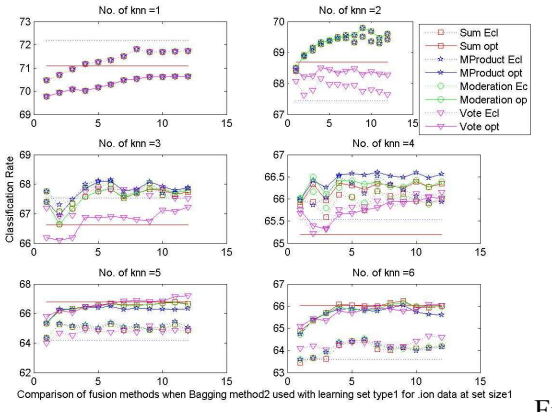

Figure 9 Ionosphere Classification rate for the fusion rules using bagging method 2 at set size 1.

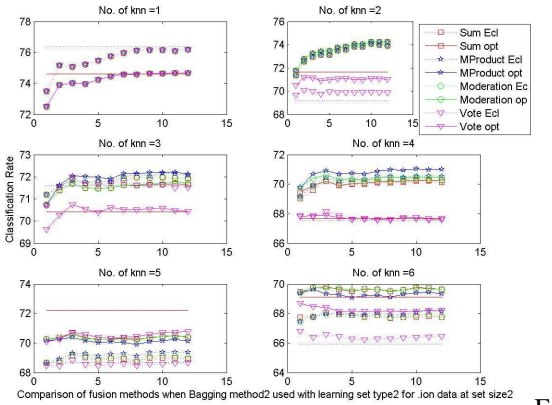

Figure 10 Ionosphere Classification rate for the fusion rules using bagging method 2 at set size 2.

Varying number of nearest neighbors, k:

We experimented with a varying number of nearest neighbors, k, ranging between 1 and 10. We found that at very small value of k = 1, the rules show similar results. The relative performances become different as k increases. The best values of k depend on the set size. Large values of k at the smallest set size would not yield good results if regular learning set is used. This is due to the fact that when a small number of classifiers exist the large number of nearest neighbors would include samples from the wrong class, which could be close to the boundary and lead to misclassification.

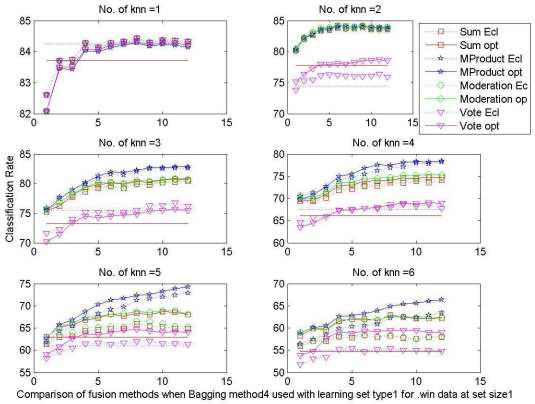

Figure 11 Wine Classification rate for the fusion rules using bagging method 4 at set size 1.

7 Conclusions:

We experiment with four fusion methods under varying bagging kNN conditions. Experiments with four bagging methods using two training set types were made at four training set sizes, using two metrics. Fusion methods used are Sum, Modified Product (MProduct) [2], Vote and Moderation [1].

Results show that the performance depends on the data used, number of nearest neighbors (k), number of fused classifiers and size of training set. We found that the relative performance of the rules was not affected by the change in the metric. Mostly bagging using the optimal metric outperformed the Euclidean metric. The same was true for the single classifier. To find the best rule we compare the rules under varying conditions. At k=1 all rules yield similar results. At other values of k the three rules derived from Product show similar results while Vote shows an opposite performance. Between the three rules MProduct outperforms the rest most often. At few instances bagging did not outperform the single classifier except when MProduct was used. Moderation either follows Sum or MProduct. While at very few instances outperforms all. On few instances Vote outperforms the rest otherwise it mostly falls behind. It was consistently inferior at even numbers of k. This is because at even number of nearest neighbors ties are randomly resolved. In general MProduct is the best choice. Further investigation of the cases where MProduct outperforms all will help understand when it is an optimum choice.

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. 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 2001. SPIE Volume 4385, pp36.

4. E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms : Bagging, boosting and variants. Machine Learning, pages 1-38, 1998.

5. C.L. Blake, E. Keogh, C.J. Merz. UCI repository of machine learning databases, http://www.ics.uci.edu/mlearn/MLRepository.html, Department of Information and Computer Science, University of California, Irvine, CA, 1998.

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

7. L. Breiman. Bias, variance and arcing classifiers. Technical Report 460, Statistics Depart˜ent, University of California at Berkeley, 1996 .

8. P. deChazal and B. Celler. Improving ecg diagnostic classification by combining multiple neural networks. In Computers in Cardiology, volume 24, pages 473{476. IEEE, 1997

9. T. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine Learning, pages 1-22, 1998

10.B. Draper and K. Baek. Bagging in computer vision. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 144-149. IEEE Comp Soc, Los Alamos, CA, USA, 1998 .

11.B. Efron and R. Tibshirani. An Introduction to the Bootstrap. Chapman and Hall, 1993.

12.T.K. Ho, J.J. Hull, and S.N. Srihari. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):66–75, 1994.

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

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

15.J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis and Applications*, 1:18–27, 1998.

16.J. Quinlan. Bagging, boosting and c4.5. In Proceedings of the 13th National Conference on Artificial Intelligence, volume 1, pages 725{730, Portland, OR, USA,, 1996. AAAI, Menlo Park, CA, USA

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

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

19.Robert Short and Keinosuke Fukunaga. A new nearest neighbor distance measure. In IEEE conference, pages 81 to 86, 1980 .

20.D.H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–260, 1992.

21.K Woods, W P Kegelmeyer, and K Bowyer. Combination of multiple experts using local accuracy estimates. *IEEE Trans PAMI*, 19:405–410, 1997.

22.L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. SMC*, 22(3):418–435,1992.