Appears in the Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), Madison, WI.

Boosting Classifiers Regionally

Richard Maclin
Computer Science Department
University of Minnesota-Duluth
Duluth, MN 55812
email: rmaclin@d.umn.edu

Abstract

This paper presents a new algorithm for Boosting the performance of an ensemble of classifiers. In Boosting, a series of classifiers is used to predict the class of data where later members of the series concentrate on training data that is incorrectly predicted by earlier members. To make a prediction about a new pattern, each classifier predicts the class of the pattern and these predictions are then combined. In standard Boosting, the predictions are combined by weighting the predictions by a term related to the accuracy of the classifier on the training data. This approach ignores the fact that later classifiers focus on small subsets of the patterns and thus may only be good at classifying similar patterns. In RegionBoost, this problem is addressed by weighting each classifier's predictions by a factor measuring how well that classifier performs on similar patterns. In this paper we examine several methods for determining how well a classifier performs on similar patterns. Empirical tests indicate RegionBoost produces gains in performance for some data sets and has little effect on others.

Introduction

Boosting (Breiman 1996b; Drucker & Cortes 1996; Freund & Schapire 1996) is a technique for creating *ensemble* classifiers, classifiers that combine the predictions of multiple component classifiers. It has proven extremely effective over a number of different domains (Breiman 1996b; Freund & Schapire 1996; Maclin & Opitz 1997; Quinlan 1996). This paper presents RegionBoost, a modification of existing Boosting techniques. The focus of RegionBoost is to do a better job of combining the predictions of the classifiers in the ensemble by considering the performance of each of these classifiers in different regions of problem space. Boosting techniques generally either average the predictions of the classifiers or produce a weighted average of the classifiers where the weight is a single value for each classifier. The advantage of RegionBoost is that a classifier that performs well in only one portion of problem space will be weighted more highly on data points from that portion of problem space (and low on other points). Quinlan (1996) and Woods et al. (1997) present approaches similar to RegionBoost, but Quinlan's technique is specific to decision trees and Woods et al.'s technique selects a single classifier rather than combining the predictions of the classifiers.

To test the effect of RegionBoost, I present experiments using ensembles of neural networks on 19 data sets from the UCI repository. These experiments show that RegionBoost often significantly outperforms standard Boosting, though it has little effect on performance for some problems. These results are especially interesting in that Boosting often produces the most accurate classifiers for many domains.

Background: Ensembles

An ensemble classifier consists of a set of individual classifiers (components) and a mechanism for combining the predictions of the components (see Figure 1). To classify a point using an ensemble, the input vector is passed to the component classifiers, each of which predicts the class of the point. These predictions are then merged by the combiner mechanism into a single prediction for the classifier. For example, the combiner could take a majority vote of the components. Research on ensembles generally focuses on: (1) which component classifiers to combine; and (2) how to com-

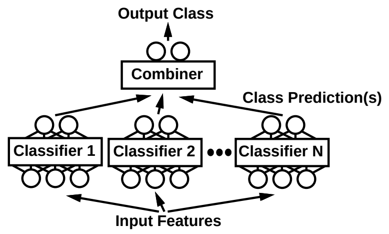

Figure 1: A classifier ensemble made up of N component classifiers and a combiner mechanism.

bine the predictions of the classifiers.

Empirical evidence suggests that an ensemble classifier almost always outperforms an average classifier from the ensemble (Maclin & Opitz 1997; Quinlan 1996). Often the ensemble outperforms *all* of its component classifiers. Theoretical work (Hansen & Salamon 1990; Krogh & Vedelsby 1995) on ensemble methods indicates that ensembles are effective when the components are accurate and differ in their predictions. Krogh and Vedelsby (1995) demonstrated that the performance of an ensemble is mathematically based on the average error rate of its components minus a term that measures the differences between the components.

Research on selecting classifiers to use as components generally focuses (if only indirectly) on the selection of component classifiers that are accurate and differ in their predictions. Examples of mechanisms that have been used to create component classifiers include using different training parameters with a single learning method (Alpaydin 1993; Maclin & Shavlik 1995), using different subsets of the training data with a single learning method (Breiman 1996a; Freund & Schapire 1996), using different learning methods (Zhang, Mesirov, & Waltz 1992), and explicitly searching for a set of classifiers that is both accurate and diverse (Opitz & Shavlik 1996). Boosting is a method that creates component classifiers by using different subsets of the training data.

Numerous methods have been suggested for combining the predictions of classifiers. Sample combination functions include voting (Hansen & Salamon 1990), simple averages (Lincoln & Skrzypek 1989), weighted averages (Freund & Schapire 1996; Rogova 1994), using a voting sequence of combiners (Asker & Maclin 1997), and *learning* a combiner function (Wolpert 1992). Wolpert's (1992) Stacking mechanism is a powerful general mechanism for producing an effective combining function by training a learner to predict the corrections needed for each of the individual components, though it is limited in that for it to be effective it needs a set of held-out training data that was not used in creating the component classifiers. But Clemen (1989) suggests that simple mechanisms are often as effective as any complex combining method.

Background: Boosting

Boosting (Freund & Schapire 1996) encompasses a family of methods. The focus of these methods is to produce a *series* of classifiers. The training set used for each member of the series is chosen based on the performance of the earlier classifier(s) in the series. In Boosting, examples that are incorrectly predicted by previous classifiers in the series are chosen more often than examples that were correctly predicted. Thus Boosting attempts to produce new classifiers that are better able to predict examples for which the current ensemble's performance is poor.

In this work we will be building on a powerful form of Boosting called Ada-Boosting (Freund & Schapire 1996). In Ada-Boosting, a training set of size N is selected for classifier $K+1$ by probabilistically selecting (with replacement) N examples from the original N training examples (since there is replacement and the probabilities differ, some examples may be selected more than once and some not at all). For classifier $K+1$, the probability depends on how often that example was misclassified by the previous K classifiers. Initially the probability of picking each example is set to $1/N$. After a classifier is added to the ensemble, the probabilities of selecting examples are adjusted by a factor based on ϵ_k, the sum of the probabilities for those examples that are incorrectly classified by classifier K. The probability of selecting each of the misclassified examples is multiplied by the value $(1-\epsilon_k)/\epsilon_k$. The probabilities of selecting each of the examples are then renormalized so that they sum to 1. This process has the effect of increasing the probability of misclassified examples and reducing (through the normalization) the probability of the correctly classified examples. In this work we use Breiman's (1996b) variation where the probabilities are all reset to $1/N$ if ϵ_k equals 0 or becomes greater than 0.5. In the latter case, multiplying by $(1-\epsilon_k)/\epsilon_k$ would *decrease* the probability of misclassified examples, while in the former case the value $(1-\epsilon_k)/\epsilon_k$ is undefined.

A critical factor in Ada-Boosting is that the predictions of the component classifiers are not simply averaged. In later classifiers the training set may focus heavily on certain examples and ignore others, producing a classifier that is very good at classifying a small subset of the points but is not effective at classifying all points. For example, Figure 2 shows the average *training* error of the different classifiers in the Boosting ensembles used in the experiments later in this paper. Even with Breiman's resetting variation, the accuracy of later classifiers in the ensemble drops off quickly. To address this problem, Ada-Boosting weights the pre-

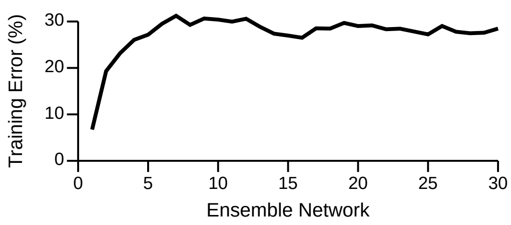

Figure 2: Composite *training* error from the classifiers in the Boosting experiments from the Results section.

dictions of classifiers by $log((1-\epsilon_k)/\epsilon_k)$. Classifiers with small values of ϵ_k are weighted higher than classifiers with large values. Although this strategy is effective, it does not make full use of the data available, a limitation addressed by RegionBoost.

RegionBoost

Ada-Boosting (Freund & Schapire 1996) addresses the problem of combining classifiers by weighting the classifiers with respect to their accuracy. But by weighting each of the classifiers by a single value, Ada-Boosting loses some of the value of the Boosting approach. In Boosting, later classifiers concentrate on correctly classifying examples that were not correctly classified by previous classifiers. Thus it is reasonable to assume that the later classifiers will be better at some types of examples and worse at other types. To address this limitation using only a single weight *undervalues* a later classifier's ability to classify examples similar to the ones it was trained with and *overvalues* its ability to classify other examples. A better approach is to weight the predictions of each classifier by how well it predicted examples similar to the one being examined.

Intuitively, the idea behind RegionBoost's approach to weighting is to use the training data to estimate the accuracy of the classifier. Although this estimate is likely to suffer from overfitting, it does have an advantage in that Boosting leave some of the data out (and in the case of later classifiers, possibly significant portions may not be included in the training set for that classifier). Rather than using a single measure of the accuracy on the training data, in RegionBoost we take advantage of the idea that some classifiers perform well only for certain regions of problem space by estimating the likely accuracy of the classifier for each new point. In this work we will examine two approaches to estimating the accuracy of the classifier.

One approach to estimating accuracy will use a simple k-Nearest-Neighbor (Cover & Hart 1967) approach to find the k points in the training set nearest to the new point we are trying to classify and assessing how well each classifier performed for each of those points. The weighting for that classifier's prediction is the sum of how well the classifier predicted each of these k training data points divided by k. In the limit, if we choose $k = N$, this method approximates the standard Ada-Boosting approach to weighting predictions. A second approach will be to train a second classifier for each classifier in the ensemble, in this work a separate neural network, to predict the accuracy for new points. For both approaches we will need to estimate the accuracy of a classifier for each training data point.

One estimate of the accuracy for a training data point (the *Discrete* method) will simply assign the value of one if the pattern is correctly predicted and zero otherwise. A second method (*Continuous*) will estimate accuracy based on the output produced by the classifier. In this approach the accuracy estimate will be based on the average distance between the actual and expected output for a pattern. For example, in a three-class problem where the expected prediction is {0,1,0} (i.e., in class 2) and the actual prediction is {0.1,0.8,0.1}, the accuracy would be 0.867 (1 - 0.133) where 0.133 is the average of the distances between 0 and 0.1, 1 and 0.8, and 0 and 0.1 divided by 3.

Table 1: Summary of the data sets used in this paper. Shown are the number of examples and output classes, plus the number of inputs, outputs, hidden units and training epochs used for each network.

Data Set	Case	Out	In	Hid	Epch
breast-cancer-w	699	2	9	5	20
credit-a	690	2	47	10	35
credit-g	1000	2	63	10	30
diabetes	768	2	8	5	30
glass	214	6	9	10	80
heart-cleveland	303	2	13	5	40
hepatitis	155	2	32	10	60
house-votes-84	435	2	16	5	40
hypo	3772	5	55	15	40
ionosphere	351	2	34	10	40
iris	159	3	4	5	80
kr-vs-kp	3196	2	74	15	20
labor	57	2	29	10	80
promoters-936	936	2	228	20	30
segmentation	2310	7	19	15	20
sick	3772	2	55	10	40
sonar	208	2	60	10	60
soybean	683	19	134	25	40
vehicle	846	4	18	10	40

Results

This section presents experiments to evaluate the effect of the RegionBoost approach on classification accuracy. The tests use 19[1] of the 23 data sets from the UCI data set repository (Murphy & Aha 1994) used in a previous study of Boosting (Maclin & Opitz 1997). Table 1 gives details for these data sets. For each data set we report error rates for Ada-Boosting and RegionBoost.

Methodology

All results are averaged over ten standard 10-fold cross validation experiments. For each 10-fold cross validation, the data set is first partitioned into 10 equal-sized sets, and each set is in turn used as the test set while the classifier trains on the other nine sets. For each fold an ensemble of 30 networks are created

[1]The 4 data sets excluded require more computational resources than were available.

Table 2: Results of experiments shown to test the effects of RegionBoost on the data sets listed in Table 1. The second, eleventh and thirteenth columns (Ada, Perceptron-Ada and NearN) show error rates using these methods. The results in the other columns represent percentage point reductions in error rate (or increases for negative numbers). For example, in the third column next to breast-cancer-w, the number is 0.3 which indicates that the error rate using this method was 3.7%. See the text for descriptions of the experiments producing these numbers.

		RegionBoost								Perceptron		NearN
		NearN:Discrete			NearN:Continuous			Net Accuracy				
Data Set	Ada	k-7	k-11	k-15	k-7	k-11	k-15	Disc	Cont	Ada	RB Disc k-15	k-15
breast-cancer-w	4.0	*0.3*	*0.3*	*0.3*	*0.2*	*0.3*	*0.2*	*0.7*	*0.4*	4.0	0.1	3.8
credit-a	15.9	*1.1*	*0.8*	*0.7*	*0.8*	*0.9*	*0.7*	*1.9*	*1.3*	14.9	0.4	16.2
credit-g	25.5	0.0	*0.3*	*0.3*	0.2	0.3	*0.3*	*0.4*	*0.8*	25.6	*0.6*	27.0
diabetes	22.9	*-0.3*	-0.2	-0.1	0.0	0.4	0.2	0.0	0.3	23.0	*-0.6*	25.1
glass	32.4	*1.2*	*1.0*	*1.3*	*1.0*	*1.0*	*1.3*	*1.0*	*1.0*	36.7	*1.8*	38.2
heart-cleveland	19.8	0.4	*0.6*	*0.7*	0.1	0.1	*0.2*	0.3	0.0	19.4	0.0	20.6
hepatitis	18.7	*0.8*	*0.8*	*0.8*	*0.8*	*1.3*	*0.7*	*0.5*	*0.6*	18.3	0.4	19.4
house-votes-84	5.1	0.2	*0.2*	*0.2*	0.2	*0.3*	*0.3*	*0.2*	0.2	5.6	0.0	6.5
hypo	6.2	0.1	0.0	0.0	0.1	0.0	0.0	0.0	0.0	6.2	0.1	7.0
ionosphere	7.9	-0.4	-0.4	-0.3	0.0	0.0	0.0	*0.8*	*0.8*	12.8	*1.2*	16.3
iris	3.8	-0.1	-0.1	0.0	0.0	0.1	0.0	-0.1	-0.1	3.5	0.2	3.4
kr-vs-kp	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.9	*0.2*	7.6
labor	4.2	0.2	0.2	0.2	0.3	0.3	0.2	*1.7*	*1.7*	3.3	0.1	24.2
promoters-936	4.4	0.0	0.0	0.1	0.0	0.1	0.0	-0.1	-0.1	5.5	0.1	8.5
segmentation	3.5	*0.2*	*0.2*	*0.2*	0.1	*0.2*	*0.2*	0.0	0.0	7.1	*2.8*	5.6
sick	4.3	0.2	0.1	0.1	*0.2*	*0.2*	*0.4*	*0.2*	0.2	4.5	0.0	4.3
sonar	12.7	0.1	0.2	0.1	*0.4*	*0.5*	*0.4*	0.2	*0.4*	18.3	*0.5*	31.2
soybean	7.0	0.1	0.0	0.0	0.0	0.0	0.1	0.1	-0.1	6.9	-0.1	12.6
vehicle	19.7	-0.2	-0.1	-0.1	*0.6*	*0.6*	*0.7*	*0.7*	*0.7*	22.6	*1.1*	31.0

(for a total of 300 networks for each 10-fold cross validation). Parameter settings for the neural networks include a learning rate of 0.15, a momentum term of 0.9, and weights are initialized randomly to be between -0.5 and 0.5. The parameters used in training were the same as used by Maclin and Opitz (1997).

Experimental Results

Table 2 incorporates the results for several of the experiments discussed in this section. The second column of this table shows the baseline Ada-Boosting results produced for each of these data sets. The next three columns show the results for the RegionBoost approach with the k-Nearest-Neighbor method (for k=7,11,15) and the Discrete method for estimating training data accuracy. The following three columns show similar results using the Continuous method for estimating accuracy. These results show reductions (or increases) in error rate for each of the data sets by percentage point (e.g., 1.1 in the third column for credit-a indicates the error rate goes from 15.9% to 14.8%). Statistically significant changes in error are shown in *italics.*

Several interesting results can be observed from these experiments. First, RegionBoost significantly decreases the error for a number of the data sets (six to ten of the 19), and it produces little or no change for the other data sets. RegionBoost only significantly increases the error for one case. There also seems to be a relation between the performance of the different methods. Several data sets always see reductions in error rate. One difference between the two methods for weighting the confidence of predictions is that the Continuous method produces significant gains for two data sets, sonar and vehicle, for which the Discrete method does not perform well.

In a second set of experiments we tested the idea of using RegionBoost where the estimated accuracy for a new point is predicted by a neural network. A network is trained after each component is created and is used to predict the estimated accuracy of the training data points for that component. This network is trained using the training data in which the expected output signal is either the Discrete (correct or incorrect) value associated with each training point (see ninth column) or the Continuous estimate of the accuracy of the training point (tenth column). The number of hidden units in these networks is the same as the number of hidden units in the component network. The results of these experiments are similar to those obtained using the nearest neighbor methods, and produce significant gains for two other data sets (labor and ionosphere).

Together, these experiments indicate that the overall RegionBoost approach can produce significant gains for many (though not all) data sets. One question which might be raised is how RegionBoost affects Boosting's performance on different learning approaches. To answer this question, we apply Boosting and RegionBoost to a much simpler form of neu-

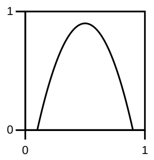

Figure 3: The "arch" problem. Two dimensional data points are randomly generated with points above the line being labeled as positive examples and points below negative examples.

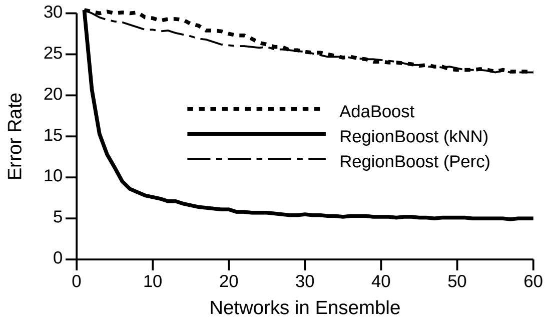

Figure 4: Error rates for Ada-Boosting and two RegionBoost approaches to the "arch" problem. One RegionBoost approach uses a Nearest Neighbor measure of accuracy; the other a perceptron.

ral networks, perceptrons. The two columns labeled Perceptron show Ada-Boosting and RegionBoost using the Discrete, Nearest Neighbor approach results for these data sets. For many of these data sets, RegionBoost again produces significant gains in performance, often on the same data sets where it previously worked well. This result holds even though the performance of the perceptron approach is often significantly different from the performance of the other neural networks.

An interesting question now arises, namely how is RegionBoost affected by the different methods used for estimating the classifiers' accuracy for a new data point? Since the two different approaches (nearest neighbor and network) often work well on the same data sets this might suggest that RegionBoost's effects are entirely dependent on aspects of the data set. To test this notion we looked at a standard Boosting problem, the "arch" problem shown in Figure 3. This problem is interesting since it has been shown that a Boosting approach can combine a sequence of classifiers that use only linear decision surfaces to solve this problem. But approaches that are not restricted to linear decision surfaces (such as a Nearest Neighbor approach) can perform extremely well on this problem (a simple Nearest Neighbor approach with k=15 has a 3.1% error rate on this problem). Figure 4 shows results for a standard Boosting approach on perceptron component classifiers, a RegionBoost approach using the Discrete, Nearest Neighbor (k=15) accuracy estimate and a RegionBoost approach using a perceptron (linear) accuracy estimate. These results indicate that for this problem, RegionBoost achieves much better performance when using Nearest Neighbor estimation. This suggests the effect of RegionBoost is not entirely dependent on the data set but also on the method used for estimating the accuracy of the classifier.

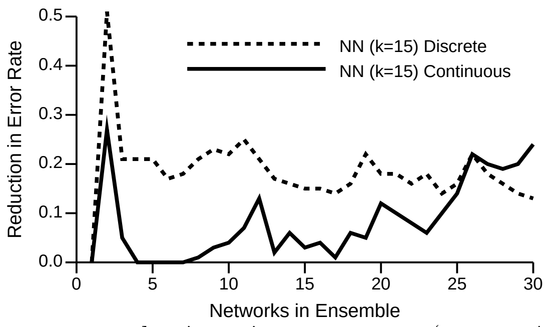

Figure 5: Reductions in error rate (versus Ada-Boosting) plotted as a function of number of classifiers combined using two Nearest Neighbor (k=15) RegionBoost approaches, one using Discrete and the other using a Continuous measure of error.

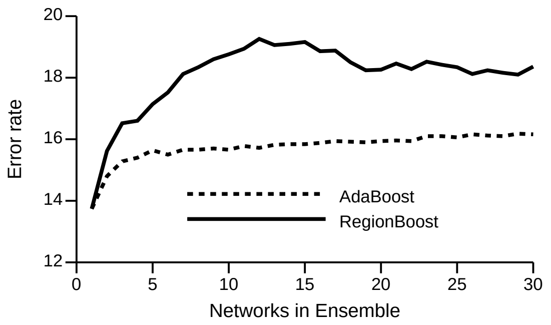

Figure 6: Error rate for a problem with "one-sided noise" using Ada-Boosting and RegionBoost.

Which approach for assessing the accuracy of an example is more appropriate? Figure 5 plots the reduction in error over all of the data sets for two approaches: the Nearest Neighbor (k=15) method of measuring accuracy with (1) Discrete and (2) Continuous measures. These results indicate that the effect of the Discrete method is strongest for the first few classifiers but tails off, whereas the Continuous method shows an early gain which quickly disappears and only after a large number of classifiers again does significantly better.

Finally, we consider the question of whether RegionBoost is subject to overfitting as described in Opitz and Maclin (1998). They constructed artificial data sets with "one-sided noise." These data sets have two relevant features and four irrelevant features where the concept consisted of a simple hyperplane based on the two relevant features. The data consists of a set of randomly generated points from each side of the hyperplane. Then a certain percentage of the points on one side of the hyperplane are mislabeled (creating "one-sided" noise). Results from experiments with this type of data set indicate Boosting often produces significant increases in error. Figure 6 shows the performance of Ada-Boosting and RegionBoost on one of these domains. These results indicate that RegionBoost is even more susceptible to overfitting than Ada-Boosting (this is not surprising given that RegionBoost depends on the training data to estimate classifier accuracy).

Future Work

The experiments presented indicate a significant effect for RegionBoost in many cases. Further experiments will evaluate the effect of RegionBoost on other learning methods (e.g., decision trees). It would also be useful to examine the value of using RegionBoost to combine different types of classifiers (e.g., decision trees and neural networks). Comparisons will also be made between RegionBoost and Boosting with Stacking (Wolpert 1992) combining.

One major advantage of RegionBoost is its ability to produce effective ensembles using components that are only effective for small regions of problem space. This advantage makes it possible to look at Boosting approaches that focus much more strongly on incorrectly labeled points. For examples, Breiman's (1996b) method includes a parameter that can be used to increase the likelihood that incorrectly labeled patterns are focused on in later classifiers. With RegionBoost it is possible to use much larger values of this parameter since the resulting classifiers (which may focus on small numbers of patterns) will be highly weighted only for problems on which they appear to be effective.

Conclusions

This paper presents RegionBoost, a new algorithm for producing a Boosting ensemble of classifiers. RegionBoost differs from standard Boosting in how it combines the predictions of the classifiers making up the ensemble. Other Boosting methods usually weight the predictions of the components by a single value, whereas RegionBoost attempts to estimate the accuracy of each of the classifiers based on the accuracy of the classifiers for similar data points in the training set. Two approaches for estimating the accuracy of each classifier are explored, a k-Nearest-Neighbor and a neural network method. Empirical results on several data sets indicate both RegionBoost approaches often produce statistically significant gains in accuracy. Detailed experiments investigating various aspects of RegionBoost seem to indicate that the effects of RegionBoost are based partially on the data set being examined and partially on the method used for estimating classifier accuracy. Experiments also show that while RegionBoost is often effective, it can suffer from overfitting problems just like Boosting.

References

Alpaydin, E. 1993. Multiple networks for function learning. In *Proc IEEE Int Conf Neur Nets*, 27–32.

Asker, L., and Maclin, R. 1997. Ensembles as a sequence of classifiers. In *IJCAI-97*.

Breiman, L. 1996a. Bagging predictors. *Machine Learning* 24(2):123–140.

Breiman, L. 1996b. Bias, variance, and arcing classifiers. Technical Report TR 460, UC-Berkeley, Berkeley, CA.

Clemen, R. 1989. Combining forecasts: A review and annotated bibliography. *Journal of Forecasting* 5:559–583.

Cover, T., and Hart, P. 1967. Nearest neighbor pattern classification. *IEEE Trans on Info The* 13:21–27.

Drucker, H., and Cortes, C. 1996. Boosting decision trees. In Touretsky, D.; Mozer, M.; and Hasselmo, M., eds., *NIPS-8*, volume 8, 479–485. Cambridge, MA: MIT Press.

Freund, Y., and Schapire, R. 1996. Experiments with a new boosting algorithm. In *ICML-96*, 148–156.

Hansen, L., and Salamon, P. 1990. Neural network ensembles. *IEEE Trans on Pat Ana and Mach Int* 12:993–1001.

Krogh, A., and Vedelsby, J. 1995. Neural network ensembles, cross validation, and active learning. In Tesauro, G.; Touretzky, D.; and Leen, T., eds., *NIPS-7*, 231–238. Cambridge, MA: MIT Press.

Lincoln, W., and Skrzypek, J. 1989. Synergy of clustering multiple back propagation networks. In Touretzky, D., ed., *NIPS-2*, 650–659. San Mateo, CA: Morg. Kauf.

Maclin, R., and Opitz, D. 1997. An empirical evaluation of bagging and boosting. In *AAAI-97*.

Maclin, R., and Shavlik, J. 1995. Combining the predictions of multiple classifiers: Using competitive learning to initialize neural networks. In *IJCAI-95*, 524–530.

Murphy, P., and Aha, D. 1994. UCI repository of machine learning databases (machine-readable data repository). UCI, Dept of Info and Comp Sci.

Opitz, D., and Maclin, R. 1998. Popular ensemble methods: An empirical study. *Machine Learning (submitted)*. (Also appears as UMD CS TR 98-1).

Opitz, D., and Shavlik, J. 1996. Generating accurate and diverse members of a neural-network ensemble. In Touretsky, D.; Mozer, M.; and Hasselmo, M., eds., *NIPS-8*, 535–541. Cambridge, MA: MIT Press.

Quinlan, J. R. 1996. Bagging, boosting, and c4.5. In *AAAI-96*, 725–730.

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

Wolpert, D. 1992. Stacked generalization. *Neural Networks* 5:241–259.

Woods, K.; Kegelmeyer, W.; and Bowyer, K. 1997. Combination of multiple classifiers using local accuracy estimates. *IEEE Trans on Pat Ana Mach Int* 19:405–410.

Zhang, X.; Mesirov, J.; and Waltz, D. 1992. Hybrid system for protein secondary structure prediction. *Journal of Molecular Biology* 225:1049–1063.