Appears in *The Fourteenth National Conference on Artificial Intelligence*,
Providence, Rhode Island, 1997, AAAI Press.

An Empirical Evaluation of Bagging and Boosting

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

David Opitz
Department of Computer Science
University of Montana
Missoula, MT 59812
email: opitz@cs.umt.edu

Abstract

An ensemble consists of a set of independently trained classifiers (such as neural networks or decision trees) whose predictions are combined when classifying novel instances. Previous research has shown that an ensemble as a whole is often more accurate than any of the single classifiers in the ensemble. Bagging (Breiman 1996a) and Boosting (Freund & Schapire 1996) are two relatively new but popular methods for producing ensembles. In this paper we evaluate these methods using both neural networks and decision trees as our classification algorithms. Our results clearly show two important facts. The first is that even though Bagging almost always produces a better classifier than any of its individual component classifiers and is relatively impervious to overfitting, it does not generalize any better than a baseline neural-network ensemble method. The second is that Boosting is a powerful technique that can usually produce better ensembles than Bagging; however, it is more susceptible to noise and can quickly overfit a data set.

Introduction

Many researchers have investigated the technique of combining the predictions of multiple classifiers to produce a single classifier (Breiman 1996c; Clemen 1989; Perrone 1993; Wolpert 1992). The resulting classifier (hereafter referred to as an *ensemble*) is generally more accurate than any of the individual classifiers making up the ensemble. Two popular methods for creating ensembles are Bagging (Breiman 1996a) and Boosting (or Arcing) (Freund & Schapire 1996). These methods rely on "resampling" techniques to obtain different training sets for each of the classifiers. While previous work has demonstrated that these two methods are very effective for decision trees (Drucker & Cortes 1996; Breiman 1996a; 1996b; Quinlan 1996), there has been little empirical testing with neural networks (especially with the new Boosting algorithm). In this paper we present a comprehensive evaluation of Bagging and Boosting as methods for creating ensembles of neural networks and compare these results with similar tests for decision trees.

We tested these algorithms with 23 data sets and find many interesting results. The first is that Bagging produces more accurate ensembles than any of its component classifiers, and is relatively impervious to overfitting. On the other hand, with neural networks a simple ensemble technique where the component networks differ only in their initial random weight settings surprisingly does about the same as Bagging. Boosting has more varied results. Sometimes it shows little to no gain in performance over individual classifiers, while other times showing significant gains even over Bagging. In further tests we demonstrate that the varied performance of Boosting can be partially explained by its sensitivity to noise in the set of training examples, thus Boosting may be susceptible to overfitting.

Classifier Ensembles

Figure 1 illustrates the basic framework for a classifier ensemble. In this example, neural networks are the basic classification method, though conceptually any classification method can be substituted in place of the networks. Each network in Figure 1's ensemble (network 1 through network N in this case) is trained using the training instances for that network. Then, for each example, the predicted output of each of these networks (o_i in Figure 1) is combined to produce the output of the ensemble ($\hat{o}$ in Figure 1). Many researchers (Alpaydin 1993; Breiman 1996c; Krogh & Vedelsby 1995; Lincoln & Skrzypek 1989) have demonstrated that an effective combining scheme is to simply average the predictions of the network.

Combining the output of several classifiers is useful only if there is disagreement. Obviously, combining several identical classifiers produces no gain. Hansen and Salamon (1990) proved that if the average error

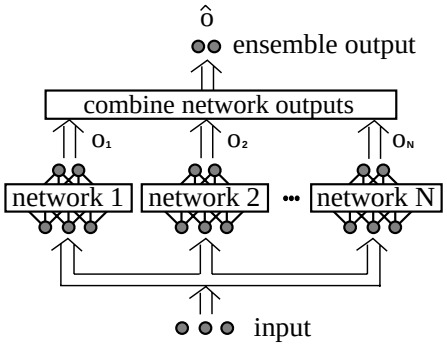

Figure 1: A classifier ensemble of neural networks.

rate for an example is less than 50% and the component classifiers in the ensemble are independent in the production of their errors, the expected error for that example can be reduced to zero as the number of classifiers combined goes to infinity; however, such assumptions rarely hold in practice. Krogh and Vedelsby (1995) later proved that the ensemble error can be divided into a term measuring the average generalization error of each individual classifier and a term measuring the disagreement among the classifiers. What they formally showed was that an ideal ensemble consists of highly correct classifiers that disagree as much as possible. Opitz and Shavlik (1996a,1996b) empirically verified that such ensembles generalize well.

As a result, methods for creating ensembles center around producing classifiers that disagree on their predictions. Generally, these methods focus on altering the training process in the hope that the resulting classifiers will produce different predictions. For example, neural network techniques that have been employed include methods for training with different topologies, different initial weights, different parameters, and training only on a portion of the training set (Alpaydin 1993; Drucker *et al.* 1994; Hansen & Salamon 1990; Maclin & Shavlik 1995). In this paper we concentrate on two popular methods (Bagging and Boosting) that try to generate disagreement among the classifiers by altering the training set each classifier sees.

Bagging Classifiers

Bagging (Breiman 1996a) is a "bootstrap" (Efron & Tibshirani 1993) ensemble method that creates individuals for its ensemble by training each classifier on a random redistribution of the training set. Each classifier's training set is generated by randomly drawing, with replacement, N examples – where N is the size of the original training set; many of the original examples may be repeated in the resulting training set while others may be left out. Each individual classifier in the ensemble is generated with a different random sampling of the training set.

Boosting Classifiers

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 for its ensemble that are better able to correctly predict examples for which the current ensemble performance is poor. (Note that in Bagging, the resampling of the training set is not dependent on the performance of the earlier classifiers.)

In this work we examine two new and powerful forms of Boosting: Arcing (Breiman 1996b) and Ada-Boosting (Freund & Schapire 1996). Like Bagging, these methods choose a training set of size N for classifier $K+1$ by probabilistically selecting (with replacement) examples from the original N training examples. Unlike Bagging, however, the probability of selecting an example is not equal across the training set. This probability depends on how often that example was misclassified by the previous K classifiers.

Both methods initially set the probability of picking each example to be $1/N$. These methods then recalculate these probabilities after each trained classifier is added to the ensemble. For Ada-Boosting, let ϵ_k be the sum of the misclassified instance probabilities of the currently trained classifier C_k. The probabilities for the next trial are generated by multiplying the probabilities of C_k's incorrectly classified instances by the factor $\beta_k = (1-\epsilon_k)/\epsilon_k$ and then renormalizing these probabilities so that their sum equals 1. Ada-Boosting combines the classifiers $C_1, \ldots, C_k$ using weighted voting where C_k has weight $\log(\beta_k)$. We use the revision described by Breiman (1996b) where we reset all the weights to be equal and restart if either ϵ_k is not less than 0.5 or ϵ_k becomes 0.

Arcing updates these probabilities somewhat differently. For the ith example in the training set, the value m_i refers to the number of times that example was misclassified by the previous K classifiers. The probability p_i for selecting example i to be part of classifier $K+1$'s training set is defined as

$$p_i = \frac{1 + {m_i}^4}{\sum_{j=1}^{N}(1 + {m_j}^4)} \quad (1)$$

Breiman chose the value of the power (4) empirically after trying several different values (Breiman 1996b). Unlike Ada-Boosting, Arcing combines its classifiers by unweighted voting.

Table 1: Summary of the data sets used in this paper. Shown are the number of examples in the data set; the number of output classes; the number of continuous and discrete features describing the examples; the number of input, output, and hidden units used in the networks; and how many epochs each network was trained.

			Features		Neural Network			
Dataset	Cases	Classes	Continuous	Discrete	Inputs	Outputs	Hiddens	Epochs
breast-cancer-w	699	2	9	-	9	1	5	20
credit-a	690	2	6	9	47	1	10	35
credit-g	1000	2	7	13	63	1	10	30
diabetes	768	2	9	-	8	1	5	30
glass	214	6	9	-	9	6	10	80
heart-cleveland	303	2	8	5	13	1	5	40
hepatitis	155	2	6	13	32	1	10	60
house-votes-84	435	2	-	16	16	1	5	40
hypo	3772	5	7	22	55	5	15	40
ionosphere	351	2	34	-	34	1	10	40
iris	159	3	4	-	4	3	5	80
kr-vs-kp	3196	2	-	36	74	1	15	20
labor	57	2	8	8	29	1	10	80
letter	20000	26	16	-	16	26	40	30
promoters-936	936	2	-	57	228	1	20	30
ribosome-bind	1877	2	-	49	196	1	20	35
satellite	6435	6	36	-	36	6	15	30
segmentation	2310	7	19	-	19	7	15	20
sick	3772	2	7	22	55	1	10	40
sonar	208	2	60	-	60	1	10	60
soybean	683	19	-	35	134	19	25	40
splice	3190	3	-	60	240	2	25	30
vehicle	846	4	18	-	18	4	10	40

Results

To evaluate the performance of Bagging and Boosting we performed experiments on a number of data sets drawn from the UCI data set repository (Murphy & Aha 1994). We report Bagging and Boosting error rates for each data set for both neural network and decision tree ensembles along with the error rate for simply using a single network or single decision tree. We also report results for a very simple network ensemble approach – creating an ensemble of networks where each network varies only by randomly initializing the weights of the network.

Methodology

All results are averaged over five standard 10-fold cross validation experiments. For each 10-fold cross validation the data set is first partitioned into 10 equal-sized sets, then 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 10 networks is created (for a total of 100 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 number of hidden units and epochs used for training are given in the next section. We chose the number of hidden units based on the number of input and output units. This choice was based on the criteria of having at least one hidden unit per output, at least one hidden unit for every ten inputs, and five hidden units being a minimum. The number of epochs was based both on the number of examples and the number of parameters (i.e., topology) of the network. For the decision trees we used the C4.5 tool (Quinlan 1996) using pruned trees as suggested in Quinlan's work (which empirically produce better performance).

Datasets

Our data sets were drawn from the UCI repository with emphasis on ones that were previously investigated by other researchers. Table 1 gives the characteristics of the data sets we chose. The data sets chosen vary across a number of dimensions including: the type of the features in the data set (i.e., continuous, discrete, or a mix of the two); the number of output classes; and the number of examples in the data set. Table 1 also shows the architecture and training parameters used

Table 2: Test set error rates for the data sets using (1) a single neural network classifier; (2) an ensemble where each individual network is trained using the original training set and thus only differs from the other networks in the ensemble by its random initial weights; (3) an ensemble where the networks are trained using randomly resampled training sets (Bagging); an ensemble where the networks are trained using weighted resampled training sets (Boosting) where the resampling is based on the (4) Arcing method and (5) Ada method; (6) a single decision tree classifier; (7) a Bagging ensemble of decision trees; and (8) a Boosting ensemble of decision trees.

	Neural Network					C4.5		
				Boosting				Boosting
Dataset	Standard	Simple	Bagging	Arcing	Ada	Standard	Bagging	Ada
breast-cancer-w	3.3	3.4	3.3	3.2	3.9	5.0	3.3	3.1
credit-a	14.8	14.0	14.1	14.8	16.2	14.9	12.1	12.6
credit-g	28.3	24.4	24.3	24.8	26.4	29.6	22.8	22.9
diabetes	23.6	22.8	23.2	23.6	22.8	28.3	21.9	22.3
glass	38.5	35.5	33.7	31.5	33.2	30.9	28.4	30.5
heart-cleveland	18.2	17.3	16.7	18.9	19.1	24.3	18.1	17.4
hepatitis	19.9	19.6	18.1	18.9	18.1	21.6	16.5	13.8
house-votes-84	5.0	4.9	4.3	4.7	5.1	3.5	3.6	4.4
hypo	6.4	6.2	6.2	6.4	6.2	0.5	0.4	0.4
ionosphere	10.1	8.0	7.6	7.0	8.0	8.1	6.0	6.0
iris	4.3	4.0	4.3	2.9	3.3	6.0	4.6	5.6
kr-vs-kp	2.3	0.9	0.9	0.6	0.4	0.6	0.5	0.3
labor	5.3	4.2	4.9	3.2	5.0	15.1	13.3	13.2
letter	18.0	12.8	12.5	6.2	4.6	14.0	10.6	6.7
promoters-936	5.0	4.8	4.5	4.7	4.7	12.8	9.5	6.3
ribosome-bind	9.5	8.5	8.4	8.4	8.5	11.2	9.3	9.1
satellite	12.9	11.1	11.0	10.3	10.3	13.8	10.8	10.4
segmentation	6.7	5.6	5.3	3.7	3.7	3.7	2.8	2.3
sick	6.0	5.7	5.8	5.4	4.8	1.3	1.0	0.9
sonar	16.9	16.7	16.5	15.9	12.5	29.0	21.6	19.7
soybean	9.0	6.4	6.8	6.5	6.3	8.0	8.0	7.9
splice	4.7	4.0	3.9	4.0	4.3	5.9	5.7	6.3
vehicle	24.5	21.1	21.7	19.3	19.5	29.4	26.1	24.8

in our neural networks experiments.

Experimental Results

Table 2 shows the neural network and decision tree error rates for the data sets described in Table 1 for the five neural network methods and three decision tree methods discussed in this paper.

Discussion

One conclusion that can be drawn from the results is that both the Simple Ensemble and Bagging approaches almost always produces better performance than just training a single classifier. For some of these data sets (e.g., glass, kr-vs-kp, letter, segmentation, soybean, and vehicle) the gains in performance are quite significant. One aspect many of these data sets share is that they involve predictions for more than two classes, which suggests that ensembles may be especially important for this type of data. Another interesting conclusion is that Bagging produces results similar to those of the Simple Ensemble. What this shows is that simply changing the initial weight settings of a neural network is nearly as effective at causing necessary changes in a network's predictions as creating small changes in the training set.

For Boosting, the results are more varied. Arcing sometimes produces results that are the same or worse than using a single classifier; however, other times it not only significantly outperforms using a single classifier, but significantly outperforms Bagging (e.g., kr-vs-kp, letter, segmentation, and vehicle). Ada-Boosting's results are even more extreme. For certain data sets (kr-vs-kp, letter, sonar), Ada-Boosting produces a significant gain over any other method (including Arcing). On other data sets Ada-Boosting produces results that are even worse than using a single classifier (breast-cancer-wisconsin, credit-a, heart-cleveland). For both Boosting methods, it seems that when they work, they are extremely effective; on the other hand, when the

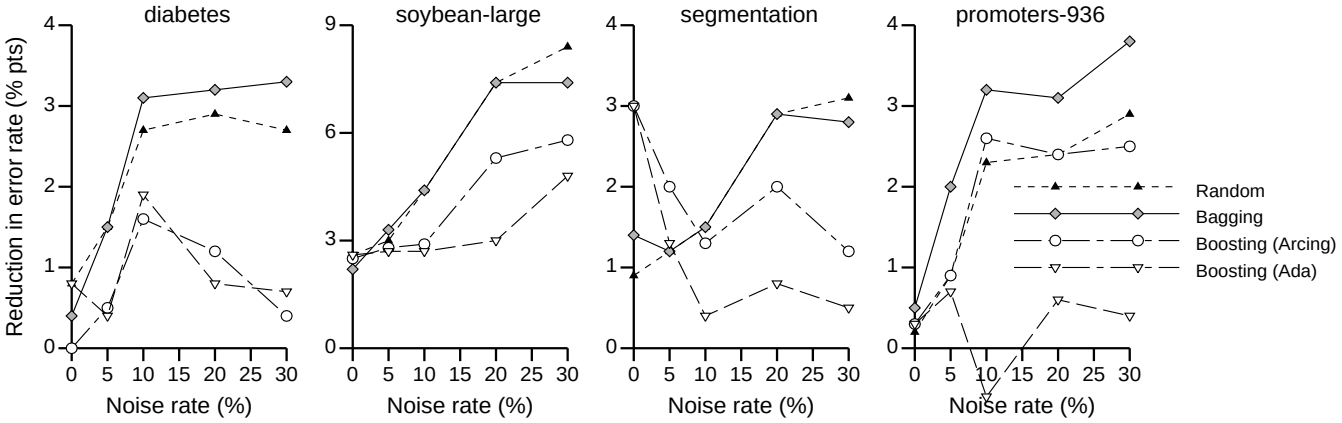

Figure 2: Simple, Bagging, and Boosting (Arcing and Ada) neural network ensembles reduction in error rate compared to using a single neural network classifier on four data sets with varying levels of noise introduced into the data set. Graphed is the percentage *point* reduction in error (e.g., for 5% noise in the segmentation data set, if the single network method had an error rate of 15.9% and the Bagging method had an error rate of 14.7%, then this is graphed as a 1.2 percentage point reduction in the error rate).

Boosting methods fail they can often hinder performance. It is also interesting to note that both Boosting methods significantly outperform Bagging on the letter, segmentation, and vehicle domains, which suggests that their positive effects may be greatest when multiple classes are predicted.

Freund & Shapire (1996) suggested that the sometimes poor performance of Boosting results from overfitting the training set since later training sets may be over-emphasizing examples that are noise (thus creating extremely poor classifiers). This argument seems especially pertinent to Ada-Boosting for two reasons. The first and most obvious reason is that its method for updating the probabilities may be over-emphasizing noisy examples. The second reason is that the classifiers are combined using weighted voting. Previous work (Sollich & Krogh 1996) has shown that optimizing the combining weights can lead to overfitting while an unweighted voting scheme is generally resilient to the problems of overfitting.

To evaluate the hypothesis that Boosting may be prone to overfitting we performed a second set of experiments using the four ensemble neural network methods. We introduced 5%, 10%, 20%, and 30% noise[2] into four different data sets. At each level we created five different noisy data sets, performed a 10-fold cross validation on each, then averaged over the five results. In Figure 2 we show the reduction in error rate for each of the ensemble methods compared to using a single neural network classifier. These results demonstrate that as the noise level grows, the efficacy of the Simple and Bagging ensembles generally increases while the Arcing and Ada-Boosting ensembles gains in performance are much smaller (or may actually decrease). Note that this effect is more extreme for Ada-Boosting which supports our hypothesis that Ada-Boosting is more effected by noise. This suggests that Boosting's poor performance for certain data sets may be explained by overfitting noise in those data sets.

Finally, we can compare the results of the neural network approaches with the C4.5 approaches. Here the results are nearly equivocal. Neural networks perform significantly better than decision trees for several data sets but decision trees also outperform neural networks for several data sets, so no strong conclusions can be drawn. One interesting thing to note is that the performance of Bagging and Boosting on neural networks generally correlates with the performance of Bagging and Boosting on C4.5. This suggests that the advantages and disadvantages of both Bagging and Boosting are independent of the classifier used and depend only on the domain on which they are being applied.

Future Work

Though our results do indeed suggest that both Boosting methods are prone to overfitting, we noted that there are at least two possible explanations for this: (1) over-emphasizing noisy examples in the later training sets; and (2) weighted voting of the networks. We plan to perform further experiments to determine if the overfitting that occurs is explained by one of these factors or by a combination of both.

Since the Boosting methods are extremely successful in many domains, we plan to investigate novel approaches that will retain the benefits of Boosting. The goal will be to create a learner where you can essentially push a start button and let it run. To do this

[2] 5% noise indicates that 5% of the features of the example, both input and output features, were randomly changed to other feature values.

we would try to preserve the benefits of Boosting while preventing overfitting on noisy data sets.

We also plan to compare Bagging and Boosting methods to other methods introduced more recently. In particular we intend to examine the use of Stacking (Wolpert 1992) as a method of *training* a combining function, so as to avoid the effect of having to weight classifiers. We will also compare Bagging and Boosting to other methods such as Opitz and Shavlik's (1996b) approach to creating an ensemble. This approach uses genetic search to find classifiers that are accurate and differ in their predictions.

Conclusions

This paper presents an empirical evaluation of Bagging (Breiman 1996a) and Boosting (Freund & Schapire 1996) for neural networks and decision trees. Our results demonstrate that a Bagging ensemble nearly always outperforms a single classifier. Our results also show that an Arcing ensemble can greatly outperform both a Bagging ensemble and a single classifier. However, for some data sets Arcing only shows a small or zero gain in performance over a single classifier. Similarly, we show that an Ada-Boosting ensemble can outperform all of the other methods (including Arcing), but that Ada-Boosting can also actually produce worse performance than using a single classifier. Further tests demonstrate that the performance of both methods of Boosting declines as the amount of noise in a data set increases, and that this result is more extreme in Ada-Boosting. In conclusion, as a general technique, Bagging is probably appropriate for most problems, but when properly applied, Boosting (either Arcing or Ada) may produce larger gains in accuracy.

Acknowledgements

This research was partially supported by University of Minnesota Grants-in-Aid to both authors. We wish to thank Mike Henderson for his work on the Bagging and Boosting versions of C4.5. David Opitz completed a portion of this work while at the University of Minnesota, Duluth.

References

Alpaydin, E. 1993. Multiple networks for function learning. In *Proceedings of the 1993 IEEE International Conference on Neural Networks*, volume I, 27–32.

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.

Breiman, L. 1996c. Stacked regressions. *Machine Learning* 24(1):49–64.

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

Drucker, H., and Cortes, C. 1996. Boosting decision trees. In Touretsky, D.; Mozer, M.; and Hasselmo, M., eds., *Advances in Neural Information Processing Systems*, volume 8, 479–485. Cambridge, MA: MIT Press.

Drucker, H.; Cortes, C.; Jackel, L.; LeCun, Y.; and Vapnik, V. 1994. Boosting and other machine learning algorithms. In *Proceedings of the Eleventh International Conference on Machine Learning*, 53–61. New Brunswick, NJ: Morgan Kaufmann.

Efron, B., and Tibshirani, R. 1993. *An Introduction to the Bootstrap*. New York: Chapman and Hall.

Freund, Y., and Schapire, R. 1996. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, 148–156. Morgan Kaufmann.

Hansen, L., and Salamon, P. 1990. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 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., *Advances in Neural Information Processing Systems*, volume 7, 231–238. Cambridge, MA: MIT Press.

Lincoln, W., and Skrzypek, J. 1989. Synergy of clustering multiple back propagation networks. In Touretzky, D., ed., *Advances in Neural Information Processing Systems*, volume 2, 650–659. San Mateo, CA: Morgan Kaufmann.

Maclin, R., and Shavlik, J. 1995. Combining the predictions of multiple classifiers: Using competitive learning to initialize neural networks. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 524–530.

Murphy, P. M., and Aha, D. W. 1994. UCI repository of machine learning databases (machine-readable data repository). University of California-Irvine, Department of Information and Computer Science.

Opitz, D., and Shavlik, J. 1996a. Actively searching for an effective neural-network ensemble. *Connection Science* 8(3/4):337–353.

Opitz, D., and Shavlik, J. 1996b. Generating accurate and diverse members of a neural-network ensemble. In Touretsky, D.; Mozer, M.; and Hasselmo, M., eds., *Advances in Neural Information Processing Systems*, volume 8, 535–541. Cambridge, MA: MIT Press.

Perrone, M. 1993. *Improving Regression Estimation: Averaging Methods for Variance Reduction with Extension to General Convex Measure Optimization*. Ph.D. Dissertation, Brown University, Providence, RI.

Quinlan, J. R. 1996. Bagging, boosting, and c4.5. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 725–730. AAAI/MIT Press.

Sollich, P., and Krogh, A. 1996. Learning with ensembles: How over-fitting can be useful. In Touretsky, D.; Mozer, M.; and Hasselmo, M., eds., *Advances in Neural Information Processing Systems*, volume 8, 190–196. Cambridge, MA: MIT Press.

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