ERROR CORRELATION AND ERROR REDUCTION IN ENSEMBLE CLASSIFIERS*

Kagan Tumer and Joydeep Ghosh
Department of Electrical and Computer Engineering,
University of Texas, Austin, TX 78712-1084
E-mail: kagan@pine.ece.utexas.edu
ghosh@pine.ece.utexas.edu
Tel: (512) 471-8980

July 22, 1996

Heading: Error correlation and reduction

Keywords: Combining, cross validation, error correlation, error reduction, ensemble classifiers, bootstrapping, resampling.

*This research was supported in part by AFOSR contract F49620-93-1-0307, NSF grant ECS 9307632, and ARO contracts DAAH 04-94-G0417 and 04-95-10494.

Abstract

Using an ensemble of classifiers, instead of a single classifier, can lead to improved generalization. The gains obtained by combining however, are often affected more by the selection of what is presented to the combiner, than by the actual combining method that is chosen. In this paper we focus on data selection and classifier training methods, in order to "prepare" classifiers for combining. We review a combining framework for classification problems that quantifies the need for reducing the correlation among individual classifiers. Then, we discuss several methods that make the classifiers in an ensemble more complementary. Experimental results are provided to illustrate the benefits and pitfalls of reducing the correlation among classifiers, especially when the training data is in limited supply.

1 Introduction

A classifier's ability to meaningfully respond to novel patterns, or *generalize*, is perhaps its most important property (Levin et al., 1990; Wolpert, 1990). In general however, the generalization is not unique, and different classifiers provide different generalizations by realizing different decision boundaries (Ghosh and Tumer, 1994). For example, when classification is performed using a multilayered, feed-forward artificial neural network, different weight initializations, or different architectures (number of hidden units, hidden layers, node activation functions, etc.), result in differences in performance. It is therefore necessary to train a multitude of networks when approaching a classification problem to ensure that a good model/parameter set is found. However, selecting the "best" classifier is not necessarily the ideal choice, because potentially valuable information may be wasted by discarding the results of less-successful classifiers. This observation leads to the concept of "combining" wherein the outputs of several classifiers are pooled before a decision is made.

Currently, the most popular way of combining multiple classifiers is via simple averaging of the corresponding output values (Lincoln and Skrzypek, 1990; Perrone and Cooper, 1993b; Tumer and Ghosh, 1996). Weighted averaging has also been proposed, and different methods for computing the proper classifier weights have been studied (Benediktsson et al., 1994; Hashem and Schmeiser, 1993; Jacobs, 1995; Lincoln and Skrzypek, 1990). Such linear combining techniques have been mathematically analyzed both for classification (Tumer and Ghosh, 1995c; Tumer and Ghosh, 1996), and regression problems (Perrone and Cooper, 1993a; Hashem and Schmeiser, 1993). Some researchers have investigated non-linear combiners using rank-based information (Ho et al., 1994; Al-Ghoneim and Vijaya Kumar, 1995), belief-based methods (Rogova, 1994; Yang and Singh, 1994; Xu et al., 1992), or voting schemes (Hansen and Salamon, 1990; Battiti and Colla, 1994). We have introduced "order statistics" combiners, and analyzed their properties (Tumer and Ghosh, 1995b; Tumer and Ghosh, 1995c). Wolpert introduced the concept of "stacking" classifiers, allowing each stage to correct the mistakes of the previous one (Wolpert, 1992). Combiners have also been successfully applied to a multitude of real world problems (Baxt, 1992; Ghosh et al., 1996; Lee et al., 1991).

Most research in this area focuses on finding the types of combiners that improve performance. Yet, it is important to note that if the classifiers to be combined repeatedly provide the same (either erroneous or correct) classification decisions, there is little to be gained from combining, regardless of the chosen scheme. Therefore, the selection and training of the classifiers that will be combined is as critical an issue as the selection of the combining method. Indeed, classifier/data selection is directly tied to the amount of correlation among the various classifiers, which in turn affects the amount of error reduction that can be achieved. For regression problems, Perrone and Cooper show that their combining results are weakened if the networks are not independent (Perrone and Cooper, 1993b). Ali and Pazzani discuss the relationship between error correlations and error reductions in the context of decision trees (Ali and Pazzani, 1995). The Boosting algorithm trains subsequent classifiers on training patterns that have been "selected" by earlier classifiers (Drucker et al., 1994), thus reducing the correlation among them. However, one can quickly run out of training data in practice if this approach is used. Twomey and Smith discuss combining and resampling in the context of a 1-d regression problem (Twomey and Smith, 1995). Meir discusses the effect of independence on combiner performance (Meir, 1995), and Jacobs reports that $N' \leq N$ independent classifiers are worth as much as N dependent classifiers (Jacobs, 1995). Breiman also addresses this issue, and discusses methods aimed at reducing the correlation among estimators (Breiman, 1993; Breiman, 1994). Krogh and Vedelsky discuss how cross-validation can be used to improve ensemble performance (Krogh and Vedelsby, 1995). The influence of the amount of training on ensemble performance is studied in (Sollich and Krogh, 1996), and the selection of individual classifier through a genetic algorithm is suggested in (Opitz and Shavlik, 1996). For classification problems, the influence of the correlation among the classifiers on the error rate of multiple classifiers was quantified by Tumer and Ghosh (Tumer and Ghosh, 1995c; Tumer and Ghosh, 1996).

In this paper we address four methods for reducing the correlations among the individual classifiers. First we use a cross-validation like partitioning of the training set and train a different classifier on each partition. The second method focuses on generating new training sets by pruning the inputs. This method relies on the redundant information among input features. By varying the criterion for pruning, a variety of training sets can be obtained for training different classifiers. The third correlation reduction method focuses on generating new training sets through resampling methods. For classification problems, Breiman first discussed this idea for combiners based on majority voting (Breiman, 1994). The final method we explore has a different feel from the previous ones, and needs a weighted averaging of the outputs. Instead of partitioning the training patterns randomly (e.g., as in cross-validation or bootstrapping), we partition them spatially. The net result is a reduction of complexity for each classifier. In all four techniques, the individual classifiers see less data, and thus there is a fundamental trade-off between decreased correlation and reduced individual performance. This trade-off is experimentally studied through simulation results on a common data set.

2 Background

In this section we outline a recently introduced mathematical framework that yields a quantitative relationship between classifier correlations and the reduction in error, when an averaging combiner is used. For more details, see (Tumer and Ghosh, 1995c; Tumer and Ghosh, 1996).

The outputs of parametric classifiers that are trained to minimize a cross-entropy or mean square error (MSE) function, given "*one-of-L*" desired output patterns, approximate the *a posteriori* probability densities of the corresponding class (Richard and Lippmann, 1991; Ruck et al., 1990). Therefore, the ith output unit of a *one-of-L* classifier network to a given input x can be modeled as[1]:

$$f_i(x) = p(c_i|x) + \eta_i(x) \ , \tag{1}$$

where $p(c_i|x)$ is the *a posteriori* probability distribution of the ith class given input x, and $\eta_i(x)$ is the error associated with the ith output.

For the Bayes optimum decision, a vector x is assigned to class i if $p(c_i|x) > p(c_k|x), \ \forall k \neq i$. Therefore, the Bayes optimum boundary is the loci of all points x^* such that $p(c_i|x^*) = p(c_j|x^*)$ where $p(c_j|x^*) = \max_{k\neq i} \ p(c_k|x)$. Since our classifier provides $f_i(\cdot)$ instead of $p(c_i|\cdot)$, the decision boundary obtained may vary from the optimum boundary (see Figure 1). The amount by which the boundary of the classifier, x_b, differs from the optimum boundary, x^*, is given by $b = x_b - x^*$. By performing a linear approximation of $p(c_k|x)$ around x^*, one can express the density function $f_b(b)$ of b in terms of the $\eta_i(x)$s (see (Tumer and Ghosh, 1995c; Tumer and Ghosh, 1996) for details).

Figure 1 shows the *a posteriori* probabilities obtained by a non-ideal classifier, and the associated added error region. The lightly shaded area provides the Bayesian error region. The

[1] If two or more classifiers need to be distinguished, a superscript is added to $f_i(x)$ and $\eta_i(x)$ to indicate the classifier number.

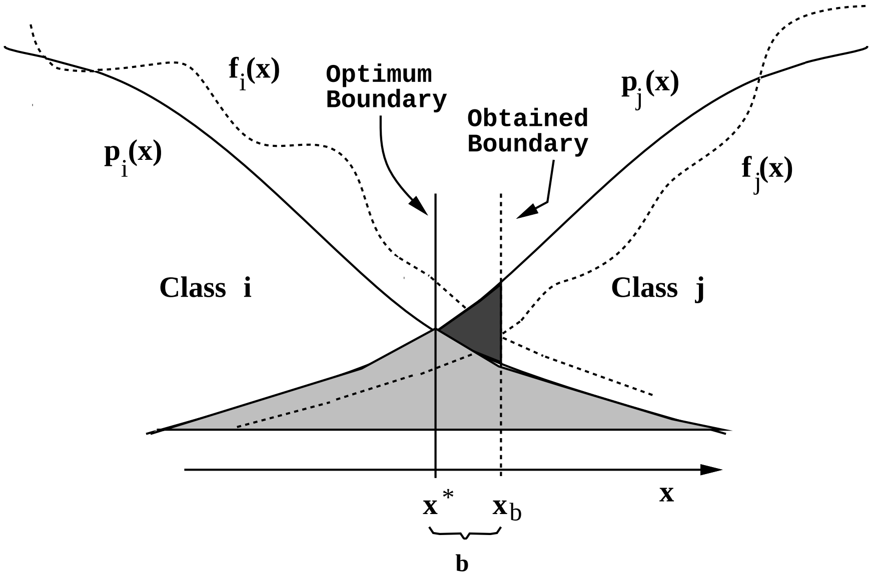

Figure 1: Decision boundaries and error regions associated with approximating the *a posteriori* probabilities (Tumer and Ghosh, 1996).

darkly shaded area is the added error region, $A(b)$, associated with selecting a decision boundary that is offset by b. Patterns corresponding to the darkly shaded region belong to class j, but are erroneously assigned to class i by the classifier. The expected added error, E_{add}, is given by:

$$E_{add} = \int_{-\infty}^{\infty} A(b) f_b(b) db, \tag{2}$$

where f_b is the density function for b. For example, if the $\eta_k(x)$s are zero-mean i.i.d., then one can show b to be a zero-mean random variable with variance, $\sigma_b^2 = \frac{2\,\sigma_{\eta_k}^2}{s^2}$; and:

$$E_{add} = \frac{s\sigma_b^2}{2} \tag{3}$$

$$E_{tot} = E_{add} + E_{bay}. \tag{4}$$

where E_{tot} is the total error, E_{bay} is the Bayes error, and s is the difference between the derivatives of the two posteriors.

For analyzing the error regions after combining, and comparing them to the single classifier case, one needs to determine the variance of the boundary obtained with the combiner. In (Tumer and Ghosh, 1996), we show that when the classifier errors are i.i.d., combining reduces the added error by N, or that $E_{add}^{ave} = \frac{1}{N} E_{add}$. In the next section we derive the added error of a combiner when the assumption that the classifiers be i.i.d is removed.

3 Combining Correlated Classifiers

Suppose the combiner denoted by *ave* performs an arithmetic average in output space. If N classifiers are available, the ith output of the *ave* combiner provides an approximation to $p(c_i|x)$ given by:

$$f_i^{ave}(x) = \frac{1}{N} \sum_{m=1}^{N} f_i^m(x) \; = \; p(c_i|x) \; + \; \bar{\eta}_i(x) \; , \tag{5}$$

where:

$$\bar{\eta}_i(x) = \frac{1}{N} \sum_{m=1}^{N} \eta_i^m(x) \; .$$

The variance of $\bar{\eta}_i$ is given by:

$$\begin{aligned} \sigma^2_{\bar{\eta}_i} &= \frac{1}{N^2} \sum_{l=1}^{N} \sum_{m=1}^{N} cov(\eta_i^m(x), \eta_i^l(x)) \\ &= \frac{1}{N^2} \sum_{m=1}^{N} \sigma^2_{\eta_i^m(x)} + \frac{1}{N^2} \sum_{m=1}^{N} \sum_{l \neq m} cov(\eta_i^m(x), \eta_i^l(x)) \end{aligned}$$

where $cov(\cdot,\cdot)$ represents the covariance. Expressing the covariances in term of the correlations ($cov(x,y) \; = \; corr(x,y)\, \sigma_x \, \sigma_y$), leads to:

$$\sigma^2_{\bar{\eta}_i} = \frac{1}{N^2} \sum_{m=1}^{N} \sigma^2_{\eta_i^m(x)} + \frac{1}{N^2} \sum_{m=1}^{N} \sum_{l \neq m} corr(\eta_i^m(x), \eta_i^l(x)) \sigma_{\eta_i^m(x)} \sigma_{\eta_i^l(x)}. \tag{6}$$

Equation 6 is significantly simplified by using the common variance σ_{η_i}, and the average correlation factor among classifiers, δ_i, given by:

$$\delta_i = \frac{1}{N\,(N-1)} \sum_{m=1}^{N} \sum_{m \neq l} corr(\eta_i^m(x), \eta_i^l(x)),$$

leading to:

$$\sigma^2_{\bar{\eta}_i} \; = \; \frac{1}{N} \sigma^2_{\eta_i(x)} \; + \; \frac{N-1}{N} \, \delta_i \, \sigma^2_{\eta_i(x)} \; .$$

In Section 2, the variance of the boundary offset was given in terms of the variances of each classifier's error. Extending that analysis to b^{ave} yields:

$$\sigma^2_{b^{ave}} \; = \; \frac{\sigma^2_{\bar{\eta}_i} + \sigma^2_{\bar{\eta}_j}}{s^2}.$$

Therefore:

$$\sigma^2_{b^{ave}} \;=\; \frac{1}{s^2}\left(\frac{1}{N}\sigma^2_{\eta_i(x)}(1+(N-1)\delta_i) \;+\; \frac{1}{N}\sigma^2_{\eta_j(x)}(1+(N-1)\delta_j)\right)$$

or:

$$\sigma^2_{b^{ave}} \;=\; \frac{\sigma^2_{\eta_i(x)} \;+\; \sigma^2_{\eta_j(x)}}{Ns^2} \;+\; \frac{N-1}{Ns^2}(\delta_i\sigma^2_{\eta_i(x)} + \delta_j\sigma^2_{\eta_j(x)}). \tag{7}$$

Recalling that the noise between classes are i.i.d. leads to[2]:

$$\begin{aligned}\sigma^2_{b^{ave}} &= \frac{1}{N}\sigma^2_b \;+\; (\frac{N-1}{N})\frac{2\sigma^2_{\eta_j(x)}}{s^2}\frac{\delta_i+\delta_j}{2} \\ &= \frac{\sigma^2_b}{N}\left(1 \;+\; (N-1)\frac{\delta_i+\delta_j}{2}\right).\end{aligned} \tag{8}$$

This correlation term in Equation 8 only applies to classes i and j. In order to extend this expression to include all the classes, we use the following expression:

$$\delta = \sum_{i=1}^{L} P_i\,\delta_i \tag{9}$$

where P_i is the prior probability of class i. The correlation contribution of each class to the overall correlation, is proportional to the prior probability of that class.

Qualitatively, the reduction in variance can be readily translated into a reduction in error rates, because a narrower boundary distribution means the likelihood that a boundary will be near the ideal one is increased. Quantitatively, the corresponding error region is given by:

$$\begin{aligned}E^{ave}_{add} &= \frac{s}{2}\,\sigma^2_{b^{ave}} \;=\; \frac{s}{2}\,\sigma^2_b\left(\frac{1+\delta(N-1)}{N}\right) \\ &= E_{add}\left(\frac{1+\delta(N-1)}{N}\right).\end{aligned} \tag{10}$$

Note that Equation 10 only relates to the *added* error rate, i.e. the error rate beyond the Bayes rate. The effect of the correlation between the errors of each classifier is readily apparent from Equation 10. If the errors are independent, then the second part of the reduction term vanishes and the combined error is reduced by N. If on the other hand, the error of each classifier has correlation 1, then the error of the combiner is equal to the initial errors and there is no improvement due to combining. Figure 2 shows how the error reduction is affected by N and δ (using Equation 10).

[2]The errors between classifiers are correlated, not the errors between classes.

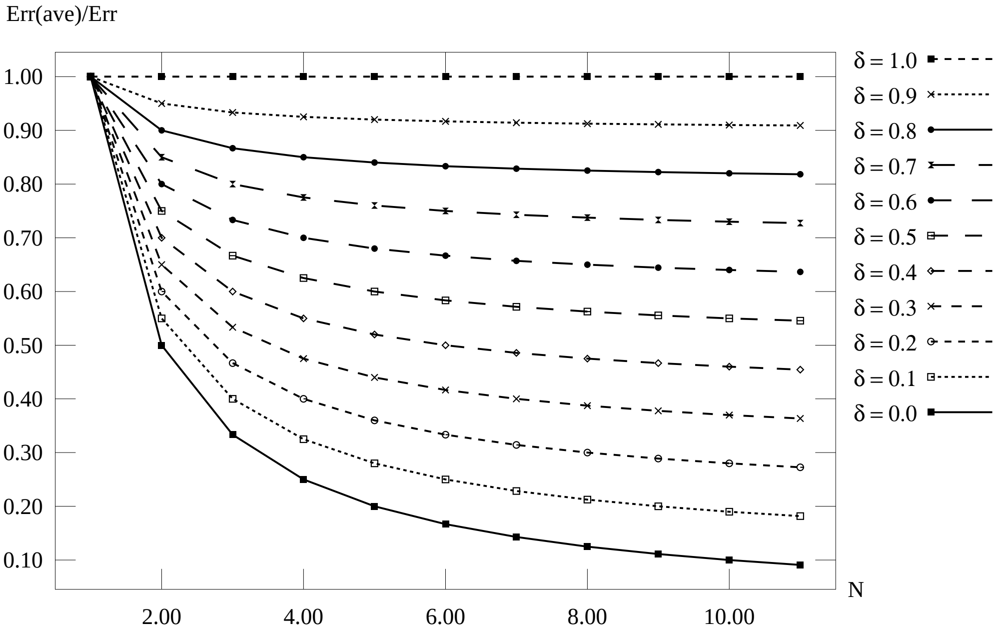

Figure 2: Error reduction ($\frac{E_{add}^{ave}}{E_{add}}$) for different classifier error correlations.

In general, the correlation values lie between these two extremes, and some reduction is achieved. It is important to understand the interaction between N and δ in order to maximize the reduction. As more and more classifiers are used (increasing N), it becomes increasingly difficult to find lowly correlated classifiers. Therefore, methods aimed at explicitly reducing the correlations must be considered.

4 Correlation Reduction Methods

The importance of the correlation values among individual classifiers in a combiner system was established in Section 3. When it is possible to extract different types of feature sets (e.g. power spectrum based vs. auto-regression based from a time signal) from the raw data, or to use radically different types of classifiers, one can avoid combining highly correlated classifiers. However, when only a single feature set is available and only one type of classifier (say, an MLP) is used, care must be taken during the combining process. In such a case, alternatives to combining N instances of the same classifier trained on the same training set are needed.

4.1 Combining $k-1$-of-k Trained Classifiers

Cross-validation, a statistical method aimed at estimating the "true' error (Friedman, 1994; Stone, 1974; Weiss and Kulikowski, 1991), provides a method for lowering correlations. In k-fold cross-validation, the training set is divided into k subsets. Then, $k-1$ of these subsets are used to train the network and results are tested on the subset that was left out of the training. By averaging the different errors obtained on the "left out" sets, a measure of the true error can be obtained. Similarly, by changing the subset that is left out of the training process, one can construct k classifiers, each of which is trained on a slightly different training set. The correlation among the k classifiers will be less than if each had been trained on the full (and identical) training set. We will call this "$k-1$-of-k" training. Note that if cross-validation is anyway being used to determine when to stop training for best generalization (Moody, 1994), we already get k trained classifiers, i.e. the extra overhead of combining is very little (Lippmann, 1995).

4.2 Input Decimation Combining

Another approach to reducing the correlation of classifiers can be found in input decimation, or in purposefully withholding some parts of each pattern from a given classifier. The method of the previous section can be called a pattern-level method, as it focuses on decoupling patterns in the training set of each classifier. In the same way, input decimation is a feature-level (or dimension-level) method as it reduces the dimensionality of a training set in order to reduce the correlation between classifiers.

The effect each single input dimension has on each output of the classifier can be measured. By selectively pruning the inputs that have the least effect on the outputs, one can train a classifier with partial inputs without compromising the overall performance. However, pruning too many inputs may lead to substantial reduction in discriminating power. Furthermore, determining which inputs to prune can be difficult. One method that provides a satisfactory solution is to train L classifiers where L is the number of output classes. For each class, a subset of inputs with low correlation to that class can be removed. The resulting L classifiers will each have seen a slightly different feature set. The degree of correlation between the classifiers can be controlled by the threshold chosen to include or dismiss a given input element.

4.3 Resampled Set Combining

The third correlation reduction method we address consists of generating different training sets for each classifier by resampling the original set. This resampling method is called bootstrapping. It is generally used for estimating the true error rate for problems with very little data (Efron, 1982; Efron, 1983; Jain et al., 1987; Weiss and Kulikowski, 1991). Breiman first used this idea to improve the performance of predictors, and dubbed it "bagging" predictors (Breiman, 1994). For regression problems, bagging uses the average of all the available predictors, whereas for classification problems, it takes a majority vote. Since in this article we deal with linear combiners, we will continue to perform the combining outlined in Section 3. However, since this is a minor variation on bagging, we will also report the bagging results, i.e. when majority voting is used.

4.4 Weighted Experts

The final method of correlation reduction that we present has a different flavor than the previous ones. The method is based on the mixture of experts framework (Haykin, 1994; Jacobs et al., 1991; Xu et al., 1995), where the output is a weighted sum of the outputs of individual networks or "experts". The weights are determined by a gating network, and are a function of the inputs. In a given region of the input space, a particular expert will be weighted more than others. Moreover, the parameter updates during training of individual networks are proportional to the gating weights. In effect, the training set is soft-partitioned according to spatial similarity, rather than the random partitions of Sections 4.1 and 4.3. Then, during training, different classifiers (experts) try to model different parts of the input space.

In the typical mixture of experts, each individual network is single layered (Haykin, 1994), and thus has limited capabilities. Consequently, a large number of experts may be needed for realistic problems (Ramamurti and Ghosh, 1996). To make a fair comparison with the training and combining schemes analyzed earlier, we use a smaller number of more powerful experts, namely MLP or RBF networks. The localized network of (Xu et al., 1995) is used for the gating network.

5 Experimental Results

In order to provide both details and insight, we have divided this section into two parts. First we will provide detailed experimental results on one difficult data set, outlining all the relevant design steps/parameters. Then we will summarize results on some other data sets taken from the UCI depository/Proben1 benchmarks (Prechelt, 1994), and discuss the implications of those results.

5.1 Underwater Sonar Data

In order to examine the benefits of combining and the effect of correlation on combining results, we use a difficult data set extracted from underwater acoustic signals. From the original passive sonar returns from four different underwater objects, a 25-dimensional feature set was extracted (Ghosh et al., 1992; Ghosh et al., 1996). Each patterns consists of 16 Gabor wavelet coefficients, 8 temporal descriptors and spectral measurements and 1 value denoting signal duration. There were 496 patterns in the training set, and 823 in the test set (Table I). The data is available at URL *http://www.lans.ece.utexas.edu.* This data set was selected because:

- The classification task is reasonably complex;
- The input dimensionality is high;
- We have an estimate of the Bayes error rate ($E_{bay} \simeq 3.61$, see (Tumer and Ghosh, 1995a)), and thus a yardstick to measure classifier performance.
- The number of training patterns is moderate, allowing various methods to be tested without biasing the experiments towards highly data intensive methods.

5.1.1 Results on Full Training Set

In this section we present the base results obtained from the oceanic data set. All classification results given in this article are the *test set results, and report the average error and standard deviation (σ) over* 20 *runs.* Each run starts from a different random initial set of weights, trains on the same 496 samples. Two types of feed forward networks, namely a multi-layered

Table I: Description of Data.

Class Description	Number of Patterns	
	Training	Test
Porpoise Sound	116	284
Cracking Ice	116	175
Whale Sound 1	116	129
Whale Sound 2	148	235
Total	496	823

perceptron (MLP) with a single hidden layer with 50 units and a radial basis function (RBF) network with 50 kernels, are used to classify the patterns. Both types of networks are trained until the classification rate of the validation set reaches a plateau.

Table II: Combining Results.

Classifier(s)		Ave		Estimated
	N	Error	σ	Correlation
MLP	1	7.47	0.44	0.89
	3	7.19	0.29	
	7	7.11	0.23	
RBF	1	6.79	0.41	0.79
	3	6.15	0.30	
	7	5.97	0.22	

Table II provides the test set results for single classifiers ($N = 1$), the combining results, as well as the estimated average error correlations between different runs of a given classifier. Because the main purpose of this experiment is to study the effect of the correlation on the improvements over the base results, MLP/RBF hybrid models are not considered.

5.1.2 $k-1$-of-k Training

Table III presents 2-of-3 training results for the sonar data presented in Section 5.1, and the corresponding combining results. Table IV shows the correlation values between classifiers trained on different partitions of the training set. The correlation values among different runs of a classifier trained on a particular set (the diagonal elements) are comparable to the values obtained from the full training set in Section 5.1.1. The correlation values between different partitions however, are noticeably lower. 2-of-3 training provides less correlated classifiers which in turn improve percentage performance gains. For example, combining three MLPs trained on different partitions of the data provides improvements of 12.4% over the average result of each classifier. In contrast, combining three MLPs trained on the full training set only provides improvements of 3.75% over single MLP results (from results in Section 5.1.1).

Table III: 2-of-3 Training Results.

Classifier(s)		Ave	
	N	Error	σ
	1	9.37	0.98
MLP	3	8.21	0.35
	1	7.98	0.98
RBF	3	6.16	0.34

Table IV: Correlations between 2-of-3 Trained Classifiers.

		CV1	CV2	CV3
	CV1	0.90	0.51	0.58
MLP	CV2		0.89	0.54
	CV3			0.89
	CV1	0.81	0.56	0.57
RBF	CV2		0.82	0.52
	CV3			0.81

It is important to note that reducing the correlation among classifiers is only helpful if the classification performance is not significantly affected by the reduced training set size. In the case where $k = 3$, only two thirds of the original training data was used in each partition. This proved to be insufficient to train the networks successfully, and resulted in poor final performance even though the correlations among individual classifiers were brought down. Therefore, we conclude that k was chosen too low, and this observation leads us to repeat the experiments with $k = 7$.

In this case, each training set possesses six sevenths of the original data, a value deemed high enough to avoid the pitfalls encountered in the previous section. The individual classifier results ($N = 1$) and the combining results are shown in Table V.

Table VI shows the correlation values between classifiers trained on different partitions of the training set. Although the between partition correlations are higher than they were for $k = 3$, they are still noticeably lower than the within partition correlations. The combining

Table V: 6-of-7 Training Results.

Classifier(s)		Ave	
	N	Error	σ
	1	8.24	0.56
MLP	7	7.39	0.24
	1	6.51	0.61
RBF	7	5.48	0.31

Table VI: Correlations between 6-of-7 Trained Classifiers.

		CV1	CV2	CV3	CV4	CV5	CV6	CV7
	CV1	0.88	0.79	0.74	0.76	0.72	0.73	0.78
	CV2		0.90	0.77	0.81	0.75	0.75	0.80
	CV3			0.89	0.80	0.77	0.74	0.79
MLP	CV4				0.89	0.76	0.75	0.79
	CV5					0.89	0.74	0.78
	CV6						0.90	0.78
	CV7							0.89
	CV1	0.78	0.70	0.70	0.71	0.70	0.67	0.73
	CV2		0.80	0.70	0.69	0.69	0.66	0.71
	CV3			0.80	0.70	0.69	0.67	0.72
RBF	CV4				0.79	0.69	0.66	0.70
	CV5					0.79	0.67	0.71
	CV6						0.82	0.66
	CV7							0.79

results for MLPs were still not up to the level of the results obtained when the full training sets were used. However, for the RBF network, 6-of-7 training and subsequent combining provided moderate improvements for $N = 7$.

The trade-off between the reduced training size and correlation is apparent with the results presented in this section. As k is increased, the training set contains more and more similar patterns and the differences between classifiers diminish, increasing the correlation among them. As k becomes smaller, the correlations decrease, but one must ensure that the partitions of the training set still contain enough patterns to properly train the individual classifiers. The balance where the gains due to decreased correlations outweigh the losses due to reduced training size must be found for this method to provide the best results.

5.1.3 Input Decimation Combining

Table VII presents the results obtained after reducing the input dimensionality of the sonar data set. The four partitions are obtained by retaining the 22 inputs that have the highest correlation to each output. Retaining more features did not result in a significant drop in correlations, whereas removing more features resulted in drops in individual classifier performance that were too large to be compensated by combining. The results indicate that the deletion of even lowly-correlated inputs affects the performance of the classifier significantly. For this experiment, we chose $N = 4, 8$ rather than $N = 3, 7$ because of the method used for decimation. Since the decimation depended on each output's correlation to the inputs, the natural number of classifiers to combine were multiples of four (this is a four-class problem).

Table VII: Input Decimation Results.

Classifier(s)		Ave	
	N	Error	σ
MLP	1	8.38	0.61
	4	7.10	0.32
	8	6.99	0.27
RBF	1	7.85	0.72
	4	6.78	0.29
	8	6.70	0.26

Table VIII: Correlations between Input Decimated Partitions.

		PAR1	PAR2	PAR3	PAR4
MLP	PAR1	0.88	0.63	0.65	0.65
	PAR2		0.88	0.74	0.71
	PAR3			0.86	0.69
	PAR4				0.87
RBF	PAR1	0.83	0.59	0.60	0.59
	PAR2		0.80	0.72	0.72
	PAR3			0.81	0.67
	PAR4				0.79

Table VIII shows the correlation values between the classifiers trained on different decimations, and underlines the correlation reduction among different partitions. There are percentage performance improvements accompanying this correlation drop. This scheme failed to significantly improve on the combining results of the classifiers trained on the full inputs for the

RBF network, but provided minor improvements over MLP results[3]. Once again, the chief barrier to improved generalization results is the drop in individual classifier performance which accompanies the drop in correlation factors.

5.1.4 Resampled Set Combining

In the first set of "resample/combine" experiments, the number of training patterns is kept at the same level as the number of patterns that were present in the original data. That is, from 496 original points, 496 are randomly picked with *resampling*. With this method, also referred to as $e0$ bootstrap, each resampled set contains 63.2% of the original training data, on the average (Weiss and Kulikowski, 1991).

Table IX: Resampling Results.

Classifier(s)		Ave		Bagging	
	N	Error	σ	Error	σ
MLP	1	8.60	0.65		
	3	7.47	0.41	7.68	0.15
	5	7.49	0.37	7.45	0.11
	7	7.47	0.27	7.42	0.12
RBF	1	8.15	0.71		
	3	6.72	0.38	6.68	0.39
	5	6.22	0.33	6.40	0.36
	7	6.01	0.35	6.29	0.41

Table IX shows classification results obtained using training sets with resampling, as well as the combining results. The correlation factors are given in Table X. The correlations between partitions were lowered, but the drop was accompanied with a significant drop in individual classifier performance, negating any potential gains.

As with the $k-1$-of-k training results, using approximately two thirds of the data proved to be insufficient. To alleviate this concern the size of the resampled set was doubled. We expect higher correlations between classifiers, because the amount of correlation is controlled through the size of the resampled set. Table XI shows classification results obtained when the training set was doubled through resampling. In order to make sure amount of training does not alter

[3]The hypothesis that input decimated error with $N = 8$ is lower than the base error with $N = 7$ is rejected at the $\alpha = .05$ level but not $\alpha = .1$ level.

Table X: Correlations between Resampled Partitions.

		RS1	RS2	RS3	RS4	RS5	RS6	RS7
MLP	RS1	0.89	0.67	0.70	0.68	0.68	0.64	0.67
	RS2		0.89	0.74	0.67	0.62	0.63	0.57
	RS3			0.90	0.70	0.65	0.65	0.64
	RS4				0.89	0.70	0.67	0.62
	RS5					0.90	0.63	0.68
	RS6						0.88	0.55
	RS7							0.89
RBF	RS1	0.82	0.49	0.56	0.51	0.59	0.57	0.56
	RS2		0.79	0.55	0.55	0.57	0.52	0.50
	RS3			0.78	0.54	0.58	0.56	0.56
	RS4				0.77	0.54	0.54	0.54
	RS5					0.81	0.56	0.59
	RS6						0.80	0.51
	RS7							0.82

the results, the classifiers in this section are trained for half as many epochs as those in the previous section (since in each epoch twice as many patterns are seen). The correlation values between classifiers trained on different resampled training sets are shown in Table XII.

Table XI: Resampling Results (training set size doubled).

Classifier(s)		Ave		Bagging	
	N	Error	σ	Error	σ
MLP	1	8.02	0.57		
	3	7.36	0.34	7.65	0.41
	5	7.31	0.13	7.35	0.28
	7	7.27	0.17	7.38	0.31
RBF	1	6.70	0.76		
	3	5.56	0.36	6.11	0.54
	5	5.29	0.28	5.51	0.36
	7	5.07	0.25	5.34	0.33

The correlations among classifiers still shows a reduction for the different partitions. However, because this reduction was not obtained at the cost of a significant decrease in the individual classifier performance, the combining results are promising. Indeed, in this case the RBF network results showed mild improvements over simply combining multiple runs of the classifiers trained on the full data.

Table XII: Correlations between Resampled Partitions (training set size doubled).

		RS1	RS2	RS3	RS4	RS5	RS6	RS7
MLP	RS1	0.88	0.76	0.78	0.80	0.80	0.77	0.78
	RS2		0.91	0.79	0.77	0.77	0.77	0.77
	RS3			0.89	0.81	0.80	0.78	0.80
	RS4				0.89	0.89	0.78	0.81
	RS5					0.88	0.78	0.81
	RS6						0.88	0.82
	RS7							0.91
RBF	RS1	0.76	0.60	0.62	0.62	0.62	0.65	0.63
	RS2		0.76	0.61	0.64	0.64	0.65	0.60
	RS3			0.76	0.64	0.64	0.63	0.60
	RS4				0.76	0.76	0.63	0.60
	RS5					0.77	0.63	0.60
	RS6						0.79	0.64
	RS7							0.77

5.1.5 Weighted Experts

Since this approach is based on the mixture of experts framework as discussed in Section 4.4, we first tried a network of eight *linear* experts trained with the EM algorithm (Ramamurti and Ghosh, 1996). A softmax based gating network, which partitions the input space using soft hyperplanes (Jacobs et al., 1991), gave poor results; a localized gating network initialized through k-means clustering (Xu et al., 1995) fared better, providing an average test set error rate of 8.97%.

Table XIII shows the performance when (i) three and (ii) seven experts (as described in Section 4.4 were used. The combining of three experts failed to surpass the base results. For seven experts, the weighted combining of MLPs provides satisfactory results, whereas the weighed combining of RBFs does not. Tables XIV and XV show the correlations between the experts. As expected, these correlations vary greatly, since the experts are trained on different spatial partitions.

The large disparity among individual experts (large standard deviations) is to be expected, because each expert focuses on a smaller section of the input space. However, because the combining result relies on a weighted average, and the weights depend on the classifier's expertize in a given region, these disparities do not disrupt the performance of the combiner.

Table XIII: Combining Weighted Experts.

Type of Experts	Number of Experts	Error	σ
MLP	1	9.00	1.25
	3	7.58	0.49
	7	7.00	0.30
RBF	1	8.34	1.23
	3	7.33	0.45
	7	6.59	0.31

Table XIV: Correlations between Three Weighted Experts.

		WE1-3	WE2-3	WE3-3
MLP	WE1-3	0.90	0.81	0.68
	WE2-3		0.91	0.72
	WE3-3			0.92
RBF	WE1-3	0.77	0.65	0.37
	WE2-3		0.76	0.47
	WE3-3			0.74

5.2 Results on Proben1 Benchmarks

In this section, we discuss results obtained from the Proben1 benchmark set[4] (Prechelt, 1994). The data sets that were included in this study are the CANCER1, GLASS1, and GENE1 sets, and the name and number combinations correspond to a specific training/validation/test set split[5]. Note that in all cases, half the data is used for training, and a quarter of the data is used for validating and testing respectively. A more detailed description of these data sets, as well as comparative studies between the Proben1 results, individual classifiers and ensembles of classifiers is available in (Tumer and Ghosh, 1995c).

CANCER1 is based on breast cancer data, obtained from the University of Wisconsin Hospitals, from Dr. William H. Wolberg (Mangasarian et al., 1990; Wolberg and Mangasarian, 1990). This set has 9 inputs, 2 outputs and 699 patterns, of which 350 are used for training. GENE1 is based on intron/exon boundary detection, or the detection of splice junctions in DNA sequences (Noordewier et al., 1991; Towell and Shavlik, 1992). 120 inputs are used to determine whether a DNA section is a donor, an acceptor or neither. There are 3175 examples, of which

[4] Available at URL *ftp://ftp.ira.uka.de/pub/papers/techreports/1994/1994-21.ps.Z.*

[5] We are using the same notation as in the Proben1 benchmarks.

Table XV: Correlations between Seven Weighted Experts.

		WE1-7	WE2-7	WE3-7	WE4-7	WE5-7	WE6-7	WE7-7
	WE1-7	0.93	0.43	0.58	0.44	0.43	0.49	0.48
	WE2-7		0.85	0.45	0.70	0.77	0.78	0.76
	WE3-7			0.97	0.52	0.48	0.50	0.50
MLP	WE4-7				0.89	0.76	0.70	0.70
	WE5-7					0.90	0.78	0.79
	WE6-7						0.90	0.87
	WE7-7							0.90
	WE1-7	0.73	0.22	0.46	0.32	0.32	0.33	0.34
	WE2-7		0.74	0.23	0.48	0.54	0.51	0.53
	WE3-7			0.81	0.51	0.39	0.32	0.35
RBF	WE4-7				0.75	0.54	0.49	0.54
	WE5-7					0.77	0.59	0.62
	WE6-7						0.75	0.68
	WE7-7							0.75

1588 are used for training. The GLASS1 data set is based on the chemical analysis of glass splinters. The 9 inputs are used to classify 6 different types of glass. There are 214 examples in this set, and 107 of them are used for training.

Before discussing the results, let us note that general classification problems can be divided into the following cases according to their complexity: (i) individual classifiers provide satisfactory results, and combining is used to reduce the standard deviation; (ii) combining provides all the potential improvements; (iii) combining provides performance gains, but further improvements are possible using the methods discussed in this paper.

The CANCER1 data falls into the first category. The error rate for a single MLP is 0.69%, whereas the error rate for an ensemble of MLPs is 0.60% (for N=3,5,7). These error rates represent an average of 1 or 2 errors per classifier, since there are 174 test patterns. In this case, combining only increases the reliability of the classifier, by reducing the standard deviation of the error.

The GLASS1 set belongs to the second category where combining improves the performance, but correlation reduction methods fail to provide added improvements. In this case, the limitation comes from the small sample size of the data. Since there are only 107 samples for a 6 class problem, each class only has about 17 examples (on the average). Any further reduction of the training set size, e.g., though cross validation or resampling, causes significant increases in the individual classifier error rates, negating any potential gains through combining.

The GENE1 data provides a good example of the third type of problem. Combining improves the results, yet there is potential for further improvements. For these experiments, we selected an MLP consisting of a single hidden layer network with 20 hidden units, and an RBF network with 15 kernels. Table XVI shows the base performance for the GENE1 data, along with the Proben1 results[6], and the estimated correlations. Table XVII shows the results of combining ensembles of MLPs and RBF networks trained on resampled training sets, the corresponding bagging results and the estimated correlations[7]. For these experiments, the architecture and sizes of the MLP and RBF networks were identical to those of the base experiments, and the training set size was doubled through resampling. The improvements are minimal for MLP ensembles, but significant for RBF ensembles for both $N = 5$ and $N = 10$.

Table XVI: Combining Results for GENE1.

Classifier(s)	N	Ave Error	σ	Estimated Correlation
MLP	1	13.47	0.44	0.74
	5	12.23	0.39	
	10	12.16	0.22	
RBF	1	14.34	0.52	0.96
	5	13.93	0.17	
	10	13.83	0.22	
Proben1:				
pivot		15.05	0.89	
no-shortcut		16.67	3.75	

6 Discussion

Major steps in the design of any ensemble based system include extracting the relevant features, determining the individual classifier architectures, and selecting the classifier training and com-

[6]Proben1 results reported here correspond to the "pivot" and "no-shortcut" architectures, discussed in (Prechelt, 1994).

[7]There are two correlations, the within sample ($N = 1$) and between sample ($N = 5, 10$) correlations, respectively.

Table XVII: Resampling/Combining Results for GENE1.

Classifier(s)		Ave		Bagging		Estimated
	N	Error	σ	Error	σ	Correlation
	1	13.97	0.43			0.76
MLP	5	12.09	0.49	12.30	0.48	0.68
	10	12.01	0.38	12.14	0.48	
	1	14.82	1.12			0.98
RBF	5	13.59	0.17	13.75	0.14	0.87
	10	13.00	0.21	13.24	0.14	

bining methodologies. In this paper, we focused on the effect of using non-identical training sets on the overall classification performance of a combiner system. In theory, reducing the correlation among classifiers that are combined increases the ensemble classification rates. In practice however, since each classifier uses a subset of the training set, individual classifier performance can deteriorate, thus offsetting any potential gains at the ensemble level. This phenomenon is even more pronounced for small data sets where reducing the training set size can lead to significant problems for individual classifiers.

The methods that we investigated can be categorized into three groups:

1. Statistical partitioning of the training sets, resulting in a change in which *patterns* a classifiers sees ($k-1$-of-k training and resampling).

2. Statistical partitioning of the feature space, resulting in the modification of which *aspects* of each pattern is seen by each classifier (input decimation).

3. Spatial partitioning of patterns using *proximity* of patterns in input space, in order to reduce the complexity of the task each individual classifier needs to perform (weighted expert combining).

Neither feature space partitioning, nor spatial partitioning provided significant improvements over the base results. Furthermore, both methods are difficult to fine-tune, as a small change in the design step (say, modifying the number of inputs that is decimated) leads to large changes in combiner performance. Statistical partitioning of the training sets on the other hand yielded promising results.

A particularly interesting observation is that for $k-1$-of-k training and resampled set combining, the RBF networks improved upon the base results. That the RBF network ensembles outperform their MLP counterparts is not surprising when one considers that RBFs are more locally "tuned" to their training sets. Indeed, the selection of the kernel locations directly influences the performance of an RBF network.

In our experiments, the final classification decision of each variation that was studied fell within a very narrow range. These results highlight the difficulties in obtaining significant improvements over combiners which use classifiers trained on all the available training data. It is therefore important to reduce the correlations without increasing error rates. If there is no shortage of data this may be possible through $k-1$-of-k training or resampling. If the amount of data is limited, as it often is the case, however, the performance gains may not justify the increased computational cost of training a large set of classifiers on slightly modified versions of the original data. An exception occurs when cross-validation is in any case being used for model selection or for estimating generalization. Here several networks trained on slightly different data sets are obtained as a byproduct, so combining them adds little to the computational burden.

Acknowledgements: The authors would like to thank Viswanath Ramamurti for providing the EM algorithm result and the data partitions in Section 4.4.

References

Al-Ghoneim, K. and Vijaya Kumar, B. V. K. (1995). Learning ranks with neural networks (Invited paper). In *Applications and Science of Artificial Neural Networks, Proceedings of the SPIE*, volume 2492, pages 446–464.

Ali, K. M. and Pazzani, M. J. (1995). On the link between error correlation and error reduction in decision tree ensembles. Technical Report 95-38, Department of Information and Computer Science, University of California, Irvine.

Battiti, R. and Colla, A. M. (1994). Democracy in neural nets: Voting schemes for classification. *Neural Networks*, 7(4):691–709.

Baxt, W. G. (1992). Improving the accuracy of an artificial neural network using multiple differently trained networks. *Neural Computation*, 4:772–780.

Benediktsson, J., Sveinsson, J., Ersoy, O., and Swain, P. (1994). Parallel consensual neural networks with optimally weighted outputs. In *Proceedings of the World Congress on Neural Networks*, pages III:129–137. INNS Press.

Breiman, L. (1993). Stacked regression. Technical Report 367, Department of Statistics, University of California, Berkeley.

Breiman, L. (1994). Bagging predictors. Technical Report 421, Department of Statistics, University of California, Berkeley.

Drucker, H., Cortes, C., Jackel, L. D., LeCun, Y., and Vapnik, V. (1994). Boosting and other ensemble methods. *Neural Computation*, 6(6):1289–1301.

Efron, B. (1982). *The Jackknife, the Bootstrap and Other Resampling Plans.* SIAM, Philadelphia.

Efron, B. (1983). Estimating the error rate of a prediction rule. *Journal of the American Statistical Association*, 78:316–333.

Friedman, J. H. (1994). An overview of predictive learning and function approximation. In Cherkassky, V., Friedman, J., and Wechsler, H., editors, *From Statistics to Neural Networks, Proc. NATO/ASI Workshop*, pages 1–55. Springer Verlag.

Ghosh, J., Beck, S., and Chu, C. (1992). Evidence combination techniques for robust classification of short-duration oceanic signals. In *SPIE Conf. on Adaptive and Learning Systems, SPIE Proc. Vol. 1706*, pages 266–276.

Ghosh, J. and Tumer, K. (1994). Structural adaptation and generalization in supervised feedforward networks. *Journal of Artificial Neural Networks*, 1(4):431–458.

Ghosh, J., Tumer, K., Beck, S., and Deuser, L. (1996). Integration of neural classifiers for passive sonar signals. In Leondes, C., editor, *Control and Dynamic Systems—Advances in Theory and Applications*, volume 77, pages 301–338. Academic Press.

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1000.

Hashem, S. and Schmeiser, B. (1993). Approximating a function and its derivatives using MSE-optimal linear combinations of trained feedforward neural networks. In *Proceedings of the Joint Conference on Neural Networks*, volume 87, pages I:617–620, New Jersey.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. Macmillan, New York.

Ho, T. K., Hull, J. J., and Srihari, S. N. (1994). Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):66–76.

Jacobs, R. (1995). Method for combining experts' probability assessments. *Neural Computation*, 7(5):867–888.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3:78–88.

Jain, A., Dubes, R., and Chen, C. (1987). Bootstrap techniques for error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:628–633.

Krogh, A. and Vedelsby, J. (1995). Neural network ensembles, cross validation and avtive learning. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, *Advances in Neural Information Processing Systems-7*. M.I.T. Press.

Lee, J., Hwang, J.-N., Davis, D., and Nelson, A. (1991). Integration of neural networks and decision tree classifiers for automated cytology screening. In *Proceedings of the International Joint Conference on Neural Networks, Seattle*, pages I:257–262.

Levin, E., Tishby, N., and Solla, S. A. (1990). A statistical approach to learning and generalization in layered neural networks. *Proc. IEEE*, 78(10):1568–74.

Lincoln, W. and Skrzypek, J. (1990). Synergy of clustering multiple back propagation networks. In Touretzky, D., editor, *Advances in Neural Information Processing Systems-2*, pages 650–657. Morgan Kaufmann.

Lippmann, R. P. (1995). Personal communication.

Mangasarian, O. L., Setiono, R., and Wolberg, W. H. (1990). Pattern recognition via linear programming: Theory and application to medical diagnosis. In Coleman, T. F. and Li, Y., editors, *Large-Scale Numerical Optimization"*, pages 22–30. SIAM Publications.

Meir, R. (1995). Bias, variance, and the combination of estimators; the case of least linear squares. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, *Advances in Neural Information Processing Systems-7*. M.I.T. Press.

Moody, J. (1994). Prediction risk and architecture selection for neural networks. In Cherkassky, V., Friedman, J., and Wechsler, H., editors, *From Statistics to Neural Networks, Proc. NATO/ASI Workshop*, pages 143–156. Springer Verlag.

Noordewier, M. O., Towell, G. G., and Shavlik, J. W. (1991). Training knowledge-based neural networks to recognize genes in DNA sequences. In Lippmann, R., Moody, J., and Touretzky, D., editors, *Advances in Neural Information Processing Systems-3*, pages 530–536. Morgan Kaufmann.

Opitz, D. W. and Shavlik, J. W. (1996). Generating accurate and diverse members of a neural-network ensemble. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems-8*. M.I.T. Press. In Press.

Perrone, M. and Cooper, L. N. (1993a). Learning from what's been learned: Supervised learning in multi-neural network systems. In *Proceedings of the World Congress on Neural Networks*, pages III:354–357. INNS Press.

Perrone, M. and Cooper, L. N. (1993b). When networks disagree: Ensemble methods for hybrid neural networks. In Mammone, R. J., editor, *Neural Networks for Speech and Image Processing*, chapter 10. Chapmann-Hall.

Prechelt, L. (1994). PROBEN1 — A set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, D-76128 Karlsruhe, Germany. Anonymous FTP: /pub/papers/techreports/1994/1994-21.ps.Z on ftp.ira.uka.de.

Ramamurti, V. and Ghosh, J. (1996). Advances in using hierarchical mixture of experts for signal classification. In *International Conference on Acoustics, Speech, and Signal Processing*, pages VI:3569–3572, Atlanta.

Richard, M. and Lippmann, R. (1991). Neural network classifiers estimate Bayesian a posteriori probabilities. *Neural Computation*, 3(4):461–483.

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

Ruck, D. W., Rogers, S. K., Kabrisky, M. E., Oxley, M. E., and Suter, B. W. (1990). The multilayer Perceptron as an approximation to a Bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1(4):296–298.

Sollich, P. and Krogh, A. (1996). Learning with ensembles: How overfitting can be useful. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems-8*. M.I.T. Press. In Press.

Stone, M. (1974). Cross-validatory choice and assessment of statistical prediction. *Journal of the Royal Statistical Society*, 36:111–147.

Towell, G. G. and Shavlik, J. W. (1992). Interpretation of artificial neural networks: Mapping knowledge-based neural networks into rules. In Moody, J., Hanson, S., and Lippmann, R., editors, *Advances in Neural Information Processing Systems-4*, pages 977–984. Morgan Kaufmann.

Tumer, K. and Ghosh, J. (1995a). Bayes error rate estimation through classifier combining. Technical Report 96-01-101, The Computer and Vision Research Center, University of Texas, Austin. Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence.

Tumer, K. and Ghosh, J. (1995b). Order statistics combiners for neural classifiers. In *Proceedings of the World Congress on Neural Networks*, pages I:31–34, Washington D.C. INNS Press.

Tumer, K. and Ghosh, J. (1995c). Theoretical foundations of linear and order statistics combiners for neural pattern classifiers. Technical Report 95-02-98, The Computer and Vision Research Center, University of Texas, Austin. (Available from URL *http://www.lans.ece.utexas.edu/* under select publications–tech reports).

Tumer, K. and Ghosh, J. (1996). Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition*, 29(2):341–348.

Twomey, J. M. and Smith, A. E. (1995). Committee networks by resampling. In Dagli, C. H., Akay, M., Chen, C. L. P., Fernández, B. R., and Ghosh, J., editors, *Intelligent Engineering Systems through Artificial Neural Networks*, volume 5, pages 153–158. ASME Press.

Weiss, S. M. and Kulikowski, C. (1991). *Computer Systems That Learn.* Morgan Kaufmann.

Wolberg, W. H. and Mangasarian, O. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. In *Proceedings of the National Academy of Sciences*, volume 87, pages 9193–9196, U.S.A.

Wolpert, D. H. (1990). A mathematical theory of generalization. *Complex Systems*, 4:151–200.

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

Xu, L., Jordan, M. I., and Hinton, G. E. (1995). An alternative model for mixtures of experts. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, *Advances in Neural Information Processing Systems-7*. M.I.T. Press.

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

Yang, J.-B. and Singh, M. G. (1994). An evidential reasoning approach for multiple-attribute decision making with uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(1):1–19.