Input Decimated Ensembles

Kagan Tumer
NASA Ames Research Center
Mail Stop 269-4
Moffett Field, CA 94035
ktumer@mail.arc.nasa.gov

Nikunj C. Oza
NASA Ames Research Center
Mail Stop 269-2
Moffett Field, CA 94035
oza@email.arc.nasa.gov

March 28, 2005

Abstract

Using an ensemble of classifiers instead of a single classifier has been shown to improve generalization performance in many pattern recognition problems. However, the extent of such improvement depends greatly on the amount of correlation among the errors of the base classifiers. Therefore, reducing those correlations while keeping the classifiers' performance levels high is an important area of research. In this article, we explore **input decimation** (ID), a method which selects feature subsets for their ability to discriminate among the classes and uses these subsets to decouple the base classifiers. We provide a summary of the theoretical benefits of correlation reduction, along with results of our method on two underwater sonar data sets, three benchmarks from the Proben1/UCI repositories, and two synthetic data sets. The results indicate that input decimated ensembles outperform ensembles whose base classifiers use all the input features; randomly selected subsets of features; and features created using principal components analysis, on a wide range of domains.

Keywords: Ensembles, dimensionality reduction, combining classifier, feature selection, correlation reduction, classification.

1 Introduction

Using an ensemble of classifiers (also known as combiners or committees) instead of a single classifier has been repeatedly shown to improve generalization performance in many pattern recognition problems [8, 16, 69]. It is well-known that, in order to obtain such improvement, one needs to simultaneously maintain a reasonable level of performance in the base classifiers that constitute the ensemble and reduce their error correlations [28, 43, 48, 63]. There are many

ensemble methods that actively promote diversity (e.g., modify error surface, lower correlations of the outputs) among their base classifiers [48, 57, 63]. Most work in this field, however, focuses on pattern-level selection (e.g., Bagging [8], Boosting [21]). These methods bring about diversity in the base models by training them with different subsets of the training set. One drawback of such methods is that by definition, only a portion of the available data is used during training. This can lead to poor performance, particularly when the data sets are small to begin with. Training the base classifiers using different subsets of features avoids this issue as all the patterns can be used in training while still yielding base model diversity.

Two possible feature selection/extraction methods are Principal Components Analysis (PCA)[30, 51] and random subspace selection [24]. PCA constructs new features such that the data has maximum variability over those features. However, PCA, when used in combining, not only generates the same features for all classifiers in the pool, but also fails to take class information into account. Random subspace selection overcomes the first shortcoming of PCA, but it too does not consider the class labels when generating the feature subsets. Consequently, these two methods do not choose features in a manner that is helpful in the classification task.

In this paper, we present input decimation—a method of choosing different subsets of the original features based on the correlations between individual features and class labels, and training classifiers on those subsets prior to combining. This method not only reduces the dimensionality of the data, but uses this dimensionality reduction to reduce the correlations among the classifiers in an ensemble, thereby improving the classification performance of the ensemble [50, 62, 67].

Our results indicate that input decimation reduces the error up to 90% over single classifiers and ensembles trained on all features, randomly-selected subsets of features, and principal components. While we expected strong ensemble performance, input decimation also provides improvements in the base classifiers in many cases by pruning extraneous or irrelevant features, thus simplifying the learning problem faced by each base classifier. In this study we use the "averaging" combiner[1] for two reasons: (i) despite its simplicity (or perhaps because of it) this combiner has been shown to outperform a wide array of more sophisticated methods [15, 16]; and (ii) by choosing a simple combiner we isolate the effects of input decimation from those of the combining method. Furthermore, pattern-level ensemble methods such as bagging, boosting, and stacking can be used *in conjunction* with input decimation which is a feature-level ensemble method (i.e., input decimation is orthogonal to those methods). Therefore, one can make meaningful comparisons between averaging combiners *with* and *without* input decimation, or say, between bagging *with* and *without* input decimation (not reported in this article), but not between input decimated ensembles and bagging or boosting.

In Section 2, we summarize a theory of classifier ensembles that highlights the connection between correlation among base classifiers and ensemble performance, along with a brief overview of different dimensionality reduction methods. In Section 3 we present the details of the input decimated ensemble, and in Section 4 we provide experimental results on two underwater sonar

[1]The output of the combiner is the average of the outputs of the base classifiers. More details are given in the next section.

data sets, three data sets from the PROBEN1/UCI benchmarks [5, 53], and two synthetic data sets which allow a systematic study of input decimation. We conclude with a discussion on the effectiveness of input decimation under various circumstances along with future research directions in Section 5.

2 Background

Model selection is an important issue in many pattern recognition problems. Neither the selection of the method (e.g., multi-layer perceptron, nearest neighbor algorithm), nor the tuning of that algorithm can yet be fully automated for all problems [14, 19, 22]. The use of ensembles provides partial relief since, by pooling the classifiers before a decision is made, the sensitivity to any single classifier is greatly reduced. Of course, the more similar the classifiers are, the less likely it is that new information will be present in the ensemble, resulting in little more than a "rubber stamping" committee. In this section we first formalize this connection between the correlation among the classifiers' errors and ensemble performance and then discuss various methods that aim to reduce that correlation.

2.1 Correlation and Ensemble Performance

In this article we focus on classifiers that model the *a posteriori* probabilities of the output classes. Such algorithms include Bayesian methods [3], and properly trained feed forward neural networks such as Multi-Layer Perceptrons (MLPs) [55]. We can model the ith output of such a classifier as follows (details of this derivation are in [62, 63]):

$$f_i(x) = P(C_i|x) + \eta_i(x),$$

where $P(C_i|x)$ is the posterior probability of the ith class given pattern x, and $\eta_i(x)$ is the error associated with the ith output. Given an input x, if we have one classifier, we classify x as being in the class i whose value $f_i(x)$ is largest.

Instead, if we use an ensemble that calculates the arithmetic average over the outputs of N classifiers $f_i^m(x)$, $m \in \{1, \ldots, N\}$, then $P(C_i|x)$ is given by:

$$f_i^{ave}(x) = \frac{1}{N} \sum_{m=1}^{N} f_i^m(x) = P(C_i|x) + \bar{\eta}_i(x), \tag{1}$$

where:

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

and $\eta_i^m(x)$ is the error associated with the ith output of the mth classifier.

The variance of $\bar{\eta}_i(x)$ is given by [63]:

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

If we express the covariances in terms of the correlations ($cov(x,y) = corr(x,y)\sigma_x\sigma_y$), assume the same variance $\sigma^2_{\eta_i}$ across classifiers, and use the average correlation factor among classifiers, δ_i, given by

$$\delta_i = \frac{1}{N(N-1)}\sum_{m=1}^{N}\sum_{l\neq m} corr(\eta_i^l(x), \eta_i^m(x)), \tag{2}$$

then the variance becomes:

$$\sigma^2_{\bar{\eta}_i} = \frac{1}{N}\sigma^2_{\eta_i(x)} + \frac{N-1}{N}\delta_i\sigma^2_{\eta_i(x)} = \frac{1+\delta_i(N-1)}{N}\sigma^2_{\eta_i(x)}. \tag{3}$$

Though this reduction in variance is reflected in a reduction in the error of the classifier, only the *model* error, i.e., the error additional to the Bayes error, is reduced.[2] Equation 4 shows the formalization of this concept based on the model error of the ensemble (E^{ave}_{model}) and that of an individual classifier (E_{model}) [62, 63]:

$$E^{ave}_{model} = \left(\frac{1+\delta(N-1)}{N}\right)E_{model} \tag{4}$$

where δ is the correlation among the classifiers' errors, and can be empirically computed [62, 63].

2.2 Correlation Reduction Methods

As summarized above, 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. As equation 4 shows, reducing δ and increasing N are two manners by which the performance of a classifier ensemble can be improved. However, these two solutions are not independent.

Figure 1 illustrates this phenomenon, where the error reduction's dependence on the correlation among the classifiers is displayed as a function of the number of classifiers (based on Equation 4). For example, even though increasing the number of classifiers from 4 to 8 does not

[2]Note that the total error, E_{total}, of a classifier is the sum of the Bayes error, E_{Bayes}, and the model error, E_{model}. In practice this fact can be used to estimate the Bayes error based on the improvements due to ensembles [64].

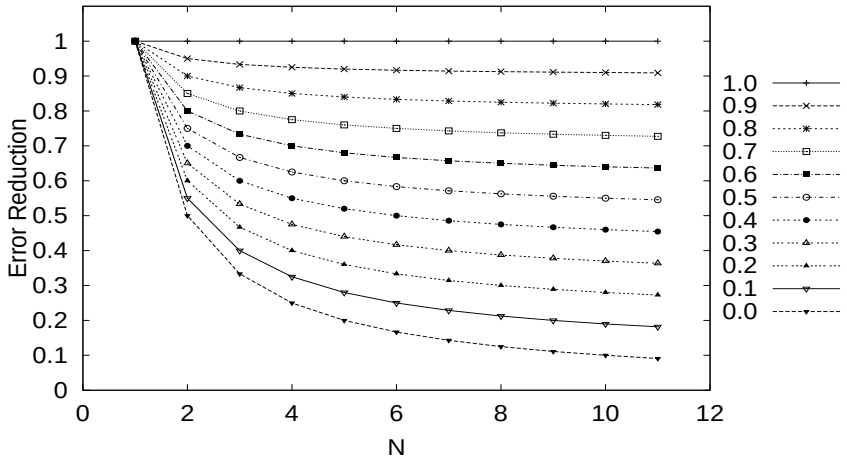

Figure 1: Effect of correlation and number of base classifiers on error reduction. Each line represents a particular average correlation indicated in the legend on the right.

provide any sizable gains when the correlation is 0.9, it provides significant gains if the correlation is 0.1. That is, keeping the correlations low not only provides better error reduction for a given number of classifiers, but provides greater gains when adding classifiers.

To improve ensemble performance, then, one must either actively promote diversity during training or achieve diversity through the selection of the data presented to the base classifier training algorithms. Examples of the former include distorting the output space through error-correcting output codes [17], using principal component analysis on the output space [44], using genetic algorithms to train the classifier [49, 60] or modifying the error function used for training [57]. Examples of the latter include bagging [8], cross-validation partitioning [39, 63] and boosting [21]. The most common data selection methods focus on the "pattern" space, though dimensionality reduction methods which manipulate the feature space can also be used. Feature space methods have the advantage that they do not reduce the number of patterns available for training each classifier. They generally fall into one of two different classes of methods: feature selection or feature extraction.

Feature extraction algorithms such as Principal Components Analysis (PCA) [4, 30, 51] or Independent Component Analysis (ICA) [27] reduce the dimensionality of the data by creating new features. Linear PCA, perhaps the most commonly used feature extraction method, creates new features that are linear combinations of the original features. The aim of PCA, however, is to devise features on which the data shows the highest variability independent of whether those features are useful for classification or not [4]. Furthermore, because all the information present in the initial features is "crammed" into fewer principal components, there is a danger that classifiers trained on the principal components will have higher rather than lower correlations

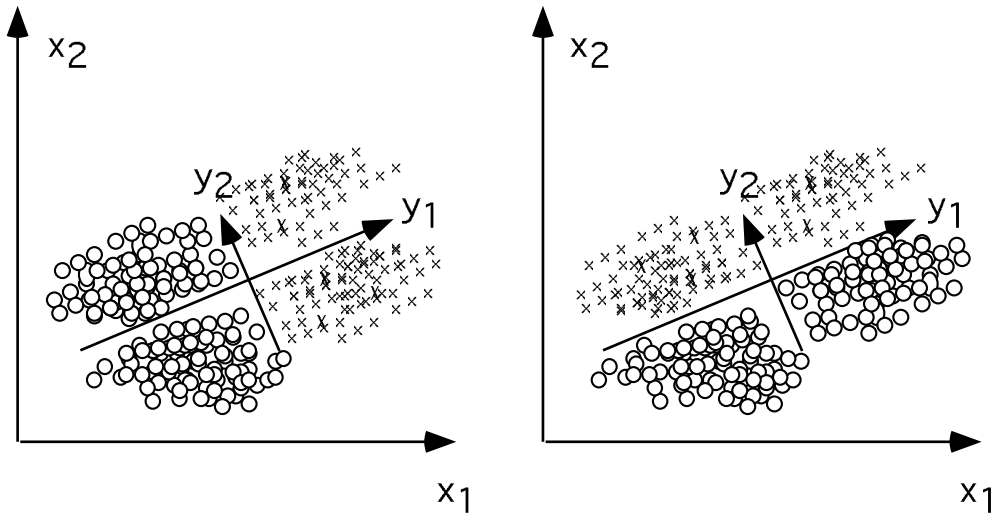

Figure 2: PCA and classification: The first principal component (y_1) can provide a good discriminating feature (left) or a poor one (right), since the class membership information is not used.

among them.

Figure 2 demonstrates the perils of not using class information. The left half of the figure shows a case in which PCA works effectively. In this case the first principal component (y_1) corresponds to the variable with the highest discriminating power. Here, there exists a c such that the examples with $y_1 < c$ are of one class and those with $y_1 > c$ are of the other class. However, there is no similar way to split the examples into two classes along the component y_2. The right half shows the same data distribution (i.e., same principal components) with different class labels. Yet, because the first principal component is not "aligned" with the class labels, selecting this principal component as an input feature is a poor choice in this case. Indeed, an input set consisting of only the first component would provide practically random decisions on this data set.[3] Yet, PCA remains one of the most frequently used dimensionality reduction methods in many classification domains, including medical and space applications [54, 59].

Feature selection algorithms focus on selecting a subset of the original features to present to the classifiers. One example is the Random Subspace (RS) method [24] where random subsets of the original features are presented to the classifiers. However, looking at y_1 and y_2 (assuming those two are the original features) in Figure 2 shows a pitfall of random feature selection. Randomly selecting feature y_1 in the class configuration shown in (a) will lead to satisfactory classification,

[3]There are variations on PCA that use local and/or nonlinear processing to improve dimensionality reduction [12, 32, 33, 46, 47, 58]. Although these methods implicitly account for some class information and therefore are better suited than global PCA methods for classification problems, they do not directly use class information.

whereas randomly selecting feature y_1 in (b) will lead to all discriminating information being lost. Many other feature selection methods use various criteria for deciding the relevance of each feature to the task at hand and choose some subset of the features according to those criteria [2, 6, 7, 9, 18, 29, 42]. The subset selection can be distinct from the learning, which is the case with *filter* methods. However, most of these feature selection methods attempt to choose features that are useful in discriminating across all classes. Using such a method within an ensemble learning scheme would have limited effectiveness since it would choose the same features for every base classifier, leading to relatively small correlation reduction. One exception is to break an L-class problem into $\binom{L}{2}$ two-class problems and perform feature selection within each of those problems [40]. In many real-world problems, there are features that are useful at distinguishing whether a pattern is of one particular class but are not useful at distinguishing among the remaining classes. In the next section we present input decimation, which takes advantage of this fact to reduce both the input dimensionality and the correlation of classifiers in an ensemble.

3 Input Decimated Ensembles

Input Decimation (ID) decouples the classifiers by exposing them to different aspects of the same data. ID trains L classifiers, one corresponding to each class in an L-class problem.[4] For each classifier, the method selects a user-determined number of the input features having the highest absolute correlation to the presence or absence of the corresponding class.[5] However, all the classifiers are trained to discriminate across all L classes in order that they model the posterior probabilities of their classes. The objective of our input selection method is to "weed" out input features that do not carry strong discriminating information for a particular class, and thereby reduce the dimensionality of the feature space to facilitate the learning process. Additionally, the classifiers' features are selected using different relevance criteria, which leads to different feature subsets for each base classifier and a reduction in their correlations.

Let the training set, $\mathcal{D}_{tr}$, take the following form:

$$\mathcal{D}_{tr} : \{(\mathbf{x_1}, \mathbf{y_1}), (\mathbf{x_2}, \mathbf{y_2}), \ldots, (\mathbf{x_m}, \mathbf{y_m})\},$$

where m is the cardinality of $\mathcal{D}_{tr}$, i.e., number of training examples. Each $\mathbf{x_i}$ is an $\|F\|$-dimensional vector (where F is the set of input features). Each $\mathbf{y_i}$ represents the class using a distributed encoding, i.e., it has L elements, where L is the number of classes, $y_{il} = 1$ if example i belongs to class l and $y_{il} = 0$ otherwise. In this study our base classifiers consist of MLPs trained with the backpropagation algorithm.[6]

[4] More generally, one trains nL classifiers where n is a positive integer.

[5] Note that this method requires the problem to have at least three classes. In a two-class problem, features strongly correlated with one class will be strongly anti-correlated with the other class, so the same features would be chosen for both classifiers.

[6] In principle, any learning algorithm that estimates the *a posteriori* class probabilities can be used.

Given such a data set, and a base classifier learning algorithm, input decimated ensemble learning operates as follows:

- For each class $l \in \{1, 2, \ldots, L\}$,
  1. Compute the *decimation coefficient* ν_{jl} for each feature $j \in F$ and class l, given by:
  $$\nu_{jl} \;=\; |corr(\mathbf{x_{ij}}, \mathbf{y_{il}})| \; .$$
  2. Select the n_l features having the highest decimation coefficient, resulting in new feature set F_l. One can either predetermine n_l based on prior information about the data set, or learn the value to optimize performance.
  3. Construct a new training set by retaining only those elements of the $\mathbf{x_i}$'s corresponding to the features F_l but retain all the outputs (the $\mathbf{y_i}$'s).
  4. Run the base classifier learning algorithm on this new training set. Call the resulting classifier f^l.[7]

Given a new example x, we classify it as follows:

- For each class $k \in \{1, 2, \ldots, L\}$, calculate $f_k^{ave}(x) = \frac{1}{L} \sum_{l=1}^{L} f_k^l(x)$, by presenting the proper features F_l of example x to each of the L classifiers f^l.[8]
- Return the class $K = \text{argmax}_k \, f_k^{ave}(x)$.

Note that, in the learning algorithm above, each classifier $f^l(x)$ returns a vector $\left[f_1^l(x) f_2^l(x) \ldots f_L^l(x)\right]$. That is, the superscript l indexes over the L *feature sets* whereas the subscript indexes over the class outputs. Feature set F_l used to train classifier f^l. Note, however that each classifier is trained to discriminate among all L classes.

Furthermore, all the training patterns are used to train all the base classifiers. Fundamentally, input decimation seeks to reduce the correlations among individual classifiers by using different subsets of input features, while pattern-level methods such as bagging and boosting attempt to do so by choosing different subsets of the training patterns.

4 Experimental Results

In this section, we present the results of input decimation on two underwater sonar data sets, three Proben1/UCI benchmark data sets and two synthetic data sets. In all the results reported

[7] If one is training nL classifiers for $n > 1$, then the algorithm calls the base classifier learning algorithm n times to create n classifiers $f^{l1}, f^{l2}, \ldots, f^{ln}$ with feature set F_l. Clearly, since these n classifiers are trained with the same training sets, the base model learning algorithm has to induce some differences among the classifiers. For example, our backpropagation neural network learning algorithm generates pseudo-random initial weights for each network.

[8] If we are instead training nL classifiers for $n > 1$, then we calculate $f_k^{ave}(x) = \frac{1}{nL} \sum_{l=1}^{L} \sum_{i=1}^{n} f_k^{ln}(x)$.

below, the base classifiers consist of Multi-Layer Perceptrons (MLPs) with a single hidden layer trained with the backpropagation algorithm. The learning rate, momentum term, and number of hidden units were experimentally determined. In all cases, we report test set error rates averaged over 20 runs, along with the differences in the mean.[9]

4.1 Passive Sonar Signals

A real world problem with all the characteristics required for a complete study is that of classifying short duration underwater signals obtained from passive sonar signals [13]. Both biological and non-biological phenomena produce such short duration sounds, and experts can determine the cause by studying their pulse signatures or spectrograms. Automating this classification process is difficult because these signals are highly non-stationary, have different spectral characteristics depending on sources and propagation paths and may have significant overlap. A more detailed description of the sonar signals and the difficulty associated with their classification can be found in [23, 62].

The two data sets used for this experiment are both extracted from short–duration passive sonar signals due to four naturally occurring oceanic sources (sound of ice cracking, porpoise and two different whale sounds), which are the four classes for our classification problem. Although there is some complementarity among the data sets, for the purposes of this study we will treat them as different data sets.[10] The first set, SONAR1, consists of four classes and 25 features, including 16 Gabor wavelet coefficients,[11] signal duration and other temporal descriptors and spectral measurements. There were 496 training and 823 test patterns. The second set, SONAR2, consists of four classes and 24 features, including reflection coefficients corresponding to the maximum broadband energy segment using both short and long time windows, signal duration and other temporal descriptors. There were 564 training and 823 test patters. For both data sets, we used an MLP with 50 hidden units.

Table 1 shows the error rates, differences in the mean, and correlation among the base classifiers for both the full feature set and the input decimated set for $N = 4$ (one base classifier per class) and $N = 8$ base classifiers (two per class). The performance of a single base classifier is also shown. In this case, each base classifier had an input decimated set of 22 features for both SONAR1 and SONAR2 after features with little correlation to each output were deleted. 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.

For SONAR1, the deletion of even lowly-correlated inputs reduces the performance of the

[9] That is, for an error with mean μ and variance σ^2, we report the $\mu \pm \sigma/\sqrt{K}$ where K is the number of repetitions (K=20 for experiments reported here). Confidence intervals of desired sensitivity can be obtained directly from the differences in the means.

[10] See [62] for a study where the two data sets were used in conjunction.

[11] Gabor wavelet coefficients provide a multiscale representation that does not assume signal stationarity [11].

Table 1: Ensemble Performance on both sonar data.

		Full Feature Set		Input Decimation	
	N	Error Rate	δ	Error Rate	δ
SONAR1	1	7.47 ± .10		8.38 ± .15	
	4	7.05 ± .07	.89	7.10 ± .07	.68
	8	7.17 ± .05		6.99 ± .06	
SONAR2	1	9.95 ± .16		9.73 ± .16	
	4	9.26 ± .15	.76	8.80 ± .06	.72
	8	8.94 ± .11		8.62 ± .06	

base classifier significantly.[12] However, due to the correspondingly large reduction in the error correlation, input decimated ensembles perform at the level of the full feature set for $N = 4$, and provide statistically significant gains for $N = 8$ classifiers (at the $\alpha = .05$ level).

For SONAR2, the gains are more significant in that even the input decimated base classifier improves slightly upon the full featured base classifier, allowing for sizable gains by the input decimated ensemble. This is achieved in spite of the relatively modest drop in the error correlation among the base classifiers. Note that the gains due to ID in this domain are quite modest. In fact, because the dimensionality of the data is low and each feature has particular significance this data set is not well suited for input decimation. As such, it is remarkable that improvements were obtained at all.

4.2 Proben1/UCI Benchmarks

In the SONAR data presented above each feature carried a significant amount of discriminating information. In fact, because each feature was carefully extracted from the raw data, one should not have expected much improvement through input decimation. In this section we perform a more detailed analysis on three benchmark data sets where we gradually decrease the dimensionality to, in some cases, 5% of the original number of features. On these benchmark sets, we expect this more extreme case of input decimation to expose the strengths and weaknesses of this method.

The three data sets from the UCI/PROBEN1 benchmarks [5, 53] selected for this study were: The Gene data set from PROBEN1 (i.e., using train/test split from PROBEN1); the Splice junction gene sequences from the UCI Repository; and Satellite Image data sets (Statlog version) from the UCI Repository. The Gene data set has 120 input features and three classes [45, 53]. The MLP has a single hidden layer of 20 units, a learning rate of 0.2 and a momentum term of 0.8. The Splice data consists of 60 input features and three classes [5]. Here we selected an MLP with a single hidden layer composed of 120 units, a learning rate of 0.05, and a momentum

[12] We use a t-test with $\alpha = 0.05$ to compare the classifiers in this paper.

term of 0.1. The Satellite Image data set has 36 input features and 6 classes [5]. We selected an MLP with a single hidden layer of 50 units, and a learning rate and momentum term of 0.5. The ensembles consisted of three classifiers for Gene and Splice and six classifiers for Satellite Image—the same as the number of classes.

Figures 3-5 show the classification performance and classifier correlations for all three data sets, averaged over 20 runs.[13] For clarity we omit the error bars, since they ranged from 0.05 to 0.25% and as such were smaller than the symbols representing the data points. Note that the rightmost point in each graph (e.g., the points corresponding to 120 features for the Gene data set) shows the performance when using the full set of features. For the Gene data, the full feature ensemble is significantly more accurate than the single classifier, while for the Satellite Image and Splice data sets, the full feature ensemble is only marginally more accurate.

4.2.1 Gene Data

For the Gene data, the input decimated ensemble with 5 inputs performed worse than the original ensemble with all the original 120 inputs. The decimated ensembles with 10 20, 30, 40, 50, and 60 inputs performed significantly better than the original ensemble while the decimated ensembles with 70 and more inputs performed comparably to the original ensemble. All the decimated ensembles significantly outperformed their PCA and RS counterparts (i.e., with the same number of inputs).

The performance of the PCA ensemble is relatively stable and inferior to Input Decimation Ensembles (IDE) across the board. This is consistent with the fact that principal components are not necessarily good discriminants. Furthermore, adding principal components beyond the first few has little effect on the classification performance. Finally, the performance of the ensemble with RS improves in random increments (e.g., step-like mini "jumps") with the addition of features, depending on how relevant those features are. On the whole RS ensemble do not provide good solutions for this problem.

4.2.2 Splice Junction Data

In the Splice data experiments, *all* the decimated feature-based ensembles significantly outperformed both the original ensemble and the PCA-based ensembles. Random feature-based ensembles performed somewhat better here than in the Gene data set. With 40 and more features, it was competitive to input decimation. However, the best performing predictor overall is clearly the input-decimated ensemble with 10 inputs per classifier. What is particularly notable in this case is that a reduction of dimensionality based on PCA has a strong negative impact on the classification performance. With 20 principal components for example, the performance of the single classifiers drops by 7% relative to the single classifier with all the input features, whereas the performance of the ID single classifier increases by 3%.

[13]For each dataset, we ran random subspace selection five times but show the results of only one run because the variations among the runs were small.

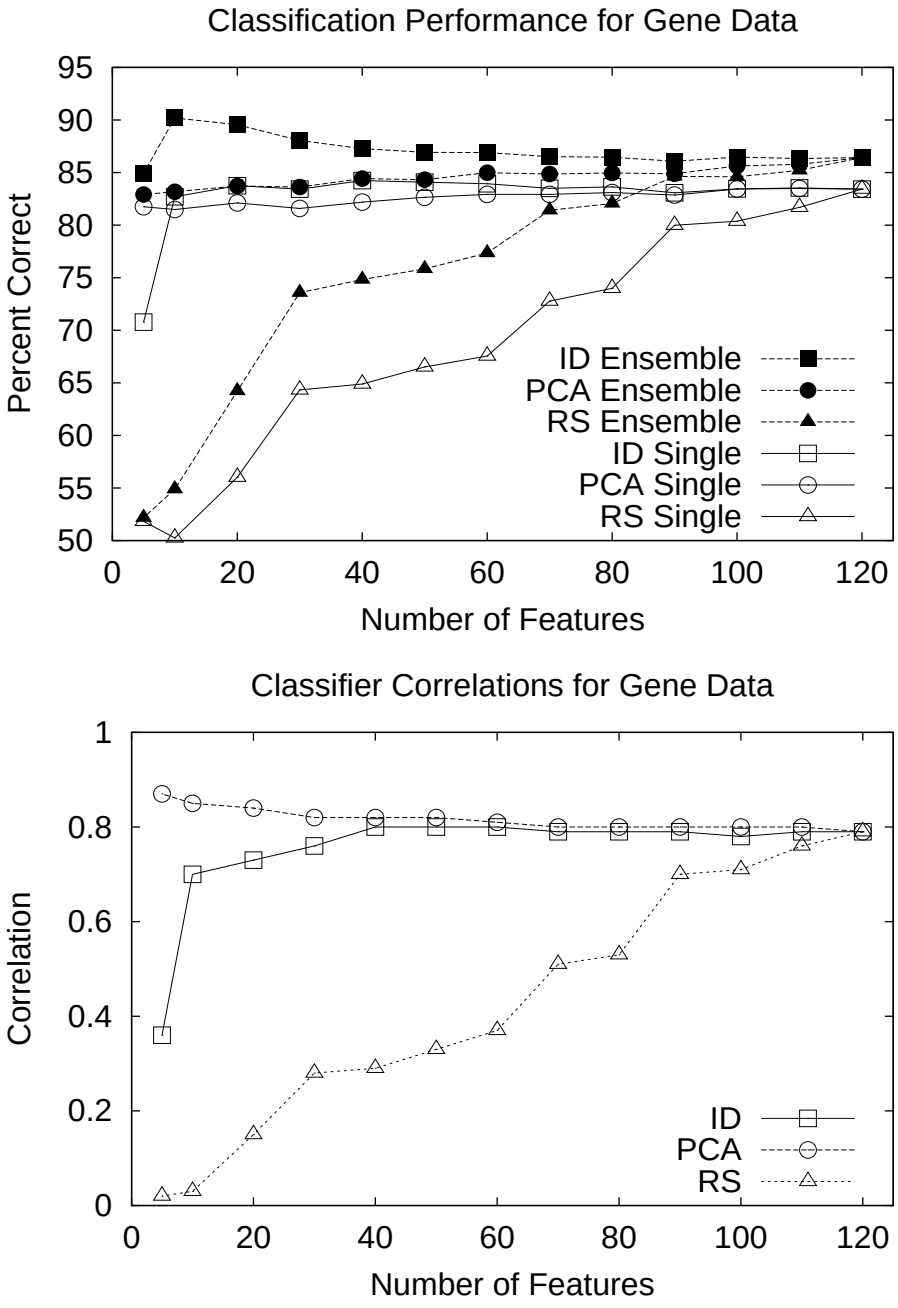

Figure 3: Performance (a) and Correlations (b) for the Gene data set.

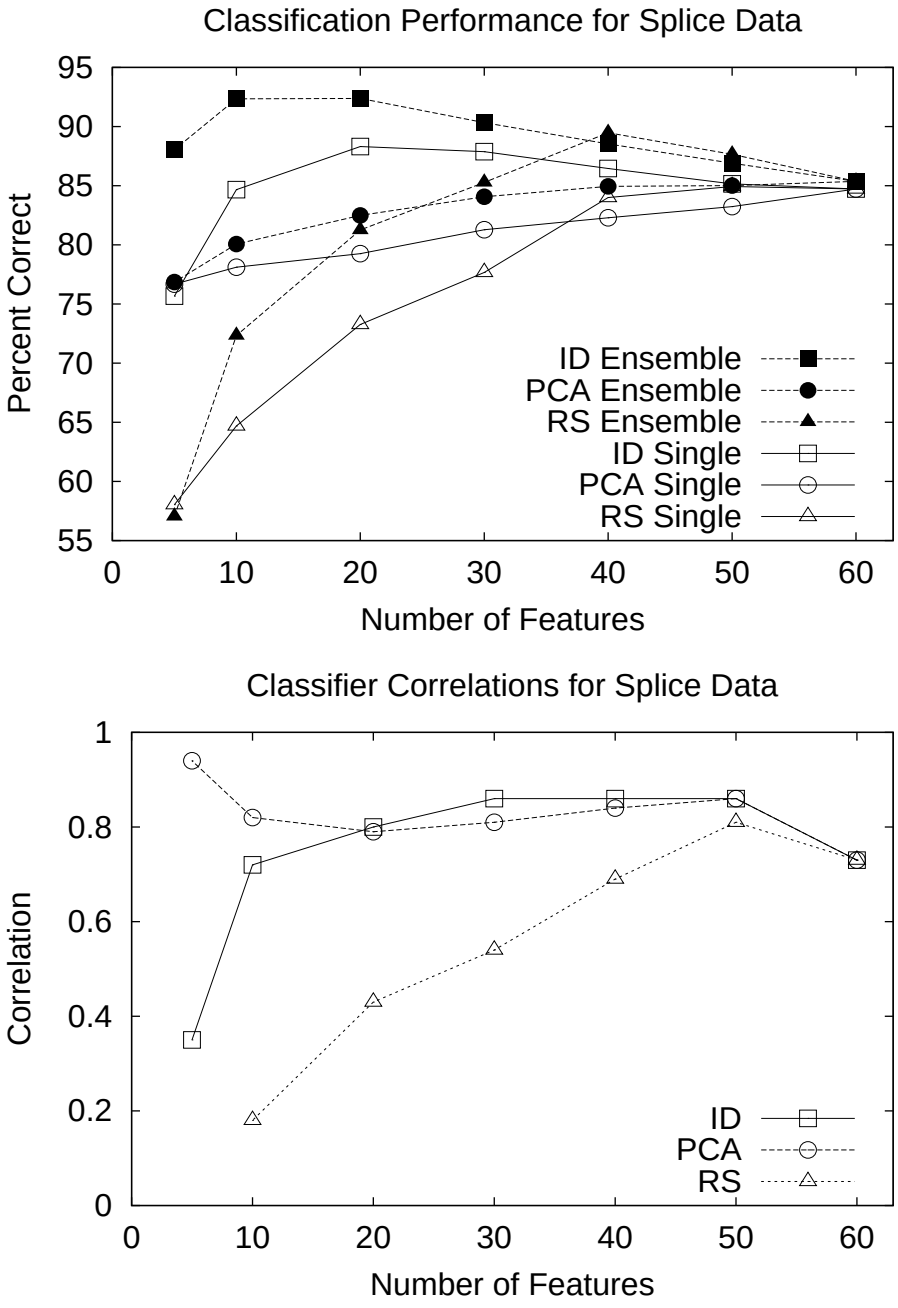

Figure 4: Performance (a) and Correlations (b) for the Splice data set.

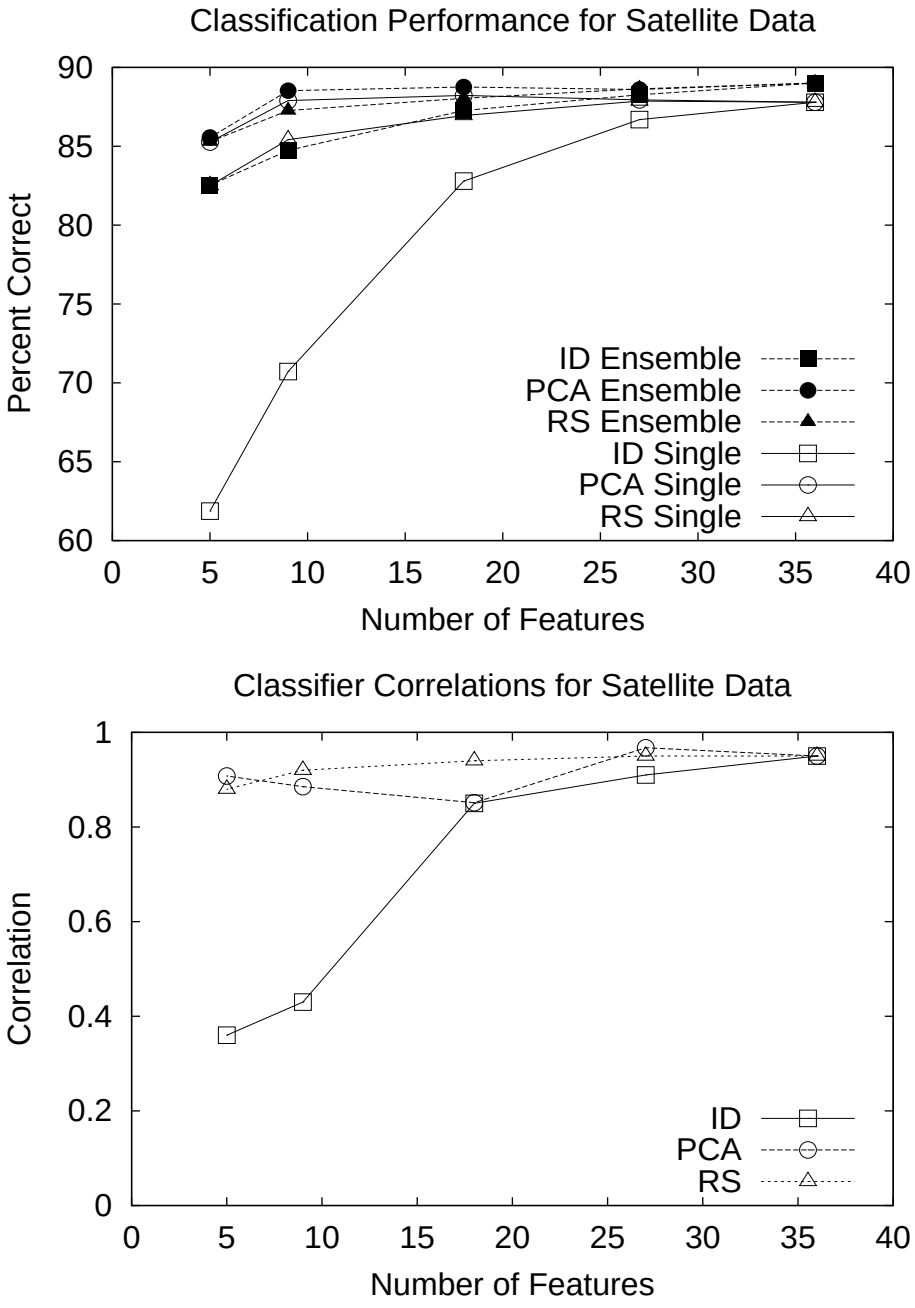

Figure 5: Performance (a) and Correlations (b) for the Satellite data set.

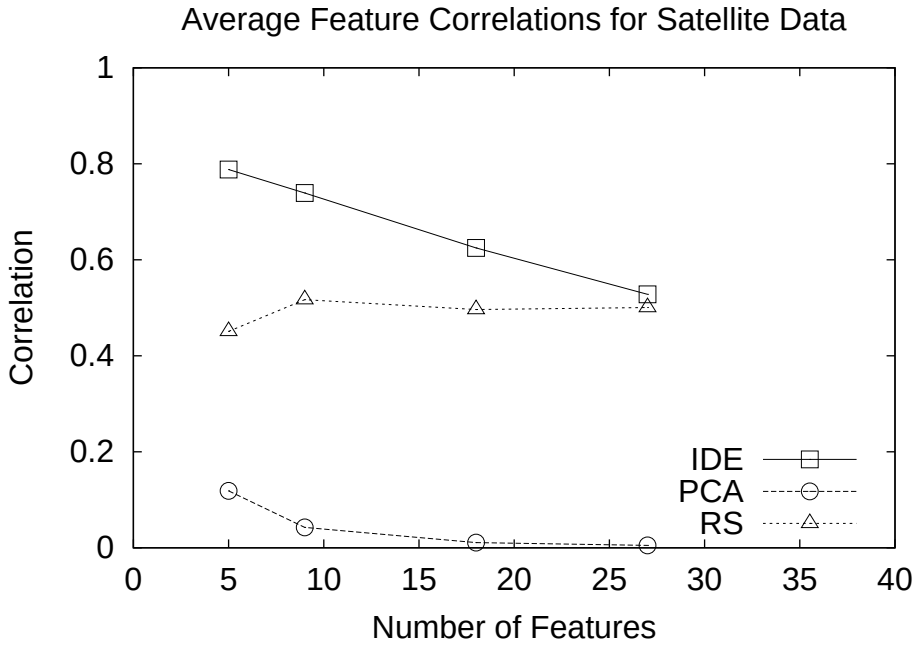

Figure 6: Average Correlation of Features in Ensembles for Satellite Image Data.

The improvement of the performance of the single classifiers due to decimation is an initially surprising aspect of these experiments since one may not expect to find too many "irrelevant" features in these real data sets. However, an analysis shows that the inputs that were decimated were in fact providing "noise" to the classifier. Although it is theoretically true that the classifier with more information will do at least as well as the classifier with less information, in practice with only a limited amount of data, extracting the "correct information" can be difficult for such classifiers, causing them to perform worse than their counterparts with less information.

4.2.3 Satellite Image Data

On the Satellite Image data however, the input decimated ensembles did not perform well. The ID performance dropped systematically as more features were removed. Both the PCA and RS ensembles outperformed IDE's. Because the single ID classifiers performed worse than the PCA and RS single classifiers, we examined the features that were chosen in each ensemble.

Figure 6 shows the average correlations among the features chosen for the base classifiers in the three types of ensembles (previous figures showed the correlation among the *outputs* of the classifiers, whereas Figure 6 looks at the correlation among the selected *input* features). The features selected by ID have a much higher correlation among themselves relative to RS and PCA ensembles, especially for smaller numbers of inputs. This means that ID often selects features

with high correlations to the class even though they may be highly correlated with one another, and therefore redundant. Random feature selection does not fall into this trap since it does not consider correlations at all. PCA's correlations are the lowest because it creates features specifically designed to have low correlations among each other.

A similar look at the Gene and Splice data sets reveals that the correlations among the chosen features for IDE, Random, and PCA ensembles are much closer to one another. In such a case, ID's use of class information gives it a significant advantage. Among the three Proben1/UCI data sets that we explored, Satellite Image is the one with the lowest dimensionality, and contains features that are redundant and relevant to all the classes, and are therefore always selected by ID. This feature set exposes a weakness of ID as currently implemented and shows three things: (i) in order to take advantage of input decimation, the initial dimensionality has to be high, as there are likely to be more irrelevant features that can be removed; (ii) if there are features that have significant meaning, they need to be included in the feature set regardless of their correlation to the particular output; and (iii) redundant features should not be chosen.

To further understand why PCA-based ensembles performed so well on this data set, we investigated the meaning of each input feature. We observed that consecutive groups of four features in the satellite image data set correspond to spectral values for a given pixel. In examining the eigenvalues and eigenvectors, we found that the highest eigenvalue contained 91.6% of the total energy (sum of the eigenvalues), and the corresponding eigenvector was a simple linear combination of the four spectral values across all the pixels. PCA ensembles perform well in this case, because the top few principal components provide very good discriminative features (i.e., the data "looks" like that in Figure 2(a)).

To remedy the shortcoming of ID as implemented (selecting features that were highly correlated with each other), we explored a variant on the decimation coefficient ν discussed in Section 3, that selects features having a combination of high absolute correlation with the output and low correlations with the features already selected. In particular, when selecting the $k+1$st feature for the lth classifier, we select the original feature j (among those not already selected for the lth classifier) maximizing $\nu_{jl}^{corrected}$, given by:

$$\begin{aligned} \nu_{jl}^{corrected} &= \nu_{jl} + \sum_{m \in F_{lk}} (1 - |corr(\mathbf{x_{ij}}, \mathbf{x_{im}})|) \\ &= |corr(\mathbf{x_{ij}}, \mathbf{y_{il}})| + \sum_{m \in F_{lk}} (1 - |corr(\mathbf{x_{ij}}, \mathbf{x_{im}})|) \end{aligned}$$

where F_{lk} is set of the first k features selected for the lth base classifier and i ranges over all the patterns so that, for example $corr(\mathbf{x_{ij}}, \mathbf{y_{il}})$ is the correlation between the jth feature and the lth output in the training set. The first term in the above sum is ν_{jl}, the decimation coefficient. The second term penalizes those features that are highly correlated with the k features F_{lk} that have already been selected.

Figure 7 gives the results of using this metric (referred to as "autocorrelation" in the figure)

as well as the results of ID and PCA repeated for comparison. We omit the results of RS ensembles to avoid overcrowding the figure. The autocorrelation method does not reach the level of performance achieved by PCA; however, it significantly improves upon IDE's for low numbers of inputs. The base classifiers' performances are much higher for IDE with autocorrelation (IDEAC) than for IDE because fewer redundant features are used.

Table 2: Satellite Image: Number of Features Chosen

Base Classifier	IDE		IDE AC	
	Total	Common	Total	Common
5	18	0	15	0
9	28	0	18	1
18	36	0	25	13
27	36	7	30	24

However, the correlations are also much higher for IDEAC than for IDE, and Table 2 provides an explanation for this. For each number of features used by each base classifier (the first column of the table), this table gives four quantities. The second column states the number of original features (out of 36 available features) used in at least one of the six base classifiers in IDE. The third column states the number of original features used in all six base classifiers. The fourth and fifth columns give the same quantities for IDE with autocorrelation. For example, in the case where each base classifier used 9 inputs, the IDE's base classifiers used 28 out of the 36 original features, and none of them were used by all the base classifiers. The IDEAC's base classifiers, on the other hand, used only 18 of the original features, and one of them was used by all six base classifiers. Compared to IDE, IDEAC tends to use fewer total features because several features that are redundant relative to others that have higher correlation with the classes remain unused. Fewer total features across the same number of base classifiers naturally leads to a greater number of common features across those base classifiers. This causes higher base classifier correlations. Note that IDE base classifiers perform worse, but get a bigger boost from combining, which is consistent with Equation 4.[14].

4.3 Synthetic Data

In the previous two subsections we applied ID to real and benchmark data sets to show the range of applicability and to stress-test it with real systems. In this section we construct synthetic data sets to enable us to study the properties of input decimated ensembles in a more systematic manner. To that end we use the following two synthetic data sets:

[14] As mentioned above, this method is a preliminary attempt at removing redundant features; it simply shows the feasibility of using feature-feature correlations in addition to feature-output correlations (see Section 5 for a discussion and future directions)

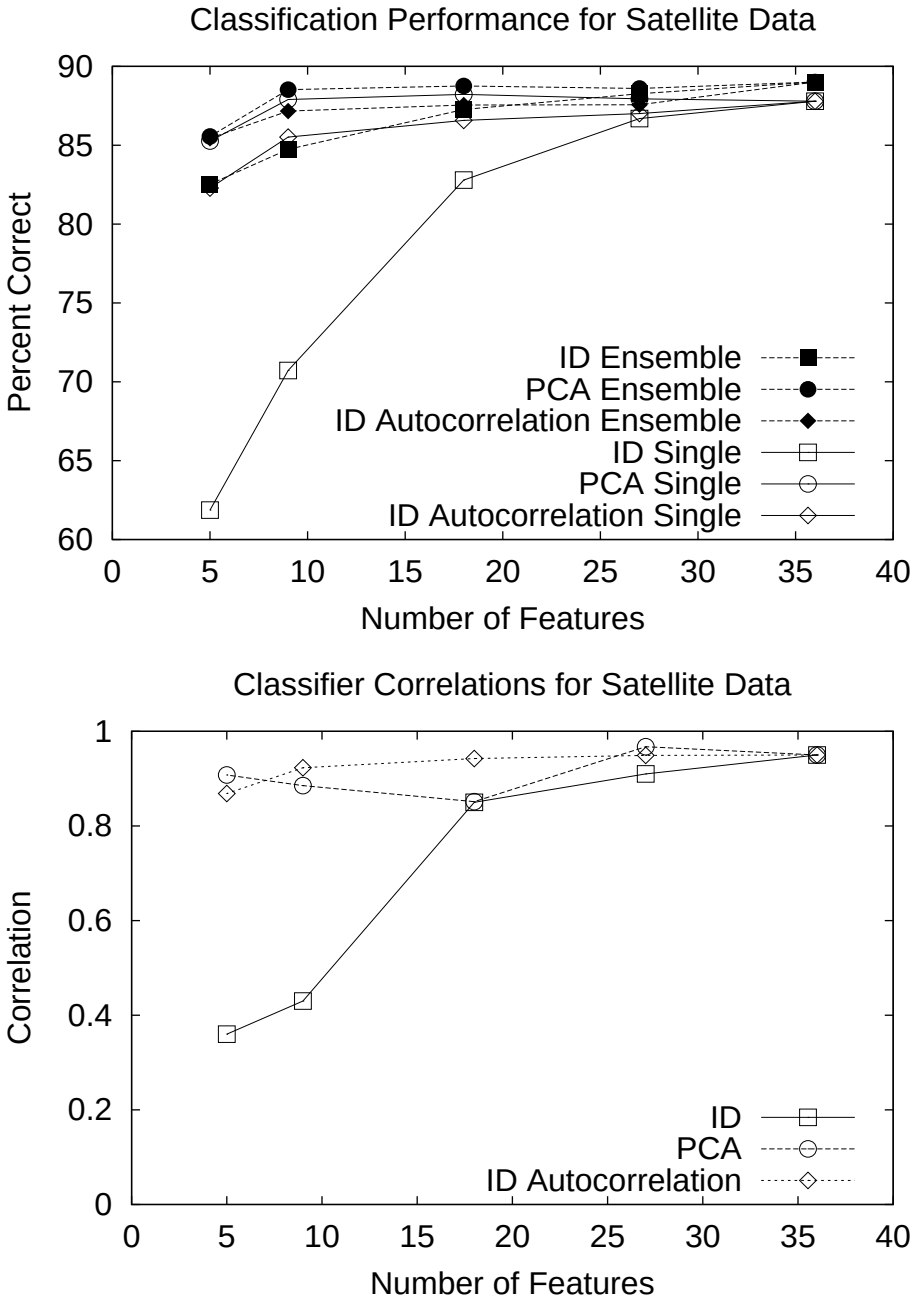

Figure 7: Performance (a) and Correlations (b) for the Satellite data set using the Autocorrelation method.

- Set A:
  - Three classes–one unimodal Gaussian per class.
  - 300 training patterns and 150 test patterns–100 training and 50 test patterns per class.
  - 100 features per pattern where there are:
    - 10 relevant features per class. Patterns that belong to a class are generated from a multivariate normal distribution in 10 independent dimensions distributed as $N(40, 5^2)$. There are no dimensions in common among the three classes. Therefore, there are 30 relevant features. For patterns in each class, the 20 features that are relevant to the other two classes are distributed as $U[-100, 100]$.[15]
    - 70 irrelevant features–distributed as $U[-100, 100]$.
- Set B: Same as Set A, except that there is overlap among the relevant features for each class. That is, each class has three relevant features in common with every other class, but there are no features that are relevant to all three classes.

In data set A there is an abundance of features that are irrelevant for the classification task. This data set was chosen to represent large data mining problems where the algorithms may get swamped by irrelevant data. In data set B the overlap among features relevant to each class provides a more difficult problem where the base classifiers are now *forced* to select some common features, reducing the potential for correlation reduction.[16]

4.3.1 Synthetic Set A

Figure 8 presents the classification accuracies and base classifier correlations on Synthetic data set A as a function of the number of inputs (which are either the number of selected principal components or the number of features selected for each base classifier through input decimation or random selection). The original single classifier and original ensemble use all the input features.[17] The points for the maximum number of features (e.g., 100 features in this data set), always represent the performance of the original classifier/ensemble. The performance and correlation for IDE with 100 features differs from those for PCA and RS ensembles because of the difference in the number of hidden units in their base classifiers.

[15]Clearly, because of this, all 30 features have some relevance to all three classes; however, the 10 features used to generate patterns belonging to each class are clearly substantially more relevant than the other 20 features.

[16]The presence of a large amount of irrelevant features biases these results against RS methods, which require *redundant* rather than *irrelevant* features. As such, the relative performance of ID vs. PCA is the main showcase of this section, though we also include results from RS ensembles for completeness.

[17]The base classifier used was an MLP with a single hidden layer and trained using a learning rate of 0.2 and a momentum term of 0.5. For IDE's, the hidden layer had 15 hidden units while for PCA and RS, we used 95 hidden units. The number of hidden units chosen was the number of hidden units in the best performing ensemble of that type. For example, the best performing IDE had 10 inputs and 15 hidden units; therefore, we chose to show the IDE results for all numbers of inputs with 15 hidden units.

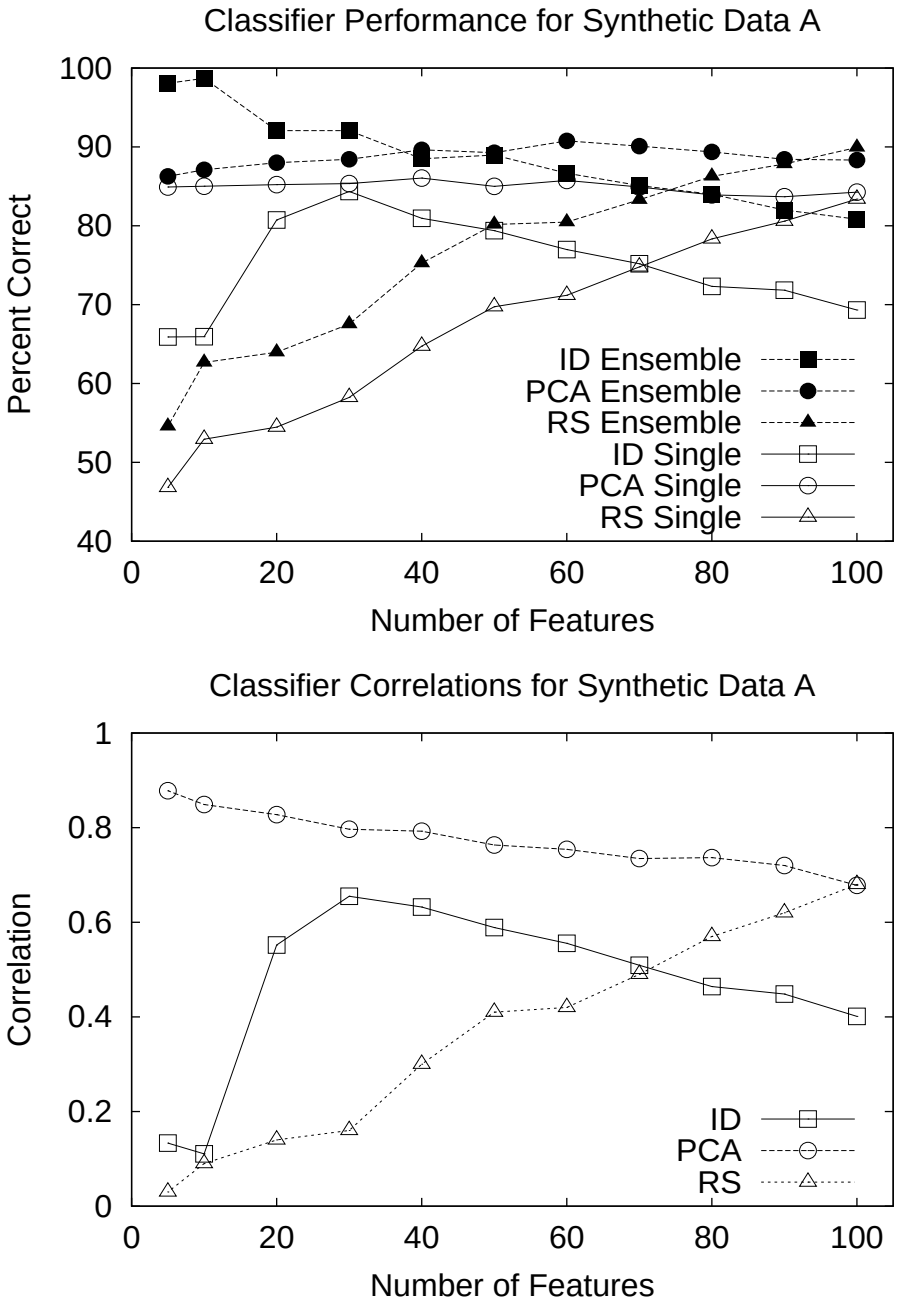

Figure 8: Data set A Performances and Correlations

An important observation that is apparent from these results is that neither PCA ensembles nor PCA base classifiers are particularly sensitive to the number of inputs. The correlations among the base classifiers reinforce this conclusion. Fewer input features in PCA means the base classifiers are more correlated since they all share the same principal features. The RS ensembles' performances, base models' performances, and correlations follow the same pattern: with an increasing number of features, the performances and correlations increase in random increments, presumably because the additional features being included are of random relevance to the outputs.

However note that input decimated base classifiers have a low correlation for small numbers of features, increasing correlation up to 30 features, and decreasing correlation after that. Upon reflection, this interesting observation is not surprising: each class has a set of 30 features that are most relevant to it. The other features add nothing but noise, slowly reducing the correlation to the outputs. The base classifiers' average performance follows a similar pattern. Note that input decimated ensembles are not adversely affected by the poor performance of the base classifiers (e.g., input decimated ensembles with 10 features outperformed input decimated ensembles with 50 features even though base classifiers with 10 features gave significantly worse results than base classifiers with 50 features). In fact, the input decimated ensemble with 10 inputs was the best performing classifier on this data set among all the classifiers tested.

In cases where more than 20 features were used, the performance of the ensemble declined with the addition of features, i.e., as more and more irrelevant features were included. The input decimated ensembles with less than 40 features significantly outperformed their PCA ensemble counterparts, those with 40 and 50 features performed comparably, and those with 60 or more features performed significantly worse. The input decimated ensembles with 70 and fewer features significantly outperformed RS ensembles while those with 80 and more features performed significantly worse. However, all the input decimation ensembles provided statistically significant improvements over the original full-featured ensembles.

The single decimated classifiers with 20 and more features outperformed the original single classifier. This perhaps surprising result (as one might have expected only the ensemble performance to improve when using subsets of the features) is mainly due to the simplification of the learning tasks, which allows the classifiers to learn the mapping more efficiently, as also observed in the SONAR and UCI data sets.

Interestingly, the average correlation among classifiers does not decrease until a small number of features remains. We attribute this to the removal of noise—removing noise increases the amount of information shared between the base classifiers. Indeed, the correlation increases steadily as features are removed until we reach 30 features (which corresponds to the actual number of relevant features). After that point, removing features reduces the correlation because the base classifiers' feature sets have a decreasing number of common features. The base classifiers' performances also decline; however, the ensemble performance still remains high. This experiment clearly shows a typical trade-off in ensemble learning: one can either increase individual classifier performance (as for input decimation with 20 or more features) or reduce the correla-

tion among classifiers (as for input decimation with less than 20 features) to improve ensemble performance.

4.3.2 Synthetic Set B

Figure 9 presents the results for the second synthetic data set, which is similar to the first data set except that there is overlap among the relevant features for the classes.[18] Because of this overlap, this feature set has fewer total relevant features and thus it constitutes a more difficult problem (as indicated by comparing the results on the full feature base classifiers and ensembles on this data set to the previous one).

The correlations for RS ensembles are comparable to those for data set A. Note that the correlations for ID and PCA-based ensembles on this data set remained in a narrower range than for data set A. Input decimation did not reduce the correlations dramatically for small feature sets in data set B the way it did in case of data set A. This is mainly caused by the "coupling" among the base classifiers due to their common input features.

In spite of these difficulties, input decimation ensembles perform extremely well. For less than 60 features, they significantly outperform PCA ensembles and for less than 80 features they significantly outperform RS ensembles. IDE's outperforms the original ensemble for less than 90 features. This is particularly heartening since this feature set is a more representative abstraction of real data sets (data sets with "clean" separation among classes are quite rare). This experiment demonstrates that when there is overlap among classes, class information becomes particularly relevant. PCA and RS operate without this vital information, therefore they are unlikely to provide competitive performance.

5 Discussion

This paper discusses input decimation, a dimensionality reduction-based ensemble method that provides good generalization performance by reducing the correlations among the classifiers in the ensemble. Through controlled experiments, we show that the input decimated single classifiers often outperform the single original classifiers (trained on the full feature set), demonstrating that simply eliminating irrelevant features can improve performance. In addition, eliminating irrelevant features in each of many classifiers using *different relevance criteria* (in this case, relevance with respect to different classes) yields significant improvement in ensemble performance through correlation reduction, as seen by comparing our decimated ensembles to the original ensembles. Selecting the features using class label information also provides significant performance gains over PCA and RS ensembles.

[18]The base classifier used was an MLP with a single hidden layer and trained using a learning rate of 0.2 and a momentum term of 0.5. For IDE's, the hidden layer had 40 hidden units while for PCA and RS, we used 95. Again, this difference in the number of hidden units causes the $N = 100$ performance to be different for IDE and PCA/Random subset selection.

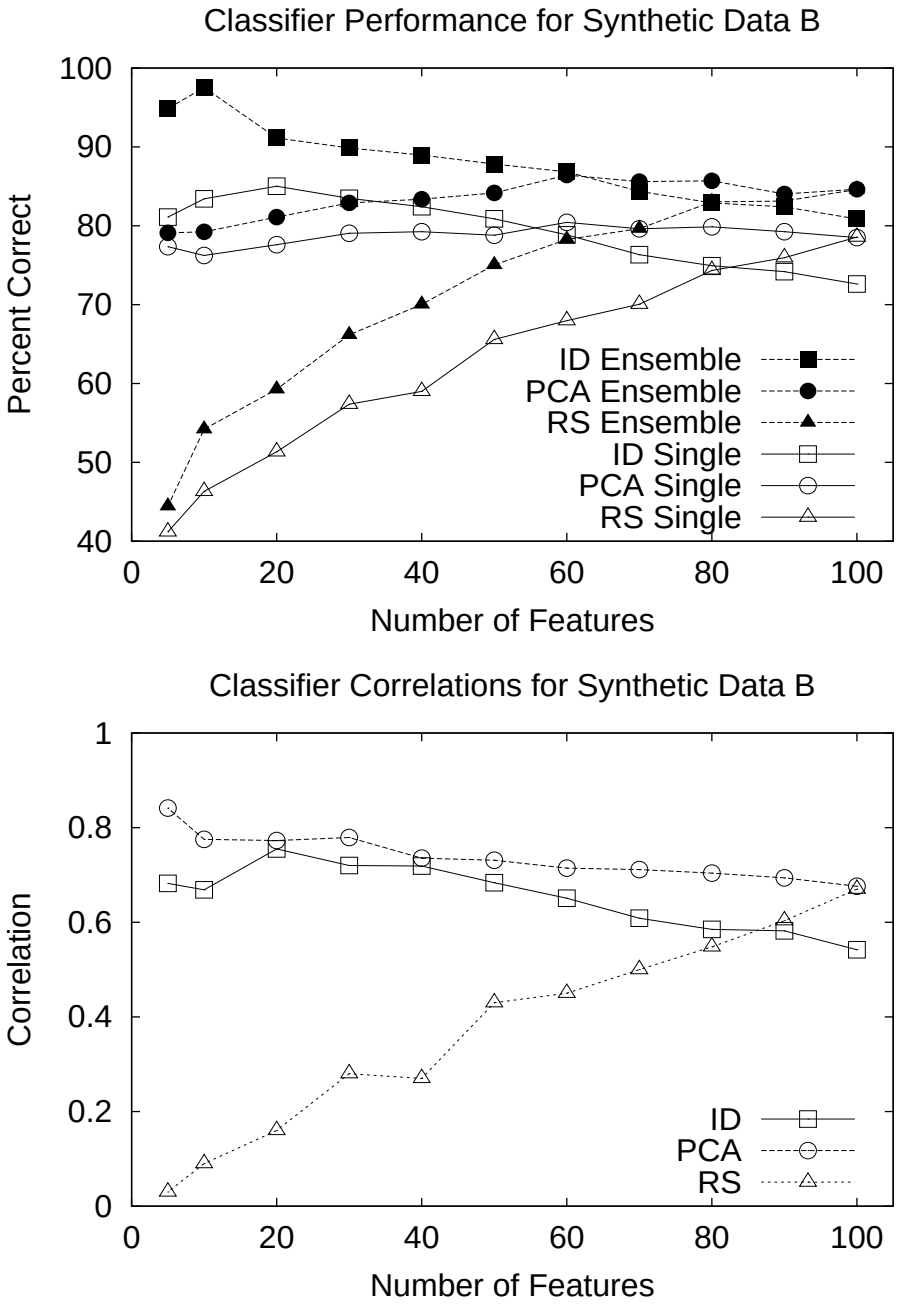

Figure 9: Data set B Performances and Correlations

Through our tests on synthetic and real data sets, we examined the characteristics that data sets need to have to fully benefit from input decimation. We observed that input decimation yields the greatest improvements over the original ensemble when there are a large number of irrelevant features. In such cases, by removing the extraneous features, input decimation reduces noise and thereby reduces the number of training examples needed to produce a meaningful model (i.e., alleviating the curse of dimensionality).

Our experiments with real data sets—especially the Satellite Image data set—showed that input decimation may suffer when there are redundant features that are highly correlated with a class. In such situations, input decimation chooses these redundant features that are equivalent to just one feature, leading to relatively poor performing base models. Our preliminary experiments in section 4.2.3 show that input decimation can yield better results if our feature selection process penalizes features that are highly correlated with those already selected. Furthermore, including some features that have a high correlation with all classes on average even though they do not have high correlation with any one class may also help in cases where particular features are important (e.g., pixel intensity).

Input decimation shares its fundamental idea of generating a diverse pool of classifiers for the ensemble with many methods such as bagging and boosting. However, by focusing on the input *features* rather than the input *patterns*, input decimation focuses on a different "axis" of correlation reduction than bagging and boosting do and is therefore orthogonal to them. We plan to experiment with using input decimation in conjunction with bagging and boosting in the future.

A final observation is that input decimation works well *in spite* of our rather crude choice for ν, the decimation coefficient (i.e., using statistical correlation of each feature individually with each class). Input decimation works because we have greatly simplified the relevance criterion: In most cases (the satellite data being the exception) we only look at one feature at a time, and instead of considering the discriminatory ability across all classes, we only consider the relevance of the features to detecting *a single* class. This typically causes each classifier in the ensemble to get a different subset of features, leading to the superior performance we have demonstrated. Nevertheless, we are currently extending this work in four directions: considering cross-correlations among the features; investigating mutual information-based relevance criteria; incorporating global relevance into the selection process; and selecting a different number of features for each classifier.

Acknowledgements: The authors would like to thank Joydeep Ghosh for helpful discussions and the reviewers for their detailed comments.

References

[1] F.M. Alkoot and J. Kittler. Improving product by moderating k-NN classifiers. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier*

Systems, pages 429–439. Springer, Berlin, 2001.

[2] R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5:4:537–550, July 1994.

[3] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. (2nd Ed.), Springer, New York, 1985.

[4] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[5] C. Blake, E. Keogh, and C.J. Merz. UCI repository of machine learning databases, 1998. (URL: http://www.ics.uci.edu/∼mlearn/MLRepository.html).

[6] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–272, 1997.

[7] K. D. Bollacker and J. Ghosh. Linear feature extractors based on mutual information. In *Proceedings of the 13th International Conference on Pattern Recognition*, pages pp. IV:720–724, 1996.

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

[9] K. J. Cherkauer. Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks. In *Working Notes of the AAAI Workshop on Integrating Multiple Learned Models*, pages 15–21, 1996.

[10] S. Cohen and N. Intrator. Automatic model selection in a hybrid perceptron/radial network. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 440–454. Springer, Berlin, 2001.

[11] J.M. Combes, A. Grossman, and Ph. Tchamitchian (Eds.). *Wavelets: Time-Frequency Methods and Phase Space*. Springer-Verlag, 1989.

[12] D. de Ridder and R. P. W. Duin. Sammon's mapping using neural networks: A comparison. *Pattern Recognition Letters*, 18:1307–1316, 1997.

[13] L. Deuser and D. Middleton. On the classification of underwater acoustic signals: An environmentally adaptive approach. *The Acoustic Society of America*, 65:438–443, 1979.

[14] P.A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice-Hall, 1982.

[15] T.G. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1998.

[16] T.G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15. Springer, Berlin, 2000.

[17] T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[18] P. Domingos. Context-sensitive feature selection for lazy learners. *Artificial Intelligence Review*, 11:227–253, 1997.

[19] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, New York, NY, second edition, 2001.

[20] R.P.W. Duin and D.M.J. Tax. Experiments with classifier combining rules. In J. Kittler and F. Roli, editors, *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 16–29. Springer, Berlin, 2000.

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

[22] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, second edition, 1990.

[23] J. Ghosh, L. Deuser, and S. Beck. A neural network based hybrid system for detection, characterization and classification of short-duration oceanic signals. *IEEE Journal of Ocean Engineering*, 17(4):351–363, October 1992.

[24] T. K. Ho. The random space method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

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

[26] T.K. Ho. Data complexity analysis for classifier combination. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 53–67. Springer, Berlin, 2001.

[27] A. Hyvarinen. Survey on independent component analysis. *Neural Computing Surveys*, 2:94–128, 1999.

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

[29] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the International Conference on Machine Learning (ICML-94)*, pages 121–129, July 1994.

[30] I.T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[31] T.M. Jorgensen and C. Linneberg. Feature weighted ensemble classifiers – a modified decision scheme. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 218–227. Springer, Berlin, 2001.

[32] N. Kambhatla and T. K. Leen. Fast non-linear dimension reduction. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems-6*, pages 152–153. Morgan Kaufmann, 1994.

[33] N. Kambhatla and T. K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9:1493, 1997.

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

[35] J. Kittler and F.M. Alkoot. Relationship of sum and vote fusion strategies. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 339–348. Springer, Berlin, 2001.

[36] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[37] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence Journal*, 1-2:273–324, 1997.

[38] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, pages 284–292, 1996.

[39] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation and active learning. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems-7*, pages 231–238. M.I.T. Press, 1995.

[40] S. Kumar, M. Crawford, and J. Ghosh. A versatile framework for labelling imagery with a large number of classes. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN-99)*, 1999.

[41] L.I. Kuncheva and C.J. Whitaker. Feature subsets for classifier combination: An enumerative experiment. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 228–237. Springer, Berlin, 2001.

[42] P. Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall Symposium on Relevance*, 1994.

[43] R. Meir. Bias, variance, and the combination of estimators; the case of least linear squares. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems-7*, pages 295–302. M.I.T. Press, 1995.

[44] C. J. Merz. A principal component approach to combining regression estimates. *Machine Learning*, 36:9–32, 1999.

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

[46] E. Oja. *Subspace Methods of Pattern Recognition*. Research Studies Press, Letchworth, England, 1983.

[47] E. Oja. Pricipal components, minor components, and linear neural networks. *Neural Networks*, 5:927–936, 1992.

[48] D. W. Opitz and J. W. Shavlik. Actively searching for an effective neural network ensemble. *Connection Science, Special Issue on Combining Artificial Neural Networks: Ensemble Approaches*, 8(3 & 4):337–354, 1996.

[49] D. W. Opitz and J. W. Shavlik. Generating accurate and diverse members of a neural-network ensemble. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems-8*, pages 535–541. M.I.T. Press, 1996.

[50] N. C. Oza and K. Tumer. Input decimated ensembles: Decorrelation through dimensionality reduction. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 238–249. Springer, Berlin, 2001.

[51] M. Partridge and R. A. Calvo. Fast dimensionality reduction and simple pca. *Intelligent Data Analysis*, 2:203–214, 1998.

[52] E. Pekalska and R.P.W. Duin. On combining dissimilarity representation. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 248–257. Springer, Berlin, 2001.

[53] Lutz Prechelt. 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, September 1994. Anonymous FTP: /pub/papers/techreports/1994/1994-21.ps.Z on ftp.ira.uka.de.

[54] N. Ramanujam, M. F. Mitchell, A. Mahadevan, S. Thomsen, A. Malpica, T. Wright, and N. Atkinson R. Richards-Kortum. Development of a multivatirate statistical algorithm to analyze human cervical tisue fluorescence spectra acquired in vivo. *Lasers in Surgery and Medicine*, 19:46–62, 1996.

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

[56] F. Roli, G. Giacinto, and G. Vernazza. Methods for designing multiple classifier systems. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 78–87. Springer, Berlin, 2001.

[57] B. Rosen. Ensemble learning using decorrelated neural networks. *Connection Science, Special Issue on Combining Artificial Neural Networks: Ensemble Approaches*, 8(3 & 4):373–384, 1996.

[58] J.W. Sammon Jr. A nonlinear mapping for data structure analysis. *IEEE transactions on Computers*, 18:401–409, 1969.

[59] N. Short. Remote sensing tutorial, 2000. URL: http://rst.gsfc.nasa.gov/starthere.html.

[60] K. Sirlantzis, M.C. fairhurst, and M.S. Hoque. Genetic algorithms for multi-classifier system configuration: A case study in character recognition. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 99–108. Springer, Berlin, 2001.

[61] D. Tax, M. van Breukelen, R. Duin, and J. Kittler. Combining multiple classifiers by averaging or by multiplying. *Pattern Recognition*, 29(2):341–348, February 1996.

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

[63] K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science, Special Issue on Combining Artificial Neural Networks: Ensemble Approaches*, 8(3 & 4):385–404, 1996.

[64] K. Tumer and J. Ghosh. Estimating the Bayes error rate through classifier combining. In *Proceedings of the International Conference on Pattern Recognition, Vienna, Austria*, pages IV:695–699, 1996.

[65] K. Tumer and J. Ghosh. Linear and order statistics combiners for pattern classification. In A. J. C. Sharkey, editor, *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*, pages 127–162. Springer-Verlag, London, 1999.

[66] K. Tumer and J. Ghosh. Robust order statistics based ensembles for distributed data mining. In H. Kargupta and P. Chan, editors, *Advances in Distributed and Parallel Knowledge Discovery*, pages 185–210. AAAI/MIT Press, 2000.

[67] K. Tumer and N. C. Oza. Decimated input ensembles for improved generalization. In *Proceedings of the International Joint Conference on Neural Networks*, 1999.

[68] D. Windridge and J. Kittler. Classifier combination as a tomographic process. In J. Kittler and F. Roli, editors, *Proceedings of the Second International Workshop on Multiple Classifier Systems*, pages 248–257. Springer, Berlin, 2001.

[69] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.