Pruning in Ordered Bagging Ensembles

Gonzalo Martínez-Muñoz GONZALO.MARTINEZ@UAM.ES
Alberto Suárez ALBERTO.SUAREZ@UAM.ES
Escuela Politécnica Superior, Universidad Autónoma de Madrid, F. Tomás y Valiente, 11, 28049 Madrid, Spain

Abstract

We present a novel ensemble pruning method based on reordering the classifiers obtained from bagging and then selecting a subset for aggregation. Ordering the classifiers generated in bagging makes it possible to build subensembles of increasing size by including first those classifiers that are expected to perform best when aggregated. Ensemble pruning is achieved by halting the aggregation process before all the classifiers generated are included into the ensemble. Pruned subensembles containing between 15% and 30% of the initial pool of classifiers, besides being smaller, improve the generalization performance of the full bagging ensemble in the classification problems investigated.

1. Introduction

The construction of classifier ensembles is an active field of research in machine learning because of the improvements in classification accuracy that can be obtained by combining the decisions made by the units in the ensemble. Ensemble generation algorithms usually proceed in two phases: In a first step a pool of diverse classifiers is trained or selected according to some prescription. Different prescriptions lead to different types of ensembles (bagging, boosting, etc. (Freund & Schapire, 1995; Breiman, 1996a; Dietterich, 2000; Webb, 2000; Breiman, 2001; Martínez-Muñoz & Suárez, 2005)). In a second step, a combiner articulates the individual hypotheses to yield the final decision.

An important drawback of classification ensembles is that both the memory required to store the parameters of the classifiers in the ensemble and the processing time needed to produce a classification increase linearly with the number of classifiers in the ensemble. Several strategies have been proposed to address these shortcomings. One approach is to prune the ensemble by selecting the classifiers that lead to improvements in classification accuracy and discarding those that are either detrimental to the performance of the ensemble or contain redundant information (Domingos, 1997; Margineantu & Dietterich, 1997; Prodromidis & Stolfo, 2001; Giacinto & Roli, 2001; Zhou et al., 2002; Zhou & Tang, 2003; Bakker & Heskes, 2003; Martínez-Muñoz & Suárez, 2004). Besides being smaller, pruned ensembles can perform better than the original full ensemble (Zhou et al., 2002; Zhou & Tang, 2003; Martínez-Muñoz & Suárez, 2004).

Pruning an ensemble of size T requires searching in the space of the $2^T - 1$ non-empty subensembles to minimize a cost function correlated with the generalization error. The search problem can be shown to be NP-complete (Tamon & Xiang, 2000). In order to render the solution feasible various heuristic methods for ensemble pruning have been developed. In (Margineantu & Dietterich, 1997) several heuristics are proposed to reduce the size of an adaboost ensemble. This study reports reductions up to 60-80% of the full ensemble without a significant increase in the generalization error. In (Zhou et al., 2002; Zhou & Tang, 2003) the selection of the classifiers is made using a genetic algorithm. This procedure reduces the size of an ensemble composed of 20 trees to 8.1 trees (on average) slightly reducing the error of the full bagging ensemble (3% on average). Other techniques aim to emulate the full ensemble by building new classifiers: In Ref. (Domingos, 1997) the full ensemble is replaced by a single classifier trained to reproduce the classification produced by the original ensemble. The objective is to build a comprehensible learner that retains most of the accuracy gains achieved by the ensemble. A further processing of this emulator can also be used to select the ensemble classifiers (Prodromidis & Stolfo, 2001). This article shows that the size of the ensemble can be be reduced up to 60-80% of its original size without a significant

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

deterioration of the generalization performance of the pruned ensemble. Adopting a different strategy, one can apply clustering to the classifiers/regressors in the ensemble and select a single representative classifier for every cluster that has been identified (Giacinto & Roli, 2001; Bakker & Heskes, 2003).

Our approach to ensemble pruning is to modify the original random aggregation ordering in the ensemble assuming that near-optimal subensembles of increasing size can be constructed incrementally by incorporating at each step the classifier that is expected to produce the maximum reduction in the generalization error (Margineantu & Dietterich, 1997; Martínez-Muñoz & Suárez, 2004). After ordering, only a fraction of the inducers in the ordered ensemble is retained. The pruned ensemble obtained in this manner shows significant improvements in classification accuracy on test examples.

In this work we propose a new criterion to guide the ordering of the units in the ensemble. The goal is to select first those classifiers that bring the ensemble closer to an ideal classification performance. In order to accomplish this, each inducer is characterized by a signature vector whose dimension is equal to the size of the training set. The components of this vector are calculated in terms of the error made by the corresponding classifier on a particular labeled example (+1 if the example is correctly classified, -1 if it is incorrectly classified). The classifier is then incorporated into the ensemble according to the deviation of the orientation of the corresponding signature vector from a reference vector. This reference vector represents the direction toward which the signature vector of the ensemble (calculated as the average of the signature vectors of the ensemble elements) should be modified to achieve a perfect classification performance on the training set. In an ensemble of size T, the ordering operation can be performed with a quick-sort algorithm, which has an average running time of $O(T\log(T))$. If we are only interested in the selection of the τ-best classifiers a quick-select algorithm can also be applied. Thus, the complexity of the ordering or of the selection operation is linear, in contrast to the quadratic time-complexity of the algorithms proposed in (Martínez-Muñoz & Suárez, 2004), where the selection of each classifier involves an evaluation over all the remaining classifiers. The proposed ordering method also makes it possible to give a criterion for selecting a subset of classifiers to be considered for the inclusion in the final ensemble. This avoids the use of a pruning percentage that is fixed beforehand.

The article is structured as follows: In Section 2 we introduce the ordering procedure in bagging ensembles. Section 3 presents the proposed criterion for ordering. The results of experiments that illustrate the performance of the pruned ensembles on several UCI datasets (Blake & Merz, 1998) are discussed in Section 4. Finally, the conclusions of this research are presented.

2. Ordering Bagging Ensembles

Let $L = (\mathbf{x}_i, y_i)$, $i = 1, 2, ..., N$ be a collection of N labeled instances. The training examples are characterized by a vector of attributes $\mathbf{x}_i \in \chi$ and a discrete class label $y_i \in \phi \equiv \{1, 2, \ldots, C\}$. Consider a learning algorithm that constructs a classifier, h, from a given training set L. This classifier produces a classification $y \in \phi$ of a new instance $\mathbf{x} \in \chi$ by a mapping $h : \chi \rightarrow \phi$.

In bagging (Breiman, 1996a) a collection of classifiers is generated by training each inducer with a different dataset. These datasets are obtained by sampling with replacement from the original training set L with N extractions (bootstrap sampling). The final classification is obtained by combining with equal weights the decisions of the individual classifiers in the ensemble. An instance $\mathbf{x}$ is thus classified according to the rule

$$\underset{k}{argmax}\left(\sum_{t=1}^{T} I(h_t(x) = k)\right) : \; k = 1, 2, \ldots, C \; , \quad (1)$$

where T is the number of classifiers and I is the indicator function such that $I(True) = 1$, $I(False) = 0$. The order in which classifiers are aggregated in bagging is irrelevant for the classification given by the full ensemble.

The rationale for modifying the aggregation ordering in a bagging ensemble is to construct small subensembles with good classification performance. As stated in the introduction, the combinatorial problem of identifying the optimal subensemble is NP-complete (Tamon & Xiang, 2000) and becomes intractable for relatively small ensembles ($T > 30$). Instead of solving the optimization problem exactly, we use an approximate procedure: Assume we have identified a subensemble composed of $\tau - 1$ classifiers, which is close to being optimal. A near-optimal ensemble of size τ is built by selecting from the pool of remaining classifiers (of size $T - (\tau - 1)$) the classifier that maximizes a quantity that is correlated with the generalization performance of the ensemble of size τ. We have performed extensive experiments using exhaustive search for small ensembles (up to 31 classifiers) and global optimization tools, such as genetic algorithms, for larger ensembles, that show that this greedy search is efficient in finding near-optimal ensembles

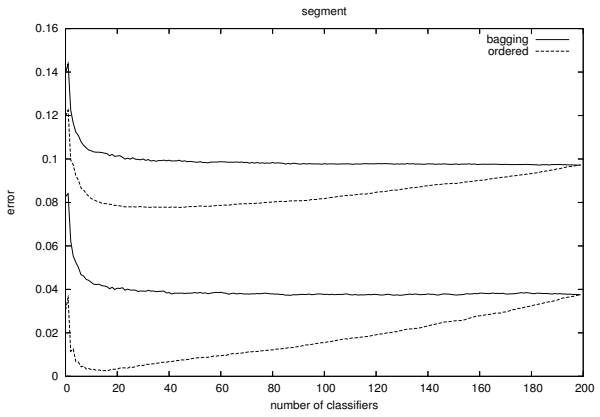

Figure 1. Average test and train error for the *Segment* dataset for bagging and ordered bagging according to the proposed heuristic.

Figure 1 shows the typical dependence of the classification error with the ensemble size in randomly ordered bagging (the standard version of bagging, where the order in which classifiers are aggregated is dictated by the bootstrap procedure) and in ordered bagging ensembles. This figure displays the error curves for both the training and the test set (*Segment* dataset). Results are averaged over 100 executions of bagging with random ordering (solid line) and of ordered bagging (long trait line). The error for bagging ensembles with random ordering generally decreases monotonically as the number of classifiers included in the ensemble increases, approaching saturation at a constant error level for large ensembles. For the ordered ensembles both the test and training error curves reach a minimum at an intermediate number of classifiers. The error rate at this minimum is lower than the error of the full ensemble. Note that the minimum of the error curve for the training set is achieved for smaller ensembles than in the test set. This means that the location of the minimum in the training error curve cannot be directly used to determine the optimal subensemble size. In this example the minimum in the training set error is achieved with 14 classifiers, whereas the best results for the test set are obtained in ensembles that contain 44 classifiers. It is in general difficult to give a reliable estimate of the optimal number of classifiers to get the best generalization accuracies. Nonetheless, Fig. 1 also shows that the minimum in the test error is fairly broad, and that, for a large range of sizes, the ordered subensembles have a generalization error that is under the final bagging error. This implies that it should be easy to identify pruned ensembles with improved classification accuracy.

3. Orientation Ordering

For the ordering procedure to be useful the quantity that guides the selection of the classifiers should be a reliable indicator of the generalization performance of the ensemble. Measures based on individual properties of the classifiers (for instance, selecting first classifiers with a lower training error) are not well correlated with the classification performance of the subensemble. It is necessary to employ measures, such as diversity (Margineantu & Dietterich, 1997), that contain information of the complementariness of the classifiers. In this work, the quantity proposed measures how a given classifier maximizes the alignment of a signature vector of the ensemble with a direction that corresponds to perfect classification performance on the training set.

Consider a dataset L_{tr} composed of N_{tr} examples. Define $\boldsymbol{c}_t$, the signature vector of the classifier h_t for the dataset L_{tr}, as the N_{tr}-dimensional vector whose components are

$$c_{ti} = 2I(h_t(\mathbf{x}_i) = y_i) - 1,\ i = 1, 2, \ldots, N_{tr}\ , \qquad (2)$$

where c_{ti} is equal to $+1$ if h_t (i.e. the t^{th} unit in the ensemble) correctly classifies the i^{th} example of L_{tr} and -1 otherwise. The average signature vector of the ensemble is

$$\boldsymbol{c}_{ens} = \frac{1}{T}\sum_{t}^{T} \boldsymbol{c}_t\ . \qquad (3)$$

In a binary classification problem, the i^{th} component of this ensemble signature vector is equal to the classification margin for the i^{th} example (the margin is defined as the difference between the number of votes for the correct class and the number of votes for the most common incorrect class, normalized to the interval $[-1, 1]$ (Schapire et al., 1998)). In general multiclass classification problems, it is equal to $1 - 2edge(i)$ of the ensemble for the i^{th} example (the edge is defined as the difference between the number of votes for the correct class and the number of votes for all incorrect classes, normalized to the interval $[0, 1]$ (Breiman, 1997)). The i^{th} example is correctly classified by the ensemble if the i^{th} component of the average vector $\boldsymbol{c}_{ens}$ is positive. That is, an ensemble whose average signature vector is in the first quadrant of the N_{tr}-dimensional space will correctly classify all examples of the L_{tr} dataset.

This study presents an ordering criterion based on the orientation of the signature vector of the individual classifiers with respect to a reference direction. This direction, coded in a reference vector, $\boldsymbol{c}_{ref}$, is

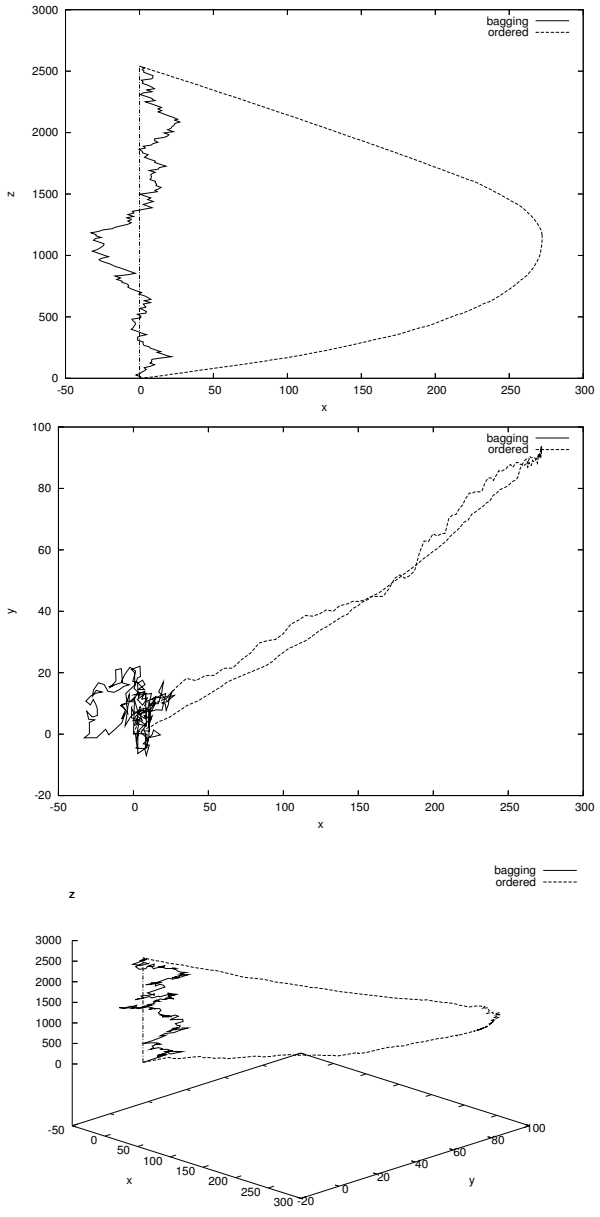

Figure 2. Projection of the unordered and ordered bagging signature vectors onto: two dimensions $\boldsymbol{c}_{ens}$ (z axis) and $\boldsymbol{c}_{ref}$ (x axis) (*top plot*), two dimensions $\boldsymbol{c}_{ref}$ and an axis perpendicular to $\boldsymbol{c}_{ref}$ and $\boldsymbol{c}_{ens}$ (y axis) (*middle plot*) and in the three dimensions previously defined (*bottom plot*). Plots are for the *Waveform* problem.

the projection of the first quadrant diagonal onto the hyper-plane defined by $\boldsymbol{c}_{ens}$. The classifiers are ordered by increasing values of the angle between the signature vectors of the individual classifiers and the reference vector $\boldsymbol{c}_{ref}$. Finally, the fraction of the classifiers whose angle is less than $\pi/2$ (i.e. those within the quadrant defined by $\boldsymbol{c}_{ref}$ and $\boldsymbol{c}_{ens}$) are included in the final ensemble. The reference vector, $\boldsymbol{c}_{ref}$, is chosen to maximize the torque on $\boldsymbol{c}_{ens}$ (which represents the central tendency of the full ensemble) along the direction that corresponds to the ideal classification performance. This effect is obtained by choosing $\boldsymbol{c}_{ref} = \boldsymbol{o} + \lambda \boldsymbol{c}_{ens}$, where $\boldsymbol{o}$ is a vector oriented along the diagonal of the first quadrant, and λ is a constant such that $\boldsymbol{c}_{ref}$ is perpendicular to $\boldsymbol{c}_{ens}$ ($\boldsymbol{c}_{ref} \perp \boldsymbol{c}_{ens}$).

As an example, consider a training set composed of three examples and an ensemble with $\boldsymbol{c}_{ens} = \{1, 0.5, -0.5\}$, meaning that the first example is correctly classified by all the classifiers of the ensemble, the second by 75% of classifiers and the third by 25% of classifiers. Then the projection is calculated considering that $\boldsymbol{c}_{ref} = \boldsymbol{o} + \lambda \boldsymbol{c}_{ens}$ and $\boldsymbol{c}_{ref} \perp \boldsymbol{c}_{ens}$, which gives $\lambda = -\boldsymbol{o} \cdot \boldsymbol{c}_{ens} / |\boldsymbol{c}_{ens}|^2$. Hence, $\lambda = -2/3$ and $\boldsymbol{c}_{ref} = \{1/3, 2/3, 4/3\}$. In the ordering phase, a stronger pull will be felt along the dimensions corresponding to examples that are harder to classify by the full ensemble (i.e. the third and second examples). However, $\boldsymbol{c}_{ref}$ becomes unstable when the vectors that define the projection (i.e. $\boldsymbol{c}_{ens}$ and the diagonal of the first quadrant) are close to each other. This makes the selection of $\boldsymbol{c}_{ref}$ less reliable and renders the ordering process less efficient. This is the case for ensembles that quickly reach zero training error, such as boosting or bagging composed of unpruned trees, which do not show significant improvements in classification performance when reordered according to the proposed heuristic.

In Figure 2 the learning processes in bagging and in ordered bagging are depicted. A 200 classifier ensemble is trained to solve the *Waveform* problem (Breiman et al., 1984) using 300 data examples (i.e. the signature vectors have 300 dimensions). These plots show 2 and 3-dimensional projections of the walks followed by the incremental sum of the signature vectors ($\sum_{t=1}^{\tau} \boldsymbol{c}_t$; $\tau = 1, 2, \ldots, T$) in the randomly ordered ensemble (solid line) and in the ordered ensemble (long trait line). In the top plot the ensemble vectors are projected onto the plane defined by $\boldsymbol{c}_{ens}$ (z axis) and by $\boldsymbol{c}_{ref}$ (x axis). The middle plot shows a 2-dimensional projection onto a plane perpendicular to $\boldsymbol{c}_{ens}$, defined by $\boldsymbol{c}_{ref}$ (x axis) and a vector perpendicular to both $\boldsymbol{c}_{ens}$ and $\boldsymbol{c}_{ref}$ (y axis). This plot is a projection into a plane that is perpendicular to the vector that defines the ensemble, $\boldsymbol{c}_{ens}$; therefore any path including all classifiers starts and finishes at the origin. Finally, the bottom plot shows a 3-dimensional projection onto the previously defined x, y and z axis. For bagging (solid lines) it can be observed that the incremental sum of the signature vectors follows a path

that can be seen as a Brownian bridge starting at the origin and with a final value of $T \times \boldsymbol{c}_{ens}$. The ordering algorithm (long trait line) rearranges the steps of the original random path in such a way that the first steps are the ones that approximate the walker the most to $\boldsymbol{c}_{ref}$: Hence the characteristic form of the ordered path, which appears elongated toward the direction of $\boldsymbol{c}_{ref}$. These plots show the stochastic nature of the bagging learning process ((Breiman, 2001; Esposito & Saitta, 2004)) and how this process can be altered by re-ordering its classifiers.

4. Experimental Results

In order to assess the performance of the ordering procedure described in the previous section, experiments are carried out in 18 classification problems from the UCI-repository (Blake & Merz, 1998) and from Refs. (Breiman, 1996b; Breiman et al., 1984). The datasets have been selected to test the performance of the pruning procedure on a wide variety of problems, including synthetic and real-world data from various application fields with different numbers of classes and attributes. Table 1 shows the characteristics of the sets investigated. For each dataset this table presents the number of examples used to train and test the ensembles, the number of attributes and the number of classes. The subdivisions into training and testing are made using approximately 2/3 of the set for training and 1/3 for testing except for the *Image Segmentation* set, where the sizes specified in its documentation are used. For the synthetic sets (*Waveform* and *Twonorm*) different training and testing sets were generated in every execution of the algorithm.

For each dataset 100 executions were carried out, each involving the following steps: (i) Generate a stratified random partition (independent sampling for the synthetic datasets) into training and testing sets whose sizes are given in Table 1. (ii) Using bootstrap sampling, 200 CART decision trees are generated from the training set. The decision trees are pruned according to the CART 10-fold cross validation procedure (Breiman et al., 1984). The ensemble generalization error is estimated in the unseen test set. This test error is calculated for a bagging ensemble that uses the first 100 classifiers generated and for a bagging ensemble containing all 200 trees. (iii) The trees in the ensemble are then ordered according to the rule described in Section 3 using the training subset. Finally, we calculate the average of the signature vector angles for vectors whose angle with respect to $\boldsymbol{c}_{ref}$ is lower than $\pi/2$. Only classifiers whose signature vector angle is less than this average are included in the pruned subensemble. This rule gives a reasonable estimate of the number of classifiers needed for an optimal generalization performance. In fact, any rule that selects the first 20-30% of the classifiers in the ordered ensemble achieves a similar generalization accuracy.

Table 1. Characteristics of the datasets used in the experiments.

Dataset	Train	Test	Attribs.	Classes
Audio	140	86	69	24
Australian	500	190	14	2
Breast W.	500	199	9	2
Diabetes	468	300	8	2
German	600	400	20	2
Heart	170	100	13	2
Horse-Colic	244	124	21	2
Ionosphere	234	117	34	2
Labor	37	20	16	2
New-thyroid	140	75	5	3
Segment	210	2100	19	7
Sonar	138	70	60	2
Tic-tac-toe	600	358	9	2
Twonorm	300	5000	20	2
Vehicle	564	282	18	4
Vowel	600	390	10	11
Waveform	300	5000	21	3
Wine	100	78	13	3

Figure 3 displays ensemble error curves for 3 of the classification problems considered. The behavior of the ensemble error curves is similar for the remaining problems. These plots show the dependence of the average train (bottom curves in each plot) and test (top curves in each plot) errors on the number of classifiers included in the subensemble for randomly ordered bagging (solid line) and for orientation ordered bagging using 100 (dotted line) and 200 trees (trait line). As expected, in unordered bagging the error decreases monotonically as the number of classifiers in the ensemble grows and reaches a constant error rate asymptotically. In contrast to this monotonic behavior, error curves in ordered bagging ensembles exhibit a typical shape where the error initially decreases with the number of classifiers, reaches a minimum, and eventually rises and reaches the error of the full bagging ensemble. This characteristic shape is reproduced in both the train and test error curves and for ensembles of 100 and 200 trees. It is important to note that this minimum is achieved for a smaller number of classifiers in the training set than in the test set. In the training set the minima appear generally for a fraction of classifiers that is under 10% of the initial pool. For the test curves the minima appear for subensemble whose

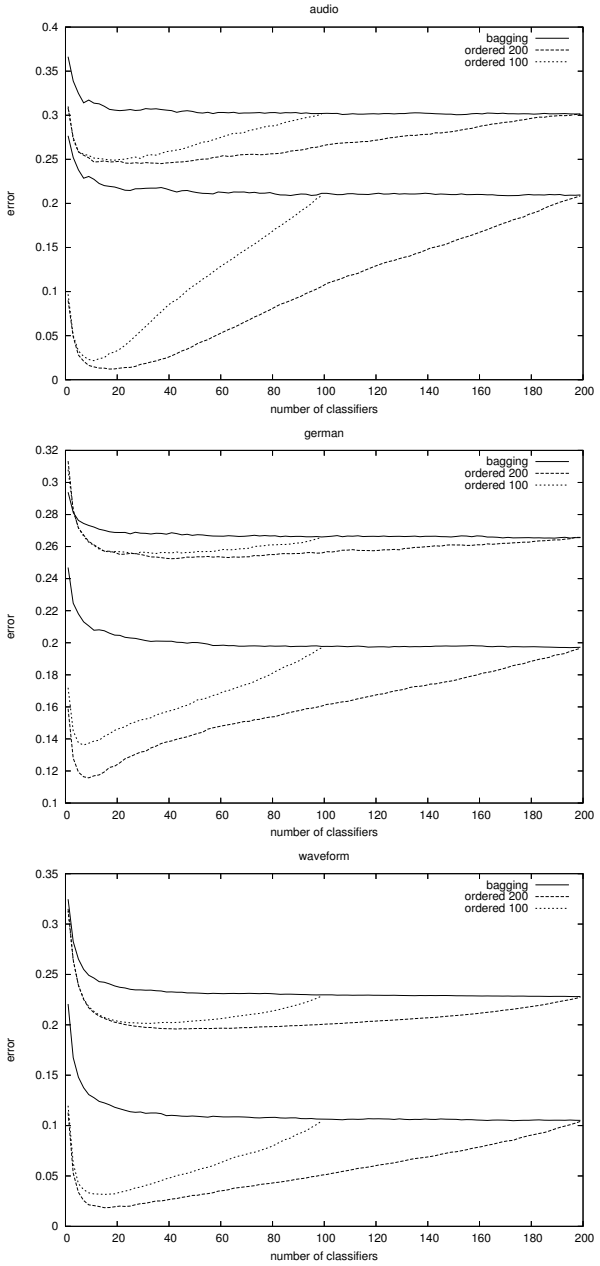

Figure 3. Train and test error curves for Bagging (*solid line*), ordered bagging with 200 trees (*trait line*) and 100 trees (*dotted line*) for *Audio*, *German* and *Waveform* classification problems.

size ranges between 20%-40% of the original classifiers. This fact makes it difficult to use directly the training error curve minimum to estimate the number of classifiers that produce the best generalization error. Furthermore, this estimation becomes more difficult when considering each execution individually (instead of the smooth averaged curves) since the curves are more bumpy and do not always show clear minima. In any case, given that the minima are fairly flat, the range of valid pruning values that lead to a reduction the mean generalization error of bagging is broad.

Table 2 shows the results for the classification problems investigated. The values reported are averages over 100 executions. Note that the figures displayed in Table 2 and the values of the test curves shown in figure 3 do not always coincide, since the former is an average for different subensemble sizes, whereas the latter is an average for a fixed number of classifiers. The second column displays the test error when considering the full ensemble of size 200 and the third column gives the test error for the ordered ensemble using the corresponding fraction of classifiers. The average number of classifiers used for calculating the generalization accuracy of the ordered ensembles is shown in the fourth column. As a reference, we run the reduce-error (RE) pruning algorithm without back-fitting [1] (Margineantu & Dietterich, 1997), using the same ensembles and with a near-optimal pruning rate of 80% (i.e. 41 classifiers of 200 and 21 of 100). This heuristic chooses at each ordering step the classifier that reduces most the training error of the already selected subensemble. The test error of the reduce-error algorithm is shown in fifth column.

The proposed method always reduces the average generalization error for the studied datasets using a small subset of the classifiers of the full ensemble. This number of classifiers in the pruned ensembles varies from 33 of 200 for the *German* dataset to 58 of 200 for *Vowel*. The improvements in classification accuracy of the presented method with respect to bagging are statistically significant at a 99.99% confidence level (using a paired two tailed Student's t-test) in all problems investigated, with the exception of *Australian* and *Diabetes*. For these sets the differences are significant for ensembles of 200 trees but with a lower confidence level (95%). However, we should be cautious about the confidence levels in the real-world datasets since the statistical test may overestimate its significance (Nadeau & Bengio, 2003). For the synthetic datasets the confidence levels are perfectly valid as the experiements were carried out using independent sampling. In comparison with reduce-error pruning the proposed method obtains similar or slightly better results. Its generalization error is lower in 11 out of 18 datasets, equal on 3 and worse on 4.

[1] We choose not to show the results with back-fitting because for this experiment configuration not using back-fitting is the most efficient selection. The generalization error with and without back-fitting are equivalent (within $\pm 0.3\%$) and the execution time increases substantially when using back-fitting.

Table 2. Average test error in %.

	BAGGING (200 TREES)				BAGGING (100 TREES)			
	FULL	ORDERED	SIZE	RE-41	FULL	ORDERED	SIZE	RE-21
AUDIO	30.2±4.1	24.4±3.7	38.6	24.4±3.9	30.2±3.9	24.8±3.7	19.1	25.0±4.0
AUSTRALIAN	14.5±2.1	14.1±2.2	38.0	13.7±2.3	14.5±2.1	14.3±2.2	18.9	14.0±2.3
BREAST W.	4.7±1.5	4.1±1.3	40.9	4.1±1.3	4.7±1.5	4.2±1.3	20.2	4.1±1.3
DIABETES	24.9±1.8	24.5±2.0	36.6	24.4±1.9	24.9±1.7	24.7±1.9	18.5	24.6±2.1
GERMAN	26.6±1.6	25.4±1.7	32.9	25.1±1.7	26.6±1.7	25.6±1.7	16.8	25.5±1.7
HEART	20.4±4.3	18.5±3.7	40.9	18.9±3.6	20.3±4.2	19.0±3.3	20.0	19.6±3.4
HORSE-COLIC	17.7±2.9	16.0±2.8	32.9	15.5±2.4	17.5±2.9	16.3±2.9	16.4	15.8±2.5
IONOSPHERE	9.3±2.5	7.4±2.3	38.5	7.6±2.5	9.4±2.4	7.7±2.4	19.3	7.6±2.4
LABOR	14.4±7.8	10.0±6.7	45.7	12.3±7.6	14.6±7.7	10.0±6.7	23.0	12.1±7.6
NEW-THYROID	7.3±3.1	5.7±2.6	44.2	6.2±2.6	7.5±3.1	5.8±2.5	22.0	6.1±2.8
SEGMENT	9.7±1.7	7.8±1.1	41.7	8.0±1.1	9.8±1.7	8.0±1.1	21.1	8.2±1.1
SONAR	24.7±4.7	20.7±5.1	47.1	21.5±4.8	24.6±4.7	21.7±4.6	23.2	21.9±4.7
TIC-TAC-TOE	2.7±1.1	2.0±0.8	48.8	2.3±1.0	2.7±1.1	2.3±0.9	24.5	2.6±1.1
TWONORM	9.3±3.1	6.5±1.0	51.3	8.7±2.0	9.5±3.1	7.5±1.0	25.6	9.6±1.8
VEHICLE	29.6±2.2	26.5±2.1	41.0	26.5±1.9	29.5±2.2	26.9±2.0	20.7	26.9±2.0
VOWEL	13.7±2.2	12.1±2.0	58.3	13.6±2.1	14.0±2.2	12.8±2.1	29.2	14.1±2.2
WAVEFORM	22.8±2.5	19.6±1.2	42.0	20.0±1.3	23.0±2.4	20.3±1.2	20.7	20.6±1.3
WINE	6.5±4.0	4.8±2.9	44.7	5.8±3.5	6.6±4.2	5.1±2.9	22.4	6.2±3.6

In a second batch of experiments we investigate how the number of classifiers in the original bagging ensemble affects the performance of the ordered ensembles. For these experiments the generated ensembles were re-evaluated using the first 100 trees of the randomly ordered bagging ensemble of size 200. The ordering algorithm is then applied to this smaller pool of classifiers. The average generalization error curve for the randomly ordered ensemble of size 100 and for the ordered one are shown in Table 2 in the sixth and seventh columns, respectively. The number of classifiers in the pruned subensemble selected from the ordered ensembles of size 100 are shown in the eighth column. In the datasets investigated, a bagging ensemble with 100 trees seems to be large enough to achieve the best possible classification performance of bagging. Slight improvements are observed for some sets (*New-thyroid*, *Sonar*, *Waveform*,...) when using the larger ensemble but also small error increases (*Heart* and *Horse-colic*). For ordered ensembles, the results reported in Table 2 show that there are small but systematic improvements in classification accuracy for the larger ensembles, at the expense of using pruned ensembles with approximately twice as many classifiers. The curves plotted in Figure 3, show that initially there is a steep parallel descent of the error curves for both ordered ensembles of size 100 and 200 and for both train and test curves up to a point that depends on the dataset. From this point onwards, the curves of the smaller ensembles slow their descent until they reach a minimum. The error curves for the ordered ensemble of size 200 continue to decrease with a smaller negative slope and finally reach a flatter minimum.

Table 3. Average ordering time (s) for orientation ordering (OO) and reduce-error (RE) for different ensemble sizes.

SIZE	50	100	200	400	800	1600
OO	0.012	0.025	0.048	0.089	0.177	0.355
RE	0.268	1.053	4.181	16.69	66.83	268.5

Finally, Table 3 shows the time needed for pruning using orientation ordering (OO) and reduce-error ordering without back-fitting (RE) (Margineantu & Dietterich, 1997) for 50, 100, 200, 400, 800 and 1600 trees and for the *Pima Indian Diabetes* set. The values reported in this table are averages over 10 executions using a Pentium IV at 3.2 MHz. The results displayed in this table clearly show the approximately linear behavior of the proposed method, in contrast to longer execution times and quadratic dependence on the size of the ensemble for the reduce-error pruning.

5. Conclusions

This article presents a novel method for pruning bagging ensembles that consistently reduces the generalization error for the studied datasets by using a fraction of the classifiers of the complete bagging ensemble. The fraction of selected classifiers varies from 15% to

30% (i.e. 30-60 classifiers of 200) of the full generated bagging ensemble. This is a clear improvement over full bagging both in storage needs and in classification speed, which is a crucial point for on-line applications. Furthermore, for applications where the classification speed is a critical issue the number of selected classifiers can be further reduced (up to about 10 classifiers, depending on the dataset) without a significant deterioration of the generalization performance.

The presented method executes in quick-sort running time (or quick-select time if only the best classifiers are selected). This compares favorably to the quadratic running time of previous ordering methods or to the running times of genetic algorithms used in earlier methods for ensemble pruning.

Acknowledgments

The authors acknowledge financial support from the Spanish *Dirección General de Investigación*, project TIN2004-07676-C02-02.

References

Bakker, B., & Heskes, T. (2003). Clustering ensembles of neural network models. *Neural Networks*, *16*, 261–269.

Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases.

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

Breiman, L. (1996b). *Bias, variance, and arcing classifiers* (Technical Report 460). Statistics Department, University of California.

Breiman, L. (1997). *Arcing the edge* (Technical Report). University of California, Berkeley, CA.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. New York: Chapman & Hall.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, *40*, 139–157.

Domingos, P. (1997). Knowledge acquisition from examples via multiple models. *Proc. 14th International Conference on Machine Learning* (pp. 98–106). Morgan Kaufmann.

Esposito, R., & Saitta, L. (2004). A monte carlo analysis of ensemble classification. *ICML '04: Proceedings of the twenty-first international conference on Machine learning* (pp. 265–272). New York, NY, USA: ACM Press.

Freund, Y., & Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. *Proc. 2nd European Conference on Computational Learning Theory* (pp. 23–37).

Giacinto, G., & Roli, F. (2001). An approach to the automatic design of multiple classifier systems. *Pattern Recognition Letters*, *22*, 25–33.

Margineantu, D. D., & Dietterich, T. G. (1997). Pruning adaptive boosting. *Proc. 14th International Conference on Machine Learning* (pp. 211–218). Morgan Kaufmann.

Martínez-Muñoz, G., & Suárez, A. (2004). Aggregation ordering in bagging. *Proc. of the IASTED International Conference on Artificial Intelligence and Applications* (pp. 258–263). Acta Press.

Martínez-Muñoz, G., & Suárez, A. (2005). Switching class labels to generate classification ensembles. *Pattern Recognition*, *38*, 1483–1494.

Nadeau, C., & Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, *52*, 239–281.

Prodromidis, A. L., & Stolfo, S. J. (2001). Cost complexity-based pruning of ensemble classifiers. *Knowledge and Information Systems*, *3*, 449–469.

Schapire, R. E., Freund, Y., Bartlett, P. L., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *12*, 1651–1686.

Tamon, C., & Xiang, J. (2000). On the boosting pruning problem. *Proc. 11th European Conference on Machine Learning* (pp. 404–412). Springer, Berlin.

Webb, G. I. (2000). Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, *40*, 159–196.

Zhou, Z.-H., & Tang, W. (2003). Selective ensemble of decision trees. *Lecture Notes in Artificial Intelligence* (pp. 476–483). Berlin: Springer.

Zhou, Z.-H., Wu, J., & Tang, W. (2002). Ensembling neural networks: Many could be better than all. *Artificial Intelligence*, *137*, 239–263.