Ensemble Pruning via Individual Contribution Ordering

Zhenyu Lu*, Xindong Wu*+, Xingquan Zhu@, Josh Bongard*
*Department of Computer Science, University of Vermont, Burlington, VT 05401, USA
+School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China
@ QCIS Center, Faculty of Engineering and Information Technology, University of Technology, Sydney, NSW 2007, Australia
zlu@uvm.edu; xwu@cems.uvm.edu; xqzhu@it.uts.edu.au; jbongard@uvm.edu

ABSTRACT

An ensemble is a set of learned models that make decisions collectively. Although an ensemble is usually more accurate than a single learner, existing ensemble methods often tend to construct unnecessarily large ensembles, which increases the memory consumption and computational cost. Ensemble pruning tackles this problem by selecting a subset of ensemble members to form subensembles that are subject to less resource consumption and response time with accuracy that is similar to or better than the original ensemble. In this paper, we analyze the accuracy/diversity trade-off and prove that classifiers that are more accurate and make more predictions in the minority group are more important for subensemble construction. Based on the gained insights, a heuristic metric that considers both accuracy and diversity is proposed to explicitly evaluate each individual classifier's contribution to the whole ensemble. By incorporating ensemble members in decreasing order of their contributions, subensembles are formed such that users can select the top p percent of ensemble members, depending on their resource availability and tolerable waiting time, for predictions. Experimental results on 26 UCI data sets show that subensembles formed by the proposed EPIC (Ensemble Pruning via Individual Contribution ordering) algorithm outperform the original ensemble and a state-of-the-art ensemble pruning method, Orientation Ordering (OO) [16].

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Database Applications - Data Mining

General Terms

Algorithms

Keywords

ensemble learning, ensemble pruning

1. INTRODUCTION

The construction of classifier ensembles is an active research field in machine learning and data mining communities [3]. Rather than relying on a single classifier, an ensemble is a set of classifiers that make decisions collectively. It is well accepted that an ensemble usually generalizes better than a single classifier [22]. Dietterich stated [8] "A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse." Since the diversity of the ensemble decreases with the increase in accuracies of ensemble members, the key to the success of any ensemble learning method is the appropriate handling of the trade-off between accuracy and diversity.

Many approaches have been proposed to create accurate and diverse ensembles. Examples include bagging [4], boosting [11], random forests [5], the random subspace method [12] and random decision trees [10]. In most ensemble methods, the diversity and accuracy are acquired by manipulating subsets of data points or features. One problem with these ensembling approaches is that they tend to construct unnecessarily large ensembles, which requires a large amount of memory to store the trained classifiers and decreases the response time for prediction. Ensemble pruning, or selective ensembles, is a technique that tackles this problem by choosing a subset of individual classifiers from a trained ensemble to form a subensemble for prediction. The classifiers in the subensemble need to be carefully chosen so that it is small enough to reduce the memory requirement and response time with predictive accuracy that is similar to or better than the original ensemble.

Given an ensemble of size M, the problem of finding the subset of ensemble members with the optimal generalization ability involves searching the space of $2^M - 1$ non-empty subensembles, which was proved to be an NP-complete problem [19]. Like other approaches in ensemble learning, the performance gain of the ensemble pruning methods stems from the balanced accuracy/diversity trade-off, where choosing only the most accurate individual classifiers to form the subensemble is theoretically unsound. As an informative example, Zhang et al. [23] has demonstrated that "an ensemble of three identical classifiers with 95% accuracy is worse than an ensemble of three classifiers with 67% accuracy and least pairwise correlated error (which is perfect!)." At the other extreme, as a strategy that only considers diversity for pruning, Kappa pruning [14] was shown to have "the poorest overall performance on the data sets investigated" of the ensemble pruning methods compared in [15].

Many approaches [15] exist for selecting good subensembles to achieve the accuracy gain and reduce the computational cost, where existing efforts roughly fall into the following two categories: (1) regarding ensemble pruning as a mathematical programming and optimization problem [23][25]; and (2) reordering the original ensemble members based on some predefined criteria [16], such as the classifiers' prediction accuracies, and selecting a subset of ensem-

KDD'10, July 25–28, 2010, Washington, DC, USA.

ble members from the sorted list. For solutions in the first category, the selected subensembles are unscalable to the users' request on the subensemble size, whereas for solutions in the second category, they fail to effectively integrate both accuracy and diversity, which are two critical measures for ensemble pruning. Indeed, while the correlation between diversity and ensemble learning have been observed for many years [13], the emphasis has been traditionally on the characterization of the diversity and the creation of the diverse classifiers. We currently lack effective pruning principles to assess the amount of "contribution" that each ensemble member can bring to the ensemble. Existing approaches fall short in answering some fundamental questions, such as, given two ensemble members, (1) what are the differences between their impact to the ensemble? (2) which one is better for ensemble learning if we have to prune out one of them? and (3) what are the effective criteria if both accuracy and diversity are considered for ensemble pruning?

To address the above concerns, we report, in this paper, our recent work on individual contribution ordering for scalable ensemble pruning. We argue that although obtaining the optimal ensemble pruning is NP-complete, two general properties can still be observed: (1) when the accuracies of individual members of two ensembles are similar, the more diverse ensemble should perform better than the other one; and (2) when two ensembles are similarly diverse, the one with more accurate individual members should perform better. To assert that these two properties are genuinely true for ensemble construction, we limit our analysis, in the paper, to a disagreement based diversity measure [12] and two-class learning problems and carry out a theoretical study to prove the validity of the first property. Because a similar proof can also be obtained for the second property, our theoretical analysis concludes that an individual classifier makes a more important contribution to the ensemble not only when it is more accurate, but also when its predictions more often disagree with the majority vote of the ensemble.

Based on this insight, a heuristic metric for evaluating contribution of each ensemble member that explicitly considers both accuracy and diversity is proposed in this paper. By reordering the ensemble members according to the proposed metric, an algorithm named Ensemble Pruning via Individual Contribution ordering (EPIC) that incorporates ensemble members with decreasing order of individual contributions into subensembles is introduced. Experimental results on 26 UCI data sets show that subensembles formed by EPIC are significantly more accurate than the original ensembles. In all experiments, bagging is chosen to be the method for constructing the original ensemble because it has been shown to be a safe and robust method [7]. To compare EPIC with peer ensemble pruning methods, Orientation Ordering (OO) [16] is chosen to be the control case, because it was shown to exhibit good performance with a low computational cost in [15]. Our experiments show that EPIC outperforms OO with a similar pruning time, which is $O(M\log(M))$ for an ensemble with size M.

The remainder of the paper is structured as follows. Section 2 reviews related work. Section 3 carries out theoretical study on the accuracy/diversity trade-off, proposes a metric for evaluating individual contribution, and introduces the EPIC algorithm. Section 4 reports the experimental settings and results, and we conclude the paper in Section 5.

2. RELATED WORK

Ensemble methods usually have two stages [16]: (1) training a set of accurate and diverse classifiers; and (2) combining individual classifiers, using strategies such as majority voting [4], weighted voting [11] or stacking [21], for predictions. Ensemble pruning can be viewed as a special case of the weighted voting approach, where each individual member is associated with a binary number indicating whether or not it should be included in the predicting process [23]. The major difference between ensemble pruning and other ensemble methods is that its main focus is to reduce memory and computational costs by choosing small subensembles for predictions, as well as maintaining or increasing the predictive accuracy.

In 1997, Margineantu and Dietterich [14] observed that although ensemble methods can achieve better generalization accuracy than a single learner, the resulting ensembles are often unnecessarily large. Sometimes the memory for storing the ensemble members is even larger than the size of the training set. A question was then raised: are all classifiers generated essential for the ensemble's performance? Through investigation of five pruning heuristics, they concluded that the size of the ensemble can be substantially reduced without serious impact on performance, and sometimes the pruned ensemble can even outperform the original ensemble. Tamon and Xiang [19] proved the problem of pruning boosting ensembles to be NP-complete.

In the work carried out by Zhou et al. [25], analysis showed that ensembling a subset of classifiers could be better than using all the classifiers. Further, a genetic algorithm (GA) based algorithm called GASEN was proposed to evolve the voting weights of individual classifiers in an ensemble of neural networks. Only the classifiers with weights above a certain threshold were chosen to form the subensembles for prediction. Experimental results of GASEN confirmed the conclusion that ensembles can be pruned to yield similar or better performance. A similar approach was used to prune ensembles of decision trees [24].

Zhang et al. [23] formulated the ensemble pruning problem as a quadratic integer programming problem, which is NP-hard. By applying semi-definite programming techniques, approximate solutions can be obtained in polynomial time for this NP-hard problem. Another work [6] first obtained an extensive library of more than 2000 classifiers trained by existing methods such as C4.5 [18] with different parameters, and then ensembles were constructed by selecting classifiers using metrics such as accuracy and ROC area.

The approach this paper takes belongs to a family of ensemble pruning methods based on ordered aggregation, where the original ensembles are reordered according to some criteria, and subensembles are constructed according to this order. In ensemble learning methods, the generalization error generally monotonically decreases with the increase of the ensemble size. Exceptions are when applying algorithms such as boosting to noisy data sets, the generalization error might increase because of overfitting. By reordering ensemble members, highest accuracy can be obtained for subensembles with an intermediate number of classifiers. A representative of this approach is [16], in which classifiers were reordered by increasing values of angles between the signature vectors (binary vectors that indicate each classifier's correctness in predicting each data point) and a reference vector. The algorithm introduced here differs from this work in that EPIC explicitly considers both accuracy and diversity.

In [17], Partalas et al. identified that an ensemble member's prediction on one data point can be divided into four subsets: (1) the subset in which the individual classifier predicts correctly and is in the minority group; (2) the subset in which the individual classifier predicts correctly and is in the majority group; (3) the subset in which the individual classifier predicts incorrectly and is in the minority group; and (4) the subset in which the individual classifier predicts incorrectly and is in the majority group. The paper concluded that considering all four cases is crucial to designing ensemble diversity measures and ensemble selection techniques.

Although the designing of EPIC is also based on the four cases, it

differs from [17] in two ways. Firstly, while both algorithms agree that considering the four cases is crucial, a more important question is understanding the relationship among the four cases and using it for pruning. [17] made an intuitive observation and concluded that ensemble members near to change status (correct/incorrect classification) are more important, which did not show the intrinsic relationship among the four cases. In this paper however, our theoretical analysis shows an unintuitive conclusion that ensemble members with more correct predictions in the minority contribute more for subensemble construction, and ensemble members with more incorrect predictions in the minority are less harmful for a subensemble's predictions. Thus EPIC is different from [17] because we show for the first time a strict order of importance for the four cases. Secondly, [17] is an iterative algorithm with a polynomial pruning time, while EPIC's pruning time is $O(M\log(M))$ for an ensemble with size M.

[2] is another work that considers the four cases. Like [17], [2] did not show the relationship among the four cases, and its pruning time is polynomial.

Recent work [15] gives a review and comparative study of the ensemble pruning literature.

3. ALGORITHM DESIGN

In this section, we formally study the accuracy/diversity trade-off and derive proper theories to quantify and characterize the importance of the individual classifier for subensemble construction. Based on the results of the analysis, a heuristic metric for evaluating a classifier's individual contribution is proposed, through which our EPIC algorithm can order and prune ensemble members to build subensemble in an effective way.

3.1 Individual Contribution Assessment

In this subsection, we first introduce the notations and the diversity measures used in this paper, and then derive properties to characterize ensemble members which are important for subensemble construction. As introduced in Section 1, two observations can be made from the accuracy/diversity trade-off: 1) when members of two ensembles are similarly accurate, the more diverse ensemble should perform better than the other one; and (2) when two ensembles are similarly diverse, the one with more accurate individual members should perform better. These observations mean that for a subensemble to outperform the original ensemble, incorporating ensemble members that are accurate but different from the peer members is crucial. In this paper, individual contribution of an ensemble member is defined in the context of ensemble pruning to indicate its expected performance in a subensemble.

The two observations can be transformed into another form to shed light on the evaluation of individual contributions: (1) when two individual classifiers in an ensemble are similarly accurate, the one that increases the diversity of the ensemble more should have a higher individual contribution; and (2) when two individual classifiers in an ensemble contributes similar diversity to the ensemble, the one with the higher accuracy should have a higher individual contribution. The first observation indicates how well a classifier is expected to work with other classifiers, and the second observation shows the individual property of a classifier. In our study, we formally prove the first observation and will show that similar proof can apply to the second observation. The two observations will therefore act as ensemble punning principles for the proposed EPIC algorithm.

Let $D = \{d_1, ..., d_N\}$ be a set of N data points where $d_i = \{(\mathbf{x}_i, y_i) \mid i \in [1, N]\}$ is a pair of input features and label that represents the ith data point, $C = \{c_1, ..., c_M\}$ be a set of M classifiers where $c_i(\mathbf{x}_j)$ gives the prediction of the ith classifier on the jth data point, $V = \{v^{(1)}, ..., v^{(N)} | v^{(i)} = [v_1^{(i)}, ..., v_L^{(i)}], i \in [1, N]\}$ be a set of vectors where $v_j^{(i)}$ is the number of predictions for the jth label of the ith data point of an ensemble combined with majority voting, and L is the number of output labels. Denote the accuracy of the ensemble as ACC_{en}, and the accuracy of the ith classifier as ACC_i.

Many measures [13] exist to evaluate the diversity (or distance) of classifiers or the diversity of an ensemble. In this paper, we employ a 0/1 loss based disagreement measure, which was proposed by Ho [12], to characterize the pair-wise diversity for ensemble members. The same analysis apply for others, such as the Q statistics or κ statistics [9], based measures.

Definition 1: Given two classifiers c_i and c_j, where $N^{(01)}$ denotes the number of data points incorrectly predicted by c_i but correctly predicted by c_j, and $N^{(10)}$ is the opposite of $N^{(01)}$, the *diversity* of c_i and c_j, denoted by $Div_{i,j}$, is the ratio between the sum of the number of data points correctly predicted by one of the classifiers only and the total number of data points, as given in Eq. (1).

$$\text{Div}_{i,j} = \frac{N^{(01)} + N^{(10)}}{N} \tag{1}$$

Definition 2: A classifier c_i's *diversity contribution* to an ensemble, denoted by $ConDiv_i$, is the sum of the diversities between c_i and each other classifier in the ensemble (excluding c_i because according to Eq. (1 a classifier's diversity to itself is zero), as given in Eq.(2).

$$\text{ConDiv}_i = \sum_{j=1}^{M} \text{Div}_{i,j} \tag{2}$$

Lemma 1: Given a two-class classification problem, the *diversity contribution* of a classifier c_i, defined in Eq. (2), is equal to Eq. (3)

$$\text{ConDiv}_i = \frac{1}{N} \sum_{k=1}^{N} (M - v_{c_i(\mathbf{x}_k)}^{(i)}) \tag{3}$$

Proof: For each data point d_k, denotes $v_{c_i(\mathbf{x}_k)}^{(i)}$ the number of classifiers that agree with c_i in prediction (including itself), and $(M - v_{c_i(\mathbf{x}_k)}^{(i)})$ is the number of classifiers that disagree with c_i in prediction. In a two-class learning task, each disagreement is counted once when calculating the diversity contribution of c_i, as defined in Eq. (2). Thus the sum of the disagreements on all data points divided by N is exactly equal to Eq.(2). ■

Lemma 2: Given a two-class classification problem, the expected *diversity contribution* of a classifier c_i, denoted by $E(\text{ConDiv}_i)$, is equal to Eq.(4), where $N_{\text{ma}}^{(i)}$ and $N_{\text{ma}}^{(i)}$ are the total number of data points that c_i votes in the majority and minority groups, respectively, and AVG_{ma} and AVG_{mi} are the average of the number of votes in the majority and minority groups, respectively.

$$E(\text{ConDiv}_i) = \frac{(N_{\text{ma}}^{(i)} \text{AVG}_{\text{mi}} + N_{\text{mi}}^{(i)} \text{AVG}_{\text{ma}})}{N} \tag{4}$$

Proof: For diversity contribution defined in Eq.(3), there are two cases for $(M - v_{c_i(\mathbf{x}_k)}^{(i)})$ in total: (1) when $v_{c_i(\mathbf{x}_k)}^{(i)} > (M - v_{c_i(\mathbf{x}_k)}^{(i)})$, which means the majority of classifiers vote for class $c_i(\mathbf{x}_k)$; and (2) when $v_{c_i(\mathbf{x}_k)}^{(i)} < (M - v_{c_i(\mathbf{x}_k)}^{(i)})$, which means the minority classifiers vote for class $c_i(\mathbf{x}_k)$. Ties are not considered here because it

is irrelevant to the conclusion and considering it will unnecessarily complicate the understanding of this paper. For a classifier c_i which is uniformly randomly drawn from the classifier pool, the expected sum of the $(M - v^{(i)}_{c_i(\mathbf{x}_k)})$ in case (1) is equal to $N^{(i)}_{\text{ma}}\text{AVG}_{\text{mi}}$. The expected sum of the $(M - v^{(i)}_{c_i(\mathbf{x}_k)})$ in case (2) equals $N^{(i)}_{\text{mi}}\text{AVG}_{\text{ma}}$, where $N^{(i)}_{\text{mi}}$ and AVG_{ma} are the counterparts of $N^{(i)}_{\text{ma}}$ and AVG_{mi} in case (1), respectively. Thus the expected diversity contribution of c_i is equal to Eq.(4). ■

Lemma 3: For classifiers c_i and c_j, the difference between their expected diversity contributions is equal to Eq.(5).

$$\frac{(N^{(i)}_{\text{ma}} - N^{(j)}_{\text{ma}})\text{AVG}_{\text{mi}} + (N^{(i)}_{\text{mi}} - N^{(j)}_{\text{mi}})\text{AVG}_{\text{ma}}}{N} \tag{5}$$

Proof: omitted due to simplicity.

In short, Lemma 3 provides an effective means to compares the diversity contributions of two equally accurate ensemble members, through which we are able to quantify each ensemble member and measure their importance for subensemble construction.

THEOREM 1: Given a two-class learning problem, and two classifiers c_i and c_j of a classifier ensemble E, if c_i and c_j are equally accurate, the classifier which disagrees more with other ensemble members is expected to contain more useful knowledge, in terms of diversity contribution, for constructing the classifier ensemble.

Proof: An individual classifier's prediction on the data points can be divided into four exclusive subsets: (1) the subset in which the individual classifier predicts correctly and is in the minority group; (2) the subset in which the individual classifier predicts correctly and is in the majority group; (3) the subset in which the individual classifier predicts incorrectly and is in the minority group; and (4) the subset in which the individual classifier predicts incorrectly and is in the majority group. Again, ties are not considered because of the irrelevance to the conclusion.

For a two-class classification problem, these four subsets are equivalent to: (1) the subset in which the individual classifier predicts correctly and the ensemble predicts incorrectly; (2) the subset in which the individual classifier predicts correctly and the ensemble predicts correctly; (3) the subset in which the individual classifier predicts incorrectly and the ensemble predicts correctly; and (4) the subset in which the individual classifier predicts incorrectly and the ensemble predicts incorrectly.

For the ith classifier,denote the fraction of data points that fall into these four sets as a_i, b_i, u_i, and t_i respectively (henceforth referred to as the a, b, u, t notation), their relations are:

$$a_i + b_i = \text{ACC}_i \tag{6}$$

$$u_i + t_i = 1 - \text{ACC}_i \tag{7}$$

where $a_i + b_i + u_i + t_i = 1$.

Eq. (5) can be rewritten using the a, b, u, t notation as:

$$\frac{(b_i + t_i - b_j - t_j)\text{AVG}_{\text{mi}} + (a_i + u_i - a_j - u_j)\text{AVG}_{\text{ma}}}{N} \tag{8}$$

because for classifier c_i, $N^{(i)}_{\text{ma}}$ is equal to $(b_i + t_i)$ and $N^{(i)}_{\text{mi}}$ is equal to $(a_i + u_i)$.

Assume the two classifiers have the same accuracy. From Eq. (6) and (7), we know that $a_i + b_i = a_j + b_j$ and $u_i + t_i = u_j + t_j$, which in another form is $a_i - a_j = b_j - b_i$ and $u_i - u_j = t_j - t_i$. Eq. (8) can therefore be rewritten as:

$$\frac{(a_i + u_i - a_j - u_j)(\text{AVG}_{\text{ma}} - \text{AVG}_{\text{mi}})}{N} \tag{9}$$

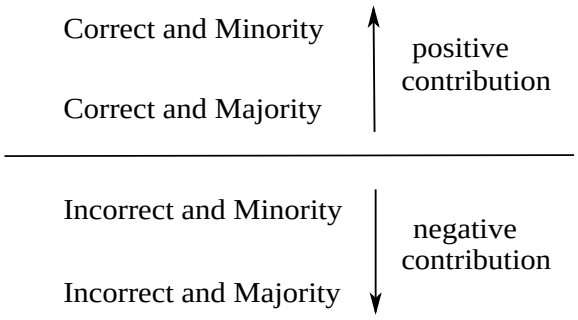


Figure 1: Rules for evaluating contributions of predictions

Since $(\text{AVG}_{\text{ma}} - \text{AVG}_{\text{mi}}) > 0$, it is clear that if $a_i + u_i > a_j + u_j$, formula (9) is greater than 0, which means that if two classifiers have the same accuracy, the classifier with more votes that are in the minority group is expected to bring more diversity contribution to the ensemble. In other words, the classifier which disagrees more with other ensemble members is expected to contain more useful knowledge for constructing subensembles. ■

Following the proof of the THEOREM 1, we can also assert that if two classifiers contribute equal diversity to the ensemble, the more accurate one is more important for constructing subensembles. The proof of the claim is omitted due to space constraints. Note that this claim is intuitive, though, because each correct prediction on a data point increases the probability of the ensemble to predict it correctly, thus making a positive contribution, and each incorrect prediction on a data point makes a negative contribution.

We note that although the derivation of the THEOREM 1 is based on the two-class learning problem, it can be generalized for multi-class problems. The assumptions in this paper are made to assist the analysis so that better understanding of the problem can lead to better design of individual contribution measures. Thorough analysis is of future interest but beyond the scope of this paper.

3.2 Evaluation Metric for Ensemble Members

The analysis in the previous section concludes following two rules for designing a heuristic metric for evaluating individual contributions of ensemble members: (1) correct predictions make positive contributions, incorrect predictions make negative contributions; and (2) correct predictions that are in the minority group make more positive contributions than correct predictions that are in the majority group, and incorrect predictions that are in the minority group make less negative contributions than incorrect predictions that are in the majority group. Fig. 1 gives an illustration of the rules.

The individual contribution of a classifier c_i is therefore defined as:

$$\text{IC}_i = \sum_{j=1}^{N} \text{IC}_i^{(j)} \tag{10}$$

where $\text{IC}_i^{(j)}$ is c_i's contribution on the jth data point d_j.

When $c_i(\mathbf{x}_j)$ equals y_j, which means c_i makes correct predictions on d_j, if $c_i(\mathbf{x}_j)$ is in the minority group, $\text{IC}_i^{(j)}$ is defined as:

$$\text{IC}_i^{(j)} = 2v^{(j)}_{\text{max}} - v^{(j)}_{c_i(\mathbf{x}_j)} \tag{11}$$

where $v^{(j)}_{\text{max}}$ is the number of majority votes on d_j, and $v^{(j)}_{c_i(\mathbf{x}_j)}$ is

the number of predictions on $c_i(\mathbf{x}_j)$, as defined before. In this equation, $(v^{(j)}_{\text{max}} - v^{(j)}_{c_i(\mathbf{x}_j)})$ is an estimation of the "degree of positive contribution". For example, given an ensemble of size 100, d_j and d_k are two data points that have been incorrectly predicted by the whole ensemble in data set D where the number of labels L equals 3. Suppose $v^{(j)} = [35,\ 33,\ 32]$, $v^{(k)} = [68,\ 0,\ 32]$, and $y_j = y_k = 3$, therefore $v^{(j)}_{\text{max}} = 35$ and $v^{(k)}_{\text{max}} = 68$. Although $v^{(j)}_3 = v^{(k)}_3$, it is harder for a subensemble to correct the wrong prediction of the ensemble for d_k, because the difference between $v^{(k)}_{max}$ and $v^{(k)}_3$ is larger than the difference between $v^{(j)}_{\text{max}}$ and $v^{(j)}_3$. The harder the correction is, the more valuable the predictions in the minority group are. Intuitively, the less classifiers that make the correct predictions with a classifier on d_j, the more valuable a classifier is. The "degree of positive contribution", which indicates how hard it is for a subensemble to correct the incorrect predictions of the whole ensemble, can therefore be estimated by $(v^{(j)}_{\text{max}} - v^{(j)}_{c_i(\mathbf{x}_j)})$. The extra $v^{(j)}_{\text{max}}$ in this equation is a normalization term that is used together with Eq. (12).

When $c_i(x_j)$ equals y_j and $c_i(x_j)$ is in the majority group (in this case $v^{(j)}_{c_i(\mathbf{x}_j)} = v^{(j)}_{\text{max}}$), $\text{IC}^{(j)}_i$ is defined as:

$$\text{IC}^{(j)}_i = v^{(j)}_{\text{sec}} \tag{12}$$

where $v^{(j)}_{\text{sec}}$ is the second largest number of votes on the labels of d_j. $(v^{(j)}_{\text{sec}} - v^{(j)}_{\text{max}})$ is an estimation of the "degree of positive contribution" in this case. Intuitively, if the majority of classifiers predicts correctly with a classifier on d_j, this classifier's contribution is not very valuable because without its prediction, the ensemble would still be correct on d_j (assuming no tie). Note that $(v^{(j)}_{\text{sec}} - v^{(j)}_{\text{max}})$ is negative. According to our rules for designing the individual contribution measure, all correct predictions make positive contributions. Thus a term $v^{(j)}_{\text{max}}$ is added to $(v^{(j)}_{\text{sec}} - v^{(j)}_{\text{max}})$ to normalize it to always be positive, which gives Eq. (12). And $v^{(j)}_{\text{max}}$ is added to $(v^{(j)}_{\text{max}} - v^{(j)}_{c_i(\mathbf{x}_j)})$ to maintain their relative order, which gives Eq. (11).

When $c_i(\mathbf{x}_j)$ does not equal y_j, $\text{IC}^{(j)}_i$ is defined as:

$$\text{IC}^{(j)}_i = v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(\mathbf{x}_j)} - v^{(j)}_{\text{max}} \tag{13}$$

where $v^{(j)}_{\text{correct}}$ is the number of votes for the correct label of d_j. The two negative cases shown in Fig. 1 can be considered together. Similar to the discussion of "degree of positive contribution", the "degree of negative contribution" is estimated by $(v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(\mathbf{x}_j)})$, which is the difference between the number of votes on the correct label and the number of votes on $c_i(\mathbf{x}_j)$. For example, given an ensemble of size 100, d_j and d_k are two data points in data set D where L = 3. Suppose $v^{(j)} = [40,\ 30,\ 30]$, $v^{(k)} = [40,\ 50,\ 10]$, $y_j = y_k = 2$, and $c_i(\mathbf{x}_j) = c_i(\mathbf{x}_j) = 1$. Therefore $v^{(j)}_{\text{correct}} = 30$ and $v^{(k)}_{\text{correct}} = 50$. Although $v^{(j)}_{c_i(\mathbf{x}_j)} = v^{(j)}_{c_i(\mathbf{x}_k)} = 40$, c_i makes a more negative contribution on d_j than d_k because $v^{(j)}_{c_i(\mathbf{x}_j)} > v^{(j)}_{\text{correct}}$, thus c_i makes a negative contribution that partially leads to the fact that the ensemble predicts incorrectly on d_i. The "degree of negative contribution", which indicates how much a classifier's predictions might negatively affect a subensemble, can therefore be estimated by $(v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(\mathbf{x}_j)})$. Note that $(v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(\mathbf{x}_j)})$ could give a positive value, but according to our designing rules incorrect predictions should make negative contributions. So a term $(-v^{(j)}_{\text{max}})$ is added to $(v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(\mathbf{x}_j)})$ to normalize it to always be negative.

Combining Eq. (11), Eq. (12), Eq. (13) with Eq. (10), the individual contribution of the classifier c_i is:

$$\text{IC}_i = \sum_{j=1}^{N} (\alpha_{ij}(2v^{(j)}_{\text{max}} - v^{(j)}_{c_i(\mathbf{x}_j)}) + \beta_{ij} v^{(j)}_{\text{sec}} + \theta_{ij}(v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(\mathbf{x}_j)} - v^{(j)}_{\text{max}})) \tag{14}$$

where:

$$\alpha_{ij} = \begin{cases} 1 & \text{if } c_i(\mathbf{x}_j) = y_j \text{ and } c_i(\mathbf{x}_j) \text{ is in the minority group;} \\ 0 & \text{otherwise.} \end{cases}$$

$$\beta_{ij} = \begin{cases} 1 & \text{if } c_i(\mathbf{x}_j) = y_j \text{ and } c_i(\mathbf{x}_j) \text{ is in the majority group;} \\ 0 & \text{otherwise.} \end{cases}$$

$$\theta_{ij} = \begin{cases} 1 & \text{if } c_i(\mathbf{x}_j) \neq y_j; \\ 0 & \text{otherwise.} \end{cases}$$

Table 1: EPIC

Input: the training set D_{tr}, the testing set D_{te}, the pruning set D_{pr}, an ensemble $C = \{c_1, ..., c_M\}$ that is trained on D_{tr}, and the parameter p which is the desired percentage of classifiers in C that should be kept in the output subensemble.

Initialize: a list of vectors $V = \{v^{(1)}, ..., v^{(N_{\text{pr}})} | v^{(i)} = [v^{(i)}_1, ..., v^{(i)}_L], i \in [1, N_{\text{pr}}]\}$, where $v^{(i)}_j = 0$ is initial number of predictions in label j on the ith data point in D_{pr}, N_{pr} is the size of D_{pr}, L is the number of class labels, and OL is an empty list.

Pruning:

1. For each c_i in C:
   for each d_j in D_{pr}:
   get $c_i(\mathbf{x}_j)$, which is c_i's predictions on d_j;
   $v^{(j)}_{c_i(\mathbf{x}_j)} = v^{(j)}_{c_i(\mathbf{x}_j)} + 1$;
2. For each c_i in C:
   $\text{IC}_i = \sum_{j=1}^{N} (\alpha_{ij}(2v^{(j)}_{\text{max}} - v^{(j)}_{c_i(x_j)}) + \beta_{ij} v^{(j)}_{\text{sec}}$
   $+\theta_{ij}(v^{(j)}_{\text{correct}} - v^{(j)}_{c_i(x_j)} - v^{(j)}_{\text{max}}))$;
   append pair (c_i, IC_i) to OL;
3. Order OL in decreasing order of IC_i;

Output: the classifiers in the first p percent of OL is outputted as the pruned subensemble.

3.3 Ensemble Pruning via Individual Contribution Ordering

The inputs to EPIC are a training set D_{tr}, a testing set D_{te}, a pruning set D_{pr} (it is called the selection set in [15]), which is an

Table 2: Characteristics of data sets used in the experiments

Data sets	Size	#Features	#Classes
Anneal	898	38	6
Autos	205	25	6
Balance Scale	625	4	3
Breast-w	699	9	2
Car	1728	6	4
Glass	214	9	7
Heart-h	294	13	5
Hypothyroid	3772	29	4
Ionosphere	351	34	2
Kr-vs-kp	3196	36	2
Letter	20000	16	26
Mfeat-factors	2000	216	10
Mfeat-fourier	2000	76	10
Mfeat-karhunen	2000	64	10
Mfeat-pixel	2000	240	10
Mfeat-zernike	2000	47	10
Nursery	12960	8	5
Optdigits	5620	64	10
Pendigits	10992	16	10
Primary tumor	339	17	22
Segment	2310	19	7
Soybean	683	35	19
Spambase	4601	57	2
Splice	3190	60	3
Vehicle	846	18	4
Vowel	990	12	11

independent set for calculating the individual contributions of ensemble members, a predefined parameter p which is the percentage of the desired subensemble in terms of the size of the original ensemble, and an ensemble $C = \{c_1, ..., c_M\}$ that is trained on D_{tr}.

The pruning process starts by collecting the predictions of each ensemble on each data point in D_{pr}. The results are recorded in $V = \{v^{(1)}, ..., v^{(N)} | v^{(i)} = [v_1^{(i)}, ..., v_L^{(i)}], i \in [1, N]\}$. Then the individual contribution of each classifier in C on the pruning set D_{pr} is calculated using Eq. (14), this step takes $O(MN_{\text{pr}})$ time, where N_{pr} is the size of D_{pr}. Then the classifiers in C are ordered by decreasing values of their contributions. The result is stored in a list OL. This step takes $O(M\log(M))$ time.

The output of EPIC is the first p percent of the classifiers in the ordered list OL. The running time of EPIC is $O(MN_{\text{pr}}+M\log(M))$. Since N_{pr} is a constant for a given data set, EPIC's time complexity is $O(M\log(M))$. Table 1 gives the pseudo-code of EPIC.

4. EXPERIMENTAL EVALUATION

This section first introduces the settings of experiments and the characteristics of the data sets tested in this paper, and then the experimental results of the comparison between EPIC, OO and bagging are given.

4.1 Experimental Settings and Data Sets

26 data sets were chosen from the UCI machine learning repository [1] to evaluate the performance of EPIC. Each data set was randomly divided into three subsets with equal sizes. There are six permutations of the three subsets. Experiments on each data set consisted of six sets of sub-experiments. Each set of sub-experiments used one of the subsets as the training set, one of the subsets as the testing set, and the other one as the pruning set, corresponding to the order of one of the six permutations. And each set of sub-experiments consisted of 50 independent trials. Therefore a total of 300 trials of experiments were conducted on each data set. In the experiments of this paper, both EPIC and OO used the independent pruning set for pruning.

A bagging ensemble was trained in each trial. The base learner was J48, which is a Java implementation of C4.5 [18] from Weka [20]. In all experiments, the ensemble size was 200. Both EPIC and OO were used to reorder the same ensemble in each trial. The predictive accuracies of the original ensemble, the EPIC-ordered ensemble and the OO-ordered ensemble were recorded. Table 2 summarizes the characteristics of the data sets.

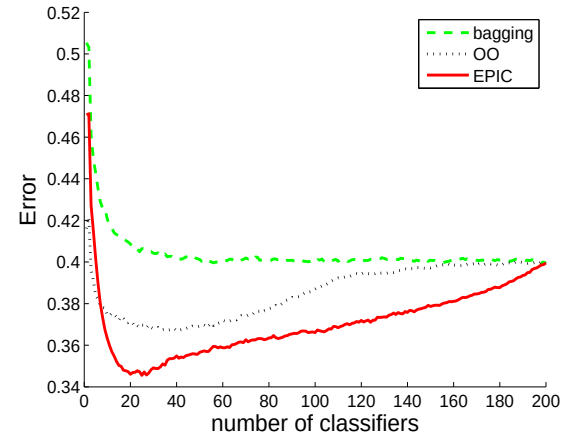
4.2 Experimental Results

The experimental results of the 26 tested data sets have four cases: (1) EPIC outperforms both the original ensemble and OO; (2) EPIC outperforms the original ensemble, and EPIC and OO perform comparably; (3) EPIC outperforms the original ensemble, and OO outperforms EPIC; and (4) the original ensemble outperforms both EPIC and OO. The first case contains 15 data sets, the second case contains eight data sets, the third case contains two cases, and the last case contains one data set. The results are reported both in figures and a summarizing table, where the figures show representative results from the four cases, and the table reports results for all tested data sets. The calculation method of the four cases will be introduced when describing the summarizing table.

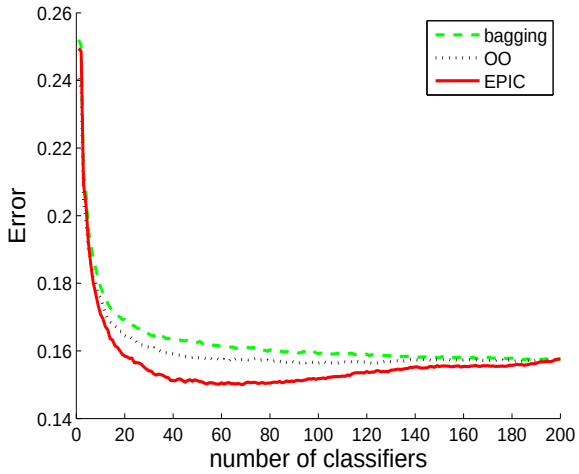
Figure 2 reports the error curves of the three compared methods for six representative data sets in the first case. Results in the figures are reported as curves of average errors with regard to the number of classifiers. The standard deviations are not reported in the figures for reasons of clarity, but are reported in the table. Classifiers in bagging ensembles are trained on training sets that are uniformly sampled from the initial training set with replacement. As the number of aggregated classifiers increases, the errors of bagging ensembles typically monotonically decrease at a decreasing rate and reach an asymptotic error when the ensemble size is large. When aggregating classifiers according to the orders given by EPIC or OO, a typical error curve drops rapidly, reaches the minimum error in the intermediate steps of aggregation which is lower than the error of the whole original ensemble, and then increase until the error is the same as the whole ensemble. Figure 2(a) shows typical error curves of bagging, EPIC and OO in the first case, where both EPIC and OO reach accuracies that are better than the whole ensemble at intermediate ensemble sizes, and EPIC's error curve is below the error curve of OO. The remaining five data sets, "Balance Scale", "Mfeat-pixel", "Primary-tumor", "Vehicle" and "Vowel" have similar error curves to "Autos".

Figure 3 reports results for two representative data sets from the second case, where EPIC outperforms the original ensemble, and EPIC and OO perform comparably. The error curves for "Car" is reported in Figure 3(a). Although EPIC achieves a lower error curve than OO, the two algorithms have comparable performance because the difference is not statistically significant. Detailed accuracies are reported later in the summarizing table. As reported in figure 3(b), the error curves for EPIC and OO are superimposed on each other.

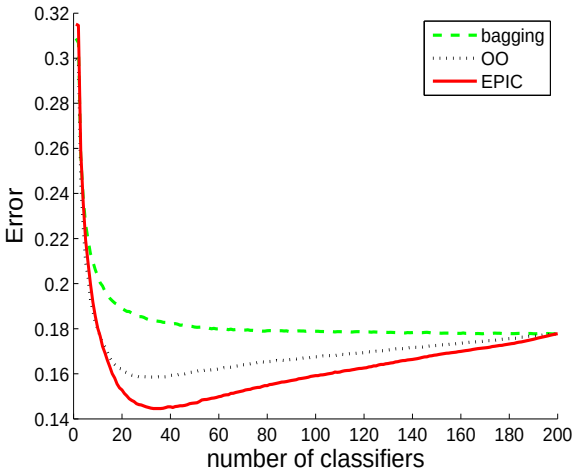
Figure 4 reports the error curves for the two data sets in third case, where EPIC outperforms the original ensemble, and OO outperforms EPIC. Figure 4(a) shows the result for "Anneal", in which OO achieves lower errors than EPIC when the size of the subensembles are smaller than 50. EPIC's average errors are better than OO when the subensemble size is larger than 50. This is a negative case for EPIC because OO reaches lower errors with less classi-



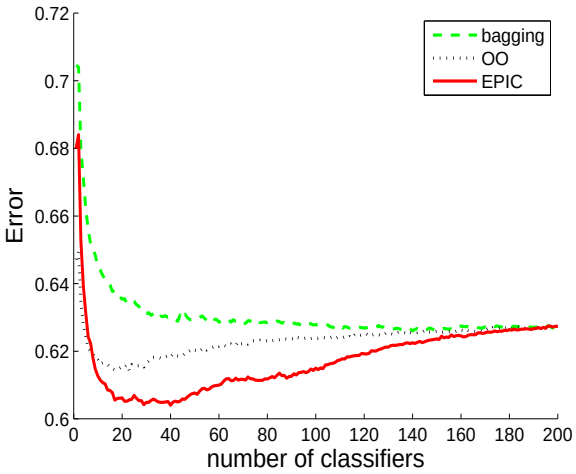
(a) “Autos”



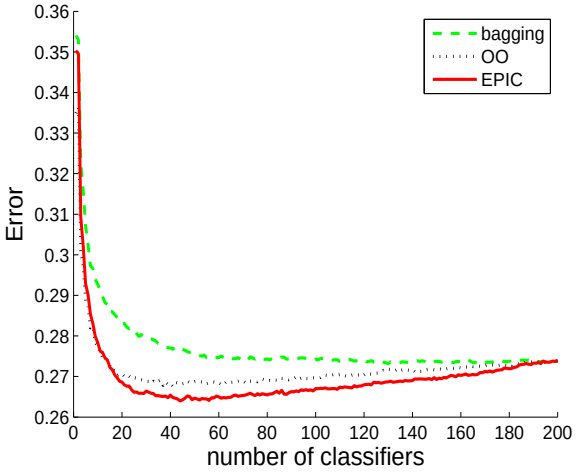
(b) “Balance Scale”



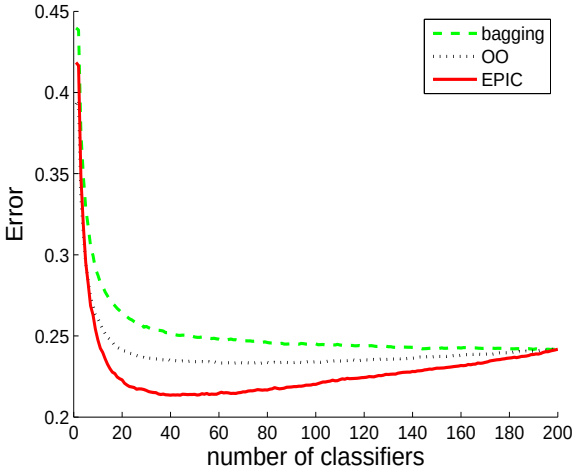
(c) “Mfeat-pixel”



(d) “Primary-tumor”

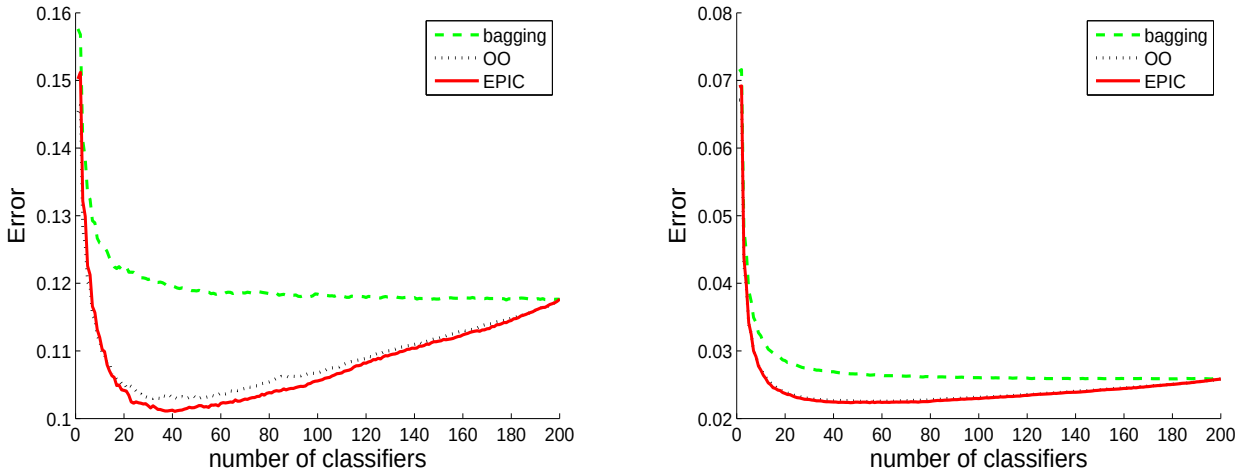


(e) “Vehicle”



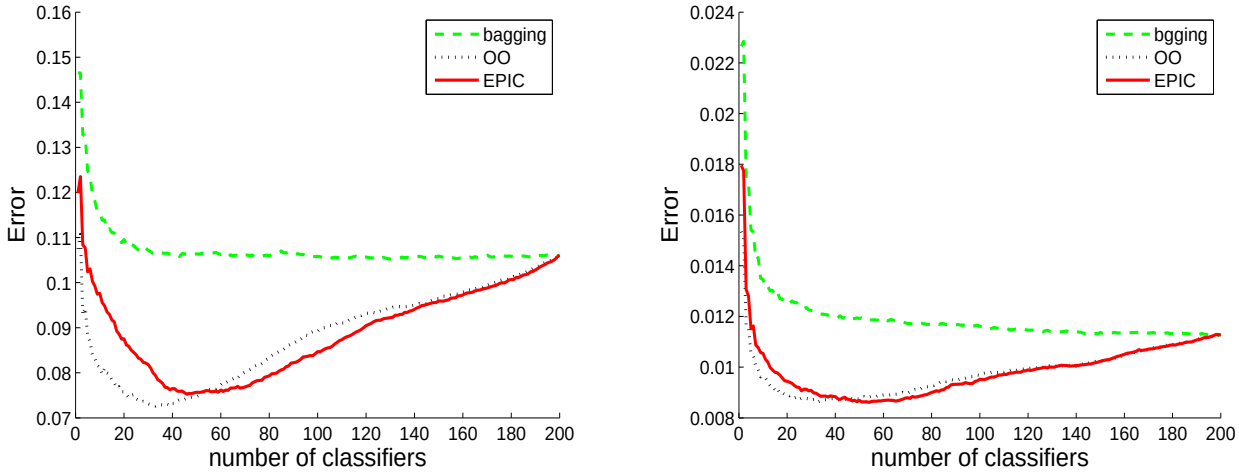
(f) “Vowel”

Figure 2: Comparative results for six data sets in the first case.



(a) "Car" (b) "Pendigits"

Figure 3: Comparative results for two data sets in the second case.



(a) "Anneal" (b) "Kr-vs-kp"

Figure 4: Comparative results for all data sets in the third case.

fiers, which is crucial for ensemble pruning methods. Figure 4(b) reports the result for "Kr-vs-kp", which is similar to the result for "Anneal" although the difference between EPIC and OO is smaller.

Figure 5 reports the result for "Heart-h". As is shown by the error curves, when the size of the subensembles are smaller than 100, the original bagging ensemble achieve lower errors than both EPIC and OO, and EPIC and OO achieve lower errors when the ensemble size is larger than 100. This means that both EPIC and OO fail to efficiently prune the original ensemble. Of the two pruning methods, the errors of EPIC are lower than OO when the subensemble size is smaller than 40, then OO achieves lower errors than EPIC until the ensemble size reaches 100. Above this ensemble size the two methods perform similarly.

Table 3 summarizes the results for the 26 data sets. Each cell in the table reports the mean and standard deviation of 300 trials. Experimental results in this paper empirically show that EPIC and OO generally reach minimum errors when the size of the subensembles are between 15% and 30%. [16] also reported similar observations. For this reason, in this table results of EPIC and OO are reported as the predictive accuracies of the subensembles constructed by using the first 15% and 30% of classifiers in the ordered lists. Results of bagging ensembles are reported as the predictive accuracies of the whole ensembles. Subensembles formed by EPIC with sizes of 15% of the original ensembles (henceforth referred to as "EPIC 15%") are compared with subensembles formed by OO with the same sizes (henceforth referred to as "OO 15%") and the whole ensembles using pair-wise t-tests at the 95% significance level. Subensembles formed by EPIC with sizes of 30% of the original ensembles (henceforth referred to as "EPIC 30%") are compared with subensembles formed by OO with the same size (henceforth referred to as "OO 30%") and the whole ensembles. The meanings of the symbols $\oplus$, $\odot$, and $\ominus$ are explained in the caption of the table. For results of bagging ensembles, the first symbol at the end shows the comparative results between "EPIC 15%" and itself, and

Table 3: Classification error percentages of bagging, EPIC and OO. $\oplus$ represents that EPIC outperforms the comparing method in pair-wise t-tests at 95% significance level, $\odot$ represents EPIC is comparable to the comparing method, and $\ominus$ represents EPIC is outperformed by the comparing method.

	Bagging (%)	EPIC + 15% (%)	EPIC + 30% (%)	OO + 15% (%)	OO + 30% (%)
Anneal	10.61±2.9⊕⊕	8.19±1.97	7.62±2.15	7.31±2.1⊖	7.73±2.31⊙
Autos	39.94±7.25⊕⊕	34.91±7.36	35.88±7.51	36.79±8.11⊕	37.19±7.61⊕
Balance Scale	15.75±1.52⊕⊕	15.41±1.58	15.06±1.64	16.12±1.66⊕	15.72±1.5⊕
Breast-w	4.48±1.14⊕⊙	4.19±0.95	4.45±0.88	4.46±0.91⊕	4.47±0.83⊙
Car	11.76±2.13⊕⊕	10.19±1.93	10.23±1.94	10.31±1.92⊙	10.38±1.97⊙
Glass	30.71±2.99⊕⊕	29.43±5.24	28.85±4.63	30.57±4.51⊕	30.81±4.39⊕
Heart-h	21.6±4.18⊖⊖	22.41±3.99	22.36±3.55	23.23±3.72⊕	21.93±3.44⊙
Hypothyroid	0.66±0.23⊕⊕	0.55±0.16	0.54±0.13	0.54±0.12⊙	0.58±0.15⊕
Ionosphere	13.22±3.09⊕⊕	11.21±2.85	10.1±3.53	11.18±2.92⊙	10.63±3.67⊙
Kr-vs-kp	1.13±0.35⊕⊕	0.91±0.17	0.87±0.16	0.87±0.14⊖	0.89±0.17⊙
Letter	9.51±0.55⊖⊕	9.65±0.64	8.99±0.48	9.58±0.47⊙	9.21±0.46⊕
Mfeat-factors	7.09±0.72⊕⊕	6.18±0.89	6.05±0.78	6.4±0.83⊕	6.31±0.82⊕
Mfeat-fourier	21.75±0.95⊕⊕	21.04±1.3	20.88±1.2	21.63±1.26⊕	21.45±1.11⊕
Mfeat-karhunen	12.07±1.72⊕⊕	10.52±1.63	10.28±1.59	11.0±1.5⊕	10.87±1.5⊕
Mfeat-pixel	17.78±4.31⊕⊕	14.54±3.0	14.98±3.65	15.87±3.34⊕	16.22±3.97⊕
Mfeat-zernike	25.22±1.54⊕⊕	24.68±1.57	24.53±1.47	25.42±1.53⊕	25.11±1.46⊕
Nursery	4.98±0.26⊕⊕	4.52±0.32	4.62±0.29	4.52±0.33⊙	4.6±0.3⊙
Optdigits	5.0±0.49⊕⊕	4.31±0.44	4.1±0.44	4.34±0.45⊙	4.25±0.43⊕
Pendigits	2.59±0.29⊕⊕	2.28±0.28	2.24±0.25	2.3±0.28⊙	2.26±0.27⊙
Primary-tumor	62.74±4.88⊕⊕	60.49±5.51	61.0±5.35	61.65±5.39⊕	62.15±5.18⊕
Segment	4.57±0.52⊕⊕	3.71±0.56	3.9±0.57	3.8±0.56⊙	3.91±0.57⊙
Soybean	14.48±3.64⊕⊕	11.08±2.95	11.25±2.81	11.37±3.09⊙	11.6±3.09⊙
Spambase	6.89±0.36⊕⊕	6.45±0.31	6.48±0.28	6.47±0.3⊙	6.5±0.27⊙
Splice	7.46±0.83⊕⊕	6.6±0.99	6.7±0.93	6.63±0.96⊙	6.72±0.89⊙
Vehicle	27.37±1.9⊕⊕	26.64±1.7	26.5±1.74	26.88±1.59⊙	26.85±1.8⊕
Vowel	24.18±2.74⊕⊕	21.65±2.17	21.47±2.34	23.59±2.3⊕	23.35±2.3⊕
EPIC 15% win/tie/loss	24/0/2			12/12/2	
EPIC 30% win/tie/loss	24/1/1				14/12/0

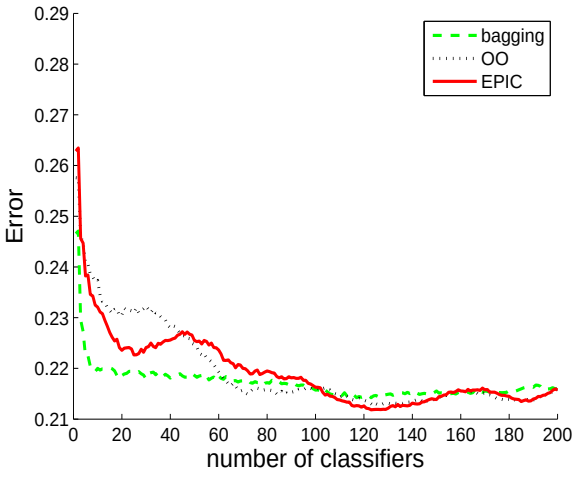


Figure 5: Comparative results for 'Heart-h" in the fourth case.

the second symbol shows the comparative results between "EPIC 30%" and itself. The cumulative win/tie/loss results of the 26 data sets are shown in the bottom rows of the table.

Data sets are divided into the four cases of results using the following rules: (1) "Heart-h" is divided into the fourth case because both EPIC and OO are outperformed by the whole ensemble; (2) each of the remaining data sets is divided into the first case if one of "EPIC 15%" and "EPIC 30%" outperforms the whole bagging ensemble and the other either outperforms or performs comparably to the whole ensemble, and one of "EPIC 15%" and "EPIC 30%" outperforms its counterpart of OO, and the other either outperforms of performs comparably to the counterpart of OO; (3) each of the remaining data sets is divided into the second case if one of "EPIC 15%" and "EPIC 30%" outperforms the whole bagging ensemble and the other either outperforms or performs comparably to the whole ensemble, and both "EPIC 15%" and "EPIC 30%" perform comparably to its counterpart of OO; and (4) the remaining data sets are divided into the third case. "Letter" is an exception in the first case, because "EPIC 15%" is outperformed by the whole ensemble. It is still divided into the first case "EPIC 30%" outperforms the whole ensemble and OO.

As is shown, EPIC outperforms bagging on 24 out of 26 data sets when using both 15% and 30% of all classifiers, which shows that EPIC efficiently performs pruning by achieving better predictive accuracies with small subensembles. EPIC is also shown to outperform OO when subensembles are formed with 15% and 30% of classifiers in the original ensemble. As introduced before, the pruning time of both EPIC and OO is $M\log(M)$.

5. CONCLUSIONS AND FUTURE WORK

In this paper, an effective ensemble pruning method termed Ensemble Pruning via Individual Contribution ordering (EPIC) is introduced, which orders individual classifiers in an ensemble in terms

of their importance to subensemble construction. We argued that although obtaining the optimal ensemble pruning solution is NP-complete, some general properties can still be derived for ensemble pruning. To achieve the goal, we limited the scope of our analysis to pair-wise diversity and two-class learning problems and have proved some general principles for ensemble pruning. Based on the derived principles, the proposed EPIC algorithm can sort ensemble members into a list, through which a subensemble with arbitrary size can be formed by selecting the top p percent of ensemble members. Experimental comparisons on 26 UCI data sets showed that EPIC outperforms both the original ensemble and a state-of-the-art ensemble pruning method, OO [16].

Although the original goal of ensemble pruning methods is to reduce the memory consumption and the computational cost with minimum impact on predictive accuracy, subensembles constructed with recent pruning methods such as OO and the method introduced in this paper are consistently more accurate than whole ensembles, which suggests that it is practically unnecessary, even without the resource consumption concerns, to keep all ensemble members for predictions.

In the paper, we limited our scope to specific diversity measures and two-class learning problems, and used Bagging to generate the initial pool of classifiers. But the general principles derived in the paper are valid for broad diversity measures, multi-class learning problems, and any other ensemble methods. Future work will involve testing EPIC's performance when working with other ensemble methods, designing better individual contribution measures and automating the selection of the sizes of the subensembles.

Acknowledgements

This research is supported in part by the US National Science Foundation (NSF) under grants EPS-0701410 and CCF-0905337, in part by the National Natural Science Foundation of China (NSFC) under award 60828005, and in part by the National Basic Research Program of China (973 Program) under award 2009CB326203.

6. REFERENCES

[1] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[2] R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer. Ensemble diversity measures and their application to thinning. *Information Fusion*, 6(1):49–62, 2005.

[3] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldá. New ensemble methods for evolving data streams. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pages 139–148, 2009.

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

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5 – 32, 2001.

[6] R. Caruana, A. Munson, and A. Niculescu-Mizil. Getting the most out of ensemble selection. In *Proceedings of the 6th International Conference on Data Mining*, pages 828–833, 2006.

[7] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 161 – 168, 2006.

[8] T. G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the 1st International Workshop on Multiple Classifier Systems*, pages 1–15, 2000.

[9] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization. *Machine Learning*, 40:2:139–157, 2000.

[10] W. Fan, H. Wang, P. S. Yu, and S. Ma. Is random model better? on its accuracy and efficiency. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 51 – 58, 2003.

[11] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

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

[13] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181 – 207, 2003.

[14] D. D. Margineantu and T. G. Dietterich. Pruning adaptive boosting. In *Proceedings of the 14th International Conference on Machine Learning*, pages 211 – 218, 1997.

[15] G. Martínez-Munoz, D. Hernández-Lobato, and A. Suárez. An analysis of ensemble pruning techniques based on ordered aggregation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):245–259, 2009.

[16] G. Martínez-Munoz and A. Suárez. Pruning in ordered bagging ensembles. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 609 – 616, 2006.

[17] I. Partalas, G. Tsoumakas, and I. Vlahavas. Focused ensemble selection: A diversity-based method for greedy ensemble selection. In *Proceeding of the 18th European Conference on Artificial Intelligence*, pages 117–121, 2008.

[18] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann, San Francisco, CA, USA, 1993.

[19] C. Tamon and J. Xiang. On the boosting pruning problem. In *Proceedings of the 11th European Conference on Machine Learning*, pages 404 – 412, 2000.

[20] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2005.

[21] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241 – 259, 1992.

[22] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.

[23] Y. Zhang, S. Burer, and W. N. Street. Ensemble pruning via semi-definite programming. *The Journal of Machine Learning Research*, 7:1315 – 1338, 2006.

[24] Z.-H. Zhou and W. Tang. Selective ensemble of decision trees. *Lecture Notes in Computer Science*, 2639:476 – 483, 2003.

[25] Z.-H. Zhou, J. Wu, and W. Tang. Ensembling neural networks: many could be better than all. *Artificial Intelligence*, 137(1-2):239 – 263, 2002.