Multi-Class Imbalance Problems: Analysis and Potential Solutions

Shuo Wang, *Member, IEEE,* and Xin Yao, *Fellow, IEEE*

***Abstract*—Class imbalance problems have drawn growing interest recently because of their classification difficulty caused by the imbalanced class distributions. Especially, many ensemble methods have been proposed to deal with such imbalance. However, most efforts so far are only focused on two-class imbalance problems. There are unsolved issues in multi-class imbalance problems, which exist in real-world applications. This paper studies the challenges posed by the multi-class imbalance problems and investigates the generalization ability of some ensemble solutions, including our recently proposed algorithm AdaBoost.NC [1], with the aim of handling multi-class and imbalance effectively and directly. We first study the impact of multi-minority and multi-majority on the performance of two basic resampling techniques. They both present strong negative effects. "Multi-majority" tends to be more harmful to the generalization performance. Motivated by the results, we then apply AdaBoost.NC to several real-world multi-class imbalance tasks and compare it to other popular ensemble methods. AdaBoost.NC is shown to be better at recognizing minority class examples and balancing the performance among classes in terms of G-mean without using any class decomposition.**

***Index Terms*—multi-class imbalance problems, ensemble learning, negative correlation learning, diversity, Boosting.**

I. Introduction

CLASS imbalance learning refers to a type of classification problems, where some classes are highly under-represented compared to other classes. The skewed distribution makes many conventional machine learning algorithms less effective, especially in predicting minority class examples. The learning objective can be generally described as "obtaining a classifier that will provide high accuracy for the minority class without severely jeopardizing the accuracy of the majority class" [2]. A number of solutions have been proposed at the data and algorithm levels to deal with class imbalance. In particular, ensemble techniques have become popular and important means, such as BEV [3], SMOTEBoost [4], etc. However, the efforts so far are focused on two-class imbalance problems in the literature.

In practice, many problem domains have more than two classes with uneven distributions, such as protein fold classification [5] [6] [7] and weld flaw classification [8]. These multi-class imbalance problems pose new challenges that are not observed in two-class problems. Zhou et al. [9] showed that dealing with multi-class tasks with different misclassification costs of classes is harder than dealing with two-class ones. Further investigations are necessary to explain what problems multi-class can cause to existing class imbalance learning techniques and how it affects the classification performance. Such information would help us to understand the multi-class issue better, and can be utilized to develop better solutions.

Most existing imbalance learning techniques are only designed for and tested in two-class scenarios. They have been shown to be less effective or even cause a negative effect in dealing with multi-class tasks [9]. Some methods are not applicable directly. Among limited solutions for multi-class imbalance problems, most attention in the literature was devoted to class decomposition – converting a multi-class problem into a set of two-class sub-problems [10]. Given a c-class problem ($c > 2$), a common decomposing scheme is to choose one class labeled as positive and merge the others labeled as negative for forming a sub-problem. Each class becomes the positive class once, and thus c binary classifiers are produced to give a final decision (known as one-against-all, one-vs-others) [11]. However, it aggravates imbalanced distributions [7], and combining results from classifiers learnt from different sub-problems can cause potential classification errors [12] [13]. It is desirable to develop a more effective and efficient method to handle multi-class imbalance problems.

This paper aims to provide a better understanding of why multi-class makes an imbalanced problem harder and new approaches to tackling the difficulties. We first study the impact of multi-class on the performance of random oversampling and undersampling techniques by discussing "multi-minority" and "multi-majority" cases in depth. We show that both "multi-minority" and "multi-majority" negatively affect the overall and minority-class performance. Particularly, the "multi-majority" case tends to be more harmful. Random oversampling does not help the classification and suffers from overfitting. The effect of random undersampling is weakened as there are more minority classes. When multiple majority classes exist, random undersampling can cause great performance reduction to those majority classes. Neither strategy is satisfactory. Based on the results, we propose to use our recently developed ensemble algorithm AdaBoost.NC [1] to handle multi-class imbalance problems. Our earlier work showed that it has good generalization ability under two-class imbalance scenarios by exploiting ensemble diversity [14]. As a new study of multi-class imbalance problems, the experiments in this paper reveal that AdaBoost.NC combined with oversampling can better recognize minority class examples and better balance the performance across multiple classes with high G-mean without using any class decomposition schemes.

The rest of this paper is organized as follows: section II briefly introduces the research progress in learning from multi-

The authors are with the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science, The University of Birmingham, Edgbaston, Birmingham B15 2TT, UK. E-mail: {S.Wang, X.Yao}@cs.bham.ac.uk

class imbalance problems in the literature and describes the AdaBoost.NC algorithm. Section III investigates the impact of class number in the presence of imbalanced data under some artificial settings. Section IV discusses the effectiveness of AdaBoost.NC in comparison with the state-of-the-art class imbalance learning methods on real-world multi-class imbalance tasks. Finally, section V concludes the paper.

II. Related Work

In this section, we first review the related studies concerning multi-class imbalance learning. There is a lack of systematic research on this topic and existing solutions are very limited. Then, we introduce the AdaBoost.NC algorithm and describe the main conclusions obtained in our earlier studies including its performance in two-class imbalance problems. This work will be extended to multi-class scenarios in this paper.

A. Multi-Class Imbalance Problems

Most existing solutions for multi-class imbalance problems use class decomposition schemes to handle multi-class and work with two-class imbalance techniques to handle each imbalanced binary sub-task. For example, protein fold classification is a typical multi-class imbalance problem. Tan et al. [7] used both one-against-all (OAA) [11] and one-against-one (OAO) [15] schemes to break down this problem, and then built rule-based learners to improve the coverage of minority class examples. OAA and OAO are two most popular schemes of class decomposition in the literature. Zhao et al. [5] used OAA to handle multi-class, and undersampling and SMOTE [16] techniques to overcome the imbalance issue. Liao [8] investigated a variety of oversampling and undersampling techniques used with OAA for a weld flaw classification problem. Chen et al. [6] proposed an algorithm using OAA to deal with multi-class and then applying some advanced sampling methods that decompose each binary problem further so as to rebalance the data. Fernandez [17] integrated OAO and SMOTE in their algorithm. Instead of applying data-level methods, Alejo et al.'s algorithm [18] made the error function of neural networks cost-sensitive by incorporating the imbalance rates between classes to emphasize minority classes, after decomposing the problem through OAA. Generally speaking, class decomposition simplifies the problem. However, each individual classifier is trained without full data knowledge. It can cause classification ambiguity or uncovered data regions with respect to each type of decomposition [7] [12] [13].

Different from the above, a cost-sensitive ensemble algorithm was proposed [19], which addresses multi-class imbalance directly without using class decomposition. Its key focuses are how to find an appropriate cost matrix with multiple classes and how to introduce the costs into the algorithm. A genetic algorithm (GA) was applied to searching for the optimum cost setup of each class. Two kinds of fitness were tested, G-mean [20] and F-measure [21], the most frequently used measures for performance evaluation in class imbalance learning. The choice depends on the training objective. The obtained cost vector was then integrated into a cost-sensitive version of AdaBoost.M1 [22], namely AdaC2 [23] [24], which is able to process multi-class data sets. However, searching the best cost vector is very time-consuming due to the nature of GA. No existing methods can deal with multi-class imbalance problems efficiently and effectively yet to our best knowledge.

We now turn our attention to the assessment metrics in class imbalance learning. Single-class performance measures evaluate how well a classifier performs in one class, particularly the minority class. Recall, precision and F-measure [21] are widely discussed single-class measures for two-class problems, which are still applicable to multi-class problems. Recall is a measure of completeness; precision is a measure of exactness. F-measure incorporates both to express their trade-off [2]. For the overall performance, G-mean [20] and AUC [25] are often used in the literature, but they are originally designed for two-class problems. Therefore, they have to be adapted to multi-class scenarios: an extended G-mean [19] is defined as the geometric mean of recall values of all classes; a commonly accepted extension of AUC is called M measure or MAUC [26], the average AUC of all pairs of classes.

B. AdaBoost.NC for Two-Class Imbalance Problems

We proposed an ensemble learning algorithm AdaBoost.NC [1] that combines the strength of negative correlation learning [27] [28] and Boosting [29]. It emphasizes ensemble diversity explicitly during training and shows very encouraging empirical results in both effectiveness and efficiency in comparison with the conventional AdaBoost and other NCL methods in general cases. We then exploited its good generalization performance to facilitate class imbalance learning [14], based on the finding that ensemble diversity has a positive role in solving this type of problem [30]. Comprehensive experiments were carried out on a set of two-class imbalance problems. The results suggest that AdaBoost.NC combined with random oversampling can improve the prediction accuracy on the minority class without losing the overall performance compared to other existing class imbalance learning methods. It is achieved by providing less overfitting and broader classification boundaries for the minority class. Applying oversampling is simply to maintain sufficient number of minority class examples and guarantee that two classes receive equal attention from the algorithm. The algorithm is described in Table I.

AdaBoost.NC penalizes classification errors and encourages ensemble diversity sequentially with the AdaBoost [29] training framework. In step 3 of the algorithm, a penalty term p_t is calculated for each training example, in which amb_t assesses the disagreement degree of the classification within the ensemble at current iteration t. It is defined as,

$$amb_t = \frac{1}{t}\sum_{i=1}^{t}\left(\|H_t = y\| - \|h_i = y\|\right),$$

where H_t is the class label given by the ensemble composed of the existing t classifiers. The magnitude of amb_t indicates a "pure" disagreement. p_t is introduced into the weight-updating step (step 5). By doing so, training examples with small $|amb_t|$ will gain more attention. The expression of α_t in step 4 is

TABLE I: AdaBoost.NC algorithm [1].

```
Given training data set ${(x_1, y_1), ..., (x_i, y_i), ..., (x_m, y_m)}$
with labels $y_i \in Y = \{1, \ldots, k\}$ and penalty strength $\lambda$,
initialize data weights $D_1(x_i) = 1/m$; penalty term $p_1(x_i) = 1$.

For training epoch $t = 1, 2, \ldots, T$:
Step 1. Train weak classifier $h_t$ using distribution $D_t$.
Step 2. Get weak classifier $h_t$: $X \rightarrow Y$.
Step 3. Calculate the penalty value for every example $x_i$:
        $p_t(x_i) = 1 - |amb_t(x_i)|$.
Step 4. Calculate $h_t$'s weight $\alpha_t$ by error and penalty using
        $\alpha_t = \frac{1}{2}\log\left(\frac{\sum_{i, y_i = h_t(x_i)} D_t(x_i)(p_t(x_i))^{\lambda}}{\sum_{i, y_i \neq h_t(x_i)} D_t(x_i)(p_t(x_i))^{\lambda}}\right)$
        for the discrete label outcome.
Step 5. Update data weights $D_t$ and obtain new weights $D_{t+1}$
        by error and penalty:
        $D_{t+1}(x_i) = \frac{(p_t(x_i))^{\lambda} D_t(x_i) exp(-\alpha_t \|h_t(x_i) = y_i\|)}{Z_t}$,
        where $Z_t$ is a normalization factor.

Output the final ensemble:
$H(x) = \arg\max_{y} \sum_{t=1}^{T} \alpha_t \|h_t(x) = y\|$.
(Define $\|\pi\|$ to be 1 if $\pi$ holds and 0 otherwise.)
```

derived by using the inferring method in [31] [24] to bound the overall training error. The pre-defined parameter λ controls the strength of applying p_t. The optimal λ depends on problem domains and base learners [32]. In general, $(0, 4]$ is deemed to be a conservative range of setting λ. As λ becomes larger, there could be either a further performance improvement or a performance degradation.

III. Challenges of Multi-Class Imbalance Learning

Two types of multi-class could occur to an imbalanced data set: one majority and multiple minority classes (multi-minority cases); one minority and multiple majority classes (multi-majority cases). A problem with multiple minority and multiple majority classes can be treated as the case when both types happen. Several interesting research questions are raised here: *Are there any differences between multiple minority and multiple majority classes? Would these two types of problem pose the same or different challenges to a learning algorithm? Which one would be more difficult to tackle? For such multi-class imbalance problems, which aspects of a problem would be affected the most by the multi-class? Would it be a minority class, a majority class or both?*

With these questions in mind, we will give separate discussions for each type under a set of artificial scenarios. For a clear understanding, two kinds of empirical analyses are conducted: 1) Spearman's rank correlation analysis, which shows the relationship between the number of classes and every evaluated performance measure, provides the evidence of the classification difficulty brought by "multi-minority" and "multi-majority". It will answer the question of *if* the difficulties exist. 2) Performance pattern analysis, which presents the performance changing tendencies of all existing classes as more classes are added into training, reveals the different performance behaviors of each class among different training strategies. It will tell us *what* difficulties are caused to the recognition of each class and *what* the differences between the two types of multi-class are.

A. Artificial Data Sets and Experimental Settings

To have sufficient number of classes for our study, we generate some artificial imbalanced data sets by using the method in [33]. In multi-minority cases, the number of minority classes is varied from 1 to 20, and only one majority class exists. Similarly, the number of majority classes is varied from 1 to 20 in multi-majority cases, and only one class is generated as the minority. Data points in each class are generated randomly from Gaussian distributions, where the mean and standard deviation of each attribute are random real values in [0,10].

We consider two different imbalanced contexts here. The first context includes a group of data sets having a relatively small size. Every generated example has 2 attributes. In training data, each minority class has 10 examples and each majority class has 100 examples. In the second context, we consider larger data sets. We enlarge the feature space to 20 attributes. In the training data of this group, each minority class contains 100 examples and each majority class contains 1000 examples. The two groups of data are denoted by "10-100" and "100-1000" respectively. Discussing both "10-100" and "100-1000" is to find out whether data size is a factor of affecting our results in the experiments.

We also generate a set of balanced multi-class data sets with the number of classes increasing from 2 to 21. They are used as the baseline to clearly show the "multi-class" difficulty in both balanced and imbalanced scenarios. The balanced training data have 2 attributes and 100 examples in each class, denoted by "100-100".

The data generation procedure is randomly repeated 20 times for each setting with the same numbers of minority and majority classes. Every training data set has a corresponding testing set, where each class contains 50 examples.

In the experiments, three ensemble training methods are compared: the conventional AdaBoost that is trained from the **or**iginal imbalanced data and used as the default baseline method (abbr. OrAda); random **ov**ersampling + AdaBoost (abbr. OvAda), where all the minority classes get their examples replicated randomly until each of them has the same size as the majority class before training starts; random **un**dersampling + AdaBoost (abbr. UnAda), where all the majority classes get rid of some examples randomly until each of them has the same size as the minority class before training starts. Every method is run 10 times independently on the current training data. Therefore, the result in the following comparisons is based on the average of 200 output values (20 training files*10 runs). As the most popular techniques in class imbalance learning, oversampling and undersampling are discussed to understand the impact of "multi-class" on the basic strategies of dealing with imbalanced data sets and examine their robustness to "multi-class".

C4.5 decision tree [34] is chosen as the base learner and implemented in Weka [35] – an open source data mining tool. The default Weka parameters are used, resulting in a pruned tree without the binary split. Each ensemble contains 51 such trees.

For the performance evaluation, single-class metrics recall, precision and F-measure are calculated for each class. To assess the overall performance across all classes, the generalized versions of AUC and G-mean, i.e. MAUC [26] and extended G-mean [19], are adopted in the following discussions.

B. Balanced Cases

In order to find out whether the "multi-class" issue is due to the type of multi-class imbalance or the multi-class itself, we first examine how it affects the classification performance on balanced data. We carry out Spearman's rank correlation analysis between performance measures and the number of classes by adding new classes of data with the equal size. Only the conventional AdaBoost (i.e. OrAda) is applied, since resampling is not necessary for balanced data. The three single-class measures are tracked for the first generated class that joins all the training sessions. Table II presents the correlation coefficients, where each entry ranges in $[-1, 1]$ (in %). A positive (negative) value indicates a monotone increasing (decreasing) relationship and a coefficient of zero indicates no tendency between them.

TABLE II: Rank correlation coefficients (in %) between the number of classes and 5 performance measures for OrAda on balanced data "100-100". Recall, precision and F-measure are calculated for the first generated class.

100-100	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-100	-100	-100	-99	-100

Almost all the measures have coefficients of -1 in Table II, which means that they present very strong negative correlations with the number of classes. As more classes are added into the training data, the performance of AdaBoost in predicting any single class and its overall ability to discriminate between classes become worse. Multi-class itself increases the data complexity and negatively affects the classification performance regardless of whether data is imbalanced. It may imply that multi-class imbalance problems cannot be simply solved by rebalancing the number of examples among classes. Next, let us examine the imbalanced cases.

C. Multi-Minority Cases

The correlation analysis and performance pattern analysis are conducted on the multi-minority cases in this subsection. The number of minority classes is varied from 1 to 20. The impact of multi-minority on the performance of oversampling and undersampling techniques is illustrated and analyzed in depth.

1) Correlation Analysis: Five performance measures and three ensemble training methods (i.e. OrAda, OvAda, UnAda) permit 15 pairwise correlations with respect to the number of minority classes. They show if multi-minority degrades the classification performance of the three ensemble training methods and which performance aspects are affected. The three single-class measures are recorded for the minority class that joins all the training sessions from 1 to 20. Table III summarizes the correlation coefficients for "10-100" and "100-1000" data groups.

TABLE III: Rank correlation coefficients (in %) between the number of minority classes and 5 performance measures for 3 ensemble methods on "10-100" and "100-1000" data. Recall, precision and F-measure are calculated for the minority class.

10-100	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-89	-94	-91	-92	-97
OvAda	-88	-93	-91	-94	-98
UnAda	-93	-93	-93	-91	-99
100-1000	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-99	-99	-100	-98	-100
OvAda	-99	-99	-99	-94	-100
UnAda	-99	-99	-100	-93	-100

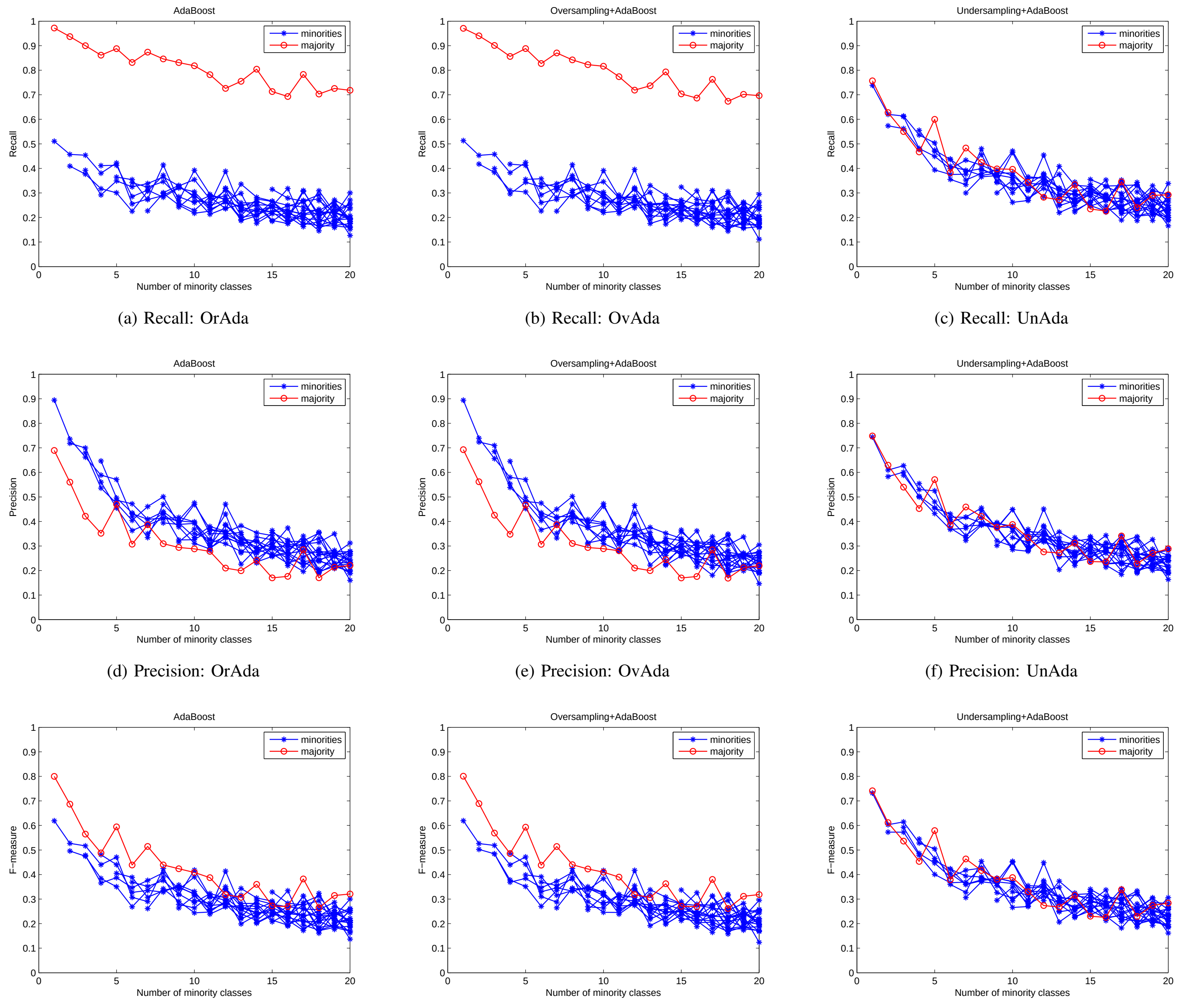
All pairs present very strong negative correlations on both groups of small and large data sets. It implies a strong monotone decreasing relationship between the measures and the number of minority classes. All of them are decreasing as more minority classes are added into the training data, regardless of the size of the training data and whether resampling is applied. In other words, multi-minority reduces the performance of these ensembles consistently and data resampling seems not to be helpful. Next, we will give a further investigation into the performance degradation caused by multi-minority classes from the level of every single class.

2) Performance Pattern Analysis: We now focus on the "10-100" group of data sets and illustrate the changing tendencies of single-class measures for all classes as the class number increases. In Fig. 1, the presented pattern reveals detailed information about how the classification performance of each class is affected and the differences among ensemble methods and evaluated measures. All the following pattern plots are scaled in the same range.

According to Fig. 1, every class's performance is decreasing. No evidence shows which class suffers from more performance degradation than other classes. The classification gets equally difficult on all classes. For each class, corresponding to one curve in the plot, the measure value drops faster at the first few steps, when the number of minority classes is approximately smaller than 10. As it gets larger, the reduction slows down.

Among the three performance measures, the drop of precision (Fig. 1(d)(e)) is more severe than that of recall (Fig. 1(a)(b)) in OrAda and OvAda. Precision is the main cause of the decrease in F-measure. The reason is that multi-minority increases the risk of predicting an example into a wrong class. As to recall, it seems that the difficulty of recognizing examples within each class is less affected by multi-minority as compared to precision, because the proportion of each class of data in the whole data set is hardly changed by adding a small class. In UnAda, each class is reduced to have a small size. Adding minority classes changes the proportion of each class significantly. It explains why UnAda's recall (Fig. 1(c)) presents higher sensitivity to multi-minority than the recall produced by OrAda and OvAda (Fig. 1(a)(b)).

Among the three ensemble methods, OrAda and OvAda



(a) Recall: OrAda (b) Recall: OvAda (c) Recall: UnAda

(d) Precision: OrAda (e) Precision: OvAda (f) Precision: UnAda

(g) F-measure: OrAda (h) F-measure: OvAda (i) F-measure: UnAda

Fig. 1: Single-class performance patterns among classes in multi-minority cases of “10-100” (x-axis: number of minority classes, y-axis: performance output).

have similar performance patterns, where the majority class obtains higher recall and F-measure than the minority classes, but lower precision values. Oversampling does not alleviate the multi-class problem. Although oversampling increases the quantity of minority class examples to make every class have the same size, the class distribution in data space is still imbalanced, which is dominated by the majority class. In UnAda, undersampling counteracts the performance differences among classes. During the first few steps, UnAda presents better recall and F-measure on minority classes (Fig. 1(c)(i)) than OrAda and OvAda (Fig. 1(a)(b)(g)(h)). From this point of view, it seems that using undersampling might be a better choice. However, its advantage is weakened as more minority classes join the training. When the class number reaches 20, three ensemble algorithms have very similar minority-class performance. The reason could be that undersampling explicitly empties some space for recognizing minority classes by removing examples from the majority class region. When there is only one minority class, a classifier is very likely to assign the space to this class. When there are many minority classes, they have to share the same space. Hence, the effect of undersampling is reduced. Undersampling seems to be more sensitive to multi-minority. For this consideration, it would be better to expand the classification area for each minority class, instead of shrinking the majority class. To achieve this goal, advanced techniques are necessary to improve the classification generalization over minority classes.

D. Multi-Majority Cases

We proceed with the same analyses for the multi-majority data "10-100" and "100-1000". The number of majority classes is varied from 1 to 20. The impact of multi-majority is studied here.

1) Correlation Analysis: Table IV summarizes the correlation coefficients. Single-class performance measures are recorded for the only minority class of each data set. Similar to the multi-minority cases, strong negative correlations between five performance measures and the number of majority classes are observed in both groups of small and large data sets, which indicate a monotone decreasing relationship. All three ensemble training methods suffer from performance reduction caused by "multi-majority".

TABLE IV: Rank correlation coefficients (in %) between the number of majority classes and 5 performance measures in 3 ensemble methods on "10-100" and "100-1000" data. Recall, precision and F-measure are calculated for the minority class.

10-100	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-79	-89	-85	-92	-97
OvAda	-84	-93	-86	-92	-97
UnAda	-92	-94	-94	-95	-99
100-1000	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-100	-96	-100	-90	-100
OvAda	-100	-93	-100	-86	-100
UnAda	-99	-99	-100	-83	-100

2) Performance Pattern Analysis: To gain more insight, we focus on the "10-100" group of data sets and present the changing tendencies of single-class measures for each class along with the increase of the number of majority classes in Fig. 2. All plots are in the same axis scale.

Among the classes in each plot, adding majority classes makes the recognition of examples of each class (i.e. recall presented in Fig. 2(a)-(c)) equally difficult. In OrAda and OvAda, minority-class precision drops faster than that of the majority classes (Fig. 2(d)(e)), because the large quantity of new majority class examples overwhelm the minority class even more. Minority class examples are more likely to be misclassified than before compared to majority class examples.

All performance measures present a drastic decrease. Especially in recall plots of OrAda and OvAda (Fig. 2(a)(b)), more and more majority class examples take the recognition rate of the minority class down to nearly 0. For every existing majority class, adding more majority classes can make it appear to be in minority. Therefore, the recall of majority classes also shows a fast drop.

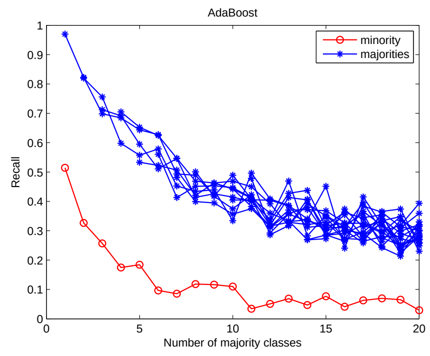
Among the three ensemble methods, UnAda produces better minority-class F-measure than OrAda and OvAda, but the recall of majority classes is sacrificed greatly. It causes the concern that using undersampling will lose too much data information when multiple majority classes exist, and can lead to severe performance reduction in majority classes.

Based on all the observations in this section, we conclude that: 1) As no new information is introduced into the minority class to facilitate the classification in OrAda and OvAda, overfitting minority-class regions happens with low recall and high precision values when compared with those measures obtained from the majority classes. Oversampling does not help for both multi-minority and multi-majority cases. 2) UnAda performs the same under multi-minority and multi-majority cases due to undersampling. In the multi-minority case, UnAda can be sensitive to the class number; in the multi-majority case, there is a high risk of sacrificing too much majority-class performance. 3) Between multi-minority and multi-majority, the multi-majority case seems to be more difficult than the multi-minority case. OrAda and OvAda present much worse minority-class performance in Fig. 2(g)(h) compared to Fig. 1(g)(h). This is because adding majority class examples aggravates the imbalanced situation. 4) Between balanced and imbalanced data, multi-class leads to performance degradation in both scenarios. We believe that learning imbalanced data is much harder than learning balanced one, for the performance difference between the types of classes shown in the performance pattern analysis and the particular performance requirement for minority classes, which would not happen in the balanced case. Because of different learning objectives, different treatments should be considered.

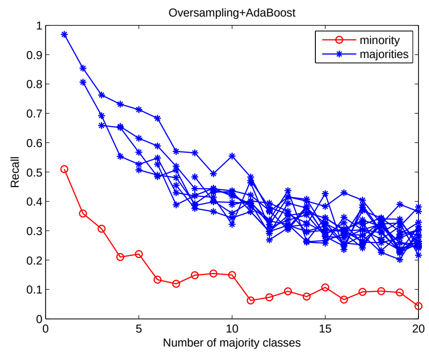
IV. A Comparative Study of Ensemble Algorithms on Multi-Class Imbalance Problems

Armed with a better understanding of multi-class imbalance problems, this section aims to find a simple and effective ensemble learning method without using class decomposition. The presence of multiple minority classes increases data complexity. In addition to more complex data distributions, the presence of multiple majority classes makes a data set even more imbalanced. Balancing the performance among classes appropriately is important. Suggested by the analysis in the previous section, we attempt to improve the generalization of the learning method by focusing on minority classes, instead of shrinking majority classes through undersampling, in order to avoid losing useful data information and keep the learning method less sensitive to the number of minority classes.

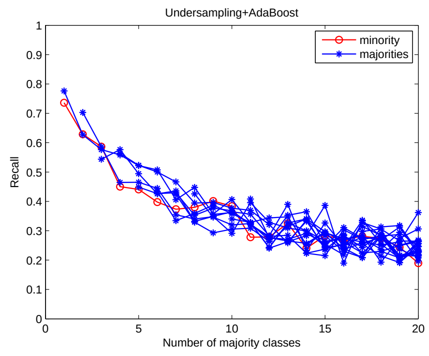
In our previous study [14], we found that the "random oversampling + AdaBoost.NC" tree ensemble is effective in handling two-class imbalance problems. It shows a good recognition rate of the minority class and balances the performance between minority and majority classes well by making use of ensemble diversity. Besides, its training strategy is flexible and simple without removing any training data. For the above reasons, we look into this algorithm and extend our study to multi-class cases in this section. The main research question here is *whether AdaBoost.NC is still effective in solving multi-class imbalance problems*. In order to answer the question and find out if class decomposition is necessary, AdaBoost.NC is compared with other state-of-the-art methods in cases of using and not using class decomposition, including the conventional AdaBoost, resampling-based AdaBoost and SMOTEBoost [4]. AdaBoost is discussed as the baseline method, because the AdaBoost.NC algorithm is in the Boosting training framework. Resampling techniques and the SMOTEBoost algorithm are chosen for their widely use in the multi-class imbalance learning literature [5] [17]. More ensemble solutions exist for



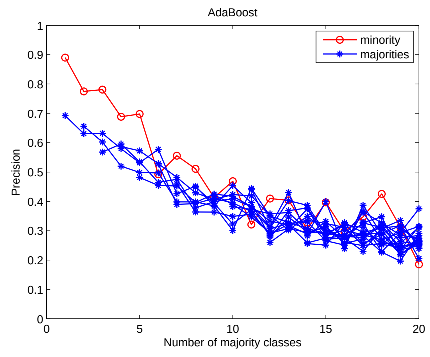
(a) Recall: OrAda



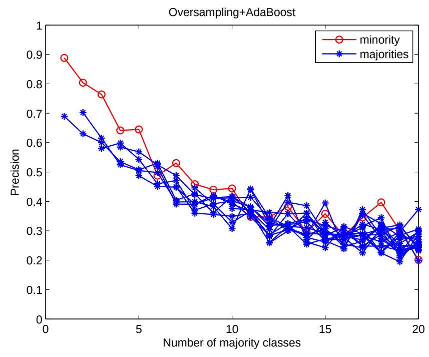
(b) Recall: OvAda



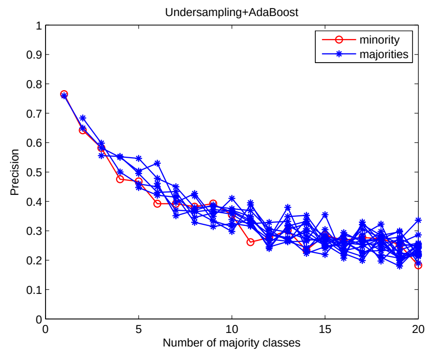
(c) Recall: UnAda



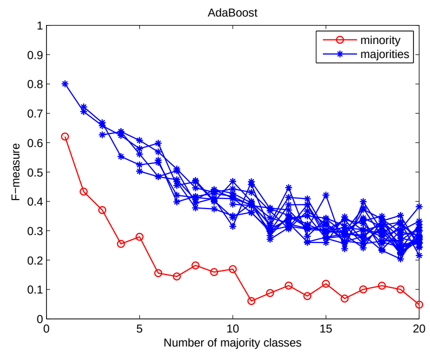
(d) Precision: OrAda



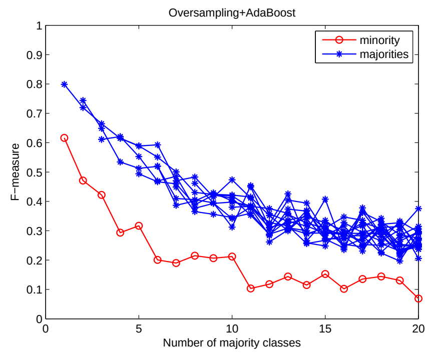
(e) Precision: OvAda



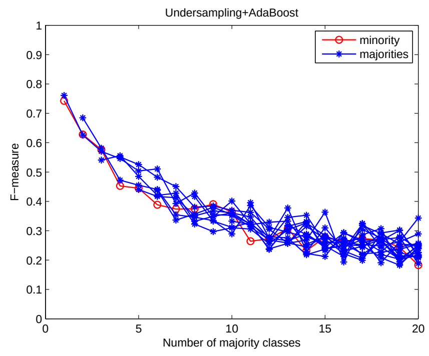
(f) Precision: UnAda



(g) F-measure: OrAda



(h) F-measure: OvAda



(i) F-measure: UnAda

Fig. 2: Single-class performance patterns among classes in multi-majority cases of “10-100” (x-axis: number of majority classes, y-axis: performance output).

two-class imbalance problems, such as RAMOBoost [36] and JOUS-Boost [37], which will be studied as our next step on how they perform when dealing with multi-class cases and what advantages they might have over the methods we discuss in this paper. Cost-sensitive algorithms are another class of solutions. They are not considered in our experiments, since we do not assume the availability of explicit cost information in our algorithm.

A. Data Sets and Experimental Settings

In the experiments, we evaluate our candidate methods on 12 classification benchmark problems from the UCI repository [38]. Each data set has more than two classes. At least one of them is significantly smaller than one of the others. The data information with class distributions is summarized in Table V.

Six ensemble models are constructed and compared, including AdaBoost without resampling applied (OrAda, the baseline model), “random oversampling + AdaBoost” (OvAda), “random undersampling + AdaBoost” (UnAda), “random oversampling + AdaBoost.NC” with $\lambda = 2$ (OvNC2), “random oversampling + AdaBoost.NC” with $\lambda = 9$ (OvNC9) and SMOTEBoost [4] (SMB).

With respect to parameter settings, the penalty strength λ in AdaBoost.NC is set to 2 and 9 based on our previous findings [32]. $\lambda = 2$ is a relatively conservative setting to show if AdaBoost.NC can make a performance improvement

TABLE V: Summary of benchmark data sets.

Data	Class	Size	Distribution
New-thyroid	3	215	150/35/30
Balance	3	625	49/288/288
Car	4	1728	1210/384/69/65
Nursery	4	12958	4320/328/4266/4044
Glass	6	214	70/76/17/13/9/29
Annealing	5	898	8/99/684/67/40
Solarflare2	6	1066	147/211/239/95/43/331
Page	5	5473	4913/329/28/88/115
Ecoli	5	327	143/77/52/35/20
Cleveland	5	303	164/55/36/35/13
Yeast	10	1484	463/429/244/163/51/44/35/30/20/5
Satimage	6	6435	1533/703/1358/626/707/1508

and $\lambda = 9$ encourages ensemble diversity aggressively, as we have explained in section II. For a better understanding of how AdaBoost.NC can facilitate multi-class imbalance learning, both values (representing two extremes) are discussed here. Applying random oversampling is necessary for AdaBoost.NC not to ignore the minority class [14]. For SMOTEBoost, the nearest neighbor parameter k is set to 5, the most accepted value in the literature. The amount of new data for a class c is roughly the size difference between the largest class and class c, considering that the other models also adjust the between-class ratio to one. For now, we use fixed parameters for the algorithms we compare with, in order to single out the impact of different algorithmic features on the performance. Once we have a better understanding of the algorithms, we can then tune and optimize parameters by using some existing parameter optimizing methods in our future studies [39].

We still employ C4.5 decision tree as the base learner, following the same settings as in the previous section. Each ensemble consists of 51 trees. As some data sets have very small classes, we perform a 5-fold cross-validation (CV) with 10 runs instead of the traditional 10-fold CV, to guarantee that each fold of data contains at least one example from every minority class.

MAUC [26] and extended G-mean [19] are used to evaluate the overall performance as before. Recall and precision are recorded as the single-class performance measures to explain how an overall performance improvement or degradation happens.

B. Ensemble Algorithms for Multi-Class Imbalance Problems

In this section, we first give respective discussions on the performance of ensemble algorithms without using any class decomposition and the ones using the "one-against-all" (OAA) scheme of class decomposition – the most frequently used scheme in the literature. Based on the observations and analysis, an improved combination strategy for OAA-based ensembles is then proposed. In order to evaluate the significance of the results from the comparative methods, we carry out the Friedman test [40] with the corresponding post-hoc test recommended by Demsar [41]. The Friedman test is a non-parametric statistical method for testing whether all the algorithms are equivalent over multiple data sets. If the test result rejects the null hypothesis, i.e., there are some differences between the algorithms, we then proceed with a post-hoc test to find out which algorithms actually differ. This paper uses an improved Friedman test proposed by Iman and Davenport [42]. The Bonferroni-Dunn test [43] is used as the post-hoc test method. Finally, we show whether class decomposition is necessary for multi-class imbalance learning based on the student T-test with 95% confidence level.

1) Ensemble Models without Using OAA: Tables VI - VII present the Friedman and post-hoc test results on MAUC, extended G-mean, recall and precision for the six ensemble models without using class decomposition. Considering the existence of multiple minority and majority classes, we only discuss recall and precision for the smallest class and the largest class, which should be the most typical ones in the data set. The minority-class recall and precision are denoted by R_{min} and P_{min}, and the majority-class recall and precision are denoted by R_{maj} and P_{maj}. The Friedman test compares the mean ranks of algorithms, which are shown in Table VI. A smaller value indicates a higher rank, i.e. better performance. Table VII gives the Friedman and post-hoc test results by choosing OvNC9 as the "control" method, in which each number is the difference of mean ranks between the "control" method and one of the other methods in the corresponding column. The CD value shown at the bottom is the critical difference value [41]. It is determined by the number of algorithms and data sets, and a critical value q_α that is equal to 2.326 in our case. The performance of any two methods is significantly different if their difference of mean ranks is larger than CD.

TABLE VI: Mean ranks of the 6 ensemble models without using OAA over 12 data sets, including OrAda, OvAda, UnAda, OvNC2, OvNC9 and SMB.

Rank	OrAda	OvAda	UnAda	OvNC2	OvNC9	SMB
MAUC	1.833	3.500	5.167	3.417	4.500	2.583
G-mean	5.000	4.333	3.917	3.667	1.833	2.250
R_{min}	5.083	4.625	1.375	3.917	2.000	3.833
P_{min}	2.750	2.917	4.833	3.250	4.250	3.000
R_{maj}	2.500	2.250	5.500	2.750	4.500	3.500
P_{maj}	3.500	4.250	3.583	3.667	3.333	2.667

TABLE VII: Friedman test with the corresponding post-hoc test (Bonferroni-Dunn) for the 6 ensemble models without using OAA. The difference of mean ranks between OvNC9 and every other method is calculated. CD is the critical difference [41]. A value larger than CD indicates a significant difference between the methods, highlighted in boldface.

	Friedman	OrAda	OvAda	UnAda	OvNC2	SMB
MAUC	Reject	**2.667**	1.000	0.667	1.083	**1.917**
G-mean	Reject	**3.167**	**2.500**	**2.084**	**1.834**	0.417
R_{min}	Reject	**3.083**	**2.625**	0.625	**1.917**	**1.833**
P_{min}	Reject	1.500	1.333	0.583	1.000	1.250
R_{maj}	Reject	**2.000**	**2.250**	1.000	1.750	1.000
P_{maj}	Not reject					

* CD = 1.776

For the overall performance measures MAUC and G-mean, we make the following observations. AdaBoost.NC does not show much advantage over other methods in terms of MAUC, where the mean ranks of OvNC2 and OvNC9 are 3.417 and 4.5 respectively. OrAda has the highest rank. The Friedman test rejects the null hypothesis of MAUC, which means the existence of significant differences among the algorithms. Concretely, OrAda and SMB produce significantly better MAUC than OvNC9, and OvNC9 produces comparable MAUC to the others. However, different observations happen to G-mean. OvNC9 achieves the best G-mean and OrAda gives the worst. OvNC9 is significantly better than resampling-based AdaBoost and comparable to SMB.

It is interesting to observe that the conventional AdaBoost without using any specific rebalancing technique is good at MAUC and bad at G-mean. It is known that AdaBoost itself cannot handle class imbalance problems very well. It is sensitive to imbalanced distributions [44] [24]. Meanwhile, our experiments in the previous section show that it suffers from multi-class difficulties significantly. It means that the low G-mean of AdaBoost results from its low recognition rates on the minority classes and the high MAUC is probably attributed to the relatively good discriminability between the majority classes. The other ensemble methods, AdaBoost.NC, SMOTEBoost and resampling-based AdaBoost, seem to be more effective in improving G-mean than MAUC.

To explain this observation, let's recall the definitions of MAUC and G-mean. G-mean is the geometric mean of recall over all classes. If any single class receives very low recall, it will take the G-mean value down. It can tell us how well a classifier can balance the recognition among different classes. A high G-mean guarantees that no class is ignored. MAUC assesses the average ability of separating any pair of classes. A high MAUC implies that a classifier is good at separating most pairs, but it is still possible that some classes are hard to be distinguished from the others. G-mean is more sensitive to single-class performance than MAUC. From this point of view, it may suggest that those ensemble solutions for class imbalance learning, especially OvNC9, can better recognize examples from the minority classes, but are not good at discriminating between some majority classes. To confirm our explanations, we look into single-class performance next.

Not surprisingly, UnAda performs the best in recall but produces the worst precision for the minority class because of the loss of a large amount of majority class data. OvNC9 ranks the second in minority-class recall, which is competitive with UnAda and significantly better than the others. OvNC9 produces higher minority-class recall than OvNC2, which implies that a large λ can further generalize the performance of AdaBoost.NC on the minority class. More minority class examples can be identified by setting a large λ. Meanwhile, it is encouraging to see that OvNC9 does not lose too much performance on minority-class precision, where no significant differences are observed. However, it sacrifices some majority-class recall when compared with OrAda and OvAda, because of the performance trade-off between minority and majority classes.

The observations on the smallest and largest classes explains that the good G-mean of OvNC9 results from the greater improvement in recall of the minority classes than the recall reduction of the majority classes. Its ineffectiveness in MAUC should be caused by the relatively poor performance in the majority classes. Based on the above results, we conclude that AdaBoost.NC with a large λ is helpful for recognizing minority class examples with high recall and capable of balancing the performance across different classes with high G-mean. From the view of MAUC and majority-class performance, it could lose some learning ability to separate majority classes. In addition, SMOTEBoost presents quite stable overall performance in terms of both MAUC and G-mean.

2) Ensemble Models Using OAA: We use exactly the same ensemble methods here, but let them work with the OAA class decomposition scheme. In this group of models, one builds a set of binary classifiers, which will then be combined for a final decision. We adopt the combining strategy used in [8], which outputs the class whose corresponding binary classifier produces the highest value of belongingness among all. These models are denoted by "-d", to indicate that OAA is used.

The comparison results for the ensembles using OAA are summarized in Tables VIII - IX. Table VIII presents the mean ranks of constructed models on all the performance measures. Based on the ranks, the Friedman and post-hoc test results are given in Table IX by choosing OvNC9-d as the "control" method. We observe that no single class imbalance learning method actually outperforms the conventional AdaBoost (i.e. OrAda-d) significantly in terms of either MAUC or G-mean. SMB-d appears to be relatively stable with slightly better MAUC and G-mean than the others. UnAda-d has the lowest rank of MAUC and OvNC9-d has the lowest rank of G-mean. Generally speaking, the overall performance of all the methods is quite close between each other. These results are different from what we have observed in the cases without using OAA, where OvNC9 yields the best G-mean. It seems that class imbalance techniques are not very effective when working with the OAA scheme.

TABLE VIII: Mean ranks of the 6 ensemble models using OAA over 12 data sets, including OrAda-d, OvAda-d, UnAda-d, OvNC2-d, OvNC9-d and SMB-d.

Rank	OrAda -d	OvAda -d	UnAda -d	OvNC2 -d	OvNC9 -d	SMB -d
MAUC	3.167	3.500	5.167	3.083	3.667	2.417
G-mean	3.333	3.167	4.000	3.417	4.333	2.750
R_{min}	3.750	3.917	2.083	3.417	4.000	3.500
P_{min}	2.083	2.583	5.000	3.167	4.583	3.500
R_{maj}	2.542	2.542	4.208	2.958	4.625	4.125
P_{maj}	3.917	3.917	3.250	3.333	3.583	3.000

As to the single-class performance in the smallest class, there is no significant difference among the methods in terms of recall, because the Friedman test does not reject the null hypothesis. Resampling techniques, SMOTEBoost and AdaBoost.NC do not show much advantage in identifying minority class examples. In terms of the minority-class precision, OvNC9-d is significantly worse than OrAda-d and OvAda-d according to Table IX. Similar happens to the largest class. We conclude that class imbalance learning techniques exhibit

TABLE IX: Friedman test with the corresponding post-hoc test (Bonferroni-Dunn) for the 6 ensemble models using OAA. The difference of mean ranks between OvNC9-d and every other method is calculated. CD is the critical difference [41]. A value larger than CD indicates a significant difference between the methods, highlighted in boldface.

	Friedman	OrAda -d	OvAda -d	UnAda -d	OvNC2 -d	SMB -d
MAUC	Reject	0.500	0.167	1.500	0.584	1.250
G-mean	Not reject					
R_{min}	Not reject					
P_{min}	Reject	**2.500**	**2.000**	0.417	1.416	1.083
R_{maj}	Reject	**2.083**	**2.083**	0.417	1.667	0.500
P_{maj}	Not reject					

* CD = 1.776

ineffectiveness in both minority and majority classes when compared with the conventional AdaBoost in this group of comparisons. Neither type of classes is better recognized. When the OAA scheme is applied to handling multi-class, they do not bring any performance improvement.

According to our results here, AdaBoost.NC does not show any significant improvement in minority-class and overall performance when working with the class decomposition scheme in multi-class imbalance scenarios, although it showed good classification ability to deal with two-class imbalance problems [14]. A possible reason for its poor performance could be that the combining step of OAA messes up the individual results. Without using OAA, AdaBoost.NC receives and learns from complete data information of all classes, which allows the algorithm to consider the difference among classes during learning with full knowledge. The OAA scheme, however, makes AdaBoost.NC learn from several decomposed sub-problems with partial data knowledge. The relative importance between classes is lost. Even if AdaBoost.NC can be good at handling each sub-problem, their combination does not guarantee good performance for the whole problem. Therefore, it may not be a wise idea to integrate class decomposition with class imbalance techniques without considering the class distribution globally. A better combining method for class decomposition schemes is needed.

3) Ensemble Models Using OAA with an Improved Combination Method: To take into account the class information of the whole data set, we improve the combination method of OAA in this section by using a weighted combining rule. Instead of the traditional way of treating the outputs of binary classifiers equally [8], we assign them different weights, determined by the size of each class. For any input example x, its belongingness value of class i from the i-th binary classifier is multiplied by the inverse of its imbalance rate. The imbalance rate is defined as the proportion of this class of data within the whole data set. The final decision of OAA will be the class receiving the highest belongingness value among all after adjusted by the weights.

We apply the same ensemble methods in the previous sections on the 12 UCI data sets. Six ensemble models are constructed. They are denoted by "-dw", indicating that class decomposition with a weighted combination is used. The Friedman and post-hoc test results are summarized in Tables X - XI.

TABLE X: Mean ranks of the 6 ensemble models using OAA with the weighted combination over 12 data sets, including OrAda-dw, OvAda-dw, UnAda-dw, OvNC2-dw, OvNC9-dw and SMB-dw.

Rank	OrAda -dw	OvAda -dw	UnAda -dw	OvNC2 -dw	OvNC9 -dw	SMB -dw
MAUC	2.750	3.167	5.750	3.167	3.583	2.583
G-mean	4.583	3.750	4.583	3.750	1.667	2.667
R_{min}	4.583	4.625	1.500	4.042	2.167	4.000
P_{min}	2.333	2.167	5.250	3.500	3.917	3.750
R_{maj}	1.958	2.125	5.708	2.792	4.542	3.875
P_{maj}	4.417	3.750	2.833	3.750	2.833	3.417

TABLE XI: Friedman test with the corresponding post-hoc test (Bonferroni-Dunn) for the 6 ensemble models using OAA with the weighted combination. The difference of mean ranks between OvNC9-dw and every other method is calculated. CD is the critical difference [41]. A value larger than CD indicates a significant difference between the methods, highlighted in boldface.

	Friedman	OrAda -dw	OvAda -dw	UnAda -dw	OvNC2 -dw	SMB -dw
MAUC	Reject	0.833	0.416	**2.167**	0.416	1.000
G-mean	Reject	**2.916**	**2.083**	**2.916**	**2.083**	1.000
R_{min}	Reject	**2.416**	**2.458**	0.667	**1.875**	**1.833**
P_{min}	Reject	1.584	1.750	1.333	0.417	0.167
R_{maj}	Reject	**2.584**	**2.417**	1.166	1.750	0.667
P_{maj}	Not reject					

* CD = 1.776

It is encouraging to observe that the ineffectiveness of AdaBoost.NC used with OAA is rectified by the weighted combination method in terms of G-mean and minority-class recall. The following observations are obtained: 1) OvNC9-dw presents significantly better MAUC than UnAda-dw and are competitive with the others. 2) OvNC9-dw produces the best G-mean with the highest mean rank, which is significantly better than the others expect for SMB-dw. UnAda-dw gives the worst G-mean. 3) Except for UnAda-dw, OvNC9-dw produces significantly better minority-class recall than the other models without sacrificing minority-class precision. OvNC9-dw shows competitive ability to recognize minority class examples with UnAda-dw. 4) OvNC9-dw loses some accuracy in finding majority class examples compared to OrAda-dw and OvAda-dw.

In summary, by applying the improved combination method to OAA, AdaBoost.NC with a large λ can find more minority class examples with a higher recall and better balance the performance across different classes with a higher G-mean than other methods. SMOTEBoost is a relatively stable algorithm in terms of overall performance that presents competitive MAUC and G-mean with AdaBoost.NC.

4) Is Class Decomposition Necessary?: The discussions in this subsection aim to answer the question of whether it is

TABLE XII: Means and standard deviations of MAUC, G-mean and minority-class recall by AdaBoost.NC with $\lambda = 9$ and SMOTEBoost methods without using OAA (i.e. OvNC9 and SMB) and using OAA with the weighted combination (i.e. OvNC9-dw and SMB-dw). Recall is computed for the smallest class of each data set. Values in boldface indicate "significant better" between OvNC9/SMB and OvNC9-dw/SMB-dw.

MAUC				
	OvNC9	OvNC9-dw	SMB	SMB-dw
New-thyroid	**0.983±0.013**	0.973±0.003	0.988±0.014	0.988±0.003
Balance	0.704±0.037	0.703±0.003	**0.703±0.027**	0.633±0.004
Car	**0.982±0.005**	0.980±0.001	0.994±0.003	**0.997±0.000**
Nursery	0.995±0.001	**0.998±0.000**	0.999±0.000	0.999±0.000
Glass	0.876±0.037	0.881±0.009	0.925±0.030	0.924±0.009
Annealing	**0.986±0.009**	0.975±0.003	0.984±0.018	0.981±0.004
Solarflare2	0.866±0.020	**0.901±0.003**	0.890±0.015	0.891±0.002
Page	**0.989±0.004**	0.984±0.001	**0.989±0.005**	0.973±0.002
Ecoli	0.952±0.020	0.957±0.002	0.954±0.019	**0.963±0.004**
Cleveland	0.727±0.046	**0.766±0.004**	0.764±0.040	0.767±0.007
Yeast	0.810±0.020	**0.857±0.004**	0.831±0.021	**0.847±0.003**
Satimage	0.984±0.002	**0.990±0.000**	0.991±0.001	0.992±0.000
G-mean				
	OvNC9	OvNC9-dw	SMB	SMB-dw
New-thyroid	0.927±0.052	0.915±0.056	0.934±0.060	0.940±0.057
Balance	0.321±0.173	0.319±0.180	0.000±0.000	0.000±0.000
Car	**0.924±0.024**	0.897±0.038	0.928±0.033	**0.944±0.031**
Nursery	0.954±0.006	**0.967±0.006**	0.992±0.004	**0.996±0.003**
Glass	0.571±0.278	0.578±0.249	0.561±0.343	0.508±0.344
Annealing	0.823±0.310	0.895±0.191	0.764±0.341	0.854±0.258
Solarflare2	0.486±0.112	**0.540±0.096**	0.520±0.120	0.514±0.094
Page	0.912±0.031	0.920±0.026	0.860±0.048	0.871±0.052
Ecoli	0.776±0.069	0.790±0.062	0.798±0.052	0.803±0.059
Cleveland	0.117±0.160	0.075±0.144	0.009±0.066	0.000±0.000
Yeast	0.237±0.270	0.190±0.257	0.140±0.240	0.060±0.164
Satimage	0.872±0.011	**0.881±0.009**	0.895±0.011	0.898±0.010
Minority-Class Recall				
	OvNC9	OvNC9-dw	SMB	SMB-dw
New-thyroid	0.897±0.134	0.910±0.117	0.900±0.121	0.906±0.135
Balance	0.144±0.112	0.150±0.112	0.000±0.000	0.000±0.000
Car	0.980±0.043	**0.997±0.021**	0.967±0.058	0.973±0.057
Nursery	0.985±0.020	0.992±0.018	0.983±0.017	**0.993±0.012**
Glass	**0.990±0.070**	0.930±0.202	0.910±0.218	0.860±0.248
Annealing	0.790±0.351	0.870±0.263	0.730±0.380	0.850±0.307
Solarflare2	0.400±0.193	0.456±0.190	0.275±0.142	**0.353±0.169**
Page	0.974±0.076	0.985±0.052	0.881±0.161	0.822±0.187
Ecoli	0.820±0.182	0.860±0.176	0.870±0.169	0.880±0.153
Cleveland	0.167±0.212	0.197±0.246	0.030±0.119	0.030±0.104
Yeast	0.097±0.117	0.070±0.101	**0.056±0.092**	0.023±0.058
Satimage	0.721±0.042	**0.800±0.037**	0.692±0.050	0.709±0.040

necessary to use class decomposition for handling multi-class imbalance problems. We compare the overall and minority-class performance of OvNC9 and SMB with the performance of OvNC9-dw and SMB-dw. AdaBoost.NC with $\lambda = 9$ and SMOTEBoost are chosen, because AdaBoost.NC performs better at G-mean and minority-class recall and SMOTEBoost presents good and stable MAUC and G-mean. Raw performance outputs from the 12 data sets are shown in Table XII. Values in boldface indicate "significantly better" between OvNC9 (SMB) and OvNC9-dw (SMB-dw) based on the student T-test with 95% confidence level.

According to the table, no consistent difference is observed between OvNC9 and OvNC9-dw in the three performance measures. In most cases, they present competitive measure values with each other. OvNC9-dw shows slightly better G-mean with more wins. The same happens between SMB and SMB-dw. It suggests that whether to apply OAA does not affect class imbalance learning methods much. Learning from the whole data set directly is sufficient for them to achieve good MAUC and G-mean and find minority-class examples effectively. Therefore, we conclude that using class decomposition is not necessary to tackle multi-class imbalance problems. Besides, Table XII further confirms our previous conclusion from the Friedman and post-hoc tests that AdaBoost.NC has better generalization especially for the minority class. For example, SMOTEBoost produces zero G-mean and zero minority-class recall on data set "Balance", which means that no examples from the minority class are found and the obtained classifier is barely useful, while AdaBoost.NC changes this situation with much better G-mean and minority-class recall.

V. CONCLUSIONS

This paper studies the challenges of multi-class imbalance problems and investigates the generalization ability of ensemble algorithms including AdaBoost.NC [1] to deal with multi-class imbalance data. Two types of multi-class imbalance problems, i.e. the multi-minority and multi-majority cases, are studied in depth. For each type, we examine overall and minority-class performance of three ensemble methods based on the correlation analysis and performance pattern analysis. Both types show strong negative correlations with the five performance measures, which are MAUC, G-mean, minority-class recall, minority-class precision and minority-class F-measure. It implies that the performance decreases as the number of imbalanced classes increases. The results from the performance pattern analysis show that the multi-majority case tends to cause more performance degradation than the multi-minority case, because the imbalance rate gets more severe. Oversampling does not help the classification, and causes overfitting to the minority classes with low recall and high precision values. Undersampling is sensitive to the number of minority classes, and suffers from performance loss on majority classes. It suggests that a good solution should overcome the overfitting problem of oversampling but not by cutting down the size of majority classes.

Based on the analysis, we investigate a group of ensemble approaches, including AdaBoost.NC, on a set of benchmark data sets with multiple minority and/or majority classes with the aim of tackling multi-class imbalance problems effectively and efficiently. When the ensembles are trained without using class decomposition, AdaBoost.NC working with random oversampling shows better G-mean and minority-class recall than the others, which indicates good generalization for the minority class and the superior ability to balance the performance across different classes.

Our results also show that using class decomposition (the one-against-all scheme in our experiments – OAA) does not provide any advantages in multi-class imbalance learning. For AdaBoost.NC, its G-mean and minority-class recall is even reduced significantly by the use of class decomposition. The reason for this performance degradation seems to be the loss of global information of class distributions in the process of class decomposition. An improved combination method for the OAA scheme is therefore proposed, which assigns different weights to binary classifiers learnt from the sub-problems after the decomposition. The weight is decided by the proportion of the corresponding class within the data set, which delivers the distribution information of each class. By doing so, the effectiveness of AdaBoost.NC in G-mean and minority-class recall is improved significantly.

In regard to other methods, SMOTEBoost shows quite stable performance in terms of MAUC and G-mean. Oversampling itself does not bring much benefit to AdaBoost. Undersampling harms majority-class performance greatly.

Finally, we compare the ensembles without using OAA and the ones using OAA with the weighted combination method. The result suggests that it is not necessary to use class decomposition, and learning from the whole data set directly is sufficient for class imbalance learning techniques to achieve good performance.

Future work of this study includes: (a) an in-depth study of conditions, including parameter values, under which an ensemble approach, such as AdaBoost.NC, is able to improve the performance of multi-class imbalance learning; currently, the parameter of λ in AdaBoost.NC is pre-defined, and a large λ shows greater benefits; some parameter optimizing methods might be helpful here [39]; (b) investigation of other two-class imbalance learning methods into how their effectiveness is affected by multi-class and their potential advantages, such as RAMOBoost [36], RareBoost [45], JOUS-Boost [37] and cost-sensitive methods; (c) a theoretical study of the advantages and disadvantages of the proposed methods for multi-class imbalance problems and how they handle the multi-class imbalance; (d) investigation of new ensemble algorithms that combine the strength of AdaBoost.NC and SMOTEBoost; (e) a theoretical framework for analyzing multi-class imbalance problems since it is unclear how an imbalance rate could be more appropriately defined.

Acknowledgment

This research was supported by an ORS Award, an EPSRC grant (No. EP/D052785/1), and an European FP7 grant (No. 270428).

References

[1] S. Wang, H. Chen, and X. Yao, "Negative correlation learning for classification ensembles," in *International Joint Conference on Neural Networks, WCCI.* IEEE Press, 2010, pp. 2893–2900.
[2] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
[3] C. Li, "Classifying imbalanced data using a bagging ensemble variation," in *ACM-SE 45: Proceedings of the 45th Annual Southeast Regional Conference*, 2007, pp. 203–208.
[4] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *Knowledge Discovery in Databases: PKDD 2003*, vol. 2838, 2003, pp. 107–119.
[5] X.-M. Zhao, X. Li, L. Chen, and K. Aihara, "Protein classification with imbalanced data," *Proteins: Structure, Function, and Bioinformatics*, vol. 70, pp. 1125–1132, 2008.
[6] K. Chen, B.-L. Lu, and J. T. Kwok, "Efficient classification of multi-label and imbalanced data using min-max modular classifiers," in *International Joint Conference on Neural Networks*, 2006, pp. 1770–1775.
[7] A. C. Tan, D. Gilbert, and Y. Deville, "Multi-class protein fold classification using a new ensemble machine learning approach," in *Genome Informatics*, vol. 14, 2003, pp. 206–217.
[8] T. W. Liao, "Classification of weld flaws with imbalanced class data," *Expert Systems with Applications*, vol. 35, no. 3, pp. 1041–1052, 2008.
[9] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, 2006, pp. 63– 77.
[10] G. Ou and Y. L. Murphey, "Multi-class pattern classification using neural networks," *Pattern Recognition*, vol. 40, no. 1, pp. 4–18, 2007.
[11] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *Journal of Machine Learning Research*, vol. 5, pp. 101–141, 2004.
[12] R. Jin and J. Zhang, "Multi-class learning by smoothed boosting," *Machine Learning*, vol. 67, no. 3, pp. 207–227, 2007.
[13] H. Valizadegan, R. Jin, and A. K. Jain, "Semi-supervised boosting for multi-class classification," *Machine Learning and Knowledge Discovery in Databases*, vol. 5212, pp. 522–537, 2008.
[14] S. Wang and X. Yao, "Negative correlation learning for class imbalance problems," *IEEE Transactions on Neural Networks*, p. (in preparation), 2011.
[15] T. Hastie and R. Tibshirani, "Classification by pairwise coupling," *The Annals of Statistics*, vol. 26, no. 2, pp. 451–471, 1998.
[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 341–378, 2002.

[17] A. Fernández, M. J. del Jesus, and F. Herrera, "Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning," *Computational Intelligence for Knowledge-Based System Design*, vol. 6178, pp. 89–98, 2010.
[18] R. Alejo, J. M. Sotoca, R. M. Valdovinos, and G. A. Casañ, "The multi-class imbalance problem: Cost functions with modular and non-modular neural networks," in *The Sixth International Symposium on Neural Networks*, vol. 56, 2009, pp. 421–431.
[19] Y. Sun, M. S. Kamel, and Y. Wang, "Boosting for learning multiple classes with imbalanced class distribution," in *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, 2006, pp. 592–602.
[20] M. Kubat, R. Holte, and S. Matwin, "Learning when negative examples abound," in *The 9th European Conference on Machine Learning*, vol. 1224, 1997, pp. 146–153.
[21] C. V. Rijsbergen, *Information Retrieval*. London, Butterworths, 1979.
[22] 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*, vol. 55, pp. 119–139, 1997.
[23] Y. Sun, A. K. Wong, and Y. Wang, "Parameter inference of cost-sensitive boosting algorithms," in *Proceedings of 4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, 2005, pp. 21–30.
[24] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.
[25] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.
[26] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine Learning*, vol. 45, no. 2, pp. 171–186, 2001.
[27] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble," *IEEE Transactions on Systems, Man and Cybernetics, PartB: Cybernetics*, vol. 29, no. 6, pp. 716–725, 1999.
[28] M. M. Islam, X. Yao, S. M. S. Nirjon, M. A. Islam, and K. Murase, "Bagging and boosting negatively correlated neural networks," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 38, no. 3, pp. 771–784, 2008.
[29] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. of the 13th. Int. Conf. on Machine Learning*, 1996, pp. 148–156.
[30] S. Wang and X. Yao, "Relationships between diversity of classification ensembles and single-class performance measures," *IEEE Transactions on Knowledge and Data Engineering*, 2011 (DOI: 10.1109/TKDE.2011.207).
[31] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine Learning*, vol. 37, no. 3, pp. 297–336, 1999.
[32] S. Wang and X. Yao, "The effectiveness of a new negative correlation learning algorithm for classification ensembles," in *IEEE International Conference on Data Mining Workshops*, 2010, pp. 1013–1020.
[33] Z.-H. Zhou and X.-Y. Liu, "On multi-class cost-sensitive learning," in *Proceedings of the 21st National Conference on Artificial Intelligence*, 2006, pp. 567–572.
[34] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.
[35] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco, CA: Morgan Kaufmann., 2005.
[36] S. Chen, H. He, and E. A. Garcia, "Ramoboost: Ranked minority oversampling in boosting," *IEEE Transactions on Neural Networks*, vol. 21, no. 10, pp. 1624–1642, 2010.
[37] D. Mease, A. J. Wyner, and A. Buja, "Boosted classification trees and class probability/quantile estimation," *The Journal of Machine Learning Research*, vol. 8, pp. 409–439, 2007.
[38] A. Frank and A. Asuncion. (2010) UCI machine learning repository: http://archive.ics.uci.edu/ml.
[39] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 225–252, 2008.
[40] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.
[41] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.
[42] R. L. Iman and J. M. Davenport, "Approximations of the critical region of the friedman statistic," *Communications in statistics*, vol. 9, no. 6, pp. 571–595, 1980.
[43] O. J. Dunn, "Multiple comparisons among means," *Journal of the American Statistical Association*, vol. 56, no. 293, pp. 52–64, 1961.
[44] M. V. Joshi, R. C. Agarwal, and V. Kumar, "Predicting rare classes: can boosting make any weak learner strong?" in *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 297–306.
[45] M. V. Joshi, V. Kumar, and R. C. Agarwal, "Evaluating boosting algorithms to classify rare classes: Comparison and improvements," in *IBM Research Report*, 2001, pp. 257–264.

Shuo Wang received the B.Sc. degree in Computer Science from the Beijing University of Technology (BJUT), China, in 2006, and was a member of Embedded Software and System Institute in BJUT in 2007. She received the Ph.D. degree in Computer Science from the University of Birmingham, U.K., in 2011, sponsored by the Overseas Research Students Award (ORSAS) from the British Government (2007). She is currently a post-doctoral Research Fellow at the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA) in the School of Computer Science, the University of Birmingham. Her research interests include class imbalance learning, ensemble learning, machine learning and data mining.

Xin Yao (M'91-SM'96-F'03) is a Chair (Full Professor) of Computer Science at the University of Birmingham, UK, since the April Fool's Day in 1999. He directs the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA). He is an IEEE Fellow, an IEEE CIS Distinguished Lecturer, a Distinguished Visiting Professor at the University of Science and Technology of China (USTC), and a former EIC of *IEEE Trans. on Evol. Comput.* His research interests include evolutionary computation and neural network ensembles. After proposing negative correlation learning for ensembles (Y. Liu and X. Yao, "Ensemble learning via negative correlation," Neural Networks, 12(10):1399-1404, 1999.), he extended it further recently (H. Chen and X. Yao, "Regularized Negative Correlation Learning for Neural Network Ensembles," IEEE TNN, 20(12):1962-1979, 2009. H. Chen and X. Yao, "Multiobjective Neural Network Ensembles based on Regularized Negative Correlation Learning," IEEE TKDE, 22(12):1738-1751, 2010.). Currently, he is working on online learning, class imbalance learning and semi-supervised learning, using ensemble approaches. He was a recipient of the 2001 IEEE Donald G. Fink Prize Paper Award, 2010 *IEEE Transactions on Evolutionary Computation* Outstanding Paper Award, 2011 *IEEE Transactions on Neural Networks* Outstanding Paper Award, and several other best paper awards.