Resampling-Based Ensemble Methods for Online Class Imbalance Learning

Shuo Wang, *Member, IEEE,* Leandro L. Minku, *Member, IEEE,* and Xin Yao, *Fellow, IEEE*

Abstract—Online class imbalance learning is a new learning problem that combines the challenges of both online learning and class imbalance learning. It deals with data streams having very skewed class distributions. This type of problems commonly exists in real-world applications, such as fault diagnosis of real-time control monitoring systems and intrusion detection in computer networks. In our earlier work, we defined class imbalance online, and proposed two learning algorithms OOB and UOB that build an ensemble model overcoming class imbalance in real time through resampling and time-decayed metrics. In this paper, we further improve the resampling strategy inside OOB and UOB, and look into their performance in both static and dynamic data streams. We give the first comprehensive analysis of class imbalance in data streams, in terms of data distributions, imbalance rates and changes in class imbalance status. We find that UOB is better at recognizing minority-class examples in static data streams, and OOB is more robust against dynamic changes in class imbalance status. The data distribution is a major factor affecting their performance. Based on the insight gained, we then propose two new ensemble methods that maintain both OOB and UOB with adaptive weights for final predictions, called WEOB1 and WEOB2. They are shown to possess the strength of OOB and UOB with good accuracy and robustness.

Index Terms—Class imbalance, resampling, online learning, ensemble learning, Bagging.

1 Introduction

Online class imbalance learning is an emerging topic that is attracting growing attention. It aims to tackle the combined issue of online learning [1] and class imbalance learning [2]. Different from incremental learning that processes data in batches, online learning here means learning from data examples "one-by-one" without storing and reprocessing observed examples [3]. Class imbalance learning handles a type of classification problems where some classes of data are heavily underrepresented compared to other classes. With both problems, online class imbalance learning deals with data streams where data arrive continuously and the class distribution is imbalanced. Although online learning and class imbalance learning have been well studied in the literature individually, the combined problem has not been discussed much. It is commonly seen in real world applications, such as intrusion detection in computer networks and fault diagnosis of control monitoring systems [4].

When both issues of online learning and class imbalance exist, new challenges and interesting research questions arise, with regards to the prediction accuracy on the minority class and adaptivity to dynamic environments. The difficulty of learning from imbalanced data is caused by the relatively or absolutely underrepresented class that cannot draw equal attention to the learning algorithm compared to the majority class. It often leads to very specific classification rules or missing rules for the minority class without much generalization ability for future prediction [5]. This problem is exaggerated when data arrive in an online fashion. First, we cannot get a whole picture of data to evaluate the imbalance status. An online definition of class imbalance is necessary to describe the current imbalance degree. Second, the imbalance status can change over time. Therefore, the online model needs to be kept updated for good performance on the current minority class without damaging the performance on the current majority class.

As one of the earliest studies that focus on online class imbalance learning, our recent work proposed an online definition of class imbalance through two indicators (i.e. time-decayed class size and recall), and a class imbalance detection method to report the real-time class imbalance status in the data stream [6]. Based on the status information, we proposed two online ensemble learning methods – Oversampling-based Online Bagging (OOB) and Undersampling-based Online Bagging (UOB) [6], which can adjust the learning bias from the majority to the minority class effectively and adaptively through resampling.

However, because the resampling rate in OOB and UOB does not consider the size ratio between classes, there exists an issue that the resampling rate is not consistent with the imbalance degree in data and varies with the number of classes. Besides, no existing work has studied the fundamental issues of class imbalance in online cases and the adaptivity of online

• *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, L.L.Minku, X.Yao}@cs.bham.ac.uk*

learners to deal with dynamic data streams with a varying imbalance rate so far. Most work focuses on online learning with concept drifts that involve classification boundary shifts. This paper will give the first comprehensive analysis of class imbalance in data streams.

First, we improve the resampling setting strategies in OOB and UOB. Second, we look into the performance of improved OOB and UOB in both static and dynamic data streams, aiming for a deep understanding of the roles of resampling and time-decayed metrics. We also give a statistical analysis of how their performance is affected by data-related factors and algorithm settings through factorial ANOVA. Generally speaking, our experiments show the benefits of using resampling and time-decayed metrics in the improved OOB and UOB. Particularly, UOB is better at recognizing minority-class examples in static data streams, and OOB is more robust against dynamic changes in class imbalance status. The data distribution is a major factor affecting their performance. Based on the achieved results, for better accuracy and robustness under dynamic scenarios, we propose two ensemble strategies that maintain both OOB and UOB with adaptive weight adjustment, called WEOB1 and WEOB2. They are shown to successfully combine the strength of OOB and UOB. WEOB2 outperforms WEOB1 in terms of G-mean.

2 Learning Imbalanced Data Streams

In this section, we define class imbalance under online scenarios, introduce the two classification methods (i.e. OOB and UOB), and review the research progress in learning from imbalanced data streams. They form the basis of this paper.

2.1 Defining Class Imbalance

To handle class imbalance online, we first need to define it by answering the following three questions: 1) is the data stream currently imbalanced? 2) Which classes belong to the minority/majority? 3) What is the imbalance rate currently? We answered the questions by defining two online indicators – time-decayed class size and recall calculated for each class [6]. Different from the traditional way of considering all observed examples so far equally, they are updated incrementally by using a time decay (forgetting) factor to emphasize the current status of data and weaken the effect of old data.

Suppose a sequence of examples (x_t, y_t) arriving one at a time. x_t is a p-dimensional vector belonging to an input space X observed at time t, and y_t is the corresponding label belonging to the label set $Y = \{c_1, \ldots, c_N\}$. For any class $c_k \in Y$, the class size indicates the occurrence probability (percentage) of examples belonging to c_k. If the label set contains only two classes, then the size of the minority class is referred to as the imbalance rate (IR) of the data stream, reflecting how imbalanced the data stream is at the current moment. The recall of class c_k indicates the classification accuracy on this class. To reflect the current characteristics of the data stream, these two indicators are incrementally updated at each time step. When a new example x_t arrives, the size of each class, denoted by $w_k^{(t)}$, is updated by [6]:

$$w_k^{(t)} = \theta w_k^{(t-1)} + (1-\theta)\left[(x_t, c_k)\right], (k = 1, \ldots, N) \quad (1)$$

where $[(x_t, c_k)] = 1$ if the true class label of x_t is c_k, otherwise 0. θ $(0 < \theta < 1)$ is a pre-defined time decay factor, which forces older data to affect the class percentage less along with time through the exponential smoothing. Thus, $w_k^{(t)}$ is adjusted more based on new data.

If x_t's real label y_t is c_i, the recall of class c_i, denoted by $R_i^{(t)}$, is updated by [6]:

$$R_i^{(t)} = \theta' R_i^{(t-1)} + (1-\theta')\left[x_t \leftarrow c_i\right]. \quad (2)$$

For the recall of the other classes, denoted by $R_j^{(t)}$ $(j \neq i)$, it is updated by:

$$R_j^{(t)} = R_j^{(t-1)}, (j = 1, \ldots, N, j \neq i). \quad (3)$$

In Eq.2, θ' $(0 < \theta' < 1)$ is the time decay factor for emphasizing the learner's performance at the current moment. $[x_t \leftarrow c_i]$ is equal to 1 if x is correctly classified, and 0 otherwise. $\theta = 0.9$ and $\theta' = 0.9$ were shown to be a reasonable setting to balance the responding speed and the estimation variance in our experiments [6].

$w^{(t)}$ and $R^{(t)}$ values are then used by a class imbalance detection method, which outputs the information of whether the data stream should be regarded as imbalanced and which classes should be treated as the minority class. It examines two conditions for any two classes c_i and c_j:

- $w_i - w_j > \delta_1$ $(0 < \delta_1 < 1)$
- $R_i - R_j > \delta_2$ $(0 < \delta_2 < 1)$

If both conditions are satisfied, then class c_j is sent to the minority class label set Y_{min} and class c_i is sent to the majority class label set Y_{maj}. If Y_{min} and Y_{maj} are not empty, it means that the data stream is imbalanced. This can then be used to invoke the class imbalance techniques running in the online model to tackle the imbalanced distribution. Recall is used as one criterion of estimating class imbalance status, because it has been agreed that the imbalance rate is not the only factor that causes the classification difficulty [7].

2.2 Online Solutions OOB and UOB

With the information from the class imbalance detection method, we proposed OOB and UOB to learn imbalanced data streams [6]. They integrate resampling

into ensemble algorithm Online Bagging (OB) [8]. Resampling is one of the simplest and most effective techniques of tackling class imbalance [9] [10]. It works at the data level independently of the learning algorithm. Two major types of resampling are oversampling – increasing the number of minority-class examples, and undersampling – reducing the number of majority-class examples. Online Bagging [8] is an online ensemble learning algorithm that extends the offline Bagging [11]. It builds multiple base classifiers and each classifier is trained K times by using the current training example, where K follows the $Poisson\,(\lambda = 1)$ distribution. Poisson distribution is used, because the Binomial distribution K in offline Bagging tends to the Poisson(1) distribution when the bootstrap sample size becomes infinity.

Once Y_{min} and Y_{maj} (the output of the class imbalance detection method) are not empty, oversampling or undersampling embedded in Online Bagging will be triggered to either increase the chance of training minority-class examples (in OOB) or reduce the chance of training majority-class examples (in UOB). Their training procedures are given in Table 1.

TABLE 1: OOB and UOB Training Procedures.

Input: label sets Y_{min} and Y_{maj}, an ensemble with M base learners, and current training example (x_t, y_t).

for each base learner f_m $(m = 1, 2, \ldots, M)$ **do**
  if $y_t \in Y_{min}$
    set $K \sim Poisson\,(\lambda_1)$
  else
    set $K \sim Poisson\,(\lambda_2)$
  end if
  update f_m K times
end for

Resampling in OOB and UOB is performed through the parameter λ of Poisson distribution to handle class imbalance. In OOB, λ_1 is set to $1/w_k^{(t)}$; λ_2 is set to 1. In UOB, λ_1 is set to 1, λ_2 is set to $\left(1 - w_k^{(t)}\right)$. If the new training example belongs to the minority class, OOB increases value K, which decides how many times to use this example for training. Similarly, if it belongs to the majority class, UOB decreases K. The advantages of OOB and UOB are: 1) resampling is algorithm-independent, which allows any type of online classifiers to be used; 2) time-decayed class size used in OOB and UOB dynamically estimates imbalance status without storing old data or using windows, and adaptively decides the resampling rate at each time step; 3) like other ensemble methods, they combine the predictions from multiple classifiers, which are expected to be more accurate than a single classifier.

2.3 Existing Research

Most existing algorithms dealing with imbalanced data streams require processing data in batches/chunks (incremental learning), such as MuSeRA [12] and REA [13] proposed by Chen et al., and Learn++.CDS and Learn++.NIE [14] proposed by Ditzler and Polikar. Among limited class imbalance solutions strictly for online processing, Nguyen et al. first proposed an algorithm to deal with imbalanced data streams through random undersampling [15]. The majority class examples have a lower probability to be selected for training. It assumes that the information of which class belongs to the minority/majority is known and the imbalance rate does not change over time. Besides, it requires a training set to initialize the classification model before learning. Minku et al. [16] proposed to use undersampling and oversampling to deal with class imbalance in online learning by changing the parameter corresponding to Online Bagging's [8] sampling rate. However, the sampling parameters need to be set prior to learning and cannot be adjusted to changing imbalance rates. Very recently, two perceptron-based methods RLSACP [17] and WOS-ELM [18] were proposed, which assign different misclassification costs to classes to adjust the weights between perceptrons. The error committed on the minority class suffers a higher cost. RLSACP adopts a window-based strategy to update misclassification costs based on the number of examples in each class at a pre-defined speed. WOS-ELM requires a validation set to adjust misclassification costs based on classification performance, which however may not be available in many real-world applications. They were tested in static scenarios with a fixed imbalance rate and shown to be effective.

3 Improved OOB and UOB

In the current OOB and UOB methods, the choice of λ for one class only depends on the size of this class. Therefore, this training strategy can be applied to multi-class cases directly, having more than one minority or majority class. However, when the data stream becomes balanced, λ will not be equal to 1. It means that the resampling keeps running even when the data is balanced, if the class imbalance detection method is not applied as the trigger. For example, given a balanced 2-class problem, where $w_k^{(t)}$ is equal to 0.5, λ for one class will be set to 2 by OOB; given a balanced 5-class problem, where $w_k^{(t)}$ is equal to 0.2, λ for one class will be set to 5 by OOB. On the one hand, the strategy of setting λ is not consistent with the imbalance degree, and varies with the number of classes. On the other hand, it is necessary to use the class imbalance detection method, to inform OOB and UOB of when resampling should be applied.

To overcome this issue, we improve OOB and UOB with a better parameter setting strategy in this section. λ in the improved versions is determined by the size ratio between classes. The pseudo-code of improved

OOB and UOB at each training step t is given in Table 2, assuming there are two possible classes $Y = \{+1, -1\}$ (denoted by '+' and '-' for simplicity).

TABLE 2: Improved OOB and UOB.

Input: an ensemble with M base learners, current training example (x_t, y_t), and current class size $w^{(t)} = \left(w_+^{(t)}, w_-^{(t)}\right)$.

for each base learner f_m $(m = 1, 2, \ldots, M)$ **do**
  if $y_t = +1$ and $\begin{cases} w_+^{(t)} < w_-^{(t)} \text{ for OOB} \\ w_+^{(t)} > w_-^{(t)} \text{ for UOB} \end{cases}$
    set $K \sim Poisson\left(w_-^{(t)}/w_+^{(t)}\right)$
  else if $y_t = -1$ and $\begin{cases} w_-^{(t)} < w_+^{(t)} \text{ for OOB} \\ w_-^{(t)} > w_+^{(t)} \text{ for UOB} \end{cases}$
    set $K \sim Poisson\left(w_+^{(t)}/w_-^{(t)}\right)$
  else
    set $K \sim Poisson\,(1)$
  end if
  update f_m K times
end for

If the size of the positive class is smaller than the size of the negative class at the current moment, λ for the positive class will be set to $w_-^{(t)}/w_+^{(t)}$ for oversampling in OOB; λ for the negative class will be set to $w_+^{(t)}/w_-^{(t)}$ for undersampling in UOB. With the same Bagging strategy as the traditional Online Bagging (OB) [8], improved OOB and UOB only sweep through each training example once. They need $O\,(M)$ time to keep the online model up-to-date, when the base classifiers are updated sequentially.

When data become balanced, these methods will be reduced to OB automatically. For any 2-class problems, therefore, it is not necessary to apply the class imbalance detection method any more for a hard partition of class labels into minority and majority label sets. For multi-class problems, however, this hard partition is still necessary for choosing $w_{min}^{(t)}$ and $w_{maj}^{(t)}$, where $w_{min}^{(t)}$ $(w_{maj}^{(t)})$ stands for the class size of the smaller (larger) class. For example, for a data stream with 4 classes $\{c_1, c_2, c_3, c_4\}$, their proportions are 0.05, 0.15, 0.25, 0.55 respectively. Class c_3 is a majority class relative to c_1, but a minority class relative to c_4. Since we do not want to overlook the performance of any minority class, the class imbalance detection method will label c_3 as minority. This information is then used by OOB and UOB to treat this class. Without recognizing this situation, it would be difficult to determine the way of sampling. This paper focuses on 2-class imbalanced data streams, so we do not use the class imbalance detection method to trigger resampling in improved OOB and UOB here. How it helps the classification in multi-class cases will be studied as our next-step work. All the following analysis will be based on the improved OOB and UOB. We will simply use "OOB" and "UOB" to indicate the improved ones for brevity.

4 Class Imbalance Analysis in Static Data Streams

This section studies OOB and UOB in static data streams without any changes. We focus on the fundamental issue of class imbalance and look into the following questions under different imbalanced scenarios: 1) to what extent does resampling in OOB and UOB help to deal with class imbalance online? 2) How do they perform in comparison with other state-of-the-art algorithms? 3) How are they affected by different types of class imbalance and classifiers? For the first question, we compare OOB and UOB with OB [8], to show the effectiveness of resampling. For the second question, we compare OOB and UOB with two recently proposed learning algorithms, RLSACP [17] and WOS-ELM [18], which also aim to tackle online imbalanced data. For the third question, we perform a mixed (split-plot) factorial analysis of variance (ANOVA) [19] to analyse the impact of three factors, including the data distribution and imbalance rate in data, and the base classifier in OOB and UOB.

4.1 Data Description and Experimental Settings

This section describes the static imbalanced data used in the experiment, including 12 artificial data streams and 2 real-world data streams, and explains the algorithm settings and experimental designs for a clear understanding in the following analysis.

4.1.1 Data

To facilitate a deep understanding and accurate analysis, artificial data sets are generated in order to obtain desired types of imbalanced data streams. We produce 12 two-class data streams with different distributions and imbalance rates. The imbalance rate (IR), i.e. the occurrence probability of the minority class, is a direct factor that affects any online learner's performance. A smaller rate means a smaller chance to collect the minority-class examples, and thus a harder case for classification. During the online processing, since it is not possible to get the whole picture of data, the frequency of minority-class examples and data sequence become more important in online learning than in offline learning. In addition to IR, complex data distributions have been shown to be a major factor causing degradation of classification performance in offline class imbalance learning, such as small sub-concepts of the minority class with very few examples [20], and the overlapping between classes [21] [22]. Particularly, authors in [23] distinguished and analysed four types of data distributions in the minority class – safe, borderline, outliers and rare examples. Safe examples are located in the homogenous regions populated by the examples from one class only; borderline examples

are scattered in the boundary regions between classes, where the examples from both classes overlap; rare examples and outliers are singular examples located deeper in the regions dominated by the majority class. Borderline, rare and outlier data sets were found to be the real source of difficulties in real-world data sets, which could also be the case in online applications [23].

In our experiment, we consider 4 imbalance levels (i.e. 5%, 10%, 20% and 30%) and 3 minority-class distributions (i.e. safe, borderline, rare/outlier). Rare examples and outliers are merged into one category, due to the observation that they always appear together [23]. Each data stream has 1000 examples (namely 1000 time steps). Each example is formed of two numeric attributes and a class label that can be +1 or -1. The positive class is fixed to be the minority class. Each class follows the multivariate Gaussian distribution. We vary the mean and covariance matrix settings of the Gaussian distributions to obtain different distributions. For each type of distribution, the data set contains more than 50% of corresponding category of examples, measured by the 5-nearest neighbour method [23]. In the borderline case, for instance, more than 50% minority-class examples meet the "borderline" definition.

For an understanding under practical scenarios, we include 2 real-world data from fault detection applications – Gearbox [24] and Smart Building [25]. The task of Gearbox is to detect faults in a running gearbox using accelerometer data and information about bearing geometry. Smart Building aims to identify sensor faults in smart buildings [25]. Both contain two classes, and the faulty class is the minority. Without loss of generality, we let the minority class be the positive class +1, and let the majority class be the negative class -1. One data stream is generated from each data source. We limit the length of the data stream to 1000 examples and fix IR at 10%.

4.1.2 Settings

In the experiment of comparing OOB, UOB and OB, each method builds an ensemble model composed of 50 Hoeffding trees [26]. Hoeffding tree is an incremental, anytime decision tree induction algorithm capable of learning from high-speed data streams, supported mathematically by the Hoeffding bound and implemented by the Massive Online Analysis (MOA) tool with its default settings [27]. It was shown to be an effective base classifier for the original OOB and UOB [28] [29].

When comparing OOB and UOB with RLSACP and WOS-ELM, we use the multilayer perceptron (MLP) classifier as the base learner of OOB and UOB, considering that RLSACP and WOS-ELM are both perceptron-based algorithms. The number of neurons in the hidden layer of MLP is set to the number of attributes in data, which is also the number of perceptrons in RLSACP and WOS-ELM. The error cost of classes is updated at every 500 time steps in RLSACP. The error cost in WOS-ELM is predefined based on the real imbalance rate in the data stream. OOB and UOB are composed of 50 MLPs. For a clear role of resampling, OOB with a single MLP is also included in the comparison as a benchmark, since RLSACP and WOS-ELM only involve one neural network and they emphasize the minority class by increasing its misclassification cost.

The decay factor for updating class size $w_k^{(t)}$ in OOB and UOB is set to 0.9, based on our preliminary experiments [6]. Every discussed method is repeated 100 times on each data stream. The average prequential performance is recorded at each time step. Prequential test is a popular performance evaluation strategy in online learning, in which each individual example is used to test the model before it is used for training, and from this the performance measures can be incrementally updated. For example, the prequential accuracy at time step t can be calculated by [1]

$$acc\,(t) = \begin{cases} acc_x\,(t)\,, & \text{if } t = 1 \\ acc\,(t-1) + \frac{acc_x(t) - acc(t-1)}{t}, & \text{otherwise} \end{cases}$$

where acc_x is 0 if the prediction of the current training example x before its learning is wrong and 1 if it is correct.

In our prequential test, we choose G-mean and minority-class recall to be the evaluation metrics. They are two most commonly used evaluation criteria in the class imbalance learning literature, as they are insensitive to the imbalance rate. Recall is defined as the classification accuracy on a single class. Minority(positive)-class recall (Rec_p) and majority(negative)-class recall (Rec_n) can be obtained through the following formulas: $Rec_p = TP/P$ and $Rec_n = TN/N$, where TP is the number of true positives, TN is the number of true negatives, and P and N are the total numbers of positive and negative examples observed so far. G-mean is defined as the geometric mean of recalls over all classes [30], e.g. $\sqrt{Rec_p \cdot Rec_n}$ for the two-class case. Recall helps us to analyse the performance within classes, but does not reflect any performance on other classes. G-mean is an overall performance metric. It helps us to understand how well the performance is balanced among classes.

When comparing any two learning algorithms, we use the Wilcoxon Sign Rank test with Holm-Bonferroni corrections to show the difference statistically, at the overall level of significance of 0.05. Holm-Bonferroni corrections are performed to counteract the problem of multiple comparisons.

4.2 Role of Resampling

We compare the final-step minority-class recall and G-mean produced from tree-based OOB, UOB and

OB. They show the final prequential performance step after the online model has gone through all the examples, averaged over multiple runs.

TABLE 3: The *final-step minority-class recall* and statistical test results from tree-based OOB, UOB and OB.

Data		OOB	UOB	OB
Distribute	IR			
safe	30%	0.970±0.002 ***(0.00000)***	0.973±0.001	0.969±0.001 ***(0.00000)***
	20%	0.981±0.003 ***(0.00000)***	0.964±0.002	0.964±0.004 ***(0.00510)***
	10%	0.912±0.007 ***(0.00000)***	0.936±0.005	0.905±0.007 ***(0.00000)***
	5%	0.840±0.000 ***(0.00000)***	0.917±0.013	0.876±0.019 ***(0.00000)***
borderline	30%	0.636±0.013 ***(0.00000)***	0.774±0.007	0.462±0.015 ***(0.00000)***
	20%	0.577±0.007 ***(0.00000)***	0.811±0.007	0.266±0.046 ***(0.00000)***
	10%	0.424±0.018 ***(0.00000)***	0.868±0.011	0.026±0.016 ***(0.00000)***
	5%	0.225±0.035 ***(0.00000)***	0.831±0.019	0.000±0.000 ***(0.00000)***
rare/ outlier	30%	0.197±0.016 ***(0.00000)***	0.662±0.033	0.033±0.004 ***(0.00000)***
	20%	0.395±0.015 ***(0.00000)***	0.755±0.014	0.142±0.015 ***(0.00000)***
	10%	0.195±0.024 ***(0.00000)***	0.699±0.014	0.195±0.024 ***(0.00000)***
	5%	0.310±0.010 ***(0.00000)***	0.519±0.021	0.008±0.009 ***(0.00000)***
Gearbox		0.045±0.009 ***(0.00000)***	0.446±0.041	0.000±0.000 ***(0.00000)***
Smart Building		0.430±0.004 ***(0.00000)***	0.764±0.011	0.234±0.103 ***(0.00000)***

Tables 3 - 4 present their means and standard deviations over the 100 runs. We can see that UOB achieves the highest minority-class recall and G-mean in almost all cases, and OB gets the lowest. Both oversampling in OOB and undersampling in UOB improve the prediction accuracy on the minority class and the overall performance greatly compared to OB. UOB is more effective. These observations are further confirmed by our statistical test, between UOB and OOB/OB. Twenty eight pairs of comparisons are involved for each performance metric. The resulting p-values are included in brackets in the tables. Those in bold italics suggest a significant difference between UOB and the method in the corresponding column.

Moreover, the three models show quite high performance in the cases with a 'safe' distribution, but the performance gets much worse in the other cases. Particularly, OB suffers a greater performance reduction. It means that 'borderline' and 'rare/outlier' are harder distributions than 'safe' in online learning. A smaller IR does not necessarily cause worse performance from the results here. More details about the impact of data distribution and IR will be given in Section 4.4.

4.3 Comparison with Other Algorithms

This section compares MLP-based OOB and UOB with two state-of-the-art methods RLSACP and WOS-ELM. OOB with a single MLP is included as a benchmark, denoted by OOB_{sg}. Tables 5 - 6 present their final-step minority-class recall and G-mean respectively. We can observe that UOB is the winner of minority-class recall in most cases, and OOB is the winner of G-mean. The Wilcoxon Sign Rank test is thus carried out between UOB and every other method for minority-class recall and between OOB and every other method for G-mean. There are 56 pairs of comparison for each metric. P-values are included in the tables.

TABLE 4: The *final-step G-mean* and statistical test results from tree-based OOB, UOB and OB.

Data		OOB	UOB	OB
Distribute	IR			
safe	30%	0.984±0.001 ***(0.00000)***	0.978±0.001	0.983±0.001 ***(0.00000)***
	20%	0.981±0.001 ***(0.00000)***	0.971±0.001	0.975±0.002 ***(0.00000)***
	10%	0.953±0.003 ***(0.00000)***	0.959±0.002	0.951±0.003 ***(0.00000)***
	5%	0.916±0.000 ***(0.00000)***	0.954±0.006	0.936±0.010 ***(0.00000)***
borderline	30%	0.705±0.005 ***(0.00000)***	0.735±0.003	0.631±0.009 ***(0.00000)***
	20%	0.712±0.003 ***(0.00000)***	0.786±0.003	0.500±0.045 ***(0.00000)***
	10%	0.636±0.013 ***(0.00000)***	0.859±0.004	0.154±0.048 ***(0.00000)***
	5%	0.470±0.036 ***(0.00000)***	0.872±0.008	0.000±0.000 ***(0.00000)***
rare/ outlier	30%	0.405±0.014 ***(0.00000)***	0.532±0.016	0.181±0.013 ***(0.00000)***
	20%	0.565±0.009 ***(0.00000)***	0.651±0.011	0.374±0.020 ***(0.00000)***
	10%	0.435±0.027 ***(0.00000)***	0.697±0.007	0.435±0.027 ***(0.00000)***
	5%	0.547±0.008 ***(0.00000)***	0.641±0.011	0.058±0.069 ***(0.00000)***
Gearbox		0.206±0.023 ***(0.00000)***	0.498±0.013	0.000±0.000 ***(0.00000)***
Smart Building		0.609±0.003 ***(0.00000)***	0.689±0.008	0.472±0.108 ***(0.00000)***

Although undersampling in MLP-based UOB improves minority-class recall greatly, we notice that the majority-class recall is dragged down too much, which explains why its overall performance is worse than OOB. RLSACP and WOS-ELM outperform OOB in terms of minority-class recall in some cases. However, because their majority-class recall is decreased more than the increase of minority-class recall, their G-mean does not beat OOB's. Besides, we notice that WOS-ELM presents very large performance variance in many cases. This is caused by the fact that the minority-class examples are overemphasized and the majority-class performance is sacrificed greatly in some runs. A possible explanation for this phenomenon is that the extreme learning machine (ELM) used in WOS-ELM was found to be sensitive to outliers in data and lacks robustness sometimes [31]. Especially after the minority class is further emphasized by a higher weight in WOS- ELM, the empirical risk minimization principle of ELM is more likely to lead to great bias towards the minority class.

TABLE 5: The *final-step minority-class recall* and statistical test results from MLP-based OOB, OOB_{sg}, UOB, RLSACP and WOS-ELM.

Data Distribute	IR	OOB	UOB	RLSACP	WOS-ELM	OOB_{sg}
safe	30%	0.973±0.004 ***(0.00000)***	0.986±0.001	0.490±0.344 ***(0.00000)***	0.332±0.466 ***(0.00000)***	0.959±0.016 ***(0.00000)***
	20%	0.979±0.003 ***(0.00000)***	0.992±0.007	0.551±0.373 ***(0.00000)***	0.283±0.443 ***(0.00000)***	0.960±0.026 ***(0.00000)***
	10%	0.923±0.005 ***(0.00620)***	0.900±0.048	0.593±0.302 ***(0.00000)***	0.586±0.466 (0.97450)	0.920±0.010 (0.0231)
	5%	0.880±0.000 ***(0.00000)***	0.741±0.120	0.027±0.025 ***(0.00000)***	0.414±0.449 (0.02530)	0.884±0.025 ***(0.00000)***
borderline	30%	0.488±0.016 ***(0.00000)***	0.828±0.036	0.512±0.020 ***(0.00000)***	0.194±0.286 ***(0.00000)***	0.515±0.074 ***(0.00000)***
	20%	0.369±0.017 ***(0.00000)***	0.854±0.055	0.535±0.028 ***(0.00000)***	0.520±0.341 ***(0.00002)***	0.392±0.079 ***(0.00000)***
	10%	0.293±0.011 ***(0.00000)***	0.806±0.106	0.481±0.064 ***(0.00000)***	0.296±0.398 ***(0.00000)***	0.315±0.060 ***(0.00000)***
	5%	0.015±0.008 ***(0.00000)***	0.456±0.212	0.039±0.017 ***(0.00000)***	0.417±0.348 (0.21640)	0.032±0.026 ***(0.00000)***
rare/ outlier	30%	0.196±0.019 ***(0.00000)***	0.727±0.047	0.493±0.115 ***(0.00000)***	0.559±0.149 ***(0.00000)***	0.303±0.072 ***(0.00000)***
	20%	0.370±0.011 ***(0.00000)***	0.853±0.052	0.468±0.015 ***(0.00000)***	0.234±0.266 ***(0.00000)***	0.391±0.040 ***(0.00000)***
	10%	0.390±0.015 ***(0.00000)***	0.839±0.088	0.521±0.144 ***(0.00000)***	0.363±0.419 ***(0.00000)***	0.408±0.044 ***(0.00000)***
	5%	0.181±0.004 ***(0.00000)***	0.479±0.200	0.111±0.030 ***(0.00000)***	0.164±0.270 ***(0.00000)***	0.212±0.045 ***(0.00000)***
Gearbox		0.008±0.005 ***(0.00000)***	0.697±0.110	0.042±0.023 ***(0.00000)***	0.888±0.022 ***(0.00000)***	0.049±0.037 ***(0.00000)***
Smart Building		0.065±0.014 ***(0.00000)***	0.552±0.075	0.161±0.280 ***(0.00000)***	0.484±0.011 ***(0.00000)***	0.109±0.058 ***(0.00000)***

TABLE 6: The *final-step G-mean* and statistical test results from MLP-based OOB, OOB_{sg}, UOB, RLSACP and WOS-ELM.

Data Distribute	IR	OOB	UOB	RLSACP	WOS-ELM	OOB_{sg}
safe	30%	0.972±0.001	0.926±0.007 ***(0.00000)***	0.493±0.345 ***(0.00000)***	0.065±0.066 ***(0.00000)***	0.963±0.006 ***(0.00000)***
	20%	0.970±0.001	0.907±0.010 ***(0.00000)***	0.548±0.365 ***(0.00000)***	0.036±0.077 ***(0.00000)***	0.960±0.008 ***(0.00000)***
	10%	0.957±0.002	0.842±0.025 ***(0.00000)***	0.593±0.304 ***(0.00000)***	0.146±0.135 ***(0.00000)***	0.955±0.005 ***(0.00240)***
	5%	0.933±0.000	0.776±0.040 ***(0.00000)***	0.123±0.105 ***(0.00000)***	0.160±0.181 ***(0.00000)***	0.936±0.013 (0.02430)
borderline	30%	0.586±0.007	0.515±0.022 ***(0.00000)***	0.515±0.079 ***(0.00000)***	0.231±0.137 ***(0.00000)***	0.580±0.026 (0.16150)
	20%	0.537±0.010	0.426±0.050 ***(0.00000)***	0.475±0.086 ***(0.00008)***	0.348±0.138 ***(0.00000)***	0.537±0.041 (0.37770)
	10%	0.500±0.009	0.374±0.096 ***(0.00000)***	0.467±0.123 (0.02940)	0.189±0.113 ***(0.00000)***	0.509±0.042 (0.09680)
	5%	0.104±0.061	0.447±0.060 ***(0.00000)***	0.183±0.055 ***(0.00000)***	0.306±0.172 ***(0.00000)***	0.149±0.093 ***(0.00001)***
rare/ outlier	30%	0.399±0.016	0.463±0.026 ***(0.00000)***	0.513±0.021 ***(0.00000)***	0.476±0.056 ***(0.00000)***	0.458±0.039 ***(0.00000)***
	20%	0.561±0.007	0.425±0.060 ***(0.00000)***	0.482±0.093 ***(0.00000)***	0.254±0.228 ***(0.00000)***	0.558±0.022 (0.93770)
	10%	0.598±0.011	0.447±0.094 ***(0.00000)***	0.516±0.153 (0.09300)	0.163±0.170 ***(0.00000)***	0.605±0.030 (0.03150)
	5%	0.416±0.004	0.450±0.081 ***(0.00000)***	0.316±0.047 ***(0.00000)***	0.162±0.208 ***(0.00000)***	0.446±0.045 ***(0.00000)***
Gearbox		0.077±0.047	0.459±0.055 ***(0.00000)***	0.189±0.063 ***(0.00000)***	0.289±0.022 ***(0.00000)***	0.198±0.088 ***(0.00000)***
Smart Building		0.243±0.027	0.485±0.020 ***(0.00000)***	0.220±0.081 ***(0.00033)***	0.527±0.004 ***(0.00000)***	0.295±0.082 ***(0.00000)***

OOB and OOB_{sg} have similar performance. OOB_{sg} shows better G-mean than RLSACP and WOS-ELM in most cases. This observation confirms that resampling is the main reason for OOB and UOB outperforming RLSACP and WOS-ELM, rather than the ensemble of classifiers.

4.4 Factorial Analysis of Data-Related Factors and Base Classifiers

We have looked into the effectiveness of OOB and UOB through algorithm comparisons. Based on the results so far, we perform a mixed (split-plot) factorial analysis of variance (ANOVA) [19], to study statistically whether and how their performance is affected by types of class imbalance and base classifiers. Three factors are analysed: data distributions, IR and base classifiers. There are 3 different levels (settings) for 'distribution' (safe, borderline, rare/outlier) and 4 different levels for IR (5%, 10%, 20% and 30%). We consider 2 levels (types) of base classifiers – decision tree and MLP. A mixed design is necessary, because the distribution and IR are between-subjects factors (their levels vary with the data being used), and the base classifier is a within-subjects factor (its levels vary within the data). The factorial design allows the effects of a factor to be estimated at several levels of the other factors. The effects of the factors on the final-step G-mean is discussed. The metric under observation is also called "response" in ANOVA.

The ANOVA results are presented in Table 7, including p-value and eta-squared (η^2). A p-value smaller than 0.05 indicates a significant difference by rejecting the null hypothesis under the significance level of 5%. η^2 is a measure in the range of $[0, 1]$ describing the effect size. The larger the η^2, the greater the effect of the factor. It is worth mentioning here that η^2 is calculated separately for between-subjects and within-subjects factors [32]. For the cases with only between-subjects factors and the cases involving within-subjects factors, the sum of η^2 of the factor effects and the associated standard errors is equal to 1 respectively.

Table 7 shows the factor effects on G-mean. All the p-values are much smaller than 0.05, suggesting that all the three factors and their interactions have a significant impact on G-mean. According to η^2, the effect of data distribution (0.842 for OOB and 0.941 for UOB) is much larger than the effects of IR (0.063 for OOB and 0.001 for UOB) and the base classifier (0.201 for OOB and 0.692 for UOB). The effect

TABLE 7: Factor effects of data distributions (abbr. Dis), imbalance rates (abbr. IR) and base classifiers (abbr. BC), and their interaction effects on *G-mean* from OOB and UOB. The symbol "*" indicates the interaction between two factors.

Between-Subjects Effects				
	OOB		UOB	
	p-val	η^2	p-val	η^2
Dis	0.000	0.842	0.000	0.941
IR	0.000	0.063	0.029	0.001
IR*Dis	0.000	0.091	0.000	0.028
Within-Subjects Effects				
	OOB		UOB	
	p-val	η^2	p-val	η^2
BC	0.000	0.201	0.000	0.692
BC*Dis	0.000	0.438	0.000	0.182
BC*IR	0.000	0.181	0.000	0.062
BC*Dis*IR	0.000	0.152	0.000	0.019

of IR on UOB is much smaller than that on OOB, suggesting that UOB is less sensitive to the imbalance rate. The effect of base classifiers on OOB is much smaller than that on UOB, suggesting that OOB is less sensitive to the choice of base classifiers. Generally speaking, the minority-class distribution is a major factor affecting the performance of OOB and UOB, compared to which their performance is quite robust to the imbalance rate.

5 Class Imbalance Analysis in Dynamic Data Streams

This section focuses on the dynamic feature of online class imbalance. Different from existing concept drift methods that mainly aim for changes in class-conditional probability density functions, we look into the performance of OOB and UOB when tackling data streams with imbalance status changes (i.e. changes in class prior probabilities). In other words, IR is changing over time. It happens in real-world problems. For example, given the task of detecting faults in a running engineering system, the faulty class is usually the minority class that rarely occurs. The faults can become more and more frequent over time, if the damaged condition gets worse; or the faults are not likely to happen from some moment, because the faulty system is repaired. Dynamic data are more challenging than static ones, because the online model needs to sense the change and adjust its learning bias quickly to maintain its performance. Different types of changes in imbalance status are designed and studied here, varying in changing speed and severity.

The adaptivity and robustness of OOB and UOB are studied by answering the following questions: 1) how does the time-decayed metric used in OOB and UOB help to handle the imbalance change? 2) How do OOB and UOB perform in comparison with other state-of-the-art algorithms under dynamic scenarios? 3) How is their performance affected by the decay factor? For the first question, OOB and UOB are compared with those applying the traditional method of updating the class size. For the second question, OOB and UOB are compared with RLSACP and WOS-ELM. For the final question, ANOVA using a repeated measure design is performed to analyse the impact of the decay factor.

5.1 Data Description and Experimental Settings

In this section, we first describe the dynamic imbalanced data used in the experiment, covering various types of changes in terms of changing severity and speed. Then, we give the algorithm settings and experimental designs.

5.1.1 Data

Three data sources are used to produce dynamic data streams – Gaussian data, Gearbox and Smart Building. By using the same data generation method as described in Section 4.1, we produce two-class Gaussian data streams formed of two numeric attributes and a class label that can be +1 or -1. Each class follows the multivariate Gaussian distribution. Some overlapping between classes is enabled for a certain level of data complexity. Gearbox and Smart Building are fault detection data from the real-world applications used in the previous section, containing a faulty class (+1) and a nonfaulty class (-1). We limit the length of each data stream to 1000 examples. Different from static data streams, the dynamic ones involve a class imbalance change right after time step 500. During the first 500 time steps, IR is fixed to 10% and the positive class is the minority. From time step 501, a change occurs at a certain speed and severity. Four data streams are generated from each data source. Each type is a combination of two changing severity levels and two changing speeds. The changing severity can be either high or low; the changing speed can be either abrupt or gradual. The concrete change settings are:

- High severity: the negative class becomes the minority with IR 10% in the new status.
- Low severity: the data stream becomes balanced (i.e. IR = 50%) in the new status.
- Abrupt change: the length of the changing period is 0. The new status completely takes over the data stream from time step 501.
- Gradual change: the change lasts for 300 time steps. During the changing period, the probability of examples being drawn from the old joint distribution decreases linearly.

5.1.2 Settings

When studying how the time-decayed class size $w_k^{(t)}$ in OOB and UOB helps the classification in dynamic data streams, we replace $w_k^{(t)}$ with the class size updated in the traditional way, which considers all observed examples so far equally. They are denoted by OOB_{tr} and UOB_{tr} and compared with OOB and

UOB. All the methods here build an ensemble model composed of 50 Hoeffding trees.

When discussing the adaptivity of OOB and UOB in comparison with RLSACP and WOS-ELM, MLP is used as the base classifier of OOB and UOB as before. Both OOB and UOB consist of 50 MLPs. OOB with a single MLP (OOB_{sg}) is included as the benchmark. The window size for updating the error cost of classes is shortened to 50 time steps in RLSACP, to encourage a faster response to the change. The error cost in the original WOS-ELM was updated based on a separate validation data set in the original article [18], which however may not be available in many real-world problems, or expire over time. To allow the same level of adaptivity as OOB and UOB, the time-decayed class sizes are used to set the costs in WOS-ELM. Specifically, the error cost of the majority class is always equal to 1; the error cost of the minority class is set to $w_{maj}^{(t)}/w_{min}^{(t)}$, where $w_{maj}^{(t)}$ and $w_{min}^{(t)}$ are the time-decayed sizes of the current majority and minority classes respectively.

The same as in Section 4, the prequential recall and G-mean are tracked to observe the performance before and after the change. For a clear and accurate understanding of OOB and UOB learning dynamic data, we divide the prequential evaluation process into three stages, old-status stage, changing stage and new-status stage, by resetting the performance metrics to 0 between the stages. In more details, for the data with an abrupt change (where there is no changing stage), recall and G-mean are reset after time step 500, when the change starts and ends; for the data with a gradual change (where the changing stage is 300 time-step long), recall and G-mean are reset after time step 500 and 800. This is to ensure that the performance observed after the change is not affected by the performance before the change. Prequential performance curves will be presented to help visualize the performance behaviors at each time step and the impacts of changes. For a quantitative understanding, we will provide the average prequential performance covering all the time steps during the new-status stage. It can better reflect the influence of the status change than the final-step performance. The Wilcoxon Sign Rank test with Holm-Bonferroni corrections is applied for statistical analysis.

5.2 Role of the Time-Decayed Metric

In this section, we aim to find out: 1) how different types of imbalance changes affect the performance of OOB and UOB; 2) whether and how the time-decayed class size in OOB and UOB facilitates learning in dynamic data streams, through the comparison with OOB_{tr} and UOB_{tr}. To understand the impact of change on each class, we monitor the behaviours of prequential recall of minority and majority classes individually along with time. Fig. 1 presents the recall curves produced from the data streams using data source Smart Building.

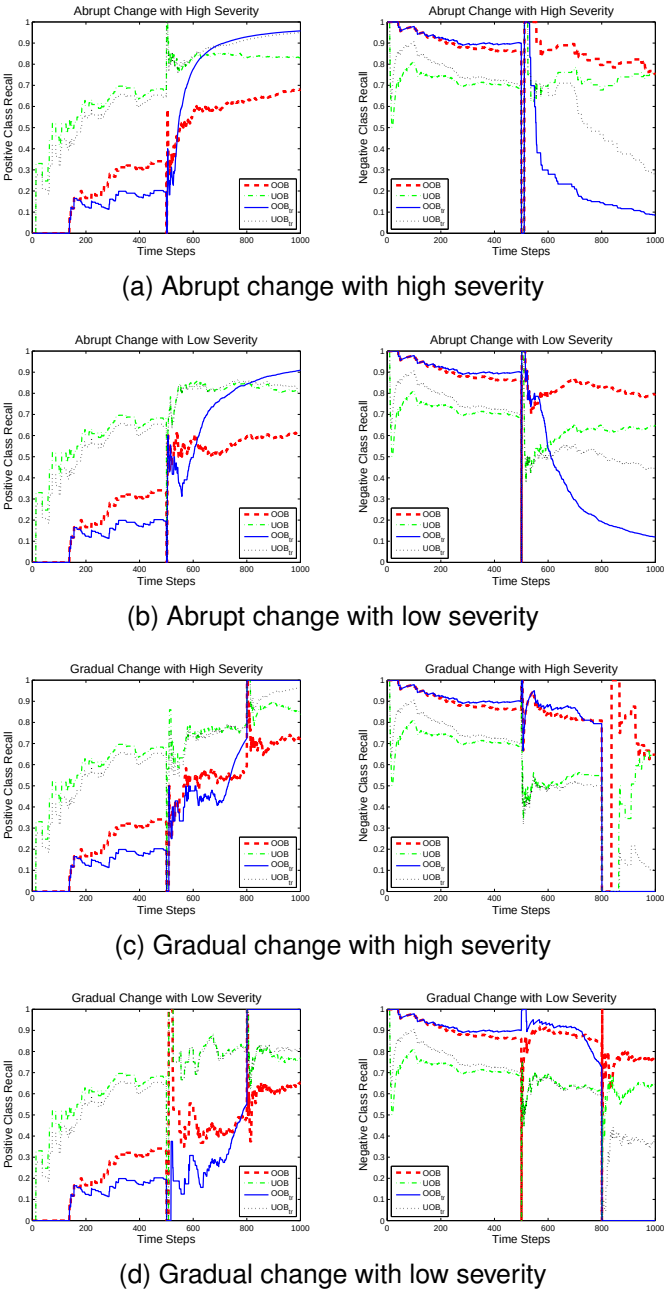

Fig. 1: Prequential recall curves of positive (left) and negative (right) classes from OOB, UOB, OOB_{tr} and UOB_{tr} on Smart Building data with 4 types of changes.

For the abrupt change with high severity, before the change happens, UOB and UOB_{tr} have better minority(positive)-class recall than OOB and OOB_{tr}, which confirms that undersampling is a more aggressive technique of emphasizing the minority class than oversampling. After time step 500 when this class becomes the majority abruptly, all methods present a rapid growth in recall, especially for OOB_{tr} and UOB_{tr}. The growth is caused by the frequent arrival of positive-class examples; OOB_{tr} and UOB_{tr} increase

faster than OOB and UOB, because the imbalance status obtained by using the traditional method still shows that this class is the minority within a period of time after the change. So, the wrong resampling technique is applied, which emphasizes this class even when it has become the majority. The time-decayed class size in OOB and UOB reflects the correct imbalance status quickly, and thus OOB and UOB adjust their focus on the correct class after the change.

In terms of negative-class recall, when this class turns from the majority to the minority, all the methods present a more or less reduced recall; OOB is the most resistant. One reason for the reduction is the performance trade-off between classes. As the positive class becomes the majority with a great performance improvement, the performance of the other class would be compromised to some extent. UOB suffers from a greater reduction than OOB right after the change, because undersampling makes the online model learn less knowledge about this class before the change. Even so, the recall from UOB recovers faster than OOB_{tr} and UOB_{tr}. When OOB_{tr} and UOB_{tr} still treat this class as the majority after the change, UOB has already realized its minority status and applied undersampling to the other class. The robustness of OOB and the adaptivity of UOB prove the benefits of using time-decayed class size and the importance of considering real-time status of data streams.

The other three types of changes present similar results. The positive-class recall presents a fast increase, and the negative-class recall is decreased, after the change occurs. Among the four methods, UOB is the best at recognizing the minority-class examples before the change; OOB is the most resistant method to changes; UOB is affected more by the changes than OOB on the old majority class, but its performance recovers faster than OOB_{tr} and UOB_{tr}. For the space consideration, the plots from Gaussian and Gearbox were omitted, from which similar results are obtained.

To understand how the overall performance is affected, we compare average G-mean over the new-status stage. Table 8 shows its means and standard deviations from 100 runs for the Smart Building data. P-values from the Wilcoxon Sign Rank test between OOB and every other method are included in brackets, among which the ones in bold italics suggest a significant difference. We can see that OOB_{tr} and UOB_{tr} are significantly worse than OOB and UOB, because of the incorrectly detected imbalance rate. Although the above results show that UOB suffers from a performance reduction on the old majority class, its G-mean still outperforms OOB in some cases, because of its fast performance recovery on this class through undersampling using the correctly estimated imbalance rate.

In conclusion, when there is an imbalance status change in the data stream, OOB is the least affected. A performance drop on the old majority class occurs to UOB after the change, but its performance recovers rapidly. It is necessary to use the time-decayed metric for choosing the correct resampling technique and estimating the imbalance status.

5.3 Comparison with Other Algorithms

This section discusses the performance of MLP-based OOB and UOB on the above dynamic data, in comparison with RLSACP and WOS-ELM. OOB containing a single MLP (OOB_{sg}) is included as the benchmark. Average G-mean over the new-status stage and the p-values produced from the Wilcoxon Sign Rank test between OOB and the others are shown in Table 9. "NaN" p-value in the table means that the two groups of samples for the statistical test are exactly the same.

The results show that MLP-based OOB and UOB are quite competitive. In Gaussian data, UOB shows worse G-mean than OOB, because the imbalance change causes larger recall degradation on the old majority class. In Gearbox and Smart Building data, OOB produces zero G-mean in 2 cases, which is caused by zero negative-class recall. In these two cases where the negative class turns from the majority to the minority gradually, because OOB is not so aggressive as UOB at recognizing minority-class examples, it could not adjust the learning bias to the new minority class quickly enough. That is not observed in tree-based OOB, as the decision tree was shown to be a more stable base classifier than MLP based on the results in Section 4. OOB and UOB outperform RLSACP and WOS-ELM in most cases. Although WOS-ELM applies the same time-decayed metric for updating misclassification costs as in OOB and UOB, its G-mean remains zero in many cases, caused by the zero recall of the old minority class. This class is not well learnt before the change, and its recall value remains low after the change because no emphasis is given to this class. Resampling is simply a better class imbalance strategy than adjusting class costs through the comparison with OOB_{sg}, which tallies with our observation in the previous section.

5.4 Factorial Analysis of the Decay Factor

Section 5.2 has shown the key role of using the time-decayed class size $w_k^{(t)}$ for deciding the resampling rate in OOB and UOB. According to our preliminary experiments [6], the decay factor θ in its definition formula (Eq. 1) is set to 0.9 in the above experiments. Too small values of θ (< 0.8) emphasize the current performance too much, which can affect the performance stability considerably. On the contrary, too large values (> 0.95) can deteriorate the adaptivity of online models to dynamic changes. Within a reasonable setting range, this section studies the impact of the decay factor on the performance of OOB and UOB statistically. One-way ANOVA using a repeated measure design is conducted.

TABLE 8: *Average G-mean* during the new-status stage and statistical test results from tree-based OOB, UOB, OOB_{tr} and UOB_{tr} on Smart Building. (Change type: H, high severity; L, low severity; A, abrupt; G, gradual)

Change		OOB	UOB	OOB_{tr}	UOB_{tr}
H	A	0.693±0.004	0.763±0.012 ***(0.00000)***	0.389±0.018 ***(0.00000)***	0.662±0.016 ***(0.00000)***
H	G	0.609±0.001	0.462±0.012 ***(0.00000)***	0.000±0.000 ***(0.00000)***	0.208±0.123 ***(0.00000)***
L	A	0.675±0.002	0.701±0.006 ***(0.00000)***	0.446±0.025 ***(0.00000)***	0.639±0.006 ***(0.00000)***
L	G	0.679±0.001	0.683±0.020 (0.29420)	0.000±0.000 ***(0.00000)***	0.511±0.022 ***(0.00000)***

TABLE 9: *Average G-mean* during the new-status stage and statistical test results from MLP-based OOB, OOB_{sg}, UOB, RLSACP and WOS-ELM. (Change type: H, high severity; L, low severity; A, abrupt; G, gradual)

Data	Change		OOB	UOB	RLSACP	WOS-ELM	OOB_{sg}
Gaussian	H	A	0.532±0.034	0.019±0.013 ***(0.00000)***	0.366±0.096 ***(0.00000)***	0.090±0.125 ***(0.00000)***	0.544±0.049 (0.07080)
	H	G	0.360±0.002	0.359±0.011 ***(0.00440)***	0.388±0.089 (0.04490)	0.037±0.095 ***(0.00000)***	0.350±0.010 ***(0.00000)***
	L	A	0.795±0.003	0.521±0.025 ***(0.00000)***	0.439±0.210 ***(0.00000)***	0.022±0.074 ***(0.00000)***	0.778±0.013 ***(0.00000)***
	L	G	0.811±0.001	0.758±0.015 ***(0.00000)***	0.613±0.118 ***(0.00000)***	0.015±0.056 ***(0.00000)***	0.813±0.010 ***(0.00160)***
Gearbox	H	A	0.293±0.009	0.201±0.032 ***(0.00000)***	0.438±0.012 ***(0.00000)***	0.000±0.000 ***(0.00000)***	0.261±0.113 (0.54110)
	H	G	0.000±0.000	0.364±0.119 ***(0.00000)***	0.140±0.126 ***(0.00000)***	0.000±0.000 (NaN)	0.161±0.177 ***(0.00000)***
	L	A	0.452±0.014	0.479±0.012 ***(0.00000)***	0.378±0.031 ***(0.00000)***	0.000±0.000 ***(0.00000)***	0.430±0.056 (0.01470)
	L	G	0.408±0.032	0.432±0.023 ***(0.00000)***	0.055±0.095 ***(0.00000)***	0.000±0.000 ***(0.00000)***	0.373±0.080 ***(0.00078)***
Smart Building	H	A	0.276±0.028	0.261±0.052 (0.04740)	0.135±0.156 ***(0.00000)***	0.123±0.008 ***(0.00000)***	0.229±0.131 (0.03530)
	H	G	0.000±0.000	0.140±0.091 (0.12090)	0.157±0.188 ***(0.00000)***	0.000±0.000 (NaN)	0.063±0.094 ***(0.00000)***
	L	A	0.451±0.019	0.435±0.016 ***(0.00000)***	0.228±0.144 ***(0.00000)***	0.000±0.000 ***(0.00000)***	0.441±0.044 (0.23750)
	L	G	0.443±0.023	0.485±0.018 ***(0.00000)***	0.080±0.078 ***(0.00000)***	0.000±0.000 ***(0.00000)***	0.374±0.112 ***(0.00000)***

We choose three levels (settings) for the factor θ – 0.8, 0.9 and 0.95. DT-based OOB and UOB are applied to the Smart Building data with an abrupt change in high severity, under each setting of θ. The response in this ANOVA is the average prequential G-mean during the new-status stage of class imbalance. It is worth nothing that the data groups under different factor levels violate the assumption of sphericity in ANOVA based on Mauchly's test, i.e. the level of dependence between pairs of groups is not equal. Therefore, Greenhouse-Geisser correction is used [33]. The ANOVA and performance results are shown in Table 10.

TABLE 10: ANOVA results and average G-mean from OOB and UOB after the change.

	ANOVA		Average G-mean		
	p-val	η^2	$\theta = 0.8$	$\theta = 0.9$	$\theta = 0.95$
OOB	0.000	0.989	0.726	0.693	0.643
UOB	0.018	0.077	0.763	0.763	0.760

According to the p-values and effect size η^2, we can see that the decay factor has a significant impact on both OOB and UOB (p-val < 0.05). The effect on OOB is much higher than the effect on UOB. For OOB, a relatively smaller θ seems to be beneficial to its performance, implying that adapting to the change sooner may help OOB to respond to the new minority class better. UOB is less sensitive to the setting of θ.

6 Ensembles of OOB and UOB

UOB has been shown to be a better choice than OOB in terms of minority-class recall and G-mean. However, it has some weaknesses when the majority class in the data stream turns into the minority class. OOB has been shown to be more robust against changes. To combine the strength of OOB and UOB, this section proposes new methods based on the idea of ensembles, which train and maintain both OOB and UOB. A weight is maintained for each of them, adjusted adaptively according to their current performance measured by G-mean. Their combined weighted vote will decide the final prediction.

Recalling the definition of time-decayed recall $R^{(t)}$ in Section 2.1, we use $R^{(t)}$ to calculate real-time G-mean for deciding the weights of online learners. It is adaptive to dynamic changes through the time decay factor, and reflects the current performance better than the prequential recall. However, we find that $R^{(t)}$ presents a large variance between time steps, which may result in misleading and unstable weights. So, we introduce the simple moving average (SMA) technique to smooth out the short-term fluctuations of $R^{(t)}$ curves. Given l values of $R^{(t)}$, smoothed recall $SR^{(t)}$ at time step t is defined as,

$$SR^{(t)} = \frac{R^{(t)} + \ldots + R^{(t-\lfloor l/2 \rfloor)} + \ldots + R^{(t-l)}}{l}.$$

$SR^{(t)}$ looks back l time steps for the smoother recall. Using $SR^{(t)}$, we next introduce the weighted ensemble methods of OOB and UOB, tested on the real-world data in comparison with individual OOB and UOB.

6.1 Weighted Ensemble of OOB and UOB

By calculating SMA of recall of both classes, we can obtain current G-mean. It is used to determine the weights of OOB and UOB. Their weighted ensemble, denoted as WEOB, is expected to be both accurate and robust in dynamic environments, as it adopts the better strategy (OOB or UOB) for different situations.

Two weight adjusting strategies are proposed and compared here. Suppose OOB has G-mean value g_o and UOB has G-mean value g_u at the current moment. Let α_o and α_u denote the weights of OOB and UOB respectively. In the first strategy, normalized G-mean values of OOB and UOB are used as their weights:

$$\alpha_o = \frac{g_o}{g_o + g_u}, \alpha_u = \frac{g_u}{g_o + g_u}.$$

In the second strategy, the weights can only be binary values (0 or 1):

$$\begin{cases} \alpha_o = 1, \alpha_u = 0 & \text{if } g_o \geq g_u \\ \alpha_o = 0, \alpha_u = 1 & \text{if } g_o < g_u \end{cases}$$

In other words, the final prediction will solely depend on the online model with the higher G-mean. Let's denote these two strategies as WEOB1 and WEOB2. WEOB1 follows the traditional idea of ensemble. From the statistical point of view, combining the outputs of several classifiers by averaging can reduce the risk of an unfortunate selection of a poorly performing classifiers and thus provide stable and accurate performance [34], although the averaging may or may not beat the performance of the best classifiers in the ensemble (used by WEOB2). Therefore, it is worth looking into both strategies.

6.2 Data Description and Experimental Settings

To compare WEOB1 and WEOB2 with OOB and UOB, we generate two data streams from real-world applications Gearbox and Smart Building. Each data stream contains 5000 time steps. At every 500 time steps, our data generation algorithm randomly chooses one class to be the minority class, and fixes the imbalance rate randomly chosen from set {5%, 10%, 20%}. By doing so, there could be an abrupt change in the class imbalance status after every 500 examples have arrived. The detailed information of the resulting data is summarized in Table 11, including which class belongs to the minority during which period of time.

TABLE 11: Data Stream Description.

Data	Class label	Time steps of being minority
Gearbox	Positive	1-500, 1501-2000, 3001-4500
	Negative	501-1500, 2001-3000, 4501-5000
Smart Building	Positive	1-500, 1501-2000, 3001-4000
	Negative	501-1500, 2001-3000, 4001-5000

WEOB1 and WEOB2 combine one OOB and one UOB composed of 50 base classifiers respectively. All methods use Hoeffding tree as the base classifier. l is set to 51 for calculating $SR^{(t)}$. All the other parameter settings remain the same. Prequential recall and G-mean are recorded at each time step. The performance metrics are reset to 0 after every 500 time steps for a clear observation.

6.3 Experimental Analysis

Prequential recall curves are shown in Fig. 2. Each plot compares WEOB1, WEOB2, OOB and UOB. We can see that, during the static periods of 500 time steps between the performance resettings, when OOB and UOB present a significant performance difference, WEOB1 and WEOB2 always locate in between OOB and UOB. When UOB suffers from a recall reduction caused by the class imbalance change, WEOB1 and WEOB2 are shown to be less affected by the change.

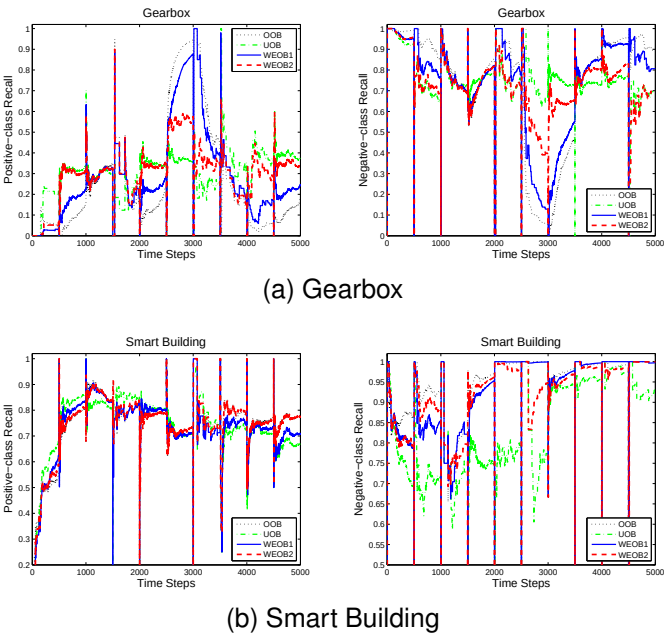

Fig. 2: Prequential recall of positive (left) and negative (right) classes.

With respect to the overall performance, Table 12 compares the average prequential G-mean during the 500 time steps after each status change, including its mean and standard deviation over 100 runs, and the p-values from Wilcoxon Sign Rank test performed between WEOB2 and every other method. The comparative results from this table tally with our results on recall. WEOB1 and WEOB2 are better than or locate in between single OOB and UOB in 9 out of 10 cases. On the Gearbox data, WEOB2 is competitive with UOB, and outperforms the other two in most cases significantly; on the Smart Building data, WEOB2 seems to be slightly worse than OOB, and better than the other two in most cases. Overall, WEOB2 outperforms WEOB1 significantly in 6 out of 10 cases, in terms of G-mean.

Generally speaking, WEOB1 and WEOB2 successfully combine the strength of OOB and UOB. They achieve high performance in terms of recall and G-mean, and show good robustness to changes in the class imbalance status. Particularly, WEOB2 shows better G-mean than WEOB1.

TABLE 12: *Average G-mean* from tree-based OOB, UOB, WEOB1 and WEOB2 during the 500 time steps after each status change, and statistical test results (in brackets). Values in ***bold italics*** indicate a statistically significant difference.

Data	Status change moments	OOB	UOB	WEOB1	WEOB2
Gearbox	500	0.237±0.045 ***(0.00000)***	0.466±0.012 (0.12160)	0.364±0.014 ***(0.00000)***	0.463±0.012
	1500	0.397±0.056 ***(0.00000)***	0.296±0.124 ***(0.00510)***	0.395±0.079 (0.00830)	0.404±0.072
	2000	0.329±0.015 ***(0.00000)***	0.494±0.024 ***(0.00002)***	0.424±0.018 ***(0.00000)***	0.485±0.024
	3000	0.371±0.022 ***(0.00000)***	0.412±0.015 ***(0.00000)***	0.437±0.015 ***(0.00000)***	0.458±0.025
	4500	0.309±0.039 ***(0.00000)***	0.480±0.010 ***(0.00000)***	0.424±0.023 ***(0.00000)***	0.443±0.024
Smart Building	500	0.841±0.011 ***(0.00000)***	0.770±0.008 ***(0.00000)***	0.821±0.016 ***(0.00200)***	0.831±0.015
	1500	0.867±0.025 (0.22690)	0.793±0.009 ***(0.00000)***	0.843±0.011 ***(0.00000)***	0.865±0.022
	2000	0.872±0.011 (0.01240)	0.779±0.016 ***(0.00000)***	0.886±0.003 ***(0.00000)***	0.868±0.015
	3000	0.865±0.009 (0.04100)	0.856±0.035 (0.67880)	0.854±0.002 (0.18780)	0.861±0.013
	4000	0.849±0.014 ***(0.00410)***	0.823±0.007 ***(0.00000)***	0.839±0.005 (0.17010)	0.842±0.017

7 Conclusions

As one of the earliest works focusing on online class imbalance learning, this paper improved and studied in depth two ensemble learning methods OOB and UOB using resampling and the time-decayed metric to overcome class imbalance online. Four major contributions have been made.

First, the original OOB and UOB were improved with a better resampling strategy, in which the sampling rate is consistent with the imbalance degree in the data stream. Second, they were analysed in a group of static data streams varying in data distributions and imbalance rates. Both oversampling in OOB and undersampling in UOB were shown to facilitate the recognition of minority-class examples with improved minority-class recall and G-mean. They also outperformed two recently proposed algorithms RLSACP [17] and WOS-ELM [18] for learning imbalanced data streams. To find out which fundamental factors affect the performance of OOB and UOB the most, three factors were investigated through factorial ANOVA, which are data distributions, imbalance rates and types of base classifiers. All of them had significant impacts on G-mean. The data distribution was found to be the most influential factor. Decision tree was a better base classifier than MLP. Among all discussed models, tree-based UOB showed the best minority-class recall and G-mean on the static data streams. Third, to examine the adaptivity and robustness, OOB and UOB were studied in dynamic data streams involving class imbalance changes with different speed and severity. The time-decayed class size played an important role in responding to the change correctly and timely. However, it was found that UOB suffers from a great performance reduction right after the change, while OOB is more robust against this situation. Finally, to combine the strength of OOB and UOB, we proposed two weighted ensemble methods – WEOB1 and WEOB2. Both showed better accuracy than OOB and better robustness than UOB. WEOB2 outperformed WEOB1 in terms of G-mean.

In the future, we would like to extend our work to multi-class cases. Second, we would like to study our methods in data streams with concept drifts. Third, the work presented in this paper relied heavily on computational experiments. More theoretical studies will be given.

Acknowledgments

This work was supported by two European Commission FP7 Grants (Nos. 270428 and 257906) and an EPSRC Grant (No. EP/J017515/1). Xin Yao was supported by a Royal Society Wolfson Research Merit Award.

References

[1] L. L. Minku, "Online ensemble learning in the presence of concept drift," Ph.D. dissertation, School of Computer Science, The University of Birmingham, 2010.

[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] N. C. Oza and S. Russell, "Experimental comparisons of online and batch versions of bagging and boosting," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 359–364.

[4] J. Meseguer, V. Puig, and T. Escobet, "Fault diagnosis using a timed discrete-event approach based on interval observers: Application to sewer networks," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 5, pp. 900–916, 2010.

[5] G. M. Weiss, "Mining with rarity: a unifying framework," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 7–19, 2004.

[6] S. Wang, L. L. Minku, and X. Yao, "A learning framework for online class imbalance learning," in *IEEE Symposium on Computational Intelligence and Ensemble Learning (CIEL)*, 2013, pp. 36–45.

[7] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429 – 449, 2002.

[8] N. C. Oza, "Online bagging and boosting," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 2340–2345, 2005.

[9] J. V. Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental perspectives on learning from imbalanced data," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 935–942.

[10] C. Elkan, "The foundations of cost-sensitive learning," in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI'01)*, 2001, pp. 973–978.

[11] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[12] S. Chen, H. He, K. Li, and S. Desai, "MuSeRA: Multiple selectively recursive approach towards imbalanced stream data mining," in *International Joint Conference on Neural Networks*, 2010, pp. 1–8.
[13] S. Chen and H. He, "Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach," *Evolving Systems*, vol. 2, no. 1, pp. 35–50, 2010.
[14] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2283 – 2301, 2013.
[15] H. M. Nguyen, E. W. Cooper, and K. Kamei, "Online learning from imbalanced data streams," in *International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2011, pp. 347–352.
[16] L. L. Minku and X. Yao, "DDD: A new ensemble approach for dealing with concept drift," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 4, pp. 619 –633, 2012.
[17] A. Ghazikhani, R. Monsefi, and H. S. Yazdi, "Recursive least square perceptron model for non-stationary and imbalanced data stream classification," *Evolving Systems*, vol. 4, no. 2, pp. 119–131, 2013.
[18] B. Mirza, Z. Lin, and K.-A. Toh, "Weighted online sequential extreme learning machine for class imbalance learning," *Neural Processing Letters*, vol. 38, no. 3, pp. 465–486, 2013.
[19] D. C. Montgomery, *Design and Analysis of Experiments*. Great Britain: John Wiley and Sons, 2004.
[20] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," in *ACM SIGKDD Explorations Newsletter*, vol. 6, 2004, pp. 40–49.
[21] R. C. Prati, G. E. Batista, and M. C. Monard, "Class imbalances versus class overlapping: An analysis of a learning system behavior," *Lecture Notes in Computer Science*, vol. 2972, pp. 312–321, 2004.
[22] V. Garcia, J. Sanchez, and R. Mollineda, "An empirical study of the behavior of classifiers on imbalanced and overlapped data sets," *Progress in Pattern Recognition, Image Analysis and Applications*, vol. 4756, pp. 397–406, 2007.
[23] K. Napierala and J. Stefanowski, "Identification of different types of minority class examples in imbalanced data," *Hybrid Artificial Intelligent Systems*, vol. 7209, pp. 139–150, 2012.
[24] "2009 PHM challenge competition data set," The Prognostics and Health Management Society (PHM Society). [Online]. Available: http://www.phmsociety.org/references/datasets
[25] M. P. Michaelides, V. Reppa, C. Panayiotou, and M. Polycarpou, "Contaminant event monitoring in intelligent buildings using a multi-zone formulation," in *8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SAFEPROCESS)*, vol. 8, 2012, pp. 492–497.
[26] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 97–106.
[27] MOA Massive online analysis: Real time analytics for data streams. [Online]. Available: http://moa.cms.waikato.ac.nz/
[28] S. Wang, L. L. Minku, D. Ghezzi, D. Caltabiano, P. Tino, and X. Yao, "Concept drift detection for online class imbalance learning," in *International Joint Conference on Neural Networks (IJCNN '13)*, 2013, pp. 1–8.
[29] S. Wang, L. L. Minku, and X. Yao, "Online class imbalance learning and its applications in fault detection," *International Journal of Computational Intelligence and Applications*, vol. 12, pp. 1 340 001(1–19), 2013.
[30] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in *Proc. 14th International Conference on Machine Learning*, 1997, pp. 179–186.
[31] S. Ding, H. Zhao, Y. Zhang, X. Xu, and R. Nie, "Extreme learning machine: algorithm, theory and applications," *Artificial Intelligence Review*, p. (In Print), 2013.
[32] C. A. Pierce, R. A. Block, and H. Aguinis, "Cautionary note on reporting eta-squared values from multifactor anova designs," *Educational and Psychological Measurement*, vol. 64, pp. 916–924, 2004.
[33] S. W. Greenhouse and S. Geisser, "On methods in the analysis of profile data," *Psychometrika*, vol. 24, no. 2, pp. 95–112, 1959.
[34] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–45, 2006.

Shuo Wang is a 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 (UK). She 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). Dr. Wang's research interests include class imbalance learning, ensemble learning, online learning and machine learning in software engineering. Her work has been published in internationally renowned journals and conferences.

Leandro L. Minku is a Research Fellow II at the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science, the University of Birmingham (UK). He received the BSc, MSc and PhD degrees in Computer Science from the Federal University of Parana (Brazil) in 2003, the Federal University of Pernambuco (Brazil) in 2006, and the University of Birmingham (UK) in 2010, respectively. He was an intern at Google Zurich for six months in 2009/2010, the recipient of the Overseas Research Students Award (ORSAS) from the British government, and of several scholarships from the Brazilian Council for Scientific and Technological Development (CNPq). Dr. Minku's main research interests include machine learning in changing environments, ensembles of learning machines, machine learning for software engineering and search-based software engineering. His work has been published in internationally renowned journals such as IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Software Engineering and ACM Transactions on Software Engineering and Methodology.

Xin Yao is a Chair (Professor) of Computer Science and the Director of CERCIA (the Centre of Excellence for Research in Computational Intelligence and Applications) at the University of Birmingham, UK. He is an IEEE Fellow and the President (2014-15) of IEEE Computational Intelligence Society (CIS). His work won the 2001 IEEE Donald G. Fink Prize Paper Award, 2010 IEEE Transactions on Evolutionary Computation Outstanding Paper Award, 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), 2011 IEEE Transactions on Neural Networks Outstanding Paper Award, and many other best paper awards. He won the prestigious Royal Society Wolfson Research Merit Award in 2012 and the IEEE CIS Evolutionary Computation Pioneer Award 2013. He was the Editor-in-Chief (2003-08) of IEEE Transactions on Evolutionary Computation. His major research interests include evolutionary computation and ensemble learning. He published 200+ refereed international journal papers.