# Dealing with Multiple Classes in Online Class Imbalance Learning

**Shuo Wang**
University of Birmingham, UK
s.wang@cs.bham.ac.uk

**Leandro L. Minku**
University of Leicester, UK
leandro.minku@leicester.ac.uk

**Xin Yao**
University of Birmingham, UK
x.yao@cs.bham.ac.uk

## Abstract

Online class imbalance learning deals with data streams having very skewed class distributions in a timely fashion. Although a few methods have been proposed to handle such problems, most of them focus on two-class cases. Multi-class imbalance imposes additional challenges in learning. This paper studies the combined challenges posed by multi-class imbalance and online learning, and aims at a more effective and adaptive solution. First, we introduce two resampling-based ensemble methods, called MOOB and MUOB, which can process multi-class data directly and strictly online with an adaptive sampling rate. Then, we look into the impact of multi-minority and multi-majority cases on MOOB and MUOB in comparison to other methods under stationary and dynamic scenarios. Both multi-minority and multi-majority make a negative impact. MOOB shows the best and most stable G-mean in most stationary and dynamic cases.

## 1 Introduction

In many real-world classification tasks, such as spam filtering [Nishida *et al.*, 2008] and fault diagnosis in computer monitoring systems [Meseguer *et al.*, 2010], data often arrive over time in streams of instances. Several online learning techniques have thus been developed to process each data example once "on arrival" without storage and reprocessing. They maintain a current model that reflects the current data concept to make a prediction at each time step. These data stream applications are also class imbalanced, i.e. some classes of data are heavily under-represented compared to other classes, e.g. the spam class and the class of faults. This skewed class distribution can cause great learning difficulties, because the traditional machine learners tend to ignore or overfit the minority class [Weiss, 2004]. With the aim of tackling the combined issue of online learning and class imbalance learning, online class imbalance learning has been drawing growing attention.

Although a few methods have been proposed to deal with class imbalance online, they assume that the data have only two classes – one minority class and one majority class. This assumption does not apply in many real-world problems. For example, in fault detection of a real-time running engineering system (discriminating fault and non-fault classes), it is likely that more than one type of faults exists and needs to be recognised. Multi-class tasks have been shown to suffer more learning difficulties than two-class ones in offline learning, because multi-class increases the data complexity and aggravates the imbalanced distribution [Wang and Yao, 2012]. We expect this difficulty to become even more aggravated in online learning scenarios, given that it is impossible to see the whole picture of data and the data may be dynamically changing.

Very little work has addressed the multi-class issue in online class imbalance learning. It is still unclear what learning difficulties multi-class can cause and how to handle it effectively. Therefore, a systematic study of multiple classes is necessary in order to gain a better understanding of its impact in online class imbalance learning, and a more effective and adaptive method suitable for both stationary and dynamic cases needs to be developed. This paper will focus on the following **research questions**: Q1. Can we develop a method able to process multi-class directly and overcome class imbalance adaptively? Q2. What is the impact of multi-class imbalance in online learning with *stationary* imbalance status? Q3. What is the impact of multi-class imbalance in online learning with *dynamic* imbalance status? It is worth mentioning that explicit concept drifts involving any changes in class-conditional probability density functions are not considered in this paper, for a clear observation of the impact of the number of classes.

For Q1 (Section 3), we propose two ensemble learning methods, MOOB and MUOB, which can process multi-class directly without using class decomposition and have an adaptive resampling technique to deal with class imbalance. For Q2 (Section 4), we fix the imbalance ratio between minority and majority classes and vary the number of minority and majority classes by generating several artificial data streams. We analyse how MOOB and MUOB are affected in comparison to other existing methods. For Q3 (Section 5), the imbalance ratio between minority and majority classes is changing gradually. Class emergence and class disappearance are considered as two major class evolution types [Sun *et al.*, 2016]. Both artificial and real-world data are discussed. In general, MOOB shows the best and most stable performance in both stationary and dynamic cases.

## 2 Background

As the basis of this paper, the research progress in online two-class imbalance learning and offline multi-class imbalance learning is reviewed in this section. Then, VWOS-ELM and CBCE are introduced. They are the only methods capable of learning multi-class imbalanced data streams so far. Research gaps that motivate our study are identified.

### 2.1 Online Two-Class Imbalance Learning

Different from incremental learning methods that store and process data in batches [Hoens *et al.*, 2012a] [Hoens and Chawla, 2012], online learning methods learn from data one by one without any pre-knowledge. Several online methods have been proposed to tackle class imbalance. For example, [Nguyen *et al.*, 2011] proposed a Naive Bayes ensemble method based on random undersampling. [Wang *et al.*, 2013] and [Wang *et al.*, 2015] proposed OOB and UOB, which can tackle imbalanced data with a dynamically changing imbalance rate. However, they cannot balance multi-class data effectively and adaptively. Although the original OOB and UOB targeted general online cases, their sampling rates were not set consistently when the number of classes changes. Cost-sensitive methods, such as cost-sensitive Bagging and Boosting [Wang and Pineau, 2013], RLSACP [Ghazikhani *et al.*, 2013], WOS-ELM [Mirza *et al.*, 2013] and ESOS-ELM [Mirza *et al.*, 2015b], set a different misclassification cost for each class. However, the costs are pre-defined, and their cost setting strategies are only designed for two-class cases.

### 2.2 Offline Multi-Class Imbalance Learning

In offline class imbalance learning, most work resorts to class decomposition schemes (e.g. One-against-all (OAA), one-against-one (OAO) and ECOC) to decompose multi-class into several two-class problems. For example, MC-HDDT [Hoens *et al.*, 2012b] is a decision tree method based on OAA and ECOC. The imECOC method [Liu *et al.*, 2013] extended the original ECOC for imbalanced data based on a BWC-weighting method. Even though class decomposition simplifies multi-class problems, it causes new issues, such as combining binary classifiers. In the online learning context, maintaining and combining multiple binary classifiers is more difficult, because the number of classes may change, and some classifiers can get outdated along with time. Furthermore, each binary classifier is trained without full data knowledge. This can cause classification ambiguity or uncovered data regions with respect to each type of decomposition [Jin and Zhang, 2007]. Therefore, using class decomposition schemes is not a desirable way to learn multi-class data online.

Different from the above, a few approaches were proposed recently to tackle multi-class imbalance directly. EDS [Cao *et al.*, 2013] used evolutionary search techniques to find the optimum cost setup of each class, which was then integrated into a multi-class cost-sensitive ensemble classifier. AdaBoost.NC [Wang and Yao, 2012] and CoMBo [Koço and Capponi, 2013] are two Boosting methods for learning multi-class imbalanced data directly. However, their core techniques are not applicable to online cases.

### 2.3 Online Multi-Class Imbalance Learning

VWOS-ELM was proposed very recently, aiming at class imbalance problems in multi-class data streams [Mirza *et al.*, 2015a]. It is an ensemble method formed by multiple WOS-ELM base classifiers [Mirza *et al.*, 2013]. WOS-ELM is a perceptron-based extreme learning machine. It requires a data set for initialisation before the sequential learning starts. Different class weights are maintained to tackle class imbalance, based on models' performance on a validation data set. However, once the validation data set does not reflect the real status of data, the class weights will not be accurate for learning. Besides, initialisation data is not always available.

CBCE is a class-based ensemble method focusing on class evolution [Sun *et al.*, 2016]. One-against-all class decomposition technique is used for handling multiple classes. Undersampling is applied to overcome class imbalance induced by the class evolution. Although CBCE is shown to handle dynamic classes well, class decomposition is not an ideal way of processing multi-class imbalanced problems.

In summary, none existing methods can handle multi-class imbalance well in online learning. Meanwhile, there is no study looking into the radical learning issues caused by multi-class.

## 3 Resampling-based Ensemble Methods

This section proposes two resampling-based ensemble methods, Multi-class Oversampling-based Online Bagging (MOOB) and Multi-class Undersampling-based Online Bagging (MUOB). As suggested by their names, they use oversampling or undersampling to overcome class imbalance, with the framework of Online Bagging (OB) [Oza, 2005]. Resampling is algorithm-independent, so that any type of base classifiers forming the ensemble is allowed. To be able to process multi-class data directly, for example, Hoeffding trees or neural networks can be used. Besides, resampling is one of the simplest and most effective imbalance techniques in both offline and online class imbalance learning [Hulse *et al.*, 2007] [Wang *et al.*, 2015]. In order to tackle class imbalance through resampling under both stationary and dynamic scenarios, a time-decayed class size (i.e. class prior probability) [Wang *et al.*, 2013] is adopted. It is a real-time indicator, reflecting the current class imbalance status. It is used to decide the sampling rate adaptively in MOOB and MUOB.

For a sequence of examples $(x_t, y_t)$ arriving one at a time, $x_t$ is an input vector belonging to the input space $X$ observed at time step $t$, and $y_t$ is the corresponding label belonging to the label set $Y = \{c_1, \ldots, c_N\}$ (N>2). For any $c_k \in Y$, the time-decayed class size $w_k^{(t)}$ indicates the occurrence probability of examples belonging to $c_k$. To reflect the current imbalance status of the data stream, $w_k^{(t)}$ is incrementally updated at each time step, by using a time decay (forgetting) factor that weakens the effect of old data. When a new example $x_t$ arrives, $w_k^{(t)}$ is updated by [Wang *et al.*, 2013]:

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

where $[(x_t, c_k)] = 1$ if the true class label of $x_t$ is $c_k$, otherwise 0. $\theta$ $(0 < \theta < 1)$ is the pre-defined time decay factor. It

forces older data to have less impact on the class percentage, so that $w_k^{(t)}$ is adjusted more based on new data. $\theta = 0.9$ was shown to be a reasonable setting to balance the responding speed and estimation variance [Wang *et al.*, 2013].

With the class imbalance information, MOOB and MUOB integrate resampling into OB. OB is an online version of the offline Bagging [Breiman, 1996], designed for balanced data. 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. In MOOB, oversampling is used to increase the chance of learning minority-class examples through $\lambda$ based on current $w_k^{(t)}$. Similarly, in MUOB, undersampling is used to reduce the chance of learning majority-class examples. At each time step $t$, their training procedures are given in Table 1.

Table 1: MOOB and MUOB Training Procedures.

**Input:** an ensemble with $M$ classifier, current training examples $(x_t, y_t)$ where $y_t$ corresponds to $c_j$ in $Y$, and current class size $w^{(t)} = \left(w_1^{(t)}, \ldots, w_k^{(t)}, \ldots, w_N^{(t)}\right)$.
$w_{min} = \min_{k=1}^{N} w_k^{(t)}$
$w_{max} = \max_{k=1}^{N} w_k^{(t)}$
**for** each base learner $f_m$ ($m = 1, 2, \ldots, M$) **do**
  **if using MOOB:** set $K \sim Poisson\left(w_{max}/w_j^{(t)}\right)$
  **if using MUOB:** set $K \sim Poisson\left(w_{min}/w_j^{(t)}\right)$
  update $f_m$ $K$ times using $(x_t, y_t)$
**end for**

The minimum and maximum class sizes ($w_{min}$ and $w_{max}$) among all classes are calculated at each time step. MOOB sets $\lambda$ to $w_{max}/w_j^{(t)}$, so that the smaller class has a larger sampling rate, and vice versa for MUOB. When the data stream is strictly balanced, MOOB and MUOB will be reduced to the traditional OB, where $\lambda = 1$.

The advantages of MOOB and MUOB are: 1) they contain mechanisms to deal with dynamic imbalance status, so that no validation data set is needed for updating class weights. 2) Resampling allows the ensemble learner to use any types of base classifiers. Neither a data set for initialising classifiers nor class decomposition for handling multi-class is required.

In the following sections, the prequential performance of MOOB and MUOB will be examined, in comparison with VWOS-ELM and OB. VWOS-ELM is chosen as the only online learning algorithm aiming to solve multi-class imbalance directly so far. OB without any class imbalance techniques is included, serving as the baseline method.

The prequential test used in our experiment is a popular performance evaluation strategy in online learning, in which each individual example tests the model before it is used for training, and from this the performance measures can be incrementally updated and recorded at each time step [Minku, 2010]. In our test, we record recall of each class and G-mean as the evaluation metrics. They are two most commonly used criteria in class imbalance learning, as they are insensitive to the imbalance rate. Recall is defined as the classification accuracy on a single class. It helps us to analyse the performance on one class. G-mean is an overall performance metric, defined as the geometric mean of recalls over all classes [Kubat and Matwin, 1997]. It helps us to understand how well the performance is balanced among classes. It is worth noting that F-score and AUC are also popular, but they are not suitable for multi-class analysis.

# 4 Multi-Class Learning from Stationary Data

In this section, we give an in-depth analysis of multi-class imbalance in stationary data streams. With a fixed imbalance ratio between minority and majority classes, the impact of multi-class will be easily observed. Two basic types of multi-class can occur: one majority and multiple minority classes (multi-minority); one minority and multiple majority classes (multi-majority) [Wang and Yao, 2012]. We look into each type by producing artificial data sets with a different number of minority/majority classes, aiming to find out the learning difficulties in each type and the differences between them.

## 4.1 Multi-Minority Data

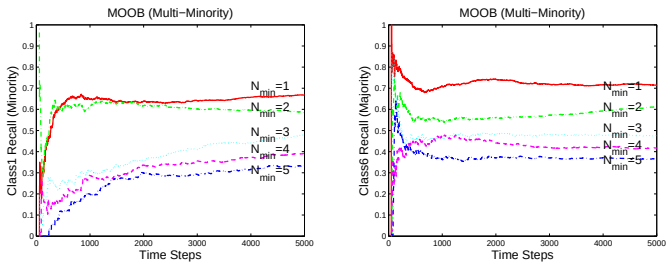
For multi-minority cases, we generate data streams with 5000 time steps. Each data example has 2 numeric attributes. The number of minority classes is varied from 1 to 5 (denoted by $c1, \ldots, c5$), and there is only one majority class (denoted by c6). The imbalance ratio between minority and majority classes remains 3:7. Data points in each class are generated randomly from Gaussian distributions, where the mean and standard deviation of each class are random real values in [0,5]. Among different data streams, the examples with the same class label follow the same Gaussian distribution. Table 2 summarizes the class number (N) and class size (P) (i.e. prior probability) of the generated multi-minority data.

Table 2: Class Settings for Multi-Minority Data.

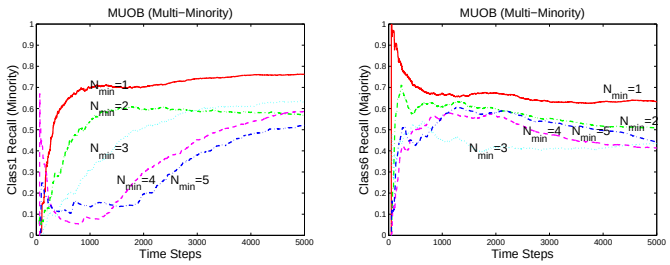
| ID | $N_{min}$ | $N_{maj}$ | $P_{min}$ | $P_{maj}$ |
|---|---|---|---|---|
| Bi | 1 (c1) | 1 (c6) | 3/10 | 7/10 |
| Min2 | 2 (c1,c2) | 1 (c6) | 3/13 | 7/13 |
| Min3 | 3 (c1-c3) | 1 (c6) | 3/16 | 7/16 |
| Min4 | 4 (c1-c4) | 1 (c6) | 3/19 | 7/19 |
| Min5 | 5 (c1-c5) | 1 (c6) | 3/22 | 7/22 |

The prequential recall curves of c1 (minority) and c6 (majority) from MOOB, MUOB, OB and VWOS-ELM are compared in Fig. 1. Every method is run 100 times independently. Because all the compared methods have an ensemble learning framework, we set the number of base classifiers to 11. Choosing an odd number is to avoid an even majority vote from base classifiers. Considering that VWOS-ELM is a perceptron-based method, we use multilayer perceptron (MLP) as the base classifier in MOOB, MUOB and OB for a fair comparison. Further discussion of using other base classifiers can be found in [Wang *et al.*, 2015] for 2-class cases. VWOS-ELM requires a data set to initialise each base classifier and a validation data set to update class weights. According to the authors, the initialisation set needs to include

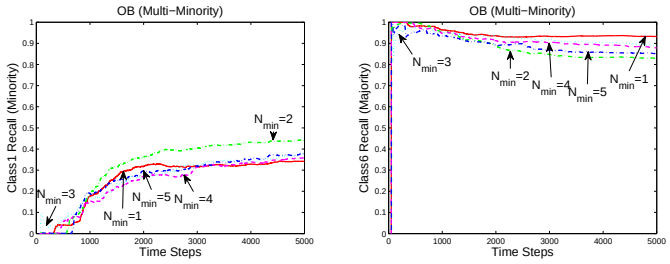
examples from all classes. To meet this requirement, we use the first 1% of data (i.e. 50 examples) as the initialisation and validation data. We start tracking the prequential performance from time step 51 for all the methods.



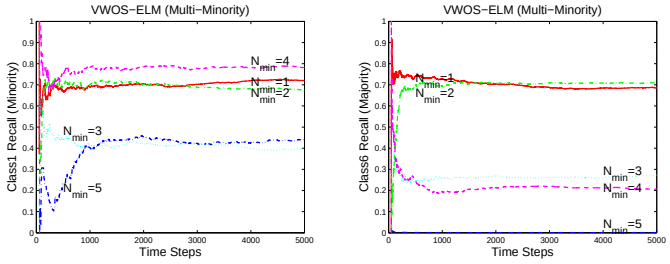

(a) MOOB




(b) MUOB




(c) OB




(d) VWOS-ELM

Figure 1: Prequential recall curves of classes c1 (minority) and c6 (majority) in multi-minority cases.

For MOOB and MUOB, it is clear to observe that increasing minority-class number reduces minority-class recall and majority-class recall. Multi-minority delays and worsens the recognition of minority-class examples, as well as majority-class examples. At the end of learning, the online learner shows worse recall of both types of classes in data with more minority classes. Because undersampling is a more aggressive method of emphasizing the minority class than oversampling, the minority-class recall recovers better in MUOB than in MOOB as more examples arrive.

The observation on VWOS-ELM is different. The negative impact of multi-minority is shown to be higher on the majority class than on the minority class. When there are five minority classes in the data stream, the majority-class recall becomes nearly zero. The likely reason is that the base learner WOS-ELM in VWOS-ELM tends to over-emphasize the minority class sometimes especially when the minority-class size is small [Wang *et al.*, 2015]. So, when there are multiple minority classes, the performance on the only majority class in the data stream can be sacrificed greatly.

Final G-mean (i.e. G-mean at the last time step) of all the methods is shown in Table 3. The significantly best value is shown in boldface, based on the Wilcoxon Sign Rank test. We can see that when the number of minority class is small ($N_{min}$ = 1,2,3), OB performs the worst; VWOS-ELM shows very good G-mean, because it is very aggressive at boosting minority-class recall. When $N_{min}$ becomes 5, MOOB shows the best G-mean. VWOS-ELM becomes the worst, caused by too much performance degradation of the majority class as shown in Fig. 1. In all the multi-minority cases, MOOB is better than or at least comparable to MUOB in G-mean. Although MUOB is shown to provide higher minority-class recall than MOOB in the above analysis, its majority-class recall is compromised more than that of MOOB. Therefore, MOOB provides the most stable performance among all.

## 4.2 Multi-Majority Data

For multi-majority cases, we use the same data generation settings as in the previous section, producing four multi-majority data streams. Table 4 summarizes their class number and class size.

Table 4: Class Settings for Multi-Majority Data.

| ID | $N_{min}$ | $N_{maj}$ | $P_{min}$ | $P_{maj}$ |
|---|---|---|---|---|
| Maj2 | 1 (c1) | 2 (c5,c6) | 3/17 | 7/17 |
| Maj3 | 1 (c1) | 3 (c4-c6) | 3/24 | 7/24 |
| Maj4 | 1 (c1) | 4 (c3-c6) | 3/31 | 7/31 |
| Maj5 | 1 (c1) | 5 (c2-c6) | 3/38 | 7/38 |

We observe the prequential recall of c1 (minority) and c6 (majority) in the multi-majority data streams. We obtain very similar observations on the impact of multi-majority to those in multi-minority cases (Fig. 1), so the recall curves in the multi-majority cases are omitted here for the space reason. Comparing the same method in multi-minority and multi-majority cases, their final recall is very similar. Multi-majority is not shown to be more difficult than multi-minority. This is an interesting result. In offline learning, multi-majority is a much more difficult case than multi-minority, because the minority class is overwhelmed by the large quantity of new majority-class examples [Wang and Yao, 2012]. It is not found in online cases. The reason may lie in other factors, e.g. the data distribution and imbalance ratio, which will be investigated in our next work.

Table 3: Mean and standard deviation of final G-mean in multi-minority cases.

| | Bi | Min2 | Min3 | Min4 | Min5 |
|---|---|---|---|---|---|
| MOOB | 0.690±0.003 | **0.625±0.013** | 0.402±0.010 | **0.302±0.005** | **0.271±0.007** |
| MUOB | 0.694±0.000 | 0.564±0.007 | 0.364±0.016 | 0.214±0.024 | 0.179±0.023 |
| VWOS-ELM | **0.700±0.008** | 0.492±0.163 | **0.419±0.038** | 0.129±0.037 | 0.026±0.038 |
| OB | 0.563±0.007 | 0.468±0.029 | 0.287±0.017 | 0.145±0.011 | 0.143±0.008 |

Table 5: Mean and standard deviation of final G-mean in multi-majority cases.

| | Bi | Maj2 | Maj3 | Maj4 | Maj5 |
|---|---|---|---|---|---|
| MOOB | 0.690±0.003 | 0.558±0.009 | **0.420±0.004** | **0.326±0.008** | **0.277±0.005** |
| MUOB | 0.694±0.000 | **0.564±0.005** | 0.330±0.019 | 0.265±0.018 | 0.201±0.015 |
| VWOS-ELM | **0.700±0.008** | 0.174±0.024 | 0.068±0.076 | 0.027±0.056 | 0.126±0.032 |
| OB | 0.563±0.007 | 0.512±0.032 | 0.315±0.033 | 0.111±0.081 | 0.038±0.063 |

These results are further reflected in final G-mean, shown in Table 5. Comparing the same method in Table 5 and Table 3, G-mean in the multi-majority cases is not worse than G-mean in the multi-minority cases, except for VWOS-ELM. MOOB shows the significantly best G-mean especially in cases with more majority classes. The fact that MUOB is worse than MOOB is interesting, as oversampling does not improve the classification performance as much as undersampling and suffers from overfitting in offline learning [Wang and Yao, 2012]. On one hand, oversampling seems to strengthen the performance stability, and is less likely to cause overfitting in online learning. On the other hand, discarding examples by undersampling is more likely to cause insufficient learning in online cases.

## 5 Multi-Class Learning from Dynamic Data

In most online learning applications, the imbalance status does not stay static. It is more likely that the change occurs in a gradual manner in a real-world scenario. For example, in a faulty gearbox system, as the part is wearing out eventually and the faulty condition gets worse, the faulty data will appear more and more frequently. Therefore, this section focuses on dynamic data streams with a gradual class imbalance change. Both artificial data and real-world data are discussed.

### 5.1 Artificial Data

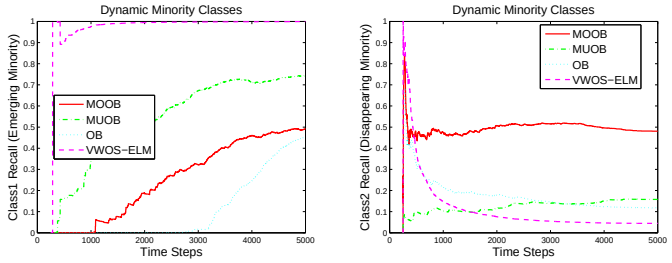
Based on the class evolution type categorized in [Sun *et al.*, 2016], we consider an emerging class and a disappearing class in both minority and majority types of classes. Concretely speaking, we generate three data streams with 5000 time steps, each of which has two minority classes (c1 and c2) and two majority classes (c3 and c4). How the class imbalance status changes in the artificial data is described in Table 6. The class imbalance change starts at time step 1 and ends at time step 5000. The first 5% data are used for initialisation and validation in VWOS-ELM. All the settings for the four learning methods remain the same as in Section 4.

Fig. 2 shows the recall of the two dynamic minority classes in DyMin and the two dynamic majority classes in DyMaj.

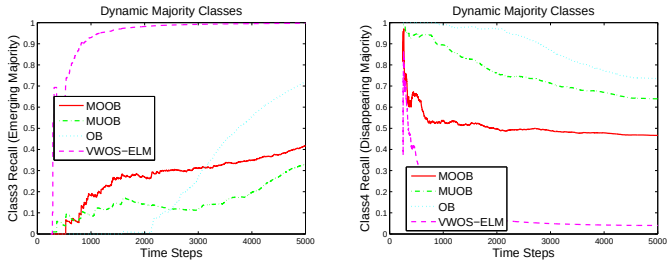
Table 6: Class Settings for Dynamic Data.

| ID | $P_{c1}$ | $P_{c2}$ | $P_{c3}$ | $P_{c4}$ |
|---|---|---|---|---|
| DyMin | $0 \rightarrow 0.3$ | $0.3 \rightarrow 0$ | 0.35 | 0.35 |
| DyMaj | 0.15 | 0.15 | $0 \rightarrow 0.7$ | $0.7 \rightarrow 0$ |
| DyAll | $0 \rightarrow 0.3$ | $0.3 \rightarrow 0$ | $0 \rightarrow 0.7$ | $0.7 \rightarrow 0$ |

The curve tendency in DyMin is quite similar to the one in DyMaj.




(a) Recall of c1 anc c2 in DyMin




(b) Recall of c3 anc c4 in DyMaj

Figure 2: Prequential recall curves of dynamic classes in DyMin and DyMaj.

For the emerging class (c1 in DyMin and c3 in DyMaj), all the methods have a growing recall, as the examples from this class arrive more and more frequently. MUOB has a better c1 recall and a worse c3 recall than MOOB, because c1

remains to be the minority in DyMin, and c3 becomes majority gradually in DyMaj, in which MUOB adjusts its focus on the minority classes eventually. For the disappearing class (c2 in DyMin and c4 in DyMaj), VWOS-ELM shows a significant drop. It is because this class is over-emphasized at the beginning with a very high recall, and the class weights are updated based on a separate validation set that does not reflect the current imbalance status. So, it causes less focus on this class later on, even though it has become a minority class. The other methods show more stable performance on the disappearing class.

After the analysis within each class, we now compare the overall performance G-mean in all the three dynamic data streams, shown in Table 7. MOOB performs the best, regardless of which class is dynamically changing. For MOOB and MUOB, their G-mean is quite similar in DyMin and DyMaj; for VWOS-ELM and OB, however, their G-mean in DyMaj is much worse than in DyMin. This is because the sampling rate in MOOB and MUOB is adaptive to the current imbalance status, whereas VWOS-ELM and OB do not have such mechanism. Besides, DyMaj involves a larger change of class imbalance than DyMin. Based on the above analysis, we can see the benefit of using an adaptive sampling in dynamic data streams in MOOB and MUOB. Among all, MOOB has the best performance.

Table 7: Mean and standard deviation of final G-mean in dynamic artificial data.

| | DyMin | DyMaj | DyAll |
|---|---|---|---|
| MOOB | **0.452±0.008** | **0.463±0.007** | **0.526±0.007** |
| MUOB | 0.272±0.032 | 0.289±0.026 | 0.349±0.043 |
| VWOS-ELM | 0.185±0.023 | 0.066±0.012 | 0.135±0.009 |
| OB | 0.336±0.012 | 0.049±0.055 | 0.304±0.013 |

### 5.2 Real-World Data

We now compare the four methods in two real-world data applications: online chess game [Žliobaitė, 2011] and UDI TweeterCrawl data [Li *et al.*, 2012]. The Chess data consist of online game records of one player from 2007 to 2010. The task is to predict if the player will win, lose or draw (3 classes). The original Tweet data include 50 million tweets posted mainly from 2008 to 2011. The task is to predict the tweet topic. In our experiment, we choose a time interval, containing 8774 examples and covering 7 tweet topics. Because real-world data hardly remain static, they both contain some gradual changes in class imbalance status. The final G-mean is shown in Table 8.

VWOS-ELM performs very well, giving the best G-mean in both data sets. By looking into its recall in each class, its majority-class performance (class2 in Chess and Class4 in Tweet) is sacrificed, but not as much as the improvement on the minority classes. Therefore, its overall performance is high. However, we notice that its good performance relies heavily on the choice of the initialisation data set. MOOB comes to the second with some improvement on the minority-class recall compared to OB and quite high

Table 8: Mean and standard deviation of final G-mean in Chess and Tweet.

| | Chess | Tweet |
|---|---|---|
| MOOB | 0.314±0.019 | 0.346±0.003 |
| MUOB | 0.237±0.120 | 0.002±0.010 |
| VWOS-ELM | **0.321±0.030** | **0.372±0.007** |
| OB | 0.000±0.000 | 0.000±0.000 |

majority-class recall. For MUOB, its majority-call recall is sacrificed too much in Chess and its minority-class recall is not good enough in Tweet, so its performance is not satisfactory. Overall, VWOS-ELM and MOOB are better choice for these real-world data, where VWOS-ELM is more aggressive at finding minority-class examples and MOOB is better at balancing the performance among classes.

## 6 Conclusions

This paper investigates the multi-class problem in online class imbalance learning. We study three research issues: Q1. effective and adaptive multi-class learning methods for online data; Q2. multi-class in stationary data streams; Q3. multi-class in dynamic data streams.

For Q1, we propose two ensemble learning methods – MOOB and MUOB. To the best of our knowledge, they are the first methods that can simultaneously process multi-class imbalance directly without using class decomposition, and handle class imbalance adaptively and strictly online without using any initialisation and validation data sets. For Q2, we investigate the online performance of MOOB and MUOB by varying the number of minority and majority classes respectively and fixing the class ratio between them, in comparison to VWOS-ELM and OB. All methods are negatively affected by both multi-minority and multi-majority, especially the minority-class recall of MOOB and the majority-class recall of VWOS-ELM; MOOB shows the best and most stable G-mean. For Q3, we generate multi-class data with gradually emerging and disappearing classes. For more convincing results, we also select two real-world data sets that are collected over time. MOOB is the best at G-mean in most cases; VWOS-ELM shows the most aggressive performance at boosting minority-class performance, which benefits the real-world data classification, but it suffers performance reduction on the disappearing class in the artificial data. We also show the benefit of using the adaptive sampling rate in MOOB and MUOB.

In the near future, we would like to study more types of multi-class data streams, e.g. with more severe imbalance status or different data distributions. It is also important to consider concept drifts in multi-class imbalanced data streams.

## Acknowledgments

This work was supported by EPSRC (Grant No. EP/K001523/1). Xin Yao was also supported by a Royal Society Wolfson Research Merit Award.

## References


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

[Cao *et al.*, 2013] P. Cao, B. Li, D. Zhao, and O. Zaiane. A novel cost sensitive neural network ensemble for multi-class imbalance data learning. In *The 2013 International Joint Conference on Neural Networks*, pages 1–8, 2013.

[Ghazikhani *et al.*, 2013] A. Ghazikhani, R. Monsefi, and H. S. Yazdi. Recursive least square perceptron model for non-stationary and imbalanced data stream classification. *Evolving Systems*, 4(2):119–131, 2013.

[Hoens and Chawla, 2012] T. Ryan Hoens and N. V. Chawla. Learning in non-stationary environments with class imbalance. In *18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 168–176, 2012.

[Hoens *et al.*, 2012a] T. R. Hoens, R. Polikar, and N. V. Chawla. Learning from streaming data with concept drift and imbalance: an overview. *Progress in Artificial Intelligence*, 1(1):89–101, 2012.

[Hoens *et al.*, 2012b] T.R. Hoens, Q. Qian, N.V. Chawla, and Z. Zhou. Building decision trees for the multi-class imbalance problem. In *Proceedings of the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 122–134, 2012.

[Hulse *et al.*, 2007] 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*, pages 935–942, 2007.

[Jin and Zhang, 2007] R. Jin and J. Zhang. Multi-class learning by smoothed boosting. *Machine Learning*, 67(3):207–227, 2007.

[Koço and Capponi, 2013] S. Koço and C. Capponi. On multi-class classification through the minimization of the confusion matrix norm. In *JMLR: Workshop and Conference Proceedings 29*, pages 277–292, 2013.

[Kubat and Matwin, 1997] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proc. 14th International Conference on Machine Learning*, pages 179–186, 1997.

[Li *et al.*, 2012] R. Li, S. Wang, H. Deng, R. Wang, and K. Chang. Towards social user profiling: unified and discriminative influence model for inferring home locations. In *Proceedings of the 18th ACM SIGKDD*, pages 1023–1031, 2012.

[Liu *et al.*, 2013] X. Liu, Q. Li, and Z. Zhou. Learning imbalanced multi-class data with optimal dichotomy weights. In *2013 IEEE 13th International Conference on Data Mining (ICDM)*, pages 478 – 487, 2013.

[Meseguer *et al.*, 2010] 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*, 40(5):900–916, 2010.

[Minku, 2010] L. L. Minku. *Online Ensemble Learning in the Presence of Concept Drift*. PhD thesis, School of Computer Science, The University of Birmingham, 2010.

[Mirza *et al.*, 2013] B. Mirza, Z. Lin, and K. Toh. Weighted online sequential extreme learning machine for class imbalance learning. *Neural Processing Letters*, 38:465–486, 2013.

[Mirza *et al.*, 2015a] B. Mirza, Z. Lin, J. Cao, and X. Lai. Voting based weighted online sequential extreme learning machine for imbalance multi-class classification. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 565 – 568, 2015.

[Mirza *et al.*, 2015b] B. Mirza, Z. Lin, and N. Liu. Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift. *Neurocomputing*, 149:316–329, 2015.

[Nguyen *et al.*, 2011] H. M. Nguyen, E. W. Cooper, and K. Kamei. Online learning from imbalanced data streams. In *International Conference of SoCPaR*, pages 347–352, 2011.

[Nishida *et al.*, 2008] K. Nishida, S. Shimada, S. Ishikawa, and K. Yamauchi. Detecting sudden concept drift with knowledge of human behavior. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 3261–3267, 2008.

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

[Sun *et al.*, 2016] Y. Sun, K. Tang, L. L. Minku, S. Wang, and X. Yao. Online ensemble learning of data streams with gradually evolved classes. *IEEE Transaction on Knowledge and Data Engineering*, (Accepted), 2016.

[Wang and Pineau, 2013] B. Wang and J. Pineau. Online ensemble learning for imbalanced data streams. *ArXiv e-prints*, 2013.

[Wang and Yao, 2012] S. Wang and X. Yao. Multi-class imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man and Cybernetics, PartB: Cybernetics*, 42(4):1119–1130, 2012.

[Wang *et al.*, 2013] 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)*, pages 36–45, 2013.

[Wang *et al.*, 2015] S. Wang, L. L. Minku, and X. Yao. Resampling-based ensemble methods for online class imbalance learning. *IEEE Transactions on Knowledge and Data Engineering*, (5):1356 – 1368, 2015.

[Weiss, 2004] G. M. Weiss. Mining with rarity: a unifying framework. *ACM SIGKDD Explorations Newsletter*, 6(1):7–19, 2004.

[Žliobaitė, 2011] I. Žliobaitė. Combining similarity in time and space for training set formation under concept drift. *Intelligent Data Analysis*, 15(4):589–611, 2011.