# A Systematic Study of Online Class Imbalance Learning with Concept Drift

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

***Abstract*—As an emerging research topic, online class imbalance learning often combines the challenges of both class imbalance and concept drift. It deals with data streams having very skewed class distributions, where concept drift may occur. It has recently received increased research attention; however, very little work addresses the combined problem where both class imbalance and concept drift coexist. As the first systematic study of handling concept drift in class-imbalanced data streams, this paper first provides a comprehensive review of current research progress in this field, including current research focuses and open challenges. Then, an in-depth experimental study is performed, with the goal of understanding how to best overcome concept drift in online learning with class imbalance.**

***Index Terms*—Online learning, class imbalance, concept drift, resampling.**

## I. Introduction

With the wide application of machine learning algorithms to the real world, class imbalance and concept drift have become crucial learning issues. Applications in various domains such as risk management [1], anomaly detection [2], software engineering [3], and social media mining [4] are affected by both class imbalance and concept drift. Class imbalance happens when the data categories are not equally represented, i.e., at least one category is minority compared to other categories [5]. It can cause learning bias towards the majority class and poor generalization. Concept drift is a change in the underlying distribution of the problem, and is a significant issue specially when learning from data streams [6]. It requires learners to be adaptive to dynamic changes.

Class imbalance and concept drift can significantly hinder predictive performance, and the problem becomes particularly challenging when they occur simultaneously. This challenge arises from the fact that one problem can affect the treatment of the other. For example, drift detection algorithms based on the traditional classification error may be sensitive to the imbalanced degree and become less effective; and class imbalance techniques need to be adaptive to changing imbalance rates, otherwise the class receiving the preferential treatment may not be the correct minority class at the current moment.

Although there have been papers studying data streams with an imbalanced distribution and data streams with concept drift respectively, very little work discusses the cases when both class imbalance and concept drift exist. Hoens et al. gave the first overview on the combined issue, but only some chunk-based learning techniques were introduced [7]. Our paper aims to provide a more systematic study of handling concept drift in class-imbalanced data streams using experimental studies. We focus on online (i.e. one-by-one) learning, because it is a more difficult case than chunk-based learning, considering that only a single instance is available at a time. Besides, online learning approaches can be applied to problems where data arrives in chunks, but chunk-based learning approaches cannot be applied to online problems where high speed and memory constraints are present. Online learning approaches are particularly useful for applications that produce high-speed data streams, such as robotic systems and sensor networks [3].

We first give a comprehensive review of current research progress in this field, including problem definitions, problem and approach categorization, performance evaluation and up-to-date approaches. It reveals new challenges and research gaps. Most existing work focuses on the concept drift in posterior probabilities (i.e. real concept drift [8], changes in $P(y \mid \mathbf{x})$). The challenges in other types of concept drift have not been fully discussed and addressed. Especially, the change in prior probabilities $P(y)$ is closely related to class imbalance and has been overlooked by most existing work. Most proposed concept drift detection approaches are designed for and tested on balanced data streams. Very few approaches aim to tackle class imbalance and concept drift simultaneously. Among limited solutions, it is still unclear which approach is better and when. It is also unknown whether and how applying class imbalance techniques (e.g. resampling methods) affects concept drift detection and online prediction.

To fill in the research gaps, we then provide an experimental insight into how to best overcome concept drift in online learning with class imbalance, by focusing on three research questions: 1) what are the challenges in detecting each type of concept drift when the data stream is imbalanced? 2) Among the proposed methods designed for online class imbalance learning with concept drift, which one performs better for which type of concept drift? 3) Would applying class imbalance techniques (e.g. resampling methods) facilitate concept drift detection and online prediction? Six recent approaches, DDM-OCI [9], LFR [10], PAUC-PH [11] [12], OOB [13], RLSACP [14] and ESOS-ELM [15], are compared and analyzed in depth under each of the three fundamental types of concept drift (i.e. changes in prior probability $P(y)$, class-conditional probability density function (pdf) $p(\mathbf{x} \mid y)$ and posterior probability $P(y \mid \mathbf{x})$) in artificial data streams,

S. Wang and X. Yao (the corresponding author) 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. X. Yao is also with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, 518055, China. E-mail: {S.Wang, X.Yao}@cs.bham.ac.uk.

L. L. Minku is with the Department of Informatics, University of Leicester, Leicester LE1 7RH, UK. E-mail: leandro.minku@leicester.ac.uk.

as well as real-world data sets. To the best of our knowledge, they are the very few methods that are explicitly designed for online learning problems with class imbalance and concept drift so far.

Finally, based on the review and experimental results, we propose several important issues that need to be considered for developing an effective algorithm for learning from imbalanced data streams with concept drift. We stress the importance of studying the mutual effect of class imbalance and concept drift.

The major contributions of this paper include: (1) this is the first comprehensive study that looks into concept drift detection in class-imbalanced data streams; (2) data problems are categorized into different types of concept drift and class imbalance with illustrative applications; (3) existing approaches are compared and analysed systematically in each type; (4) pros and cons of each approach are investigated; (5) the results provide guidance for choosing the appropriate technique and developing better algorithms for future learning tasks; (6) this is also the first work exploring the role of class imbalance techniques in concept drift detection, which sheds light on whether and how to tackle class imbalance and concept drift simultaneously.

The rest of this paper is organized as follows. Section II formulates the learning problem, including a learning framework and detailed problem descriptions and introduction of class imbalance and concept drift individually. Section III reviews the combined issue of class imbalance and concept drift, including example applications and existing solutions. Section IV carries out the experimental study, aiming to find out the answers to the three research questions. Section V draws the conclusions and points out potential future directions.

## II. Online Learning Framework with Class Imbalance and Concept Drift

In data stream applications, data arrives over time in streams of examples or batches of examples. The information up to a specific time step $t$ is used to build/update predictive models, which then predict the new example(s) arriving at time step $t+1$. Learning under such conditions needs chunk-based learning or online learning algorithms, depending on the number of training examples available at each time step. According to the most agreed definitions [6] [16], chunk-based learning algorithms process a batch of data examples at each time step, such as the case of daily internet usage from a set of users; online learning algorithms process examples one by one and the predictive model is updated after receiving each example [17], such as the case of sensor readings at every second in engineering systems. The term "incremental learning" is also frequently used under this scenario. It is usually referred to as any algorithm that can process data streams with certain criteria met [18].

On one hand, online learning can be viewed as a special case of chunk-based learning. Online learning algorithms can be used to deal with data coming in batches. They both build and continuously update a learning model to accommodate newly available data, and simultaneously maintain its performance on old data, giving rise to the stability-plasticity dilemma [19]. On the other hand, the way of designing online and chunk-based learning algorithms can be very different [6]. Most chunk-based learning algorithms are unsuitable for online learning tasks, because batch learners process a chunk of data each time, possibly using an offline learning algorithm for each chunk. Online learning requires the model being adapted immediately upon seeing the new example, and the example is then immediately discarded, which allows to process high-speed data streams. From this point of view, designing online learning algorithm can be more challenging but so far has received much less attention than the other.

First, the online learner needs to learn from a single data example, so it needs a more sophisticated training mechanism. Second, data streams are often non-stationary (concept drift). The limited availability of training examples at the current moment in online learning hinders the detection of such changes and the application of techniques to overcome the change. Third, it is often seen that data is class imbalanced in many classification tasks, such as the fault detection task in an engineering system, where the fault is always the minority. Class imbalance aggravates the learning difficulty [5]. This difficulty can be further complicated by a dynamically-changing imbalanced distribution [20]. However, there is a severe lack of research addressing the combined issue of class imbalance and concept drift in online learning.

To fill in this research gap, this paper aims at a comprehensive review of the work done to overcome class imbalance and concept drift, a systematic study of learning challenges, and an in-depth analysis of the performance of current approaches. We begin by formalizing the learning problem in this section.

### A. Learning Procedure

In supervised online classification, suppose a data generating process provides a sequence of examples $(\mathbf{x_t}, y_t)$ arriving one at a time from an unknown probability distribution $p_t(x, y)$. $\mathbf{x_t}$ is the input vector belonging to an input space $X$, and $y_t$ is the corresponding class label belonging to the label set $Y = \{c_1, \ldots, c_N\}$. We build an online classifier $F$ that receives the new input $\mathbf{x_t}$ at time step $t$ and then makes a prediction. The predicted class label is denoted by $\hat{y}_t$. After some time, the classifier receives the true label $y_t$, used to evaluate the predictive performance and further train the classifier. This whole process will be repeated at following time steps. It is worth pointing out that we do not assume new training examples always arrive at regular and pre-defined intervals here. In other words, the actual time interval between time step $t$ and $t+1$ may be different from the actual time interval between $t+1$ and $t+2$.

One challenge arises when data is class imbalanced. Class imbalance is an important data feature, commonly seen in applications such as spam filtering [21] and fault diagnosis [2] [3]. It is the phenomenon when some classes of data are highly under-represented (i.e. minority) compared to other classes (i.e. majority). For example, if prior probabilities of the classes $P(c_i) \ll P(c_j)$, then $c_j$ is a majority class and $c_i$ is a minority class. The difficulty in learning from imbalanced

data is that the relatively or absolutely underrepresented class cannot draw equal attention to the learning algorithm, which often leads to very specific classification rules or missing rules for this class without much generalization ability for future prediction. It has been well-studied in offline learning [22], and has attracted growing attention in data stream learning in recent years [7].

In many applications, such as energy forecasting and climate data analysis [23], the data generator operates in nonstationary environments. It gives rise to another challenge, called "concept drift". It means that the probability density function (pdf) of the data generating process is changing over time. For such cases, the fundamental assumption of traditional data mining – the training and testing data are sampled from the same static and unknown distribution – does not hold anymore. Therefore, it is crucial to monitor the underlying changes, and adapt the model to accommodate the changes accordingly.

When both issues exist, the online learner needs to be carefully designed for effectiveness, efficiency and adaptivity. An online class imbalance learning framework was proposed in [20] as a guide for algorithm design. The framework breaks down the learning procedure into three modules – a class imbalance detector, a concept drift detector and an adaptive online learner, as illustrated in Fig. 1.

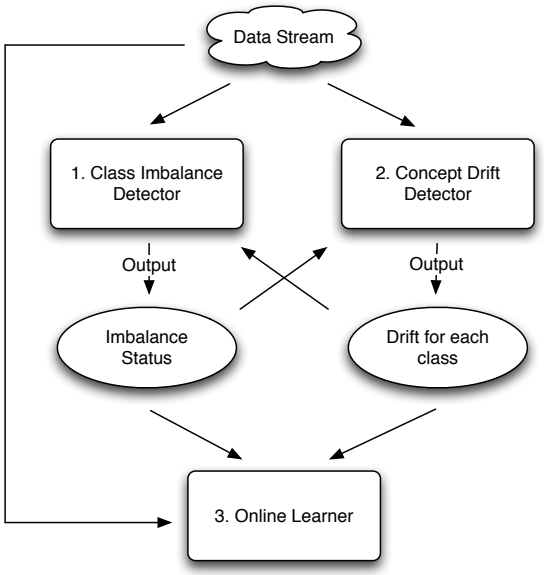



Fig. 1: Learning framework for online class imbalance learning [20].

The class imbalance detector reports the current class imbalance status of data streams. The concept drift detector captures concept drifts involving changes in classification boundaries. Based on the information provided by the first two modules, the adaptive online learner determines when and how to respond to the detected class imbalance and concept drift, in order to maintain its performance. The learning objective of an online class imbalance algorithm can be described as "recognizing minority-class data effectively, adaptively and timely without sacrificing the performance on the majority class" [20].

### B. Problem Descriptions

A more detailed introduction about class imbalance and concept drift is given here individually, including the terminology, research focuses and state-of-the-art approaches. The purpose of this section is to understand the fundamental issues that we need to take extra care of in online class imbalance learning. We also aim at understanding whether and how the current research in class imbalance learning and concept drift detection are individually related to their combined issue elaborated later in Section III, rather than to provide an exhaustive list of approaches in the literature. Among others, we will answer the following questions: *can existing class imbalance techniques process data streams? Would existing concept drift detectors be able to handle imbalanced data streams?*

*1)* ***Class imbalance****:* In class imbalance problems, the minority class is usually much more difficult or expensive to be collected than the majority class, such as the spam class in spam filtering and the fraud class in credit card application. Thus, misclassifying a minority-class example is more costly. Unfortunately, the performance of most conventional machine learning algorithms is significantly compromised by class imbalance, because they assume or expect balanced class distributions or equal misclassification costs. Their training procedure with the aim of maximizing overall accuracy often leads to a high probability of the induced classifier predicting an example as the majority class, and a low recognition rate on the minority class. In reality, it is common to see that the majority class has accuracy close to 100% and the minority class has very low accuracy between 0%-10% [24]. The negative effect of class imbalance on classifiers, such as decision trees [22], neural networks [25], k-Nearest Neighbour (kNN) [26] [27] [28] and SVM [29] [30], has been studied. A classifier that provides a balanced degree of predictive performance for all classes is required. The major research questions in this area are summarized and answered as follows:

(a) *How do we define the imbalanced degree of data?*

It seems to be a trivial question. However, there is no consensus on the definition in the literature. To describe how imbalanced the data is, researchers choose to use the percentage of the minority class in the data set [31], the size ratio between classes [32], or simply a list of the number of examples in each class [33]. The coefficient of variance is used in [34], which is less straightforward. The description of imbalance status may not be a crucial issue in offline learning, but becomes more important in online learning, because there is no static data set in online scenarios. It is necessary to have some measurement automatically describing the up-to-date imbalanced degree and techniques monitoring the changes in class imbalance status. This will help the online learner to decide when and how to tackle class imbalance. The issue of changes in class imbalance status is relevant to concept drift, which will be further discussed in the next subsection.

To define the imbalanced degree suitable for online learning, a real-time indicator was proposed – time-decayed class

size [20], expressing the size percentage of each class in the data stream. It is updated incrementally at each time step by using a time decay (forgetting) factor, which emphasizes the current status of data and weakens the effect of old data. Based on this, a class imbalance detector was proposed to determine which classes should be regarded as the minority/majority and how imbalanced the current data stream is, and then used for designing better online classifiers [13] [3]. The merit of this indicator is that it is suitable for data with arbitrary number of classes.

(b) *When does class imbalance matter?*

It has been shown that class imbalance is not the only problem responsible for the performance reduction of classifiers. Classifiers' sensitivity to class imbalance also depends on the complexity and overall size of the data set. Data complexity comprises issues such as overlapping [35] [36] and small disjuncts [37]. The degree of overlapping between classes and how the minority class examples distribute in data space aggravate the negative effect of class imbalance. The small disjunct problem is associated with the within-class imbalance [38]. Regarding the size of the training data, a very large domain has a good chance that the minority class is represented by a reasonable number of examples, and thus may be less affected by imbalance than a small domain containing very few minority class examples. In other words, the rarity of the minority class can be in a relative or absolute sense in terms of the number of available examples [5].

In particular, authors in [39] [40] 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 learning imbalanced data sets offline, which have also been shown to be the harder cases in online applications [13]. Therefore, for any developed algorithms dealing with imbalanced data online, it is worth discussing their performance on data with different types of distributions.

(c) *How can we tackle class imbalance effectively (state-of-the-art solutions)?*

A number of algorithms have been proposed to tackle class imbalance at the data and algorithm levels. Data-level algorithms include a variety of resampling techniques, manipulating training data to rectify the skewed class distributions. They oversample minority-class examples (i.e. expanding the minority class), undersample majority-class examples (i.e. shrinking the majority class), or combine both, until the data set is relatively balanced. Random oversampling and random undersampling are the simplest and most popular resampling techniques, where examples are randomly chosen to be added or removed. There are also smart resampling techniques (a.k.a guided resampling). For example, SMOTE [33] is a widely used oversampling method, which generates new minority-class data points based on the similarities between original minority-class examples in the feature space. Other smart oversampling techniques include Borderline-SMOTE [41], ADASYN [42], MWMOTE [43], to name but a few. Smart undersampling techniques include Tomek links [44], One-sided selection [45], Neighbourhood cleaning rule [46], etc. The effectiveness of resampling techniques have been proved in real-world applications [47]. They work independently of classifiers, and are thus more versatile than algorithm-level methods. The key is to choose an appropriate sampling rate [48], which is relatively easy for two-class data sets, but becomes more complicated for multi-class data sets [49]. Empirical studies have been carried out to compare different resampling methods [31]. Particularly, it is shown that smart resampling techniques are not necessarily superior to random oversampling and undersampling; besides, they cannot be applied to online scenarios directly, because they work on a static data set for the relation among the training examples. Some initial effort has been made recently, to extend smart resampling techniques to online learning [50].

Algorithm-level methods address class imbalance by modifying their training mechanism with the direct goal of better accuracy on the minority class, including one-class learning [51], cost-sensitive learning [52] and threshold methods [53]. They require different treatments for specific kinds of learning algorithms. In other words, they are algorithm-dependent, so they are not as widely used as data-level methods. Some online cost-sensitive methods have been proposed, such as CSOGD [54] and RLSACP [14]. They are restricted to the perceptron-based classifiers, and require pre-defined misclassification costs of classes that may or may not be updated during the online learning.

Finally, ensemble learning (also known as multiple classifier systems) [55] has become a major category of approaches to handling class imbalance [56]. It combines multiple classifiers as base learners and aims to outperform every one of them. It can be easily adapted for emphasizing the minority class by integrating different resampling techniques [57] [58] [59] [60] or by making base classifiers cost-sensitive [61] [62] [63] [64]. A few ensemble methods are available for online class imbalance learning, such as OOB and UOB [13] applying random oversampling and undersampling in Online Bagging [65], and WOS-ELM [66] training a set of cost-sensitive online extreme learning machines.

It is worth pointing out that, the aforementioned online learning algorithms designed for imbalanced data are unsuitable for non-stationary data streams. They do not involve any mechanism handling drifts that affect classification boundaries, although OOB and UOB can detect and react to class imbalance changes.

(d) *How do we evaluate the performance of class imbalance learning algorithms?*

Traditionally, overall accuracy and error rate are the most frequently used metrics of performance evaluation. However, they are strongly biased towards the majority class when data is imbalanced. Therefore, other performance measures have been adopted. Most studies concentrate on two-class problems. By convention, the minority class is treated to be the positive, and the majority class is treated to be the negative.

Table I illustrates the confusion matrix of a two-class problem, producing four numbers on testing data.

TABLE I: Confusion matrix for a two-class problem.

| | Predicted as positive | Predicted as negative |
|---|---|---|
| Actual positive | True positive (TP) | False negative (FN) |
| Actual negative | False positive (FP) | True negative (TN) |

From the confusion matrix, we can derive the expressions for *recall* and *precision*:

$$recall = \frac{TP}{TP + FN}, \tag{1}$$

$$precision = \frac{TP}{TP + FP}. \tag{2}$$

Recall (i.e. TP rate) is a measure of completeness – the proportion of positive class examples that are classified correctly to all positive class examples. Precision is a measure of exactness – the proportion of positive class examples that are classified correctly to the examples predicted as positive by the classifier. The learning objective of class imbalance learning is to improve recall without hurting precision. However, improving recall and precision can be conflicting. Thus, F-measure is defined to show the trade-off between them.

$$Fm = \frac{\left(1 + \beta^2\right) \cdot recall \cdot precision}{\beta^2 \cdot precision + recall}, \tag{3}$$

where $\beta$ corresponds to the relative importance of recall and precision. It is usually set to 1. Kubat et al. [45] proposed to use G-mean to replace overall accuracy:

$$Gm = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}}. \tag{4}$$

It is the geometric mean of positive accuracy (i.e. TP rate) and negative accuracy (i.e. TN rate). A good classifier should have high accuracies on both classes, and thus a high G-mean.

According to [5], any metric that uses values from both rows of the confusion matrix for addition (or subtraction) will be inherently sensitive to class imbalance. In other words, the performance measure will change as class distribution changes, even though the underlying performance of the classifier does not. This performance inconsistency can cause problems when we compare different algorithms over different data sets. Precision and F-measure, unfortunately, are sensitive to the class distribution. Therefore, recall and G-mean are better options.

To compare classifiers over a range of sample distributions, AUC (abbr. of the Area Under the ROC curve) is the best choice. A ROC curve depicts all possible trade-offs between TP rate and FP rate, where FP rate = $FP/\left(TN + FP\right)$. TP rate and FP rate can be understood as the benefits and costs of classification with respect to data distributions. Each point on the curve corresponds to a single trade-off. A better classifier should produce a ROC curve closer to the top left corner. AUC represents a ROC curve as a single scalar value by estimating the area under the curve, varying in [0, 1]. It is insensitive to the class distribution, because both TP rate and FP rate use values from only one row of the confusion matrix. AUC is usually generated by varying the classification decision threshold for separating positive and negative classes in the testing data set [67] [68]. In other words, calculating AUC requires a set of confusion matrices. Therefore, unlike other measures based on a single confusion matrix, AUC cannot be used as an evaluation metric in online learning without memorizing data. Although a recent study has modified AUC for evaluating online classifiers [11], it still needs to collect recently received examples.

The properties of the above measures are summarized in Table II. They are defined under the two-class context. They cannot be used to evaluate multi-class data directly, except for recall. Their multi-class versions have been developed [69] [70] [71]. The "multi-class" and "online" columns in the table show whether the corresponding measure can be used directly without modification in multi-class and online data scenarios.

TABLE II: Performance evaluation measures for class imbalance problems.

| Measures | Multi-class | Online | Sensitive to Imbalance |
|---|---|---|---|
| recall | yes | yes | no |
| precision | no [69] | yes | yes |
| Fm | no [69] | yes | yes |
| Gm | yes [70] | yes | no |
| AUC | no (See MAUC [71]) | no (See PAUC [11]) | no |

*2) **Concept drift**:* Concept drift is said to occur when the joint probability $P\left(\mathbf{x}, y\right)$ changes [8] [72] [73]. The key research topics in this area include:

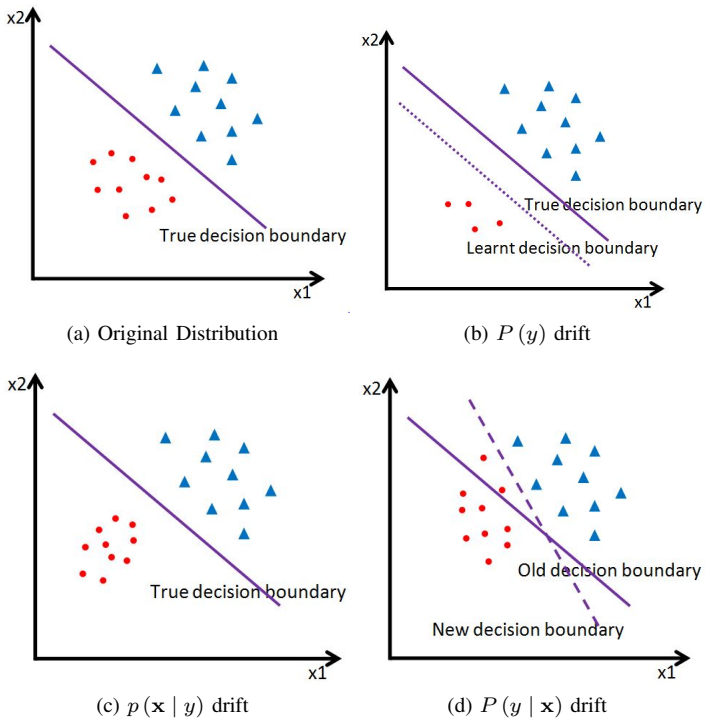
(a) *How many types of concept drift are there? Which type is more challenging?*

Concept drift can manifest three fundamental forms of changes corresponding to the three major variables in the Bayes' theorem [74]: 1) a change in prior probability $P\left(y\right)$; 2) a change in class-conditional pdf $p\left(\mathbf{x} \mid y\right)$; 3) a change in posterior probability $P\left(y \mid \mathbf{x}\right)$. The three types of concept drift are illustrated in Figure 2, comparing to the original data distribution shown in Figure 2(a).

Fig. 2(b) shows the $P\left(y\right)$ type of concept drift without affecting $p\left(\mathbf{x} \mid y\right)$ and $P\left(y \mid \mathbf{x}\right)$. The decision boundary remains unaffected. The prior probability of the circle class is reduced in this example. Such change can lead to class imbalance. A well-learnt discrimination function may drift away from the true decision boundary, due to the imbalanced class distribution.

Fig. 2(c) shows the $p\left(\mathbf{x} \mid y\right)$ type of concept drift without affecting $P\left(y\right)$ and $P\left(y \mid \mathbf{x}\right)$. The true decision boundary remains unaffected. Elwell and Polikar claimed that this type of drift is the result of an incomplete representation of the true distribution in current data, which simply requires providing supplemental data information to the learning model [75].

Fig. 2(d) shows the $P\left(y \mid \mathbf{x}\right)$ type of concept drift. The true boundary between classes changes after the drift, so that the previously learnt discrimination function does not apply any more. In other words, the old function becomes unsuitable




(a) Original Distribution (b) $P(y)$ drift

(c) $p(\mathbf{x} \mid y)$ drift (d) $P(y \mid \mathbf{x})$ drift

Fig. 2: Illustration of 3 concept drift types.

or partially unsuitable, and the learning model needs to be adapted to the new knowledge.

The posterior distribution change clearly indicates the most fundamental change in the data generating function. This is classified as *real concept drift*. The other two types belong to *virtual concept drift* [7], which does not change the decision (class) boundaries. In practice, one type of concept drift may appear in combination with other types.

Existing studies primarily focus on the development of drift detection methods and techniques to overcome the real drift. There is a significant lack of research on virtual drift, which can also deteriorate classification performance. As illustrated in Fig. 2(b), even though these types of drift do not affect the true decision boundaries, they can cause a well-learnt decision boundary to become unsuitable. Unfortunately, the current techniques for handling real drift may not be suitable for virtual drift, because they present very different learning difficulties and require different solutions. For instance, the methods for handling real drift often choose to reset and retrain the classifier, in order to forget the old concept and better learn the new concept. This is not an appropriate strategy for data with virtual drift, because the examples from previous time steps may still remain valid and help the current classification in virtual drift cases. It would be more effective and efficient to calibrate the existing classifier than retraining it. Besides, techniques for handling real drift typically rely on feedback about the performance of the classifier, while techniques for handling virtual drift can operate without such feedback [8]. From our point of view, all three types are equally important. Particularly, the two virtual types require more research effort than currently dedicated work by our community. A systematic study of the challenges in each type will be given in Section IV.

Concept drift has further been characterized by its speed, severity, cyclical nature, etc. A detailed and mutually exclusive categorization can be found in [73]. For example, according to speed, concept drift can be either abrupt, when the generating function is changed suddenly (usually within one time step), or gradual, when the distribution evolves slowly over time. They are the most commonly discussed types in the literature, because the effectiveness of drift detection methods can vary with the drifting speed. While most methods are quite successful in detecting abrupt drifts, as future data is no longer related to old data [76], gradual drifts are often more difficult, because the slow change can delay or hide the hint left by the drift. We can see some drift detection methods specifically designed for gradual concept drift, such as Early Drift Detection method (EDDM) [77].

(b) *How can we tackle concept drift effectively (state-of-the-art solutions)?*

There is a wide range of algorithms for learning in non-stationary environments. Most of them assume and specialize in some specific types of concept drift, although real-world data often contains multiple types. They are commonly categorized into two major groups: active vs. passive approaches, depending on whether an explicit drift detection mechanism is employed. Active approaches (also known as trigger-based approaches) determine whether and when a drift has occurred before taking any actions. They operate based on two mechanisms – a change detector aiming to sense the drift accurately and timely, and an adaptation mechanism aiming to maintain the performance of the classifier by reacting to the detected drift. Passive approaches (also known as adaptive classifiers) evolve the classifier continuously without an explicit trigger reporting the drift. A comprehensive review of up-to-date techniques tackling concept drift is given by Ditzler et al. [16]. They further organise these techniques based on their core mechanisms, summarized in Table III. This table will help us to understand how online class imbalance algorithms are designed, which will be introduced in details in Section III. There exist other ways to classify the proposed algorithms, such as Gama et al.'s taxonomy based on the four modules of an adaptive learning system [8], and Webb et al.'s quantitative characterization [78]. This paper adopts the one proposed by Ditzler et al. [16] for its simplicity.

The best algorithm varies with the intended applications. A general observation is that, while active approaches are quite effective in detecting abrupt drift, passive approaches are very good at overcoming gradual drift [75] [16]. It is worth noting that most algorithms do not consider class imbalance. It is unclear whether they will remain effective if data becomes imbalanced. For example, some algorithms determine concept drift based on the change in the classification error, including OLIN [79], DDM [80] and PERM [81]. As we have explained in Section II-B 1), the classification error is sensitive to the imbalance degree of data, and does not reflect the performance of the classifier very well when there is class imbalance. Therefore, these algorithms may not perform well when concept drift and class imbalance occur simultaneously. Some recent papers tried to tackle this issue by using other performance

TABLE III: Categorization of concept drift techniques. See [16] for the full list of techniques under each category.

| | | |
|---|---|---|
| **Active** | **Step1. Change detection** | **Hypothesis tests**: assess the validity of a hypothesis by comparing the distributions of two sets of fix-length data sequences. |
| | | **Change-point methods**: identify the change point by analyzing all possible partitions of a fixed data sequence. |
| | | **Sequential hypothesis tests**: provide a one-off detection of change or no change, by inspecting incoming examples one by one (sequentially). |
| | | **Change detection tests**: analyze the statistical behavior of streams of data in a fully sequential manner, such as a feature value or classification error. They are either based on a pre-defined threshold or some statistical features representing current data. |
| | **Step2. Classifier adaptation** | **Windowing**: the classifier is retrained based on a window with up-to-date examples. The window length can be either fixed or adaptive. |
| | | **Weighting**: all received examples are weighted according to time or classification error, which are then used to update the classifier. |
| | | **Random Sampling**: the examples used to retrain the classifier are randomly chosen based on certain rules. |
| | | **Ensemble**: build a new model in the classifier for the new concept. |
| **Passive** | **Single classifier**: update a single classifier, such as decision trees, online information network, and extreme learning machine. | |
| | **Ensemble**: add, remove or modify the models in an ensemble classifier. | |

metrics that are more robust to the imbalance degree. More details will be given in Section III. Some other algorithms are specifically designed for data streams coming in batches, such as AUE [82] and the Learn++ family [75]. These algorithms cannot be applied to online cases directly.

(c) *How do we evaluate the performance of concept drift detectors and online classifiers?*

To fully test the performance of drift detection approaches (especially an active detector), it is necessary to discuss both data with artificial concept drifts and real-world data with unknown drifts. Using data with artificial concept drifts allows us to easily manipulate the type and timing of concept drifts, so as to obtain an in-depth understanding of the performance of approaches under various conditions. Testing on data from real-world problems helps us to understand their effectiveness from the practical point of view, but the information about when and how concept drift occurs is unknown in most cases. The following aspects are usually considered to assess the accuracy of active drift detectors. Their measurement is based on data with artificial concept drifts where drifts are known.

- True detection rate: the possibility of detecting the true concept drift. It shows the accuracy of the detection approach.
- False alarm rate: the possibility of reporting a concept drift that does not exist (false-positive rate). It characterizes the costs and reliability of the detection approach.
- Delay of detection: an estimate of how many time steps are required on average to detect a drift after the actual occurrence. It reflects how much time would be taken before the drift is detected.

Wang and Abraham [10] use a histogram to visualize the distribution of detection points from the drift detection approach over multiple runs. It reflects all the three aspects above in one plot. It is worth nothing that there are trade-offs between these measures. For example, an approach with a high true detection rate may produce a high false alarm rate. A very recent algorithm, Hierarchical Change-Detection Tests (HCDTs), was proposed to explicitly deal with the trade-off [83].

After the performance of drift detection approaches is better understood, we need to quantify the effect of those detections on the performance of predictive models. All the performance metrics introduced in the previous section of "class imbalance" can be used. The key question here is how to calculate them in the streaming settings with evolving data. The performance of the classifier may get better or worse every now and then. There are two common ways to depict such performance over time – holdout and prequential evaluation [8].

Holdout evaluation is mostly used when the testing data set (holdout set) is available in advance. At each time step or every few time steps, the performance measures are calculated based on the valid testing set, which must represent the same data concept as the training data at that moment. However, this is a very rigorous requirement for data from real-world applications.

In prequential evaluation, data received at each time step is used for testing before it is used for training. From this, the performance measures can be incrementally updated for evaluation and comparison. This strategy does not require a holdout set, and the model is always tested on unseen data.

When the data stream is stationary, the prequential performance measures can be computed based on the accumulated sum of a loss function from the beginning of the training. However, if the data stream is evolving, the accumulated measure can mask the fluctuation in performance and the adaptation ability of the classifier. For example, consider that an online classifier correctly predicts 90 out of 100 examples received so far (90% accuracy on data with the original concept). Then, an abrupt concept drift occurs at time step 101, which makes the classifier only correctly predict 3 out of 10 examples from the new concept (30% accuracy on data with the new concept). If we use the accumulated measure based on all the historical data, the overall accuracy will be 93/110, which seems to be high but does not reflect the true performance on the new data concept. This problem can be solved by using a sliding window or a time-based fading factor that weigh observations [84].

## III. Overcoming Class Imbalance and Concept Drift Simultaneously

Following the review of class imbalance and concept drift in Section II, this section reviews the combined issue, including example applications and existing solutions. When both exist, one problem affects the treatment of the other. For example,

the drift detection algorithms based on the traditional classification error may be sensitive to imbalanced degree and become less effective; the class imbalance techniques need to be adaptive to changing $P(y)$, otherwise the class receiving the preferential treatment may not be the correct minority class at the current moment. Therefore, their mutual effect should be considered during the algorithm design.

### A. Illustrative Applications

The combined problems of concept drift and class imbalance have been found in many real-world applications. Three examples are given here, to help us understand each type of concept drift.

*1) Environment monitoring with $P(y)$ drift:* Environment monitoring systems usually consist of various sensors generating streaming data in high speed. Real-time prediction is required. For example, a smart building has sensors deployed to monitor hazardous events. Any sensor fault can cause catastrophic failures. Machine learning algorithms can be used to build models based on the sensor information, aiming to predict faults in sensors accurately and timely [3]. First, the data is characterized by class imbalance, because obtaining a fault in such systems can be very expensive. Examples representing faults are the minority. Second, the number of faults varies with the faulty condition. If the damage gets worse over time, the faults will occur more and more frequently. It implies a prior probability change, a type of virtual concept drift.

*2) Spam filtering with $p(\mathbf{x} \mid y)$ drift:* Spam filtering is a typical classification problem involving class imbalance and concept drift [85]. First of all, the spam class is the minority and suffers from a higher misclassification cost. Second, the spammers are actively working on how to break through the filter. It means that the adversary actions are adaptive. For example, one of the spamming behaviours is to change email content and presentation in disguise, implying a possible class-conditional pdf ($p(\mathbf{x} \mid y)$) change [8].

*3) Social media analysis with $P(y \mid \mathbf{x})$ drift:* In social media (e.g. twitter, facebook), consider the example where a company would like to make relevant product recommendations to people who have shown some type of interest in their tweets. Machine learning algorithms can be used to discover who is interested in the product based on the tweets [86]. The number of users who have shown the interest is always very small. So, this is a minority class. Meanwhile, users' interest changes from time to time. Users may lose their interest in the current trendy product very quickly, causing posterior probability ($P(y \mid \mathbf{x})$) changes.

Although the above examples are associated with only one type of concept drift, different types often coexist in real-world problems, which are hard to know in advance. For the example of spam filtering, which email belongs to spam also depends on users' interpretation. Users may re-label a particular category of normal emails as spam, which indicates a posterior probability change.

### B. Approaches to Tackling Both Class Imbalance and Concept Drift

Some research efforts have been made to address the joint problem of concept drift and class imbalance, due to the rising need from practical problems [87] [1]. Uncorrelated Bagging is one of the earliest algorithms, which builds an ensemble of classifiers trained on a more balanced set of data through resampling and overcomes concept drift passively by weighing the base classifier based on their discriminative power [88] [89] [90]. Selectively recursive approaches SERA [91] and REA [92] use similar ideas to Uncorrelated Bagging of building an ensemble of weighted classifiers, but with a "smarter" oversampling technique. Learn++.CDS and Learn++.NIE are more recent algorithms, which tackle class imbalance through the oversampling technique SMOTE [33] or a sub-ensemble technique, and overcome concept drift through a dynamic weighting strategy [93]. HUWRS.IP [94] improves HUWRS [95] to deal with imbalanced data streams by introducing an instance propagation scheme based on a Naïve Bayes classifier, and using Hellinger distance as a weighting measure for concept drift detection. The Hellinger weight for drift detection is calculated as the average of the minority-class and majority-class Hellinger distance between the two feature distributions. It guarantees equal weight to the Hellinger distance between the minority-class and majority-class distributions. The instance propagation scheme selects old minority-class examples that are relevant to the current data concept. It avoids the problem of using misleading examples from the old data concept. However, relevant examples may not exist in some rapid drifting cases. So, Hellinger Distance Decision Tree (HDDT) was proposed to use Hellinger distance as the decision tree splitting criteria that is imbalance-insensitive [96]. All these approaches belong to chunk-based learning algorithms. Their core techniques work when a batch of data is received at each time step, i.e., they are unsuitable for online processing. Developing a true online algorithm for concept drift is very challenging because of the difficulties in measuring minority-class statistics using only one example at a time [16].

To detect concept drift in an online imbalanced scenario, a few methods have been proposed recently. Drift Detection Method for Online Class Imbalance (DDM-OCI) [9] is one of the very first algorithms detecting concept drift actively in imbalanced data streams online. It monitors the reduction in minority-class recall (i.e. true positive rate). If there is a significant drop, a drift will be reported. It was shown to be effective in cases when minority-class recall is affected by the concept drift, but not when the majority class is mainly affected. A Linear Four Rates (LFR) approach was then proposed to improve DDM-OCI, which monitors four rates from the confusion matrix – minority-class recall and precision and majority-class recall and precision, with statistically-supported bounds for drift detection [10]. If any of the four rates exceeds the bound, a drift will be confirmed. Instead of tracking several performance rates for each class, prequential AUC (PAUC) [11] [12] was proposed as an overall performance measure for online scenarios, and was used as the concept drift

TABLE IV: Online approaches to tackling concept drift and class imbalance, and their properties.

| Approaches | Category? | Class imbalance? | Access to old data? | Additional data? | Multi-class? | Drift type? |
|---|---|---|---|---|---|---|
| DDM-OCI [9] | Active (change detection test + windowing) | No | No | No | No | $P(y \mid \mathbf{x})$ |
| LFR [10] | Active (change detection test + windowing) | No | No | No | No | $P(y \mid \mathbf{x})$ |
| PAUC-PH [11] | Active (change detection test + windowing) | No | Yes | No | No | $P(y \mid \mathbf{x})$ |
| RLSACP [14]/ONN [98] | Passive (single classifier) | Yes | Yes | No | No | all 3 types |
| ESOS-ELM [15] | Passive+Active (ensemble) | Yes | No | Yes | No | $p(\mathbf{x} \mid y)$, $P(y \mid \mathbf{x})$ |
| OOB/UOB using CID [13] | Active (weighting) | Yes | No | No | No | $P(y)$ |

indicator in Page-Hinkley (PH) test [97]. However, it needs access to historical data. DDM-OCI, LFR and PAUC-based PH test are active drift detectors designed for imbalanced data streams, and are independent of classification algorithms. They aim at concept drift with classification boundary changes by default. Therefore, if a concept drift is reported, they will reset and retrain the online model. Although these drift detectors are designed for imbalanced data, they themselves do not involve any class imbalance techniques, such as resampling, to adjust the decision boundary of the online model. It is still unclear how they perform when working with class imbalance techniques.

Besides the above active approaches, the perceptron-based algorithms RLSACP [14], ONN [98] and ESOS-ELM [15] adapt the classification model to non-stationary environments passively, and involve mechanisms to overcome class imbalance. RLSACP and ONN are single-model approaches with the same general idea. Their error function for updating the perceptron weights is modified, including a forgetting function for model adaptation and an error weighting strategy as the class imbalance treatment. The forgetting function has a pre-defined form, allowing the old data concept to be forgotten gradually. The error weights in RLSACP are incrementally updated based either on the classification performance or the imbalance rate from recently received data. It was shown that weight updating based on the imbalance rate leads to better performance.

ESOS-ELM is an ensemble approach, maintaining a set of online sequential extreme learning machines (OS-ELM) [99]. For tackling class imbalance, resampling is applied in a way that each OS-ELM is trained with approximately equal number of minority- and majority-class examples. For tackling concept drift, voting weights of base classifiers are updated according to their performance G-mean on a separate validation data set from the same environment as the current training data. In addition to the passive drift detection technique, ESOS-ELM includes an independent module – ELM-store, to handle recurring concept drift. ELM-store maintains a pool of weighted extreme learning machines (WELM) [66] to retain old information. It adopts a threshold-based technique and hypothesis testing to detect abrupt and gradual concept drift actively. If a concept drift is reported, a new WELM will be built and kept in ELM-store. If any stored model performs better than the current OS-ELM ensemble, indicating a possible recurring concept, it will be introduced in the ensemble. ESOS-ELM assumes the imbalance rate is known in advance and fixed. It needs a separate data set for initializing OS-ELMs and WELMs, which must include examples from all classes. It is also necessary to have validation data sets reflecting every data concept for concept drift detection, which can be a quite restrictive requirement for real-world data.

With a different goal of concept drift detection from the above, a class imbalance detection (CID) approach was proposed, aiming at $P(y)$ changes [20]. It reports the current imbalance status and provides information of which classes belong to the minority and which classes belong to the majority. Particularly, a key indicator is the real-time class size $w_k^{(t)}$, the percentage of class $c_k$ at time step t. When a new example $\mathbf{x_t}$ arrives, $w_k^{(t)}$ is incrementally updated by the following equation [20]:

$$w_k^{(t)} = \theta w_k^{(t-1)} + (1-\theta)\left[(\mathbf{x_t}, c_k)\right], (k = 1, \ldots, N) \quad (5)$$

where $[(\mathbf{x_t}, c_k)] = 1$ if the true class label of $\mathbf{x_t}$ is $c_k$, and 0 otherwise. $\theta$ $(0 < \theta < 1)$ is a pre-defined time decay (forgetting) factor, which reduces the contribution of older data to the calculation of class sizes along with time. It is independent of learning algorithms, so it can be used with any type of online classifiers. For example, it has been used in OOB and UOB [13] for deciding the resampling rate adaptively and overcoming class imbalance effectively over time. OOB and UOB integrate oversampling and undersampling respectively into ensemble algorithm Online Bagging (OB) [65]. Oversampling and undersampling are one of the simplest and most effective techniques of tackling class imbalance [31].

The properties of the above online approaches are summarized in Table IV, answering the following six questions in order:

- How do they handle concept drift (the type based on the categorization in Table III)?
- Do they involve any class imbalance technique to improve the predictive performance of online models, in addition to concept drift detection?
- Do they need access to previously received data?
- Do they need additional data sets for initialisation or validation?
- Can they handle data streams with more than two classes (multi-class data)?
- Which type of concept drift can it deal with?

## IV. Performance Analysis

With a complete review of online class imbalance learning, we aim at a deep understanding of concept drift detection in imbalanced data streams and the performance of existing approaches introduced in Section III-B. Three research questions will be looked into through experimental analysis: *1)*

*what are the difficulties in detecting each type of concept drift?* Little work has given separate discussions on the three fundamental types of concept drift, especially the $P(y)$ drift. It is important to understand their differences, so that the most suitable approaches can be used for the best performance. *2) Among existing approaches designed for imbalanced data streams with concept drift, which approach is better and when?* Although a few approaches have been proposed for the purpose of overcoming concept drift and class imbalance, it is still unclear how well they perform for each type of concept drift. *3) Whether and how do class imbalance techniques affect concept drift detection and online prediction?* No study has looked into the mutual effect of applying class imbalance techniques and concept drift detection methods. Understanding the role of class imbalance techniques will help us to develop more effective concept drift detection methods for imbalanced data.

### A. Data Sets

For an accurate analysis and comparable results, we choose two most commonly used artificial data generators, SINE1 [80] and SEA [100], to produce imbalanced data streams containing three simulated types of concept drift. In SINE1, each generated point has two attributes $(x_1, x_2)$, uniformly distributed in $[0, 1]$. The concept is decided by where the point is located (above the $\sin$ function or not). In SEA, each sample has three attributes $x_1$, $x_2$ and $x_3$ with values between 0 and 10. Only the first two attributes are relevant. The class label is determined by a threshold.

This is one of the very few studies that individually discuss $P(y)$, $p(\mathbf{x} \mid y)$ and $P(y \mid \mathbf{x})$ types of concept drift in depth. In addition, each generator produces two data streams with a different drifting speed – abrupt and gradual drifts. The drifting speed is defined as the inverse of the time taken for a new concept to completely replace the old one [73]. According to speed, drifts can be either abrupt, when the generating function is changed completely in only one time step, or gradual, otherwise. The data streams with a gradual concept drift are denoted by 'g' in the following experiment, i.e. SINE1g [77] and SEAg. Every data stream has 3000 time steps, with one concept drift starting at time step 1501. The new concept in SINE1 and SEA fully takes over the data stream from time step 1501; the concept drift in SINE1g and SEAg takes 500 time steps to complete, which means that the new concept fully replaces the old one from time step 2001. The detailed settings for generating each type of concept drift are included in the individual subsections.

After the detailed analysis of the three types of concept drift, three real-world data sets are included in our experiment with unknown concept drift, which are PAKDD 2009 credit card data (PAKDD) [101], Weather data [76] and UDI TweetCrawl data [102]. Data in PAKDD are collected from the private label credit card operation of a Brazilian retail chain. The task of this problem is to identify whether the client has a good or bad credit. The "bad" credit is the minority class, taking 19.75% of the provided modelling data. Because the data have been collected from a time interval in the past, gradual market change occurs. The Weather data set aims to predict whether rain precipitation was observed on each day, with inherent seasonal changes. The class of "rain" is the minority, taking 31% of the data set. The original Tweet data include 50 million tweets posted mainly from 2008 to 2011. The task is to predict the tweet topic. We choose a time interval, containing 8774 examples and covering seven tweet topics [103]. Then, we further reduce it to 2-class data by using only two out of seven topics for our experiment. These real-world data will help us to understand the effectiveness of existing concept drift and class imbalance approaches in practical scenarios, which usually have more complex data distributions and concept drift.

### B. Experimental and Evaluation Settings

The approaches listed in Table IV, which are explicitly designed for the combined problem of class imbalance and concept drift, are discussed in our experiment. For the three active drift detection methods – DDM-OCI, LFR and PAUC-PH, they need to work with online learning algorithms for classification. We choose two approaches to build the online model, the traditional Online Bagging (abbr. OB) [65] and OOB with CID [13], to build the online model. Because OOB applies oversampling to overcome class imbalance and OB does not, it can help us to observe the role of class imbalance techniques (oversampling in our experiment) in concept drift detection. UOB is not chosen, for the consideration that undersampling may cause unstable performance which may indirectly affect our observation [13]. Between RLSACP and ONN, due to their similarity and the more theoretical support in RLSACP, only RLSACP is included in our experiment.

Considering RLSACP and ESOS-ELM are perceptron-based methods, we use the Multilayer Perceptron (MLP) classifier as the base learner of OB and OOB. The number of neurons in the hidden layer of MLPs is set to the average of the number of attributes and classes in data, which is also the number of perceptrons in RLSACP and in the base learner of ESOS-ELM. All ensemble methods maintain 15 base learners. For ESOS-ELM, we disable the "ELM-Store", which is designed for recurring concept drift; we allow that its ensemble size can grow to 20. In addition, ESOS-ELM requires an initialisation data set to initialize ELMs, and validation data sets to adjust misclassification costs. When dealing with artificial data, we use the first 100 examples to initialize ESOS-ELM, and generate a separate validation data set for each concept stage. We track the performance of all the methods from time step 101.

In summary, ten algorithms join the comparison from Table IV: OB, OOB, DDM-OCI+OB/OOB, PAUC-PH+OB/OOB, LFR+OB/OOB, RLSACP and ESOS-ELM. OB is the baseline without involving any class imbalance and concept drift techniques.

To evaluate the effectiveness of concept drift detection methods and online learners, we adopt prequential test (as described in Section II) for its simplicity and popularity. Prequential recall of each class (defined in Eq. 1) and prequential G-mean (defined in Eq. 4) are tracked over time for

TABLE V: Artificial data streams with $P(y)$ concept drift.

| ID | Data | Speed | Class +1 | | | Class -1 | | |
|---|---|---|---|---|---|---|---|---|
| | | | Concept | Old $P(y)$ | New $P(y)$ | Concept | Old $P(y)$ | New $P(y)$ |
| 1 | SINE1 | Abrupt | Points below $x_2 = \sin(x_1)$ | 0.1 | 0.9 | Points above or on $x_2 = \sin(x_1)$ | 0.9 | 0.1 |
| 2 | SINE1g | Gradual | | | | | | |
| 3 | SEA | Abrupt | $x_1 + x_2 \leq 7$ | 0.5 | 0.1 | $x_1 + x_2 > 7$ | 0.5 | 0.9 |
| 4 | SEAg | Gradual | | | | | | |

comparison, because they are insensitive to imbalance rates. When discussing the generated artificial data sets with ground truth known, we also compare the true detection rate (abbr. TDR), total number of false alarms (abbr. FA) and delay of detection (abbr. DoD) (as defined in Section II) among methods using any of the three active drift detectors (i.e. DDM-OCI, LFR and PAUC-PH). The calculation of TDR, FA and DoD is the same for both of the abrupt and the gradual drifting cases, based on the following understanding: before a real concept drift occurs (before time step 1500 in our cases), all the reported alarms are considered as false alarms; after a real concept drift starts (after time step 1500 in our cases), the first detection is seen as the true drift detection; after that and before the next new real concept drift, the consequent detections are considered as false alarms.

Furthermore, because we are particularly interested in how the learner performs on the new data concept in the artificial data sets, we calculate the average recall and G-mean over all the time steps after the concept drift completely ends (time step 1500 for the abrupt drifting cases and time step 2000 for the gradual drifting cases). It is worth noting that the recall and G-mean values are reset to 0 when the drift starts and ends for an accurate analysis. We use the Wilcoxon Sign Rank test at the confidence level of 95% as our significance test in this paper.

### *C. Comparative Study on Artificial Data*

#### C.1. $\mathbf{P}(y)$ Concept Drift

This section focuses on the $P(y)$ type of concept drift, without $p(\mathbf{x} \mid y)$ and $P(y \mid \mathbf{x})$ changes. Data streams SINE1 and SINE1g have a severe class imbalance change, in which the minority (majority) class during the first half of data streams becomes the majority (minority) during the latter half. SEA and SEAg have a less severe change, in which the data stream presented to be balanced during the first half becomes imbalanced during the latter half. $P(y)$ is changed linearly during the concept transition period (time step 1501 to time step 2000) in the gradual drifting cases. The concrete setting for each data stream is summarized in Table V.

Table VI compares the detection performance of the three active concept drift detectors, in terms of TDR, FA and DoD. We can see that DDM-OCI and LFR are sensitive to class imbalance changes in data. They present very high true detection rate; especially, LFR has 100% TDR in all cases regardless of whether resampling is used to tackle class imbalance. PAUC-PH does not report any concept drift, showing 0% TDR in all cases. This is because DDM-OCI and LFR use time-decayed metrics as the indicator of concept drift, which have higher sensitivity to performance change in general than the prequential AUC used by PAUC-PH. LFR shows even higher TDR than DDM-OCI, because it tracks four rates in the confusion matrix instead of one. For the same reason, DDM-OCI and LFR have a higher chance of issuing false alarms than PAUC-PH. For DDM-OCI, oversampling in OOB increases the probability of reporting a concept drift by observing TDR in SEA and SEAg, compared to OB. This is because more examples are used for training in OOB, which improves the performance on the minority class for concept drift detection.

TABLE VI: Performance of the 3 active concept drift detectors on artificial data with $P(y)$ changes: TDR, FA and DoD. The '-' symbol indicates that no concept drift is detected.

| | Method | TDR | FA | DoD |
|---|---|---|---|---|
| SINE1 | DDM-OCI+OB | 100% | 0 | 94 |
| | DDM-OCI+OOB | 100% | 2.22 | 45 |
| | LFR+OB | 100% | 24 | 91 |
| | LFR+OOB | 100% | 26.16 | 63 |
| | PAUC-PH+OB | 0% | 1.03 | - |
| | PAUC-PH+OOB | 0% | 1.28 | - |
| SINE1g | DDM-OCI+OB | 100% | 1.09 | 281 |
| | DDM-OCI+OOB | 100% | 4.38 | 118 |
| | LFR+OB | 100% | 18.01 | 383 |
| | LFR+OOB | 100% | 21.15 | 153 |
| | PAUC-PH+OB | 0% | 1 | - |
| | PAUC-PH+OOB | 0% | 1 | - |
| SEA | DDM-OCI+OB | 45% | 11.9 | 255 |
| | DDM-OCI+OOB | 94% | 14.1 | 301 |
| | LFR+OB | 100% | 0.73 | 35 |
| | LFR+OOB | 100% | 6.51 | 45 |
| | PAUC-PH+OB | 0% | 1 | - |
| | PAUC-PH+OOB | 0% | 1 | - |
| SEAg | DDM-OCI+OB | 92% | 15.1 | 80 |
| | DDM-OCI+OOB | 100% | 16.56 | 93 |
| | LFR+OB | 100% | 2.27 | 121 |
| | LFR+OOB | 100% | 6.3 | 324 |
| | PAUC-PH+OB | 0% | 1 | - |
| | PAUC-PH+OOB | 0% | 1.01 | - |

Table VII compares recall and G-mean of all models over the new data concept, i.e. performance over time steps 1501-3000 for data streams with an abrupt change and performance over time steps 2001-3000 for data streams with a gradual change, showing whether and how well the drift detector can help with learning after concept drift is completed. In SINE1 and SINE1g, the negative class presents to be the minority after the change; in SEA and SEAg, the positive class presents to be the minority after the change.

In terms of minority-class recall, we can see that ESOS-ELM performs the significantly best, but ESOS-ELM sacrifices majority-class recall, especially in SINE1 and SINE1g. In terms of G-mean, OOB and OOB using PAUC-PH perform the significantly best, which shows they can best balance the performance between classes. It is worth noting that PAUC-PH is the drift detection method with 0% TDR based on Table VI. It means that OOB plays the main role in learning.

TABLE VII: Performance of online learners on artificial data with $P(y)$ changes: means and standard deviations of average recall of each class and average G-mean over the new data concept. The significantly best values among all methods are shown in bold italics.

| | Method | Class+1 Recall | Class-1 Recall | G-mean |
|---|---|---|---|---|
| SINE1 | DDM-OCI+OB | 0.887±0.004 | 0.170±0.009 | 0.317±0.009 |
| | DDM-OCI+OOB | 0.979±0.007 | 0.049±0.016 | 0.188±0.033 |
| | LFR+OB | 0.870±0.004 | 0.183±0.019 | 0.334±0.022 |
| | LFR+OOB | 0.952±0.011 | 0.061±0.023 | 0.221±0.042 |
| | PAUC-PH+OB | 0.889±0.004 | 0.168±0.008 | 0.316±0.007 |
| | PAUC-PH+OOB | **0.992±0.002** | 0.692±0.013 | 0.828±0.008 |
| | RLSACP | 0.962±0.004 | 0.072±0.014 | 0.217±0.026 |
| | ESOS-ELM | 0.176±0.136 | **0.999±0.001** | 0.358±0.192 |
| | OB | 0.889±0.004 | 0.170±0.009 | 0.318±0.009 |
| | OOB | **0.992±0.002** | 0.699±0.014 | **0.832±0.008** |
| SINE1g | DDM-OCI+OB | **1.000±0.000** | 0.000±0.000 | 0.000±0.000 |
| | DDM-OCI+OOB | 0.997±0.004 | 0.008±0.005 | 0.050±0.016 |
| | LFR+OB | 0.972±0.006 | 0.031±0.027 | 0.138±0.079 |
| | LFR+OOB | 0.956±0.011 | 0.036±0.026 | 0.150±0.076 |
| | PAUC-PH+OB | **1.000±0.000** | 0.000±0.000 | 0.000±0.000 |
| | PAUC-PH+OOB | 0.989±0.001 | 0.708±0.002 | **0.835±0.002** |
| | RLSACP | **1.000±0.000** | 0.000±0.001 | 0.002±0.013 |
| | ESOS-ELM | 0.109±0.102 | **0.997±0.000** | 0.273±0.165 |
| | OB | **1.000±0.000** | 0.000±0.000 | 0.000±0.000 |
| | OOB | 0.989±0.002 | 0.709±0.002 | **0.835±0.001** |
| SEA | DDM-OCI+OB | 0.003±0.031 | **0.999±0.000** | 0.007±0.055 |
| | DDM-OCI+OOB | 0.146±0.072 | 0.965±0.013 | 0.344±0.086 |
| | LFR+OB | 0.020±0.009 | 0.996±0.001 | 0.113±0.053 |
| | LFR+OOB | 0.059±0.031 | 0.981±0.007 | 0.221±0.054 |
| | PAUC-PH+OB | 0.323±0.010 | 0.995±0.001 | 0.559±0.009 |
| | PAUC-PH+OOB | 0.514±0.015 | 0.943±0.007 | **0.688±0.010** |
| | RLSACP | 0.021±0.023 | 0.993±0.007 | 0.070±0.077 |
| | ESOS-ELM | **0.608±0.214** | 0.829±0.140 | **0.681±0.142** |
| | OB | 0.324±0.009 | 0.996±0.001 | 0.561±0.008 |
| | OOB | 0.515±0.016 | 0.945±0.006 | **0.689±0.010** |
| SEAg | DDM-OCI+OB | 0.040±0.073 | 0.998±0.001 | 0.124±0.136 |
| | DDM-OCI+OOB | 0.142±0.071 | 0.973±0.014 | 0.334±0.096 |
| | LFR+OB | 0.003±0.006 | **0.999±0.000** | 0.019±0.035 |
| | LFR+OOB | 0.076±0.084 | 0.976±0.018 | 0.217±0.123 |
| | PAUC-PH+OB | 0.365±0.029 | 0.997±0.000 | 0.600±0.023 |
| | PAUC-PH+OOB | 0.489±0.024 | 0.951±0.011 | **0.679±0.017** |
| | RLSACP | 0.002±0.006 | **0.999±0.001** | 0.011±0.035 |
| | ESOS-ELM | **0.562±0.208** | 0.809±0.143 | 0.646±0.130 |
| | OB | 0.371±0.029 | 0.997±0.001 | 0.605±0.023 |
| | OOB | 0.484±0.032 | 0.951±0.012 | **0.675±0.022** |

It also explains that OOB and OOB using PAUC-PH have very close performance. None of the other OB and OOB models show competitive recall and G-mean. Especially for those using DDM-OCI and LFR, their G-mean is significantly lower than PAUC-PH with OOB models, due to their high FA. The high number of false alarms causes too much resetting and performance loss. OOB can increase the chance of producing a false alarm, based on the observation that it led to a higher FA than OB models, because more minority-class examples join the training. This explains why G-mean from DDM-OCI and LFR is even lower in OOB models than in OB models, for the case of SINE1.

Therefore, we conclude that, for $P(y)$ type of concept drift, it is not necessary to apply any drift detection techniques that are not specifically designed for class imbalance changes; the use of these drift detectors could be even detrimental to the predictive performance due to false alarms and performance resetting; the adaptive resampling in OOB is sufficient to deal with the change and maintain the predictive performance; when using OOB with other active concept drift detectors, the number of false alarms and performance resetting need to be carefully considered.

### C.2. $\mathbf{p}(\mathbf{x} \mid y)$ Concept Drift

The data streams in this section only involve $p(\mathbf{x} \mid y)$ type of concept drift, without $P(y)$ and $P(y \mid \mathbf{x})$ changes. The class imbalance ratio is fixed to 1:9 and we let the positive class be the minority, so that the data stream is constantly imbalanced. The concept drift in each data stream is controlled by $p(\mathbf{x})$ of the negative class, as shown in Table VIII. $P(x_1)$ is changed linearly during the concept transition period in the gradual drifting cases.

Table IX compares the detection performance of the three active concept drift detectors. Similar to our previous results, DDM-OCI and LFR are more sensitive to $P(x \mid y)$ changes than PAUC-PH. When DDM-OCI and LFR work with OOB, their TDR shows 100%; and LFR has higher FA and shorter DOD than DDM-OCI, due to more indicators it monitors. PAUC-PH shows 0% TDR in most cases of working with both OB and OOB. Different from $P(y)$ changes, when DDM-OCI and LFR work with OB, their TDR is rather low, which suggests that their sensitivity is dependent on the class imbalance techniques. To explain this, we observe OB's recall of each class over time. Unlike the cases with class imbalance changes, where it is possible for the minority-class examples to become more frequent, the data streams generated in this section have a fixed minority class with a constantly small prior probability. The minority-class recall remains low (e.g. 0 in SINE1 and SINE1g cases) due to the imbalanced distribution. These detectors cannot detect any concept drift, because the classification performance they monitored does not change significantly. In other words, the classification difficulty indirectly affects the detection sensitivity of DDM-OCI and LFR. When oversampling is applied, which introduces more training examples for the minority class, the performance metrics (G-mean, recall and precision) monitored by DDM-OCI and LFR can be substantially improved. It also increases the possibility of reporting a concept drift. This explains the low detection rate of DDM-OCI and LFR when working with OB and their high detection rate when working with OOB.

Table X compares recall and G-mean of all models over the new data concept. As we expected, almost all OB models show significantly worse minority-class recall and G-mean. On SINE1 and SINE1g data, minority-class recall of OB models is as low as 0, which may hinder the detection of any concept drift (as we observed in Table IX). Among the OOB models, those using DDM-OCI and LFR perform significantly worse than OOB using PAUC-PH and OOB itself, and the latter two show very close performance. This is because DDM-OCI and LFR trigger concept drift with false alarms, and cause model resetting multiple times. Along with the resetting, the useful and valid information learnt in the past is forgotten at the same time. For the two passive models, RLSACP and ESOS-ELM did not perform very well compared to OOB, showing significantly lower minority-class recall and G-mean in Table X. Generally speaking, for imbalanced data streams with $p(\mathbf{x} \mid y)$ changes, class imbalance seems to be a more important issue than concept drift, considering that the learning model without triggering any concept drift detection

TABLE VIII: Artificial data streams with $p(\mathbf{x} \mid y)$ concept drift.

| ID | Data | Speed | Class +1 | | Class -1 | |
|---|---|---|---|---|---|---|
| | | | Old concept | New concept | Old concept | New concept |
| 1 | SINE1 | Abrupt | Points below $x_2 = \sin(x_1)$ | Points below $x_2 = \sin(x_1)$ | Points above or on $x_2 = \sin(x_1)$ and $P(x_1 < 0.5) = 0.9$ | Points above or on $x_2 = \sin(x_1)$ and $P(x_1 < 0.5) = 0.1$ |
| 2 | SINE1g | Gradual | | | | |
| 3 | SEA | Abrupt | $x_1 + x_2 \leq 7$ | $x_1 + x_2 \leq 7$ | $x_1 + x_2 > 7$ and $P(x_1 < 5) = 0.9$ | $x_1 + x_2 > 7$ and $P(x_1 < 5) = 0.1$ |
| 4 | SEAg | Gradual | | | | |

TABLE IX: Performance of the 3 active concept drift detectors on artificial data with $p(\mathbf{x} \mid y)$ changes: TDR, FA and DoD. The '-' symbol indicates that no concept drift is detected.

| | Method | TDR | FA | DoD |
|---|---|---|---|---|
| SINE1 | DDM-OCI+OB | 0% | 0 | - |
| | DDM-OCI+OOB | 100% | 1.25 | 594 |
| | LFR+OB | 0% | 0.05 | - |
| | LFR+OOB | 100% | 3.99 | 528 |
| | PAUC-PH+OB | 4% | 0.45 | 232 |
| | PAUC-PH+OOB | 0% | 0.45 | - |
| SINE1g | DDM-OCI+OB | 0% | 0 | - |
| | DDM-OCI+OOB | 100% | 1.37 | 387 |
| | LFR+OB | 0% | 0 | - |
| | LFR+OOB | 100% | 5.45 | 258 |
| | PAUC-PH+OB | 0% | 1.04 | - |
| | PAUC-PH+OOB | 0% | 1 | - |
| SEA | DDM-OCI+OB | 16% | 1 | 1394 |
| | DDM-OCI+OOB | 100% | 4.03 | 473 |
| | LFR+OB | 100% | 0.31 | 52 |
| | LFR+OOB | 100% | 13.48 | 59 |
| | PAUC-PH+OB | 0% | 0 | - |
| | PAUC-PH+OOB | 0% | 0.85 | - |
| SEAg | DDM-OCI+OB | 90% | 0.15 | 238 |
| | DDM-OCI+OOB | 100% | 4.03 | 279 |
| | LFR+OB | 29% | 0 | 1154 |
| | LFR+OOB | 100% | 12.75 | 196 |
| | PAUC-PH+OB | 0% | 1 | - |
| | PAUC-PH+OOB | 0% | 1 | - |

achieves the best performance. Besides, while the adopted class imbalance technique can improve the final prediction, it can also improve the performance of active concept drift detection methods, depending on their working mechanism.

### C.3. $\mathbf{P}(y \mid \mathbf{x})$ Concept Drift

The data streams in this section only involve $P(y \mid \mathbf{x})$ type of concept drift, without $P(y)$ and $p(\mathbf{x} \mid y)$ changes. Following the settings in Section IV-C.2, we fix the class imbalance ratio to 1:9 and let the positive class be the minority, so that the data stream is constantly imbalanced. As shown in Table XI, the data distribution in SINE1 and SINE1g involves a concept swap, and this change occurs probabilistically in SINE1g; the data distribution in SEA and SEAg has a concept threshold moving, and this change occurs continuously in SEAg. The change in SEA and SEAg is less severe than the change in SINE1 and SINE1g, because some of the examples from the old concept are still valid under the new concept after the threshold moves completely. The concept drift discussed in this section belongs to the real concept drift category, which affects the classification boundary and is expected to be captured by all concept drift detectors.

According to Table XII, we can see that DDM-OCI and LFR have difficulty in detecting the concept drift when working with OB, because of the poor recall and G-mean produced by OB, which is also observed and explained in Section IV-C.2.

TABLE X: Performance of online learners on artificial data with $p(\mathbf{x} \mid y)$ changes: means and standard deviations of average recall of each class and average G-mean over the new data concept. The significantly best values among all methods are shown in bold italics.

| | Method | Class+1 Recall | Class-1 Recall | G-mean |
|---|---|---|---|---|
| SINE1 | DDM-OCI+OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | DDM-OCI+OOB | 0.036±0.025 | 0.997±0.002 | 0.145±0.052 |
| | LFR+OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | LFR+OOB | 0.061±0.036 | 0.994±0.005 | 0.200±0.066 |
| | PAUC-PH+OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | PAUC-PH+OOB | ***0.689±0.038*** | 0.985±0.004 | ***0.811±0.027*** |
| | RLSACP | 0.090±0.028 | 0.939±0.012 | 0.251±0.045 |
| | ESOS-ELM | 0.058±0.122 | ***1.000±0.000*** | 0.113±0.208 |
| | OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | OOB | ***0.696±0.020*** | 0.985±0.004 | ***0.817±0.013*** |
| SINE1g | DDM-OCI+OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | DDM-OCI+OOB | 0.035±0.064 | 0.993±0.006 | 0.096±0.135 |
| | LFR+OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | LFR+OOB | 0.038±0.062 | 0.992±0.008 | 0.111±0.132 |
| | PAUC-PH+OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | PAUC-PH+OOB | ***0.801±0.032*** | 0.988±0.003 | ***0.884±0.019*** |
| | RLSACP | 0.072±0.049 | 0.952±0.009 | 0.173±0.102 |
| | ESOS-ELM | 0.077±0.112 | 0.991±0.035 | 0.162±0.215 |
| | OB | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.000 |
| | OOB | ***0.802±0.034*** | 0.988±0.003 | ***0.884±0.021*** |
| SEA | DDM-OCI+OB | 0.001±0.000 | ***0.999±0.000*** | 0.002±0.006 |
| | DDM-OCI+OOB | 0.144±0.027 | 0.973±0.007 | 0.332±0.040 |
| | LFR+OB | 0.036±0.012 | 0.984±0.005 | 0.144±0.048 |
| | LFR+OOB | 0.085±0.039 | 0.971±0.015 | 0.243±0.069 |
| | PAUC-PH+OB | 0.130±0.027 | 0.983±0.004 | 0.341±0.042 |
| | PAUC-PH+OOB | 0.459±0.044 | 0.923±0.010 | 0.645±0.030 |
| | RLSACP | 0.000±0.001 | ***0.999±0.001*** | 0.001±0.006 |
| | ESOS-ELM | 0.202±0.158 | 0.967±0.071 | 0.394±0.167 |
| | OB | 0.130±0.027 | 0.983±0.004 | 0.341±0.042 |
| | OOB | ***0.477±0.031*** | 0.919±0.010 | ***0.657±0.021*** |
| SEAg | DDM-OCI+OB | 0.002±0.007 | ***1.000±0.000*** | 0.010±0.035 |
| | DDM-OCI+OOB | 0.100±0.040 | 0.978±0.008 | 0.257±0.066 |
| | LFR+OB | 0.101±0.027 | 0.999±0.000 | 0.269±0.058 |
| | LFR+OOB | 0.050±0.029 | 0.980±0.011 | 0.182±0.065 |
| | PAUC-PH+OB | 0.107±0.025 | 0.999±0.000 | 0.278±0.046 |
| | PAUC-PH+OOB | ***0.348±0.023*** | 0.939±0.017 | ***0.553±0.019*** |
| | RLSACP | 0.000±0.000 | ***1.000±0.000*** | 0.000±0.002 |
| | ESOS-ELM | 0.183±0.137 | 0.964±0.090 | 0.368±0.161 |
| | OB | 0.106±0.021 | 0.999±0.000 | 0.279±0.040 |
| | OOB | ***0.345±0.027*** | 0.943±0.018 | ***0.552±0.022*** |

When DDM-OCI and LFR work with OOB, their detection rate TDR is greatly improved (above 90% in most cases). This is because the improved performance metrics facilitate the detection. LFR is more sensitive to the change, which produces higher FA and shorter DoD. Different from previous observations in terms of concept drift detection performance, PAUC-PH working with OB produces 100% TDR and low FA on data streams SINE1 and SINE1g, but PAUC-PH does not work well with OOB on the same data. It is interesting to see that oversampling does not always play a positive role in drift detection. One possible reason is that oversampling sometimes lessens the performance reduction caused by the real concept drift, while it tries to maintain or improve the

TABLE XI: Artificial data streams with $P(y \mid \mathbf{x})$ concept drift.

| ID | Data | Speed | Class +1 | | Class -1 | |
|---|---|---|---|---|---|---|
| | | | Old concept | New concept | Old concept | New concept |
| 1 | SINE1 | Abrupt | Points below $x_2 = \sin(x_1)$ | Points above/on $x_2 = \sin(x_1)$ | Points above/on $x_2 = \sin(x_1)$ | Points below $x_2 = \sin(x_1)$ |
| 2 | SINE1g | Gradual | | | | |
| 3 | SEA | Abrupt | $x_1 + x_2 \leq 7$ | $x_1 + x_2 \leq 13$ | $x_1 + x_2 > 7$ | $x_1 + x_2 > 13$ |
| 4 | SEAg | Gradual | | | | |

overall predictive performance, especially for AUC type of metrics in our case. There is evidence, showing that AUC is a more stable metric than G-mean [104], as it is computed by altering a threshold value for labeling data samples [105]. When classification is significantly improved by oversampling, we observe in the experiment that PAUC in PAUC-PH is less affected by the concept drift than the monitored indicators in DDM-OCI and LFR, thus leading to a smaller TDR. On data streams SEA and SEAg, PAUC-PH does not report any concept drift, probably due to the less severe concept drift.

TABLE XII: Performance of the 3 active concept drift detectors on artificial data with $P(y \mid \mathbf{x})$ changes: TDR, FA and DoD. The '-' symbol indicates that no concept drift is detected.

| | Method | TDR | FA | DoD |
|---|---|---|---|---|
| SINE1 | DDM-OCI+OB | 0% | 0 | - |
| | DDM-OCI+OOB | 97% | 1.02 | 1166 |
| | LFR+OB | 0% | 0 | - |
| | LFR+OOB | 91% | 3.92 | 783 |
| | PAUC-PH+OB | 100% | 1.03 | 884 |
| | PAUC-PH+OOB | 2% | 1.28 | 1180 |
| SINE1g | DDM-OCI+OB | 0% | 0 | - |
| | DDM-OCI+OOB | 69% | 2.16 | 165 |
| | LFR+OB | 0% | 1 | - |
| | LFR+OOB | 85% | 6.21 | 306 |
| | PAUC-PH+OB | 100% | 1.03 | 1119 |
| | PAUC-PH+OOB | 0% | 1 | - |
| SEA | DDM-OCI+OB | 61% | 0.39 | 23 |
| | DDM-OCI+OOB | 100% | 3.87 | 151 |
| | LFR+OB | 10% | 0.02 | 865 |
| | LFR+OOB | 100% | 13.73 | 65 |
| | PAUC-PH+OB | 0% | 1 | - |
| | PAUC-PH+OOB | 0% | 1 | - |
| SEAg | DDM-OCI+OB | 100% | 0 | 71 |
| | DDM-OCI+OOB | 100% | 3.9 | 342 |
| | LFR+OB | 3% | 0.02 | 1036 |
| | LFR+OOB | 100% | 13.59 | 123 |
| | PAUC-PH+OB | 0% | 1 | - |
| | PAUC-PH+OOB | 0% | 1 | - |

The recall and G-mean over the new data concept in Table XIII further confirms the above analysis. The OB models produce very low minority-class recall and thus low G-mean. RLSACP and ESOS-ELM did not perform well on the new data concept either. By comparing the models that captures concept drifts (DDM-OCI+OOB, LFR+OOB, PAUC-PH+OB) and the models without reporting any concept drift (PAUC-PH+OOB, OOB), it seems that class imbalance causes a more difficult learning issue than the real concept drift in our cases. The models solely tackling class imbalance produce the significantly best recall and G-mean. The rather low imbalance ratio (i.e. 1:9) could be a reason. It would be worth discussing various imbalance levels in data with concept drift in our future work, in order to find out when it is worthwhile considering concept drift in imbalanced data streams. By comparing the results in Table XIII, Table X and Table VII, the $P(y \mid \mathbf{x})$ type of concept drift indeed leads to the most performance reduction. It is consistent with our understanding that the real concept drift is the most radical type of change in data. However, existing approaches do not seem to tackle it well when data streams are very imbalanced. To develop better concept drift detection methods, the key issues here include how to best have them and class imbalance techniques work together and how to tackle the performance loss brought by false alarms.

TABLE XIII: Performance of online learners on artificial data with $P(y \mid \mathbf{x})$ changes: means and standard deviations of average recall of each class and average G-mean over the new data concept. The significantly best values among all methods are shown in bold italics.

| | Method | Class+1 Recall | Class-1 Recall | G-mean |
|---|---|---|---|---|
| SINE1 | DDM-OCI+OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | DDM-OCI+OOB | 0.004±0.003 | 0.998±0.002 | 0.030±0.016 |
| | LFR+OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | LFR+OOB | 0.013±0.010 | 0.996±0.006 | 0.062±0.036 |
| | PAUC-PH+OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | PAUC-PH+OOB | **0.031±0.013** | 0.941±0.009 | **0.098±0.026** |
| | RLSACP | 0.000±0.001 | **0.999±0.001** | 0.003±0.010 |
| | ESOS-ELM | 0.000±0.000 | 0.997±0.003 | 0.000±0.000 |
| | OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | OOB | **0.033±0.012** | 0.942±0.009 | **0.102±0.022** |
| SINE1g | DDM-OCI+OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | DDM-OCI+OOB | 0.014±0.017 | 0.993±0.006 | 0.069±0.074 |
| | LFR+OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | LFR+OOB | 0.019±0.018 | 0.993±0.006 | 0.086±0.077 |
| | PAUC-PH+OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | PAUC-PH+OOB | **0.031±0.011** | 0.993±0.002 | **0.103±0.026** |
| | RLSACP | 0.000±0.001 | **1.000±0.000** | 0.001±0.008 |
| | ESOS-ELM | 0.000±0.000 | 0.907±0.140 | 0.000±0.000 |
| | OB | 0.000±0.000 | **1.000±0.000** | 0.000±0.000 |
| | OOB | 0.027±0.010 | 0.995±0.002 | 0.093±0.028 |
| SEA | DDM-OCI+OB | 0.013±0.022 | **0.999±0.001** | 0.050±0.085 |
| | DDM-OCI+OOB | 0.110±0.031 | 0.968±0.008 | 0.311±0.057 |
| | LFR+OB | 0.149±0.025 | **0.999±0.000** | 0.378±0.036 |
| | LFR+OOB | 0.031±0.022 | 0.964±0.016 | 0.144±0.071 |
| | PAUC-PH+OB | 0.153±0.023 | **0.999±0.000** | 0.384±0.031 |
| | PAUC-PH+OOB | **0.292±0.017** | 0.967±0.008 | **0.530±0.015** |
| | RLSACP | 0.013±0.013 | 0.995±0.001 | 0.072±0.063 |
| | ESOS-ELM | 0.065±0.068 | 0.997±0.022 | 0.222±0.106 |
| | OB | 0.152±0.023 | **0.999±0.000** | 0.383±0.032 |
| | OOB | 0.287±0.014 | 0.966±0.008 | 0.525±0.012 |
| SEAg | DDM-OCI+OB | 0.000±0.002 | **1.000±0.000** | 0.001±0.013 |
| | DDM-OCI+OOB | 0.042±0.022 | 0.988±0.006 | 0.163±0.059 |
| | LFR+OB | 0.145±0.032 | 0.999±0.000 | 0.356±0.066 |
| | LFR+OOB | 0.024±0.018 | 0.985±0.006 | 0.112±0.065 |
| | PAUC-PH+OB | 0.152±0.019 | 0.999±0.000 | 0.370±0.027 |
| | PAUC-PH+OOB | **0.288±0.034** | 0.974±0.010 | **0.512±0.036** |
| | RLSACP | 0.009±0.018 | **1.000±0.000** | 0.043±0.077 |
| | ESOS-ELM | 0.138±0.088 | 0.993±0.057 | 0.336±0.106 |
| | OB | 0.149±0.025 | 0.999±0.000 | 0.364±0.042 |
| | OOB | **0.282±0.032** | 0.974±0.008 | **0.506±0.034** |

### C.4. Analysis under Different Imbalance Ratios

The results so far have shown that the imbalance ratio is a crucial factor affecting concept drift detection and final classification performance. When discussing $p(\mathbf{x} \mid y)$ and $P(y \mid \mathbf{x})$

types of concept drift, we fixed the imbalance ratio to 1:9. To generalize our observations, we vary the imbalance levels in this section. We aim to find out whether and how the role of class imbalance changes and when it is worthwhile considering concept drift in imbalanced data streams. We compare DDM-OCI and PAUC-PH working with OB and OOB on SINE1 data with a different imbalance ratio (IR = 1:9, 2:8 and 3:7). Their drift detection performance (TDR, FA and DoD) and classification performance (G-mean) in the cases of IR equal to 2:8 and 3:7 are shown in Table XIV.

TABLE XIV: Drift Detection Performance and G-mean of DDM-OCI and PAUC-PH working with OB and OOB on SINE1 data with IR of 2:8 and 3:7 and $p(\mathbf{x} \mid y)$ and $P(y \mid \mathbf{x})$ drift. The '-' symbol indicates that no concept drift is detected.

| Data | Method | TDR | FA | DoD | G-mean |
|---|---|---|---|---|---|
| IR=2:8, $p(\mathbf{x} \mid y)$ | DDM-OCI+OB | 99% | 0.01 | 386 | 0.120±0.028 |
| | DDM-OCI+OOB | 100% | 6.91 | 295 | 0.164±0.040 |
| | PAUC-PH+OB | 0% | 1.41 | - | 0.444±0.015 |
| | PAUC-PH+OOB | 0% | 1.6 | - | 0.851±0.041 |
| IR=3:7, $p(\mathbf{x} \mid y)$ | DDM-OCI+OB | 100% | 1.02 | 963 | 0.031±0.009 |
| | DDM-OCI+OOB | 100% | 9.56 | 129 | 0.214±0.028 |
| | PAUC-PH+OB | 0% | 1 | - | 0.817±0.006 |
| | PAUC-PH+OOB | 0% | 1 | - | 0.885±0.003 |
| IR=2:8, $P(y \mid \mathbf{x})$ | DDM-OCI+OB | 0% | 0 | - | 0.000±0.000 |
| | DDM-OCI+OOB | 100% | 6.98 | 198 | 0.209±0.054 |
| | PAUC-PH+OB | 100% | 0.39 | 906 | 0.000±0.000 |
| | PAUC-PH+OOB | 0% | 0.67 | - | 0.309±0.015 |
| IR=3:7, $P(y \mid \mathbf{x})$ | DDM-OCI+OB | 86% | 1.1 | 1276 | 0.009±0.007 |
| | DDM-OCI+OOB | 100% | 8.5 | 310 | 0.215±0.040 |
| | PAUC-PH+OB | 100% | 1 | 952 | 0.000±0.000 |
| | PAUC-PH+OOB | 0% | 1 | - | 0.292±0.020 |

By comparing the results from the three data streams with a $p(\mathbf{x} \mid y)$ drift at different imbalance levels, we can see that drift detection gets easier (i.e. a higher TDR) as the data stream becomes less imbalanced for the OB models using DDM-OCI. It confirms our previous conclusion that the imbalance ratio affects the drift detection sensitivity. Meanwhile, FA is increased as more minority-class examples join the learning process. The TDR of PAUC-PH remains 0, regardless of the imbalance ratio. This is because of the insensitivity of AUC type of metrics to the class distribution, as explained in Section II-B.1. Similar to the results in Section IV-C.2, oversampling facilitates the drift detection of DDM-OCI, and improves G-mean on the new data concept of both DDM-OCI and PAUC-PH. The model resetting from DDM-OCI causes performance loss, so that PAUC-PH working with OOB performs the best.

For the cases with a $P(y \mid \mathbf{x})$ drift, we obtain similar observations in Table XIV compared to the results in Section IV-C.3: oversampling and a less imbalanced distribution improve TDR of DDM-OCI, but also increases its FA; PAUC-PH works better with OB than with OOB in terms of the drift detection performance, which further confirms our previous analysis; the OOB model using PAUC-PH presents the best G-mean.

### D. Comparative Study on Real-World Data

After the detailed analysis of the three types of concept drift, we now look into the performance of the above learning models on the three real-world data sets (PAKDD [101], Weather [76] and Tweet [102]) described in Section IV-A. Based on the experimental results on the artificial data, we focus on the best active (PAUC+OOB) and the best passive concept drift detection methods (ESOS-ELM) here for a clear observation, in comparison with OOB. The three methods use the same parameter settings as before. The initialisation and validation data required by ESOS-ELM is the first 2% examples of each data set.

Without knowing the true concept drifts in real-world data, we calculate and track the time-decayed G-mean by setting the decay factor to 0.995, which means that the old performance is forgotten at the rate of 0.5%. All the compared metrics are the average of 100 runs in the following figures.

Fig. 3 presents the time-decayed G-mean curves from OOB, PAUC-PH+OOB and ESOS-ELM on the three real-world data sets. The average number of reported drift by PAUC-PH is 1, 3 and 1 on Weather, PAKDD and Tweet data respectively. Compared to the artificial cases, we obtain some similar results: the passive approach ESOS-ELM does not perform as well as the other two methods; OOB and PAUC-PH show very close G-mean over time on Weather and PAKDD data, which suggests the importance of tackling class imbalance adaptively.

In the PAKDD plot, we can see that the G-mean level is relatively stable without significant drop; differently, G-mean in the Tweet plot is reducing. It may suggest that the concept drift in PAKDD is less significant or influential than that in Tweet. Compared to the gradual market and environment change in PAKDD, the tweet topic change can be much faster and more noticeable. Therefore, although PAUC-PH detects 3 concept drifts in PAKDD, the two methods, OOB and PAUC-PH+OOB, does not show much difference. In tweet, PAUC-PH+OOB presents better G-mean than using OOB alone, suggesting the positive effect of the active concept drift detector in fast changing data streams.

### E. Further Discussions

In this section, we summarize and further discuss the results in the above comparative study on the artificial and real-world data. We also answer the research questions proposed at the beginning of this paper. When dealing with imbalanced data streams with concept drift, we have obtained the following:

- When both class imbalance and concept drift exist, class imbalance status and class imbalance changes (i.e. $P(y)$ changes) are shown to be more crucial issues than the traditional concept drift (i.e. $p(\mathbf{x} \mid y)$ and $P(y \mid \mathbf{x})$ changes) in terms of the online prediction performance. It is necessary to adopt adaptive class imbalance techniques (e.g. OOB discussed in our experiment), in addition to using concept drift detection methods alone (e.g. DDM-OCI, LFR). Most existing papers that proposed new concept drift detection methods for imbalanced data so far did not consider the effect of class imbalance techniques on final prediction and concept drift detection.
- $P(y \mid \mathbf{x})$ concept drift (i.e. real concept drift) is the most severe type of change in data, compared to $p(\mathbf{x} \mid y)$ and $P(y)$ concept drift. This is based on the observation on

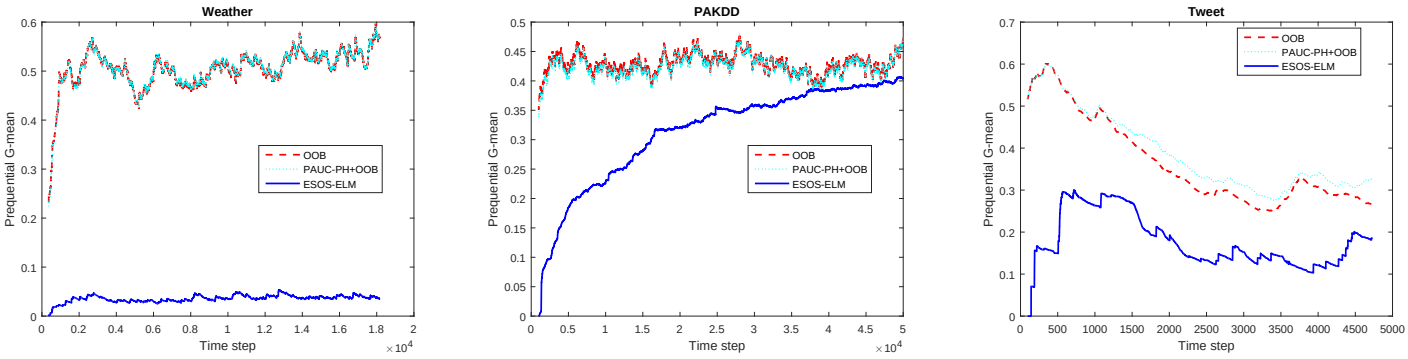



Fig. 3: Time-decayed G-mean curves (decay factor = 0.995) from OOB, PAUC-PH+OOB and ESOS-ELM on real-world data.

the final prediction performance. For all three types of concept drift, existing concept drift approaches do not show much benefit in performance improvement. Concept drift is hard to be detected when no class imbalance technique is applied. Their drift detection performance is affected by the class imbalance technique, depending on their detection mechanism.

- For $P(y)$ concept drift, it is not necessary to apply any concept drift detection methods that are not designed for class imbalance changes, due to their false alarms and model resetting. It is crucial to detect and handle the class imbalance change in time.
- From an intrinsic perspective, $P(y)$ and $p(\mathbf{x} \mid y)$ types of concept drift do not change decision boundaries, which means that the online model is still valid or partially valid. Using an appropriate class imbalance technique alone is thus expected to improve final performance effectively. $P(y \mid \mathbf{x})$ concept drift, on the other hand, affects the true decision boundary of the problem. Although those active drift detectors are designed for this type of changes, the presence of class imbalance causes poor classification performance and increases the difficulty in detecting the drift. The application of class imbalance techniques can improve the prediction performance, and indirectly facilitate drift detection.
- From the results on real-world data, we see that the effectiveness of traditional concept drift detectors (e.g. PAUC-PH) depends on the type of concept drift. For fast and significant concept drift, applying PAUC-PH seems to be more beneficial to the prediction performance.
- Among existing methods designed for imbalanced data with concept drift (4 active methods and 2 passive methods), the passive methods (i.e. ESOS-ELM and RLSACP) do not perform well in general. Although they contain both class imbalance and concept drift techniques, firstly, their class imbalance technique is not effectively adaptive to class imbalance changes, so that wrong imbalance status could be used during learning, leading to poor performance in the cases with $P(y)$ concept drift. Secondly, they are restricted to the use of certain perceptron-based classifiers, so that the disadvantages of the classifiers are also inherited by the online model. For example, OS-ELM in ESOS-ELM requires initialisation and validation data sets for training, and the weighted OS-ELM was found to over-emphasize the minority class and present large performance variance sometimes in earlier studies [13]. Thirdly, RLSACP is a single-model approach, which might be less accurate than ensemble approaches with multiple models [55].
- Among the three active methods discussed in this work, which are DDM-OCI, LFR and PAUC-PH, DDM-OCI and LFR are more sensitive to concept drift than PAUC-PH, with a higher detection rate but also higher false alarms. In addition, the detection performance of DDM-OCI and LFR can be greatly improved by OOB. The explanation can be found in the previous analysis.
- For the three active drift detectors, model resetting is triggered if a drift alert is issued. However, this is not the most appropriate technique for $P(y)$ and $p(\mathbf{x} \mid y)$ types of concept drift, because the decision boundary is not affected, and the old data concept is still useful. Other ways to handle $P(y)$ and $p(\mathbf{x} \mid y)$ types of concept drift should be investigated. Moreover, if the detector suffers from a high number of false alarms, the performance of the online model can be greatly reduced further. It can be observed in Tables VII and X. Therefore, it is important to control the number of false alarms and/or to adopt techniques to mitigate their negative effects.
- All the drift detectors discussed in this section detect concept drift based on classification performance. This might explain why their drift detection performance depends greatly on the class imbalance technique and the online learner. It is worth developing other types of drift detection methods and exploring how they work with class imbalance techniques for better classification in the future.

Overall, all these results suggest us that class imbalance and concept drift need to be studied simultaneously, when we design an algorithm to deal with imbalanced data with concept drift. Their mutual effect must be taken into consideration. Hence, we propose the following key issues to be considered for an effective algorithm:

- Is the class imbalance technique effective in predicting minority-class examples?
- Is the class imbalance technique adaptive to class imbalance changes?

- Is the concept drift technique effective in detecting different types of concept drift, in terms of detection rate, false alarms and detection promptness? Which type of concept drift is it designed for? Which type of concept drift does it perform better?
- Is the detection performance of the concept drift technique affected by the class imbalance technique? And how?
- How can we have the class imbalance technique and concept drift technique work together, to achieve better detection rate, fewer false alarms, less detection delay or better online prediction?

## V. Conclusion

This paper gives the first systematic study of handling concept drift in class-imbalanced data streams. In the context of online learning, we provide a thorough review and an experimental study of this problem.

First, a comprehensive review is given, including the problem description and definitions, the individual learning issues and solutions in class imbalance and concept drift, the combined challenges and existing solutions in online class imbalance learning with concept drift, and example applications. The review reveals research gaps in the field of online class imbalance learning with concept drift.

Second, to fill in these research gaps, we carry out a thorough empirical study by looking into the following research questions: 1) what are the challenges in detecting each type of concept drift when the data stream is imbalanced? 2) Among the proposed methods designed for online class imbalance learning with concept drift, which one performs better for which type of concept drift? 3) Would applying class imbalance techniques facilitate the concept drift detection and online prediction?

For the first research question, a $P(y)$ change can be easily tackled by an adaptive class imbalance technique (e.g. OOB used in this paper). The traditional concept drift detectors, such as LFR, DDM-OCI and PAUC-PH, do not perform well in detecting a $p(\mathbf{x} \mid y)$ change. The prediction performance on an imbalanced data stream with $p(\mathbf{x} \mid y)$ changes can be effectively improved by solely using an adaptive class imbalance technique. A $P(y \mid \mathbf{x})$ change is the most challenging case for learning, where the traditional active and passive concept drift detection methods do not bring much performance improvement. Class imbalance is shown to be a more crucial issue in terms of final prediction performance.

For the second research question, the two passive methods, RLSACP and ESOS-ELM, do not perform well in general. DDM-OCI and LFR are sensitive to different types of concept drift, with a high detection rate but also high false alarms. PAUC-PH is more conservative in terms of drift detection. Based on the observation on minority-class recall and G-mean, the combination of PAUC-PH and OOB was shown to be the best approach among all.

For the third research question, it is necessary to apply adaptive class imbalance techniques when learning from imbalanced data streams with concept drift – they bring the most prediction performance improvement. In our experiment, OOB facilitates the concept drift detection of DDM-OCI and LFR.

Finally, this paper points out several important issues for future algorithm design. There are still many challenges and learning issues in this field that are worth of ongoing research, such as more effective concept drift detection methods for imbalanced data streams, studying the mutual effect of class imbalance and concept drift, and more real-world applications with different types of class imbalance and concept drift.

## Acknowledgment

This work was supported by EPSRC (Grant Nos. EP/K001523/1 and EP/J017515/1) and the National Natural Science Foundation of China (Grant No. 61329302). Xin Yao was also supported by a Royal Society Wolfson Research Merit Award.

## References

[1] M. R. Sousa, J. Gama, and E. Brandão, "A new dynamic modeling framework for credit risk assessment," *Expert Systems with Applications*, vol. 45, p. 341351, 2016.

[2] 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.

[3] 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, no. 4, pp. 1 340 001(1–19), 2013.

[4] 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*, vol. 28, no. 6, pp. 1532 – 1545, 2016.

[5] 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.

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

[7] T. R. Hoens, R. Polikar, and N. V. Chawla, "Learning from streaming data with concept drift and imbalance: an overview," *Progress in Artificial Intelligence*, vol. 1, no. 1, pp. 89–101, 2012.

[8] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014.

[9] 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.

[10] H. Wang and Z. Abraham, "Concept drift detection for streaming data," in *International Joint Conference of Neural Networks*, 2015, pp. 1–9.

[11] D. Brzezinski and J. Stefanowski, "Prequential auc for classifier evaluation and drift detection in evolving data streams," *New Frontiers in Mining Complex Patterns*, vol. 8983, pp. 87–101, 2015.

[12] ——, "Prequential AUC: properties of the area under the roc curve for data streams with concept drift," *Knowledge and Information Systems*, vol. 52, no. 2, p. 531–562, 2017.

[13] S. Wang, L. L. Minku, and X. Yao, "Resampling-based ensemble methods for online class imbalance learning," *IEEE Transactions on Knowledge and Data Engineering*, no. 5, pp. 1356 – 1368, 2015.

[14] 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.

[15] B. Mirza, Z. Lin, and N. Liu, "Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift," *Neurocomputing*, vol. 149, pp. 316–329, 2015.

[16] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in non-stationary environments: A survey," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 12 – 25, 2015.

[17] 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.

[18] R. Polikar, L. Udpa, S. S. Udpa, and V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 31, no. 4, pp. 497 – 508, 2001.

[19] S. Grossber, "Nonlinear neural networks: Principles, mechanisms, and architectures," *Neural Networks*, vol. 1, no. 1, p. 1761, 1988.

[20] 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.

[21] 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*, 2008, pp. 3261–3267.

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

[23] C. Monteiro, R. Bessa, V. Miranda, A. Botterud, J. Wang, and G. Conzelmann, "Wind power forecasting : state-of-the-art 2009," *Technical Report (ANL/DIS-10-1), Argonne National Laboratory (ANL)*, 2009.

[24] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine Learning*, vol. 30, no. 2-3, pp. 195–215, 1998.

[25] S. Visa and A. Ralescu, "Issues in mining imbalanced data sets - a review paper," in *Proceedings of the Sixteen Midwest Artificial Intelligence and Cognitive Science Conference*, 2005, pp. 67–73.

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

[27] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *Special Issue on Learning from Imbalanced Datasets, Sigkdd Explorations*, vol. 6, no. 1, pp. 20–29, 2004.

[28] J. Zhang and I. Mani, "kNN approach to unbalanced data distributions: A case study involving information extraction," in *Workshop on Learning from Imbalanced Datasets II, ICML*, 2003, pp. 42–48.

[29] R. Yan, Y. Liu, R. Jin, and A. Hauptmann, "On predicting rare classes with svm ensembles in scene classification," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2003, pp. III – 21–4 vol.3.

[30] G. Wu and E. Y. Chang, "Class-boundary alignment for imbalanced dataset learning," in *Workshop on Learning from Imbalanced Datasets II, ICML*, 2003, pp. 49–56.

[31] 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.

[32] V. Lopez, A. Fernandez, S. Garcia, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information Sciences*, vol. 250, pp. 113–141, 2013.

[33] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 341–378, 2002.

[34] T. R. Hoens, "Living in an imbalanced world," Ph.D. dissertation, Graduate School of the University of Notre Dame, 2012.

[35] G. E. Batista, R. C. Prati, and M. C. Monard, "Balancing strategies and class overlapping," *Advances in Intelligent Data Analysis*, vol. 3646, pp. 24–35, 2005.

[36] 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.

[37] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," in *ACM SIGKDD Explorations Newsletter*, vol. 6, 2004, pp. 40–49.

[38] N. Japkowicz, "Class imbalances: are we focusing on the right issue," in *Workshop on Learning from Imbalanced Data Sets II, ICML*, 2003, pp. 17–23.

[39] 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.

[40] ——, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *Journal of Intelligent Information Systems*, vol. 46, no. 3, p. 563–597, 2016.

[41] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: A new over-sampling method in imbalanced data sets learning," in *International Conference on Intelligent Computing (ICIC)*, 2005, pp. 878–887.

[42] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008, pp. 1322–1328.

[43] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE - majority weighted minority oversampling technique for imbalanced data set learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 2, pp. 405–425, 2014, an oversampling method by generating new examples.

[44] I. Tomek, "Two modifications of cnn," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 6, no. 11, pp. 769–772, 1976.

[45] M. Kubat, R. Holte, and S. Matwin, "Learning when negative examples abound," in *9th European Conference on Machine Learning Prague*, vol. 1224, 1997, pp. 146–153.

[46] L. Jorma, "Improving identification of difficult small classes by balancing class distribution," in *8th Conference on Artificial Intelligence in Medicine in Europe, AIME 2001*, vol. 2101, 2001, pp. 63–66.

[47] M. Hao, Y. Wang, and S. H. Bryant, "An efficient algorithm coupled with synthetic minority over-sampling technique to classify imbalanced pubchem bioassay data," *Analytica Chimica Acta*, vol. 806, no. 2, p. 117127, 2014.

[48] A. Estabrooks, T. Jo, and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," *Computational Intelligence*, vol. 20, no. 1, pp. 18–36, 2004.

[49] J. A. Sáez, B. Krawczyk, and M. Woźniak, "Analyzing the over-sampling of different classes and types of examples in multi-class imbalanced datasets," *Pattern Recognition*, vol. (In press), 2016.

[50] W. Mao, J. Wang, and L. Wang, "Online sequential classification of imbalanced data by combining extreme learning machine and improved smote algorithm," in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–8.

[51] N. Japkowicz, C. Myers, and M. A. Gluck, "A novelty detection approach to classification," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1995, pp. 518–523.

[52] X.-Y. Liu and Z.-H. Zhou, "The influence of class imbalance on cost-sensitive learning: An empirical study," in *Sixth International Conference on Data Mining (ICDM'06)*, 2006, pp. 970–974.

[53] G. M. Weiss and F. Provost, "Learning when training data are costly: the effect of class distribution on tree induction," *Journal of Artificial Intelligence Research, 2003*, pp. 315–354, 2003.

[54] J. Wang, P. Zhao, and S. C. Hoi, "Cost-sensitive online classification," *IEEE Transaction on Knowledge and Data Engineering*, vol. 26, no. 10, pp. 2425 – 2438, 2014.

[55] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–45, 2006.

[56] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. PP, pp. 1–22, 2011.

[57] C. Li, "Classifying imbalanced data using a bagging ensemble variation," in *Proceedings of the 45th Annual Southeast Regional Conference*, 2007, pp. 203–208.

[58] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class imbalance learning," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 39, no. 2, pp. 539–550, 2009.

[59] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *Knowledge Discovery in Databases: PKDD 2003*, vol. 2838, 2003, pp. 107–119.

[60] J. Błaszczyński and J. Stefanowski, "Neighbourhood sampling in bagging for imbalanced data," *Special Issue on Information Processing and Machine Learning for Applications of Engineering, Neurocomputing*, vol. 150, no. Part B, p. 529–542, 2015.

[61] M. V. Joshi, V. Kumar, and R. C. Agarwal, "Evaluating boosting algorithms to classify rare classes: Comparison and improvements," in *Proceedings IEEE International Conference on Data Mining*, 2001, pp. 257–264.

[62] N. V. Chawla and J. Sylvester, "Exploiting diversity in ensembles: Improving the performance on unbalanced datasets," *Multiple Classifier Systems*, vol. 4472, pp. 397–406, 2007.

[63] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 30–39, 2004.

[64] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, "Adacost: Misclassification cost-sensitive boosting," in *Proceedings of the 16th International Conference on Machine Learning*, 1999, pp. 97–105.

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

[66] 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.

[67] M. A. Maloof, "Learning when data sets are imbalanced and when costs are unequal and unknown," in *Workshop on Learning from Imbalanced Data Sets II, ICML*, 2003.

[68] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[69] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing and Management*, vol. 45, no. 4, p. 427437, 2009.

[70] Y. Sun, M. S. Kamel, and Y. Wang, "Boosting for learning multiple classes with imbalanced class distribution," in *Sixth International Conference on Data Mining (ICDM'06)*, 2006, pp. 592–602.

[71] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine Learning*, vol. 45, no. 2, pp. 171–186, 2001.

[72] 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.

[73] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 5, pp. 730–742, 2010.

[74] M. G. Kelly, D. J. Hand, and N. M. Adams, "The impact of changing populations on classifier performance," in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999, pp. 367–371.

[75] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1517–1531, 2011.

[76] 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.

[77] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," in *ECML PKDD 2006 Workshop on Knowledge Discovery from Data Streams*, 2006.

[78] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, p. 964–994, 2016.

[79] L. Cohen, G. Avrahami-Bakish, M. Last, A. Kandel, and O. Kipersztok, "Real-time data mining of non-stationary data streams from sensor networks," *Information Fusion*, vol. 9, no. 3, pp. 344–353, 2008.

[80] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," *Advances in Artificial Intelligence*, vol. 3171, pp. 286–295, 2004.

[81] M. Harel, S. Mannor, R. El-Yaniv, and K. Crammer, "Concept drift detection through resampling," in *Proceedings of The 31st International Conference on Machine Learning*, 2014, p. 10091017.

[82] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: The accuracy updated ensemble algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 81 – 94, 2014.

[83] C. Alippi, G. Boracchi, and M. Roveri, "Hierarchical change-detection tests," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 2, pp. 246 – 258, 2017.

[84] J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine Learning*, vol. 90, no. 3, pp. 317–346, 2013.

[85] P. Lindstrom, S. J. Delany, and B. M. Namee, "Handling concept drift in text data stream constrained by high labelling cost," in *Proceeding of the 23rd International Florida Artificial Intelligence Research Society Conference*, 2010.

[86] R. Li, S. Wang, H. Deng, R. Wang, and K. C.-C. Chang, "Towards social user profiling: unified and discriminative influence model for inferring home locations," in *KDD*, 2012, pp. 1023–1031.

[87] S. Pan, J. Wu, X. Zhu, and C. Zhang, "Graph ensemble boosting for imbalanced noisy graph stream classification," *IEEE Transactions on Cybernetics*, vol. 45, no. 5, pp. 954 – 968, 2015.

[88] J. Gao, B. Ding, J. Han, W. Fan, and P. S. Yu, "Classifying data streams with skewed class distributions and concept drifts," *IEEE Internet Computing*, vol. 12, no. 6, pp. 37–49, 2008.

[89] J. Gao, W. Fan, J. Han, and P. S. Yu, "A general framework for mining concept-drifting data streams with skewed distributions," in *Proceedings of SIAM ICDM*, 2007, pp. 3–14.

[90] K. Wu, A. Edwards, W. Fan, J. Gao, and K. Zhang, "Classifying imbalanced data streams via dynamic feature group weighting with importance sampling," in *Proceedings of the 2014 SIAM International Conference on Data Mining*, 2014, pp. 722–730.

[91] S. Chen and H. He, "Sera: Selectively recursive approach towards nonstationary imbalanced stream data mining," in *International Joint Conference on Neural Networks*, 2009, pp. 522–529.

[92] ——, "Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach," *Evolving Systems*, vol. 2, no. 1, pp. 35–50, 2011.

[93] 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.

[94] T. R. Hoens and N. V. Chawla, "Learning in non-stationary environments with class imbalance," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 168–176.

[95] T. R. Hoens, N. V. Chawla, and R. Polikar, "Heuristic updatable weighted random subspaces for non-stationary environments," in *IEEE 11th International Conference on Data Mining (ICDM)*, 2011, pp. 241–250.

[96] A. D. Pozzolo, R. Johnson, and O. Caelen, "Using HDDT to avoid instances propagation in unbalanced and evolving data streams," in *International Joint Conference on Neural Networks (IJCNN)*, 2014, pp. 588 – 594.

[97] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[98] A. Ghazikhani, R. Monsefi, and H. S. Yazdi, "Online neural network model for non-stationary and imbalanced data stream classification," *International Journal of Machine Learning and Cybernetics*, vol. 5, no. 1, pp. 51–62, 2014.

[99] N. ying Liang, G. bin Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411 – 1423, 2006.

[100] W. N. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 377–382.

[101] C. Linhart, G. Harari, S. Abramovich, and A. Buchris, "PAKDD data mining competition 2009: New ways of using known methods," *New Frontiers in Applied Data Mining, Lecture Notes in Computer Science*, vol. 5669, pp. 99–105, 2010.

[102] 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*, 2012, pp. 1023–1031.

[103] S. Wang, L. L. Minku, and X. Yao, "Dealing with multiple classes in online class imbalance learning," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*, 2016, pp. 2118–2124.

[104] S. Wang, "Ensemble diversity for class imbalance learning," Ph.D. dissertation, School of Computer Science, The University of Birmingham, 2011.

[105] T. Fawcett, "ROC graphs: Notes and practical considerations for researchers," *HP Labs, Palo Alto, CA, Technical Report HPL-2003-4*, 2003.