Concept Drift Detection for Online Class Imbalance Learning

Shuo Wang, Leandro L. Minku, Davide Ghezzi, Daniele Caltabiano, Peter Tino and Xin Yao

Abstract— **Concept drift detection methods are crucial components of many online learning approaches. Accurate drift detections allow prompt reaction to drifts and help to maintain high performance of online models over time. Although many methods have been proposed, no attention has been given to data streams with imbalanced class distributions, which commonly exist in real-world applications, such as fault diagnosis of control systems and intrusion detection in computer networks. This paper studies the concept drift problem for online class imbalance learning. We look into the impact of concept drift on single-class performance of online models based on three types of classifiers, under seven different scenarios with the presence of class imbalance. The analysis reveals that detecting drift in imbalanced data streams is a more difficult task than in balanced ones. Minority-class recall suffers from a significant drop after the drift involving the minority class. Overall accuracy is not suitable for drift detection. Based on the findings, we propose a new detection method DDM-OCI derived from the existing method DDM. DDM-OCI monitors minority-class recall online to capture the drift. The results show a quick response of the online model working with DDM-OCI to the new concept.**

I. Introduction

ONLINE learning has received growing attention in machine learning in recent years, as more and more data is organized in the form of data streams rather than static databases. This type of learning aims to give timely response to incoming data. It has contributed to various real-world applications, such as sponsored search from web click data [1], credit card transactions [2] and spam filtering [3]. Strictly speaking, online learning algorithms process each training example once "on arrival" without the need for storage and reprocessing, and maintain a model that reflects the current concept to make a prediction at each time step [4]. The online learner is not given any process statistics for the observation sequence, and thus no statistical assumptions can be made in advance [5].

It is often seen that the data stream presents an imbalanced class distribution, such as fault diagnosis of control monitoring systems and intrusion detection in computer networks. In these cases, some classes of data are much more difficult or expensive to happen than the other classes, referred to as "class imbalance" in the literature. It can lead to great performance degradation, since the large number of majority-class examples overwhelms the incremental update of the model and minority-class examples are likely to be ignored [6] [7]. Learning from such imbalanced data streams is called online class imbalance (OCI) learning [8], which poses challenging problems to existing research. Concept drift is one of them, the phenomenon of unexpected change in underlying data over time. It is important as data is often nonstationary in real world. For example, new types of faults may appear during the process in fault detection. Although there exist methods for detecting and handling concept drift, class imbalance exacerbates the problem that need to be addressed.

In our previous work [8], we proposed a learning framework for online class imbalance learning, including three essential modules: a class imbalance detector used to capture the change in the prior probabilities of classes; a concept drift detector used to capture the changes in the class conditional probability density function (pdf); an adaptive online learner to handle those changes when they occur and to make a prediction on the current example. For the module of class imbalance detector, we have proposed an effective way to estimate the current class percentages and a detection method to alert the online learner to the change in class imbalance. As a continuation of the work, this paper will focus on the module of concept drift detector.

First, we analyse the impact of concept drift under seven scenarios with different types of concept drift and class imbalance change. We record the accuracy on each class (i.e. recall) and the overall accuracy at each time step, and observe their behavior. The ensemble algorithm Online Bagging is used as our online model, and three types of base classifiers are considered: decision trees, naive bayes and neural networks. The experimental results suggest that the minority-class recall is a better indicator than overall accuracy for detecting drifts involving the minority class. It presents a consistent and significant drop when a concept drift happens; meanwhile, it is less affected by the class imbalance change. Regarding the online model, decision trees and naive bayes are better base classifiers than neural networks. The neural network ensemble shows very poor minority-class recall in some cases, which can hide the effect of drift. Inspired by the observations, we next develop a drift detection method for online class imbalance problems, called DDM-OCI, based on the idea of DDM [9]. DDM-OCI makes use of the reduction in minority-class recall to capture the drift. It is shown to be able to sense the drift effectively. The online model applying DDM-OCI can respond to the new concept faster than the model applying DDM and the model without applying any drift detection methods in most cases.

The rest of this paper is organized as follows. Section II introduces the existing work in the field of concept drift detection and our previous research about online class imbalance learning. Section III studies the impact of concept

Shuo Wang, Leandro L. Minku, Peter Tino and Xin Yao are with the School of Computer Science, University of Birmingham, Birmingham, UK, B15 2TT (email: {S.Wang, L.L.Minku, P.Tino, X.Yao}@cs.bham.ac.uk).

Davide Ghezzi and Daniele Caltabiano are with STMicroelectronics SRL, Agrate Brianza (MB), Italy (email: {Davide.Ghezzi, Daniele.Caltabiano}@st.com)

drift in imbalanced data streams generated from two artificial data sets and one fault diagnosis data emulator. Section IV proposes the drift detection method and evaluates its performance through extensive experiments. Section V draws the conclusions and points out our future work.

II. Related Work

In this section, we first introduce the existing approaches for detecting concept drift, and explain why it is a difficult issue when the data stream is imbalanced. Then, we give a brief description of our previously proposed learning framework for online class imbalance learning, and propose the research questions studied in this paper.

A. Concept Drift Detection and Its Difficulties in Imbalanced Data Streams

One of the main assumptions of traditional data mining is that data is generated from a single, static and hidden function. However, it is hard to be true for data stream learning, where unpredictable changes are likely to eventually happen [10]. Concept drift is said to occur when the underlying function that generates instances changes over time. It can be formally defined as any scenario where the joint probability $p(x, c_i) = p(c_i)\, p(x \mid c_i)$ changes, in which x represents the input features and c_i represents the class label [11] [12]. So, it can manifest in the form of a change in the prior probabilities of the classes, a change in the class-conditional pdfs, or a change in both. A change in the prior probabilities of the classes is related to a change of class imbalance status. An example of such case is when a class presenting to be the minority in the data stream turns into the majority later on. In fault detection applications of engineering systems, a type of faults can appear more frequently along with time. Most existing methods aim to tackle drifts involving class-conditional changes, because they are considered to be more severe changes that affect the classification boundaries.

The learners responsive to concept drifts can be either trigger-based or evolving [10]. Trigger-based methods use an explicit detector to indicate any need for model update [13] [14] [15] [16], among which DDM (Drift Detection Method) [9] and EDDM (Early Drift Detection Method) [17] are two most popular ones strictly designed for online learning scenarios. DDM monitors overall error rate to warn the system about whether there is a drift, as it is believed that a significant error increase suggests a change in the class distribution in the PAC learning model. It computes the probability of observing a fault p_i for example i with the standard deviation s_i during the learning process. The algorithm issues a warning if $p_i + s_i \geq p_{min} + 2s_{min}$, and drift is confirmed if $p_i + s_i \geq p_{min} + 3s_{min}$. DDM requires waiting for at least 30 time steps after a drift is detected. With the similar idea, EDDM uses the distances between classification errors instead of the probability of errors to detect drift. Once a drift is reported, the current model is discarded or reset and a new one will be created. Evolving methods on the contrary do not detect changes. They usually maintain a set of models and get updated periodically based on their performance estimation, such as classifier ensembles [18] [19] [2]. Trigger-based methods can have a very quick response to drifts, but suffer from "false alarm" problems (i.e. a drift is reported when there is no drift). Evolving methods can produce more accurate results, but need some time to reflect the new concept.

Despite the aforementioned approaches, very little work has discussed changes in the prior probabilities of the classes, from which the problem of class imbalance can arise. Class imbalance in data can lead to a great performance reduction [20] [21] and poses difficult challenges to online learning including concept drift detection [22]. The difficulties lie in the following aspects, especially when a drift happens to the minority class. First, most traditional methods detect drifts based on the drop of overall accuracy or the increase of error made by the learner, such as EDDM and DDM. However, these measures are not appropriate for imbalanced data as they are sensitive to class imbalance and cannot reflect the performance on the minority class well. The minority class contributes too little to these performance measures compared to the majority class. Second, too few examples from the minority classes can make the time arbitrarily long until the concept drift is detected, which makes it difficult to infer the source of the error for the minority class – a drift or merely a result of noise [22]. Third, the minority-class performance can be very poor due to the imbalanced distribution. It may cover the effect of concept drift for detection. Fourth, a class can turn into minority or majority over time, so that drift detection methods may need to be adapted accordingly. Finally, the performance measures can vary greatly with the chosen online learner. Any of them could be the reason that affects the effectiveness of existing drift detection methods.

In the following sections, we will refer to a change affecting the class imbalance status as change in class imbalance, and a change affecting the class-conditional pdfs as concept drift, for brevity. Among very limited work solving the combined issue of concept drift and class imbalance [23] [24] [25], no drift detection methods have been developed for imbalanced data streams to the best of our knowledge. Our recent work first formulated online class imbalance learning problems by proposing an online processing learning framework [8]. More explanations about the learning framework will be given in the next section.

B. Online Class Imbalance Learning Framework

Combining class imbalance with concept drift, online class imbalance learning requires adaptive learning methods that can identify minority-class data accurately and timely without sacrificing the performance on the majority class under nonstationary environments. Accuracy (especially on the minority class), efficiency and adaptivity are desirable characteristics for a good solution. To achieve the goal, we proposed a learning framework that breaks down the learning process into three modules – a class imbalance detector, a concept drift detector and an adaptive online

leaner, as shown in Fig. 1 [8]. Each module handles one major issue of online class imbalance, and communicates with the others for the up-to-date status of data streams. The class imbalance detector reports the class imbalance status of data streams under nonstationary environments. The concept drift detector captures concept drift involving classification boundary shifts in imbalanced data streams. Based on the information provided by these two modules, the adaptive online learner determines when and how to respond to the detected class imbalance and concept drift, and makes a real-time prediction. This is a general framework for dealing with imbalanced data streams with arbitrary number of classes.

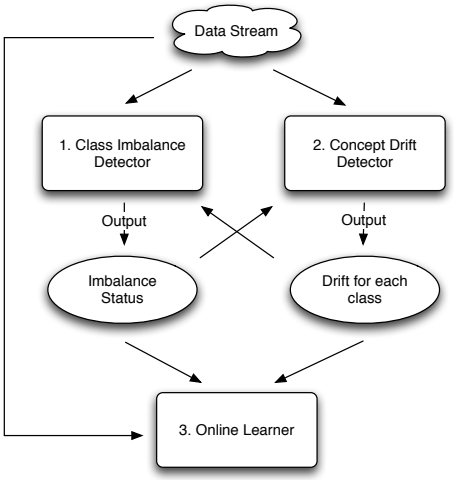


Fig. 1: Online class imbalance learning framework [8].

For the class imbalance detector within the framework, we used time decay functions to report the current distribution of class labels and current performance of the model on each class, based on which an imbalance detection algorithm was proposed to determine whether the current data stream should be regarded as "imbalanced" in real time. The main idea is, if there are any two classes having significantly different sizes and the relatively large class receives much higher recall than the small class from the model, then the small class is regarded as the minority and the large class is regarded as the majority. A class imbalance status will then be sent to the online learner – resampling-based Online Bagging. It applies either random oversampling or undersampling to adjust the learning bias from the majority towards the minority. The proposed methods are shown to be effective on data streams with class imbalance changes and a static data concept.

The concept drift detector aims to detect potential changes in the underlying distribution of data with the presence of class imbalance. Due to the importance and learning difficulty of minority classes, they should be given particular attention. Using methods based on overall accuracy are unlikely to provide accurate drift detections when the concept drift affects mainly the minority class. Nevertheless, it is still not clear how single-class performance behaves when concept drift happens and whether it can help the detection. Moreover, any change in class imbalance may disturb the detection. Therefore, the following questions will be studied in this paper as a part of the framework:

- How does concept drift affect the performance of each class when data is imbalanced? Does a class imbalance change affect the detection of concept drift?
- How to detect concept drift accurately and timely, especially when it happens to minority classes?

III. Impact of Concept Drift in Imbalanced Data Streams

This section looks into the impact of concept drift on the performance of each class in data streams with the presence of class imbalance. The behavior of recall (i.e. single-class accuracy) is observed under seven scenarios including imbalanced data streams without any change, with only class imbalance changes, with only concept drifts, and with both class imbalance changes and concept drifts at the same time. The analysis here will help us to recognize the different effect of concept drift and class imbalance, and develop effective concept drift detection methods for imbalanced data streams.

A. Data Set Descriptions

The data streams used in our experiments are generated from two artificial data sets (SEA moving hyperplane concepts [26] and STAGGER Boolean concepts [27]) and one fault diagnosis platform iNemo. SEA and STAGGER are the two most popular concept drift benchmarks in the literature.

SEA data consists of three attributes, where only the first two attributes are relevant and no class noise is introduced. All three attributes have values between 0 and 10. The class label is determined by $\alpha_1 + \alpha_2 \leq \theta$, where α_1 and α_2 represent the first two attributes and θ is a threshold value. If the condition is satisfied, then the example will be labelled as class 0; otherwise, it will be labelled as class 1. θ is adjusted for new concepts. Class 1 is chosen to be the minority class in our work.

STAGGER concepts are boolean functions of three attributes: size (small, medium and large), shape (circle, triangle and rectangle) and colour (red, blue and green). New concepts are obtained by defining different boolean functions. Class labels can be 'false' (class 0) or 'true' (class 1). Class 1 is chosen to be the minority class in our work.

iNemo is a multi-sensing platform developed by STMicroelectronics for numerous applications, such as virtual reality, augmented reality, image stabilization, human machine interfaces and robotics. It combines accelerometers, gyroscopes and magnetometers with pressure and temperature sensors to provide 3-axis sensing of linear, angular and magnetic motion in real time, complemented with temperature and barometer/altitude readings [28]. To avoid any functional disruption caused by signalling faults in iNemo, a fault emulator is developed for producing and analysing different types of faults. A fault is defined as an unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable/usual/standard condition. It can be introduced into any sensor of iNemo by using the emulator

TABLE I: Old and new concepts for each data set.

Concept	SEA	STAGGER	iNemo
Old	$\theta = 13$	size=small $\cap$ color=red	$\theta = 500$
New Low Severity	$\theta = 10$	(size=small $\cap$ color=red) $\|\|$ (color=green $\cup$ shape=square)	$\theta = 500 \,\|\|\, \theta = -500$
New High Severity	$\theta = 7$	color=green $\cup$ shape=square	$\theta = -500$

given the real data sequence coming from the sensors. In our study, we generate offset faults for the feature of gyroscope x-axis. Specifically, the erroneous signal is produced by adding an offset θ to the normal signal. The task of this data is to build and maintain an effective online learner to detect faults. Due to the rarity and importance of faults, fault detection in engineering systems is a typical problem of learning from imbalanced data streams. There are two classes in the data stream – nonfaulty (class 0) and faulty (class 1). The faulty class is the minority as it is much less likely to happen than the nonfaulty class. θ is adjusted for new concepts.

We generated seven types of data streams for each of the three data generators. Each type contains 1000 data examples. During the first half of the examples (1-500 time steps), the true size ratio between minority and majority classes is fixed to 1:9 without any concept drift. In other words, $p_{c1} = 0.1$, $p_{c0} = 0.9$, $p_{old} = 1$ and $p_{new} = 0$, where p_{c1} and p_{c0} denote the probabilities of examples belonging to class 1 and class 0, and p_{old} and p_{new} denote the probabilities of examples generated from old and new concepts. The class imbalance change or concept drift can happen abruptly at the 501 time step with either high or low severity. An abrupt concept drift means that from time step 501 onwards $p_{old} = 0$ and $p_{new} = 1$. An abrupt class imbalance change means that the class size ratio switches to another value right away. A change with high severity means that the new concept or class imbalance status is very different from the old one, whereas low severity means that the change is less severe than the high severity one, as defined in Tables I and II.

TABLE II: Old and new class imbalance status.

	p_{c0}	p_{c1}
Old	0.9	0.1
New Low Severity	0.5	0.5
New High Severity	0.1	0.9

The seven data stream scenarios with different severity of class imbalance change and concept drift are described in Table III. Each data stream contains one concept drift and one class imbalance change at the most. For all these data streams, we assume that the minority class is known at the beginning, which can be actually obtained from the class imbalance detector in the learning framework; concept drift always affects the classification rules of the minority class. Currently, we only consider two-class data with abrupt concept drifts or class imbalance changes. More complicated cases will be included in our next stage investigation.

TABLE III: Data stream scenarios – severity of changes. The symbol "–" indicates no change.

Case	Class imbalance	Concept drift
Case 1	–	–
Case 2	High	–
Case 3	Low	–
Case 4	–	High
Case 5	–	Low
Case 6	High	High
Case 7	Low	Low

B. Experimental Settings

Online Bagging (OB) [4] is adopted as our online learner, because it successfully extends the well-known offline ensemble algorithm Bagging [29] to online cases and does not present any specific behavior to handle concept drift. So, it can be used to analyse the effect of changes and give insight on how to detect them. In our experiments, each OB ensemble is formed by 50 base classifiers. Three types of classifiers are considered, including neural networks (NN), decision trees (DT) and naive bayes (NB). Particularly, Hoeffding tree [30] is used as a fast decision tree induction algorithm that is capable of learning from massive data streams. It is proved by the authors that a Hoeffding tree is 'very close' to classic decision tree learning schemes [31], such as C4.5 and CART. The neural network is a single perceptron classifier using the sigmoid function as the activation function. The implementation was provided by the Massive Online Analysis (MOA) tool with its default settings [32]. The OB algorithm is run 100 times on every data stream and outputs the average performance at each time step.

In order to observe how the performance of each class is affected by different types of changes, we monitor the current recall R_k of the online model for each class c_k, defined by n_k^+/n_k, where n_k^+ denotes the number of correctly classified examples with true label c_k and n_k denotes the total number of examples with true label c_k received so far. Following the definition in the class imbalance detector [8], R_k will be updated by $R_k^{(t)} = \eta' R_k^{(t-1)} + (1-\eta')\,[x \leftarrow c_k]$ at time step t if the current input x has true label c_k, where η' $(0 < \eta' < 1)$ is a time decay factor for emphasizing the learner's performance at the current moment, and $[x \leftarrow c_k]$ is equal to 1 if x is correctly classified and 0 otherwise. We choose time decayed R_k here rather than the traditional prequential recall without applying the time decay factor, because the former has been shown to be able to better reflect the current performance of the online model in non-stationary environments without forgetting its past performance too much. It is expected to be more sensitive to concept drift,

and thus could be a better indicator for concept drift detection. Besides, it has been used to detect class imbalance successfully [8]. η' is set to 0.9 in our experiments.

C. Results and Analysis

We plot R_k curves of each class produced from the seven cases along with the time step, which are split into two groups for comparison. Cases 1, 2, 4 and 6 are compared in one plot to show the impact of high severity changes; cases 1, 3, 5 and 7 are grouped to show the impact of low severity changes.

Recall curves of class 1 are presented in Figs. 2–4 for SEA, STAGGER and iNemo respectively, which is the original minority class during the first 500 time steps. Each figure shows the changing behavior of recall produced by OB with different base learners. As we can see, all the models have very poor recall on class 1 at first due to class imbalance. DT and NB are more appropriate base learners than NN, because their recall of class 1 gets better more quickly as more data arrive, whereas NN struggles in particular for SEA and iNemo.

We also observe that cases 4, 5, 6 and 7, which are the cases containing the concept drift, present a significant drop right after the 500 time step in DT-based OB and NB-based OB in both high and low severity comparisons. That does not happen to the other cases significantly. Cases 1, 2 and 3 show some performance fluctuation during the latter half of training, but it is not a consistent performance reduction like what happens in the concept drift ones. This is an important observation, as it may be used to discriminate concept drift from class imbalance change for drift detection.

Another interesting observation in DT-based OB and NB-based OB is, the case with both concept drift and class imbalance change produces a faster drop in recall than the case with only concept drift when the drift happens. The former then presents a faster performance recovery than the latter for high severity changes. This is because more class 1 examples with the new concept arrive, thus affecting recall more quickly. The performance drop is not observed in NN-based OB in concept drift cases of SEA and iNemo, because its recall on class 1 remains very low. This suggests that if the model has very poor performance on the minority class, it would be very difficult to detect the drift that happens to this class.

The recall of class 0, on the other hand, does not change much through all the time steps in most cases, which remains very high (nearly 1), regardless of the base classifiers, types of data streams, the changing severity and data generators, according to our observations.

Generally speaking, minority-class recall can be a useful measure for detecting concept drifts that involve the minority class in imbalanced data streams, and a good indicator to discriminate concept drifts involving the class-conditional pdfs from class imbalance changes. In our cases, the "minority class" we should monitor is the minority before any change happens. The changing severity did not influence our results much. The choice of online learners was more important. Online Bagging ensembles based on decision trees and naive bayes obtained better performance than the ensembles based on neural networks for SEA and iNemo. Poor performance on the minority class can cover the effect of concept drift and increase the difficulty in locating the drift.

To explain why overall accuracy is not appropriate for detecting drift in imbalanced data streams, Fig. 5 shows how overall accuracy produced by DT-based OB behaves in SEA data streams with high severity changes. For the fair comparison, the time decay factor 0.9 is also applied for calculating the overall accuracy.

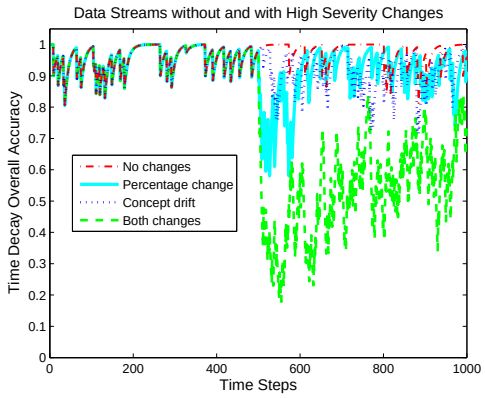
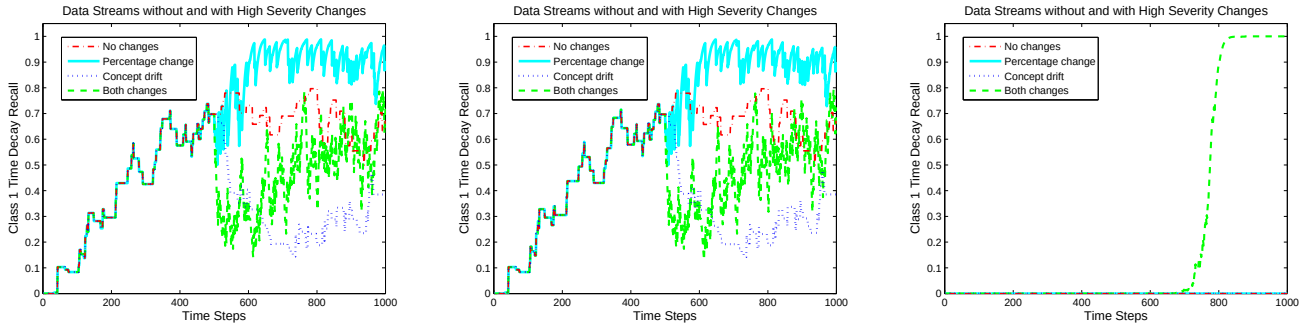


Fig. 5: SEA: time-decayed overall accuracy under the scenarios of high severity changes produced by DT-based OB.

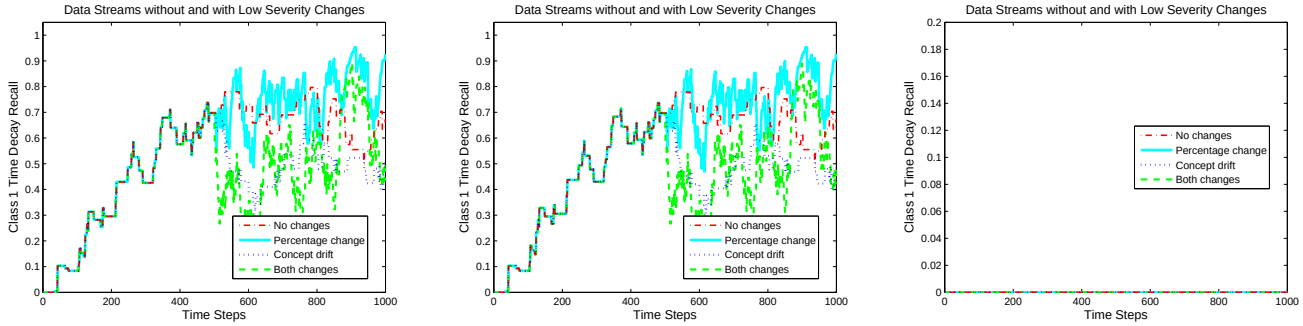
According to Fig. 5, during the first 500 time steps, the overall accuracy presents to be very high; in fact, the performance on the minority class is actually quite low based on our previous observations. From the 501 time step, a severe drop can be observed in the case containing only the class imbalance change (case 2), which is even more significant than the case containing only the concept drift (case 4). This is because overall accuracy is more sensitive to the number of examples in classes. Even though there is no concept drift in case 2, an abrupt increase in the number of minority-class examples still causes a great accuracy reduction. So, we would not recommend overall accuracy as a reliable indicator for concept drift in imbalanced data streams. It is important to emphasize here that the strategy to deal with changes in class imbalance status (e.g., resampling) should be different from the strategy to deal with drifts (e.g., resetting the model). For example, if the model is reset to deal with a change in class imbalance where a majority class becomes minority, useful information learnt when the class was majority would be lost [8]. Thus, the inadequacy of the overall accurate to distinguish between these two types of change is problematic.

IV. Concept Drift Detector

In this section, we propose a new method, called Drift Detection Method for Online Class Imbalance (DDM-OCI), to locate concept drifts actively for imbalanced data streams. Time decayed recall of the minority class is used as the

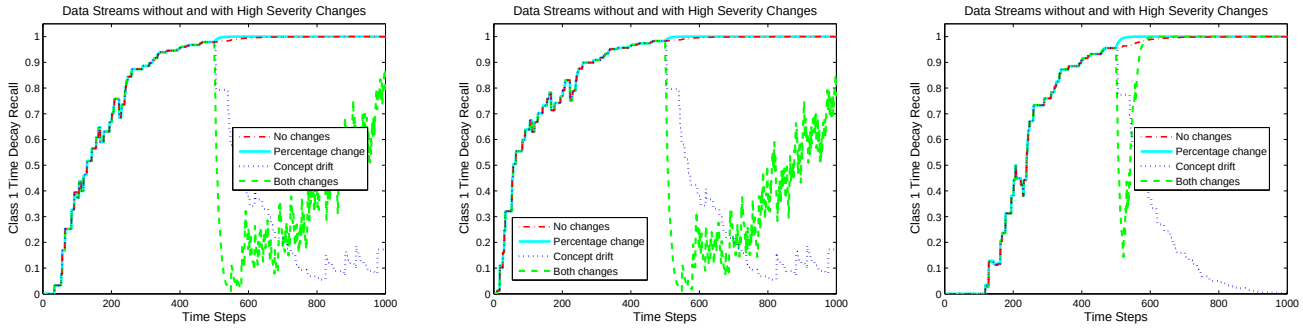


(a) Class 1 high severity changes (cases 1,2,4,6)

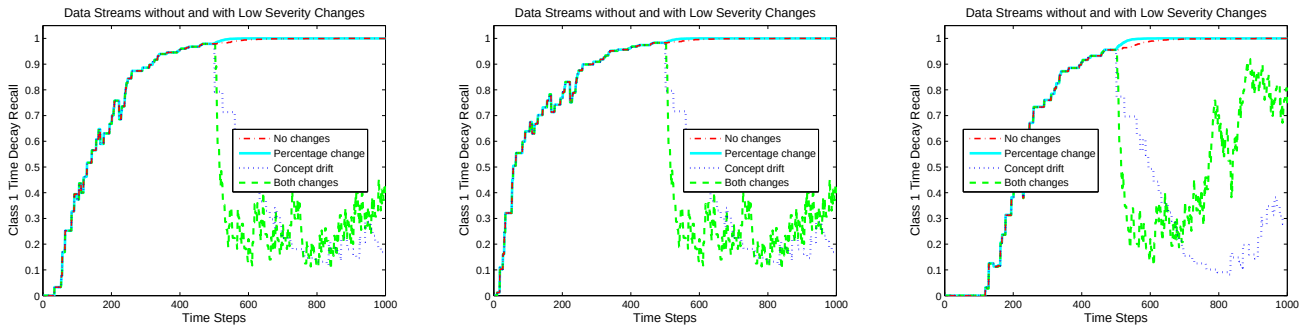


(b) Class 1 low severity changes (cases 1,3,5,7)

Fig. 2: SEA: time-decayed recall curves of class 1 under the seven scenarios produced by OB based on DT, NB and NN respectively (left: DT; middle: NB; right: NN).

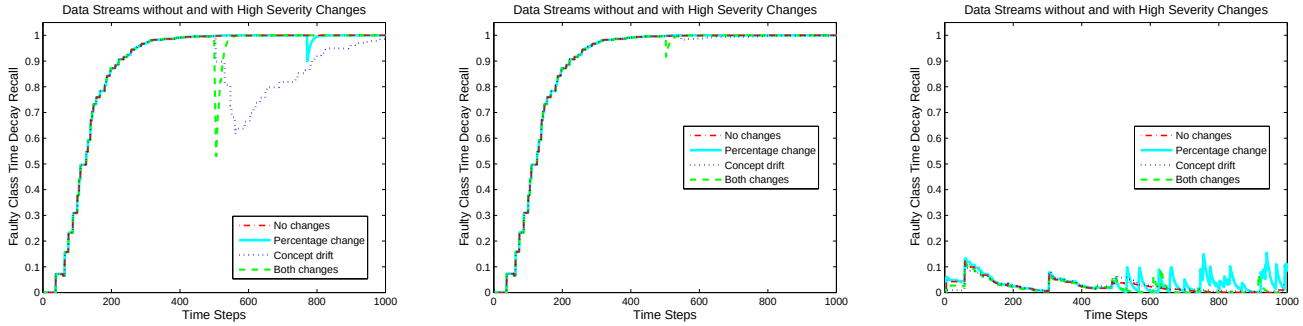


(a) Class 1 high severity changes (cases 1,2,4,6)

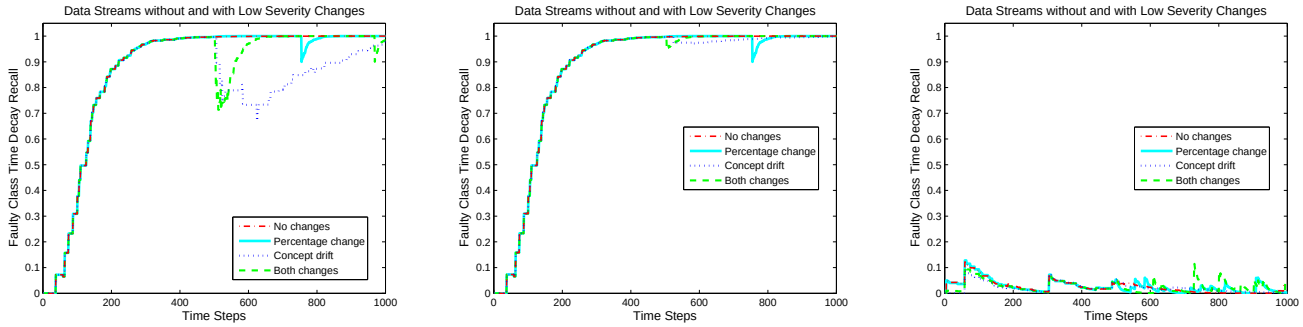


(b) Class 1 low severity changes (cases 1,3,5,7)

Fig. 3: STAGGER: time-decayed recall curves of class 1 under the seven scenarios produced by OB based on DT, NB and NN respectively (left: DT; middle: NB; right: NN).



(a) Class 1 high severity changes (cases 1,2,4,6)



(b) Class 1 low severity changes (cases 1,3,5,7)

Fig. 4: iNemo: time-decayed recall curves of class 1 under the seven scenarios produced by OB based on DT, NB and NN respectively (left: DT; middle: NB; right: NN).

drift indicator based on the results so far. The method will be integrated into the online class imbalance learning framework, which processes data one at a time on arrival. It can be easily extended to situations where data comes in batches. We assume that the minority class is known in current data for now. In the near future, we will apply the class imbalance detector to provide the information in real time and examine how well it can collaborate with the concept drift detector.

A. Drift Detection Method for Online Class Imbalance

DDM-OCI modifies the traditional detection method DDM [9] for online class imbalance scenarios. Instead of monitoring the overall error rate in DDM, DDM-OCI tracks the recall on the minority class. A significant drop in the recall suggests a drift in this class. The existing decision model thus needs to be updated. Given the class label c_k we should focus on, the method stores two values p_i and s_i, updated whenever the current example x_i belongs to c_k, where p_i is the probability of correctly classifying examples having the true label c_k, with standard deviation $s_i = \sqrt{p_i (1 - p_i) / i}$. p_i is estimated by time-decayed recall defined in the previous section. p_{max} and s_{max} are recorded to remember when $p_i + s_i$ reaches its maximum value, and the following conditions are checked during the process:

- $p_i - s_i \leq p_{max} - 2s_{max}$ for the warning level. When this level is reached, a potential drift is considered to start from this moment t_w. Examples coming after t_w are stored.
- $p_i - s_i \leq p_{max} - 3s_{max}$ for the drift level. When this level is reached, the concept drift is confirmed at this moment t_d. The online model and all the recorded values including p_i, s_i, p_{max} and s_{max} are reset. A new model is induced using the examples stored between t_w and t_d.

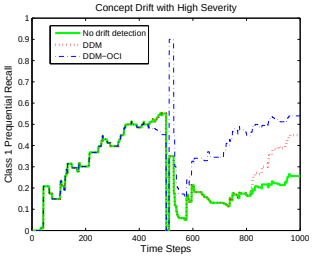
The coefficients of s_{max} for warning and drift levels reflect the confidence levels used to detect a significant drop in recall. In the inequations above, the confidence levels are 95% and 99%, respectively [9]. DDM-OCI is not allowed to perform new drift detections for 50 time steps after any drift is detected, following the setting of larger than 30 examples in the original paper. According to our observations in the previous section, DDM-OCI might trigger false positive drift detections (false alarms) due to the variance of recall on the minority class throughout the learning. How DDM-OCI performs in imbalanced data with the presence of concept drift and whether false alarms can happen and affect its performance will be examined next.

B. Performance Evaluation

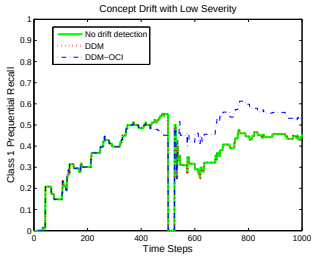
DDM-OCI is evaluated through the online learning algorithm – Online Bagging based on decision trees (DT-based OB) and Online Bagging based on naive bayes (NB-based OB), following the same settings as in Section III. Their performance is compared to those using the traditional DDM and those without applying any drift detection methods. Online Bagging based on neural networks (NN-based OB) is not discussed here, because of its very poor minority-class

performance. For the space consideration, the results from NB-based OB are not shown in this section. Its outputs are very similar to those produced by DT-based OB. Tested data streams used in the following experiments are cases 4-7 of SEA, STAGGER and iNemo, containing one concept drift.

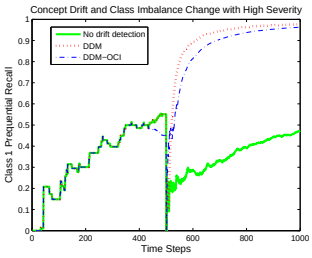
We conduct the prequential test on recall of the minority class, which is "class 1" for the three data sets. Prequential test is an evaluation technique [33], in which each individual example is used to test the model before it is used for training, and from this the performance measures can be incrementally updated. The model is always being tested on examples it has not seen. In our experiments, the minority-class recall is recorded at each time step, and a performance curve can be produced. Each point on the curve is the average of 100 runs. Because concept drift happens right in the middle of the data stream, we separate the prequential performance into two stages for the evaluation of each model by resetting the performance results to 0 at the 500th time step. This ensures that the performance observed after the drift is not affected by the performance before the drift, allowing us to analyse the behaviour of the models before and after the drift adequately.



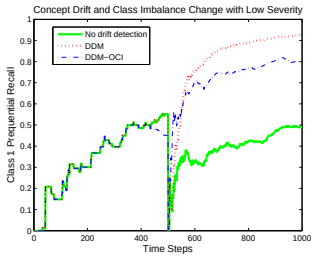
(a) Case 4: concept drift with high severity



(b) Case 5: concept drift with low severity



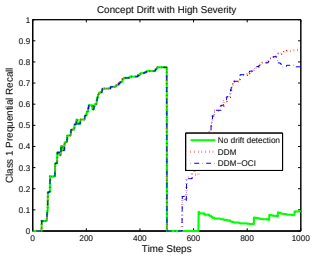
(c) Case 6: concept drift and class imbalance change with high severity



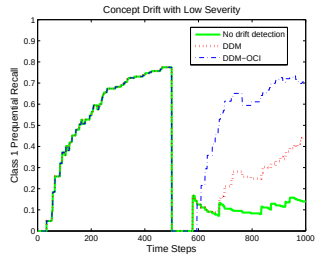
(d) Case 7: concept drift and class imbalance change with low severity

Fig. 6: SEA: prequential recall curves of class 1 produced by DT-based OB for cases 4-7.

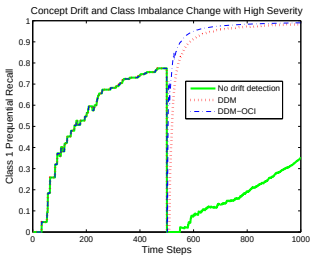
Figs. 6 - 8 present the prequential recall curves for each data set. Each plot compares the online models with different drift detection strategies. For the two artificial data SEA and STAGGER, both DDM and DDM-OCI models outperform the model without drift detection during the latter half of the learning, which shows the usefulness of applying explicit drift detecting techniques. DDM-OCI responds to the new concept earlier than DDM in most cases, even though false



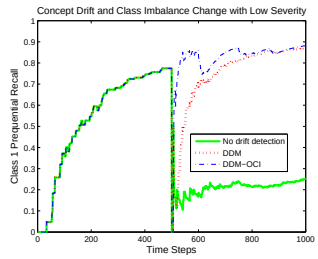
(a) Case 4: concept drift with high severity



(b) Case 5: concept drift with low severity



(c) Case 6: concept drift and class imbalance change with high severity



(d) Case 7: concept drift and class imbalance change with low severity

Fig. 7: STAGGER: prequential recall curves of class 1 produced by DT-based OB for cases 4-7.

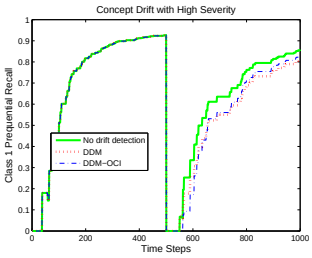
alarms can cause DDM to surpass DDM-OCI in some cases after the initial swift response (case 7 of SEA and case 4 of STAGGER). Table IV shows the average number of detected drifts.

TABLE IV: Number of drifts detected by DDM and DDM-OCI in DT-based OB.

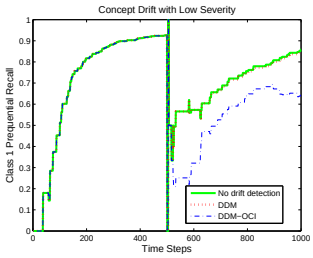
	SEA				STAGGER				iNemo			
Case	4	5	6	7	4	5	6	7	4	5	6	7
DDM	1	0	2	1	1	2	1	1	1	0	1	1
DDM-OCI	2	2	2	6	2	2	1	4	1	2	1	2

In case 6 of SEA, the number of false alarms was the same for DDM and DDM-OCI. However, in this case, the moment of the false alarms affected DDM-OCI's performance more badly than DDM's. DDM-OCI gives a false alarm at around time step 434 in most runs. This will cause a delay in the detection of the real drift, because the model is reset and the new s_{max} and p_{max} are smaller than they would be if the model had not been reset. DDM-OCI's true drift detection thus occurs around time step 541, which is later than DDM's detection at around time step 511. This will affect DDM-OCI's prequential recall badly. DDM's false alarm, on the other hand, occurs at time step 575. This false alarm is not very detrimental to DDM's prequential recall because in case 6 the percentage of examples from class 1 is very large, and a decision tree learnt from a single example of class 1 is able to provide very good class 1 recall.

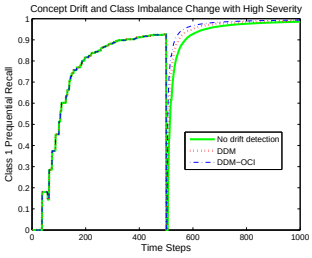
When comparing high severity drifting cases (cases 4 and 6) and low severity drifting cases (cases 5 and 7), it seems that low severity negatively affects DDM more than DDM-



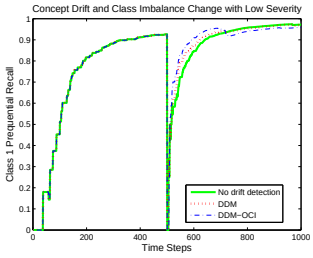
(a) Case 4: concept drift with high severity



(b) Case 5: concept drift with low severity



(c) Case 6: concept drift and class imbalance change with high severity



(d) Case 7: concept drift and class imbalance change with low severity

Fig. 8: iNemo: prequential recall curves of class 1 produced by DT-based OB for cases 4-7.

OCI, based on the observation that DDM models start to react to the drift even later in low severity cases than in high severity cases, sometimes even not providing a drift detection. The reason could be that high severity drifts in the minority class affect the overall accuracy more significantly, and are easier to be detected by DDM. Therefore, DDM tends to be less effective in detecting low severity drifts, while DDM-OCI seems to be more robust to changing severity from the point of view of early drift detection. When comparing the cases without class imbalance change (cases 4 and 5) and those with class imbalance change (cases 6 and 7), both DDM and DDM-OCI recover faster from the drift in cases 6 and 7, because more frequent occurrence of examples from class 1 facilitates the learning.

For iNemo, DDM-OCI does not show much benefit. The model without drift detection gives very good recall on the new concept already, which may imply that the new concept is quite easy to learn or related to the old concept. However, we notice that DDM-OCI detects the drift correctly and timely in these cases, which turns out to be detrimental to the performance in cases 4 and 5. The model resetting slows down the recall recovery from the old concept to the new concept, probably because it is easier to adapt a model that learnt the old concept to the new concept in this case than learning the new concept from scratch. Nevertheless, an increase in the percentage of examples from class 1 seems to make the new concept easier to learn from scratch. In cases 6 and 7, DDM-OCI becomes useful again, and provides even better performance than the approach with no drift detection. So, performing early drift detections can still be useful for a drift where the old model can itself adapt to the new concept.

In general, DDM-OCI is helpful for dealing with drifts in the minority class. It detects drifts in the minority class more quickly when using the same parameters (significance level) for the algorithm as in DDM. It also responds to the new concept better right after the drift in most cases. By mapping the result into real-world applications such as fault detection, it means that more new types of faults can be detected in time, showing the practical value of this work.

The drawback of DDM-OCI is that it is sometimes too sensitive, and thus can cause more false alarms. DDM is more conservative, and may miss the drift when it happens to the minority class. The sensitivity to the drift can be actually tuned to some extent by adjusting the two threshold parameters in the warning and drift levels. Currently, they are set to 2 and 3 in both DDM and DDM-OCI. In order to check whether DDM is able respond faster to the drift and achieve more competitive results against DDM-OCI, we tested it with several reduced parameters (0.5 and 1.5; 1 and 2; 1.5 and 2.5) and compared it to DDM-OCI with the original setting. This test was performed for case 4 of SEA and case 5 of STAGGER, in which DDM-OCI shows significantly better recall than DDM after the drift based on figures 6 and 7. Even though DDM with smaller parameters did detect the drift earlier and allowed the online model to recover its performance faster after the drift, minority-class recall was considerably reduced before the drift, because more false alarms happened to DDM than to DDM-OCI. From this point of view, using single-class recall for drift detection in DDM-OCI is more robust to the performance disruption from other classes, and is thus more suitable for imbalanced data.

The results here suggest that there is still room for us to further improve DDM-OCI by getting rid of the false alarms. Strategies to deal with false alarms such as the ones used in [11] could be used not only to deal with false alarms, but also to identify when an old model can recover faster than a new model can learn. The investigation on that is proposed as future work.

V. Conclusions

As it is rather impractical to assume that class prevalence of each class in data remains equivalent and the data concept remains static in streaming data applications, concept drift detection in imbalanced data streams becomes necessary and important, but hasn't been studied by any of the existing work so far. New challenges arise when both class imbalance and concept drift can happen during online processing. This paper looks into this problem, as a crucial component of online class imbalance learning framework.

First, we analyse the impact of concept drift on the performance of each class, especially the minority class, by observing the changing behavior of the recall measure updated incrementally with a time decay factor. Online Bagging ensembles based on three types of classifiers are examined and compared in seven learning scenarios generated from three data sets. Minority-class recall presents a consistent and

significant drop when a concept drift happens; meanwhile, it is less affected by the class imbalance change. Overall accuracy is interrupted by both class imbalance change and concept drift. This is undesirable if we use overall accuracy to detect the drift, because adopted mechanisms to deal with concept drift could be detrimental to the performance when there is only a change in class imbalance. Regarding the online model, decision trees and naive bayes are better base classifiers than neural networks. The results suggest that minority-class recall could be a good indicator for detecting drifts.

Based on the first part of work, we develop an explicit drift detection method, called DDM-OCI. It measures the reduction in minority-class recall to capture the drift. It is shown to be able to sense the drift effectively, but false alarms exist and disturb its performance. The online model applying DDM-OCI can respond to the new concept faster than the model applying DDM and the model without applying any drift detection methods in most cases.

In our future work, there is still much room to further improve our method and experiments. First, it would be necessary to reduce the impact of false alarms of detected concept drift from DDM-OCI. It might be worth to explore other types of detection methods. Second, we assume that the minority class is known in this paper. This information can be obtained in real time from the class imbalance detector in the learning framework. We would like to examine the performance when the class imbalance detector and concept drift detector work together. Third, it would be interesting to further investigate the impacts of class imbalance degree and the corresponding class imbalance techniques on concept drift detection. Currently, only two changing degrees have been discussed. Besides, we would like to extend this work to data streams with arbitrary number of classes.

ACKNOWLEDGMENT

This work was supported by the European funded project (FP7 Grant No. 270428) "iSense: making sense of nonsense" and the EPSRC funded project (Grant No. EP/J017515/1) "DAASE: Dynamic Adaptive Automated Software Engineering". Xin Yao was supported by a Royal Society Wolfson Research Merit Award.

REFERENCES

[1] M. Ciaramita, V. Murdock, and V. Plachouras, "Online learning from click data for sponsored search," pp. 227–236, 2008.
[2] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," pp. 226–235, 2003.
[3] S. J. Delany, P. Cunningham, A. Tsymbal, and L. Coyle, "A case-based technique for tracking concept drift in spam filtering," *Knowledge-Based Systems*, vol. 18, no. 4-5, pp. 187–195, 2005.
[4] N. C. Oza, "Online bagging and boosting," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 2340–2345, 2005.
[5] C. E. Monteleoni, "Online learning of non-stationary sequences," Massachusetts Institute of Technology, Tech. Rep., 2003.
[6] 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.
[7] S. Wang and X. Yao, "Multi-class imbalance problems: Analysis and potential solutions," *IEEE Transactions on Systems, Man and Cybernetics, PartB: Cybernetics*, vol. 42, no. 4, pp. 1119–1130, 2012.
[8] S. Wang, L. L. Minku, and X. Yao, "A learning framework for online class imbalance learning," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2013, p. (Accepted).
[9] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," *Advances in Artificial Intelligence*, vol. 3171, pp. 286–295, 2004.
[10] I. Zliobaite, "Learning under concept drift: an overview," Vilnius University, Tech. Rep., 2010.
[11] 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.
[12] 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.
[13] K. Nishida, "Learning and detecting concept drift," Ph.D. dissertation, 2008.
[14] A. Bifet and R. Gavalda, "Learning from time-changing data with adaptive windowing," pp. 1–17, 2007.
[15] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers–part i: Detecting nonstationary changes," *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1145–1153, 2008.
[16] C. Alippi, G. Boracchi, and M. Roveri, "Just-in-time classifiers for recurrent concepts," *IEEE Transactions on Neural Networks and Learning Systems*, vol. (To appear), 2013.
[17] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," 2006.
[18] K. O. Stanley, "Learning concept drift with a committee of decision trees," Department of Computer Sciences, University of Texas at Austin, Tech. Rep., 2003.
[19] J. Z. Kolter and M. A. Maloof, "Dynamic weighted majority: An ensemble method for drifting concepts," *The Journal of Machine Learning Research*, vol. 8, pp. 2755–2790, 2007.
[20] S. Wang and X. Yao, "Relationships between diversity of classification ensembles and single-class performance measures," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 206–219, 2013.
[21] ——, "Using class imbalance learning for software defect prediction," *IEEE Transactions on Reliability*, 2012 (DOI: 10.1109/TR.2013.2259203).
[22] 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.
[23] 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.
[24] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, 2012 (DOI: 10.1109/TKDE.2012.136).
[25] H. M. Nguyen, E. W. Cooper, and K. Kamei, "Online learning from imbalanced data streams," pp. 347–352, 2011.
[26] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," pp. 377–382, 2001.
[27] J. C. Schlimmer and R. H. Granger, "Incremental learning from noisy data," *Machine Learning*, vol. 1, no. 3, pp. 317–354, 1986.
[28] STMicroelectronics, "iNemo: iNErtial MOdule V2 demonstration board." [Online]. Available: http://www.st.com/internet/evalboard/product/250367.jsp
[29] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
[30] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 97–106.
[31] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "Data stream mining: A practical approach," 2011, MOA software.
[32] "MOA massive online analysis: Real time analytics for data streams."
[33] A. P. Dawid and V. G. Vovk, "Prequential probability: Principles and properties," *Bernoulli*, vol. 5, no. 1, pp. 125–162, 1999.