Knowledge and Information Systems manuscript No.
(will be inserted by the editor)

ADWIN-U: Adaptive Windowing for Unsupervised Drift Detection on Data Streams

Daniel Nowak Assis · Vinicius M. A. Souza

Received: 27 Sep 2023 / Accepted: 27 Sep 2023

Abstract Data streams have become increasingly important due to the massive volume of data generated by recent advances in electronic devices and sensors widely used in smart cities and Internet-of-Things applications. However, mining continuous data in non-stationary environments with concept drifts poses significant challenges, such as the need for model updates with recent patterns. A common approach for detecting concept drift is based on model performance monitoring, where a drop in performance indicates the occurrence of drift requiring a model update. The Adaptive Windowing (ADWIN) is the most representative method that follows the performance monitoring approach. However, ADWIN and similar methods often rely on labeled data, which may not be readily available in streaming scenarios. We propose ADWIN-U (Adaptive Windowing for Unsupervised Drift Detection), an unsupervised concept drift detector based on the state-of-the-art ADWIN. Our proposal is independent of labeled data for drift monitoring, making it suitable for real-world stream problems where complete labeled data is costly or impractical. Our experimental evaluation demonstrates that ADWIN-U outperforms its supervised version in various domains. Furthermore, we address the limitations of existing evaluation measures for drift detection, which tend to favor approaches that require extensive labeled data. We propose a novel evaluation metric – Balanced Accuracy by the Amount of Requested Labeled Data (BAR). BAR considers the trade-off between accuracy and the proportion of labeled data requested for model updates, favoring accurate detectors with low false alarm rates while minimizing the reliance on labeled data.

Daniel Nowak Assis
Pontifícia Universidade Católica do Paraná (PUCPR)
Curitiba, PR, Brazil
E-mail: daniel.nowak@pucpr.edu.br

Vinicius M. A. Souza
Graduate Program in Informatics (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)
Curitiba, PR, Brazil
E-mail: vinicius@ppgia.pucpr.br

Keywords Data streams · Concept drift · Unsupervised drift detector · Evaluation of drift detector

1 Introduction

Mining data streams have gained importance in the last years due to the massive amount of data generated in real-time by networks, smartphones, and all kinds of electronic devices and sensors currently available in smart cities [11]. These devices continuously produce vast amounts of data in non-stationary environments, where the underlying data distribution may change over time. This phenomenon, known as *concept drift* [16], poses a significant challenge for data stream mining, such as the need for updated models to avoid increasing error rates [36].

Modern data mining solutions are dynamic and must handle massive amounts of data with evolving patterns by updating their models with the most recent data. In order to reduce the costs of unnecessary model retraining, these solutions run a concept drift detector to decide the best times and adequate data for updates. Furthermore, updates only when needed can drastically reduce the cost of obtaining labeled data, a challenge in streaming applications involving large amounts of data [47].

The most simple and popular approach for detecting concept drifts in a data stream is based on model performance monitoring [3, 27]. These methods assume that if the classifier performance substantially degrades in a given time compared to its recent history, a drift must have occurred, and its model needs to be updated. Drift Detection Method (DDM) [15], Early Drift Detection Method (EDDM) [2], and Adaptive Windowing (ADWIN) [5] are well-known examples of drift detectors based on this assumption and employ different strategies to verify when a drop in performance is a drift event.

ADWIN is the most representative and state-of-the-art drift detection technique among the different methods that follow the performance monitoring approach [3]. The ADWIN detects drifts by tracking the average of recently seen items from the stream using a varying window. Such a window can store a stream of bits or real-valued numbers. Although ADWIN can monitor any information from the stream, virtually all work from the literature employs it in a supervised way to monitor the performance of a classifier (e.g., [6, 19, 21, 24, 25, 28, 29, 31, 34, 35, 37, 39, 50]). The monitored values can be a binary representation of the prediction $y_i = \{0, 1\}$ in which 0 means a correct prediction of an example X_i, while 1 means an incorrect prediction, or it can be real values that represent an evaluation measure given recent data, such as accuracy or f-measure [27]. In both cases, ADWIN requires labeled data to monitor such information.

The main limitation of performance monitoring approach is the requirement of fully labeled data. Since obtaining labels in a streaming scenario in which data arrives at high-speed and are potentially unbounded is costly, this supervised setting is limited or even infeasible in many real-world problems [20, 45, 48]. Although some recent proposals focus on unsupervised drift detection, the employment of the state-of-the-art ADWIN method under this setting still needs to be explored.

We propose ADWIN-U (*Adaptive Windowing for Unsupervised Drift Detection*), a concept drift detector based on ADWIN that does not depend on labeled

data for change monitoring. We evaluated different approaches and settings to employ ADWIN in an unsupervised manner by monitoring simple descriptive statistics extracted from the examples or no cost information provided by a classification model, such as the confidence score of predictions.

We experimentally demonstrate that ADWIN-U can outperform the performance of its supervised version in different real-world stream applications when adequate statistics are extracted and monitored from data, such as skewness and kurtosis. Unlike well-used mean and standard deviation statistics, skewness and kurtosis extract higher-order information related to the distribution's asymmetry and *tailedness*. These statistics proved suitable for identifying distributional changes, making them particularly effective for unsupervised drift detection.

In addition to introducing ADWIN-U, we also address the challenges of evaluating drift detectors in the unsupervised setting. The general procedure to evaluate drift detectors is to measure the accuracy performance of a classifier updated according to the detections. However, this procedure does not consider the amount of labeled data requested for model retraining, favoring methods with a high rate of false alarms, , leading to frequent and unnecessary updates that require large volumes of labeled data. To address this, we propose a novel evaluation measure – *Balanced Accuracy by the Amount of Requested Labeled Data* (BAR), that considers the trade-off between accuracy and the amount of labeled data requested for the model updates. BAR prioritizes accurate detectors with a low false alarm rate and is particularly suitable for scenarios where labeled data are costly or impractical to obtain during the drift monitoring phase.

The main contributions of this work are summarized below:

- The proposal of an unsupervised drift detector based on a comprehensive investigation of approaches and strategies to adapt the state-of-the-art ADWIN method to the unsupervised setting. We explored and compared two distinct approaches (single detector and multiple detectors) and ten statistical measures extracted from the stream. Our findings demonstrate that multiple detectors using skewness or kurtosis provide superior results, forming a key component of the proposed ADWIN-U drift detector. The evaluation was conducted on ten real-world non-stationary data stream problems across various domains;
- A novel measure to evaluate unsupervised drift detectors that considers the trade-off between accuracy and proportion of requested labeled data. Such a measure prioritizes precise detectors with a low false alarm rate, and it is more adequate for realistic scenarios where labeled data is scarce or limited.

This article is organized as follows. We discuss the related works regarding unsupervised drift detection methods in Section 2. In Section 3, we introduce the fundamentals and definitions of data streams, concept drift, and drift detection using supervised methods. Our unsupervised drift detector is introduced in Section 4, and the proposed metric BAR to evaluate unsupervised drift detectors is presented in Section 5, respectively. The experimental evaluation and discussions are presented in Section 6. Finally, Section 7 presents our conclusions and future works.

2 Related work

Most supervised drift detectors require that the ground-truth labels be available almost immediately after every prediction for performance monitoring. The assumption of label availability after predictions is very optimistic and not realistic in many situations due to the high cost of manual labeling a massive amount of fast arrival data, restricting the use of supervised detectors in practical problems [43, 44]. Even with this restriction, most works concentrate on supervised solutions that depend on labeled data. As shown in a broader review of concept drift detection [30], 85% of solutions are supervised, while 12% are semi-supervised, and only 3% of works are unsupervised methods. However, there has been an increasing interest in unsupervised detectors in the last few years [17].

While supervised methods focus on detecting changes in the relationship between features and the target variable (i.e., real drift), unsupervised methods detect changes in features (i.e., virtual drift or feature change) due to the lack of labels [16, 44]. These methods generally detect drifts by statistically comparing the distributions of two windows with past and recent data from the stream. These windows can contain the raw data, a classifier output (e.g., probabilities or distances), or the estimated class labels by a model [53]. An alternative approach, as proposed in this work, is to extract statistical measures from raw data or classifier output to populate the monitored windows. Different strategies can be considered for updating the windows, statistical tests with varying assumptions, or different strategies can be employed to detect changes according to the type of data monitored in the stream. Next, we discuss some unsupervised detectors.

The Incremental Kolmogorov-Smirnov (IKS) [40] monitors each feature individually, checking by drifts employing the non-parametric statistical test Kolmogorov-Smirnov. If detected a change in the distribution of any feature, a drift is declared. The test uses a randomized tree to insert and remove examples, making it possible to efficiently perform the statistical test incrementally without recomputing the values from scratch.

The Discriminative Drift Detection (D3) [22] is a method that uses a discriminative classifier as a proxy to distinguish recent and old instances. The classifier is trained and tested periodically, aiming to distinguish whether the new samples from the stream are from a similar distribution as the old. The drift is detected according to the classifier performance, measured by the Area Under Curve (AUC). Since it is possible to obtain the classes *old* and *new* based on the arrival time of the stream examples, it does not depend on the actual labels of the problem for performance monitoring. Thus, a drift is detected if the distributions of old and new samples vary. One Class Drift Detection (OCDD) [23] is similar to the D3, but uses a One Class Classifier instead of logistic regression. OCDD monitors the outlier percentage in the following data samples as a signal for drift.

The Image-Based Drift Detector (IBDD) [44] is an unsupervised method suitable for detecting drifts in high-dimensional data streams. IBDD represents n-dimensional data using a 2-dimensional gray-scale image. To check for changes in the distributions of old and recent data, IBDD compares the similarities between the pair of images generated for these distributions using the Mean Squared Deviation (MSD). Then, the values returned by MSD are monitored over time using a dynamic threshold.

The Student-Teacher Unsupervised Drift Detection (STUDD) [10] algorithm compares the behavior of two models, a teacher, and a student, to detect drifts. Initially, the algorithm builds a teacher model using the first batch of labeled examples from the stream. The algorithm then creates an auxiliary model called the student, which mimics the behavior of the teacher model. As new examples arrive from the stream, the algorithm compares the prediction results of the student and teacher models. It is possible to monitor the student's imitation loss to detect concept drift. Concept drift is indicated in this process when the accumulated error between the models becomes significant.

3 Background

3.1 Data streams and concept drift

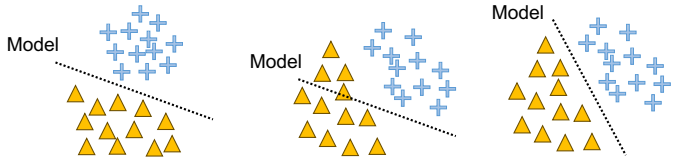
Real-world applications such as those related to Internet-of-Things (IoT) and sensors continuously generate data streams in non-stationary environments. Data streams are fast-paced and potentially infinite heterogeneous sequences of instances that arrive ordered by time. Formally, a data stream is defined by a set $\mathcal{S} = \{X_1, X_2, \ldots, X_t, X_\infty\}$, where X_t is a d-dimensional vector in the feature space that was observed at time t.

Systems modeled for mining data streams work over different constraints than those that deal with static batch data. For instance, storing all stream instances is infeasible due to the lack of memory. These systems also need to process and provide real-time responses once data arrives or at least before the arrival of the following example. Besides, many applications process high-speed streams (e.g., a seismic monitoring system processes a hundred observations per second [52]), requiring time-efficient solutions. Due to time constraints, even simple problems like duplicate item detection [12] can be challenging in the stream setting.

The most prominent challenge stream systems face is dealing with changes in the underlying data distribution, a phenomenon named *concept drift* [16]. Concept drifts affect the performance of classifiers, making them outdated when new concepts emerge due to the changes in data distributions. In classification, a concept drift occurs between the times t and $t + \Delta$, if $P_t(X, Y) \neq P_{t+\Delta}(X, Y)$, where P_t refers to the joint distribution at time t given a set of examples X and their class labels Y. Concept drifts constitute a central issue in stream learning since they are responsible for outdating predictive models, leading to degradation in accuracy.

Concept drift and its impact on a classification problem are illustrated in Figure 1. In Figure 1(a), we show a two-class problem in which a linear model successfully separates both classes. After concept drift, the class definitions change, and the current model becomes outdated, as illustrated in Figure 1(b). In Figure 1(c), we illustrate the model updated to fit the new concepts.

The changes in data distributions can occur in different manners and velocities. Figure 2 illustrates three types of drifts: incremental, abrupt, and gradual. In this figure, we represent the concepts by the color and shape of a geometric figure that starts as a light square at the beginning of the stream and becomes a dark circle at the end. The change to a new concept can have many intermediary concepts (incremental drift), change drastically (abrupt drift), or the gradual emergence of the new concept. Real-world applications can generate data with different changes



(a) Initial model (b) Outdated model (c) Updated model

Fig. 1 Impact of concept drift on a two-class classification problem.

and for different reasons. For example, a sudden shift in user preferences can lead to an abrupt concept drift in a recommender system. As users' interests evolve, their preferences may change significantly, causing drifts in the underlying patterns and relationships within the data.

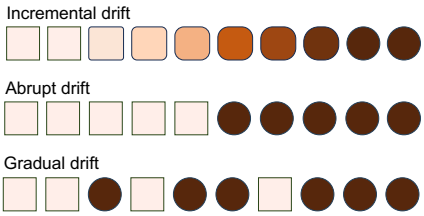


Fig. 2 Representation of different types of drift. The concepts represented by the shape of a geometric figure can evolve incrementally, abruptly, or gradually.

3.2 Drift detection

Stream classifiers use drift detectors to indicate the best times for the model update with recent data. An efficient detector must trigger model updates as soon as such changes are detected, avoiding unnecessary updates due to the costs of label requests and providing a stable accuracy over the stream.

We can categorize drift detection methods as *supervised* and *unsupervised*, according to the use of labeled data for drift monitoring [46]. Supervised methods assume the immediate availability of the actual class label of each example after processing it and use this information to monitor performance indicators, assuming that drifts cause performance drops. Some examples of popular supervised detectors based on error-monitoring are DDM [15], EDDM [2], Exponentially Weighted Moving Average for Concept Drift Detection (ECDD) [42], and ADWIN [5]. In the following, we briefly discuss how these methods work.

The DDM assumes that while the data is stationary (i.e., without drifts), the error distribution of a classifier follows a binomial distribution. To monitor changes, the method defines three distinct states for error evolution: *i) in-control*, when the error remains stable; *ii) warning*, when the error begins to rise but has not reached critical levels; and *iii) out-of-control*, when a significant increase in the error rate occurs. Specifically, a drift is detected by DDM when the condition

$p_i + s_i \geq p_{min} + 3 \times s_{min}$ is reached, where p_i represents the error rate and s_i the standard deviation at instant i, while p_{min} and s_{min} are the minimum recorded error rate and standard deviation up to that point. The warning state begins when $p_i + s_i \geq p_{min} + 2 \times s_{min}$. In this case, the method temporarily stores data in a short-term memory, which is later used to rebuild the classification model if the error transitions to the out-of-control state.

Inspired by DDM, the EDDM follows a similar approach of categorizing the error into three states, but the monitoring emphasizes the average distance between consecutive errors rather than solely the error rate. The method keeps track of a running average of the distance between errors, denoted as p'_i, and its associated standard deviation, s'_i. In this case, p'_i represents the mean distance between errors up to time i, while s'_i represents the standard deviation of these distances. Additionally, the method monitors the maximum recorded values of $p'_i + 2 \times s'_i$, referred to as $p'_{\max} + 2 \times s'_{\max}$, which are updated whenever $p'_i + 2 \times s'_i$ reaches a new maximum. A concept drift is detected when the ratio of Equation 1 is reached, in which the authors empirically find 0.9 as a threshold that indicates a significant change in the error distribution.

$$\frac{p'_i + 2 \times s'_i}{p'_{\max} + 2 \times s'_{\max}} < 0.9 \tag{1}$$

EWMA for Concept Drift Detection (ECDD) is a method that applies the Exponentially Weighted Moving Average (EWMA) [41] for concept drift detection. The EWMA technique estimates a mean of a sequence of random variables $X_1, X_2, \ldots, X_n$, progressively downweighting older data to emphasize recent observations. This exponentially weighted mean at time t is denoted by Z_t. The EWMA value Z_t at a point t is defined with a new X_t data value and weight λ in Equation 2.

$$\begin{aligned} Z_0 &= \mu_0, \\ Z_t &= (1 - \lambda) Z_{t-1} + \lambda X_t \end{aligned} \tag{2}$$

A change is flagged when the estimator satisfies the condition $Z_t > \mu_0 + L\sigma_{Z_t}$. This means a concept drift is detected when the current EWMA value Z_t exceeds the initial mean μ_0 by L times the standard deviation of the EWMA estimator (σZ_t), where L is a user-defined value named control limit. The first application of EWMA to concept drift detection was proposed in Yeh et al. [51] as a Bernoulli distribution. However, it depends on knowing the mean of the distribution in advance, which is not realistic in practical streaming applications. ECDD then formulates the EWMA using the estimated mean $\hat{\mu}_t$ and standard deviation $\hat{\sigma}_{Zt}$, as defined in Equation 3.

$$\begin{aligned} \hat{\mu}_{0,t} &= \frac{1}{t} \sum_{i=1}^{t} X_i = \frac{t-1}{t} \hat{\mu}_{0,t-1} + \frac{1}{t} X_t \\ \hat{\sigma}_{Z_t} &= \hat{\mu}_{0,t}(1 - \hat{\mu}_{0,t}) \sqrt{\frac{\lambda}{2 - \lambda}(1 - (1 - \lambda)^{2t})} \end{aligned} \tag{3}$$

ECDD alerts a concept drift when $Z_t > \hat{\mu}_{0,t} + L\hat{\sigma}_{Z_t}$. Estimating $\hat{\mu}$ online has implications for the choice of L. The authors use a Monte Carlo method [18], which defines L as a polynomial of degree 7 as a function of $\hat{\mu}$.

Since our proposal is an adaptation of the state-of-the-art supervised drift detector ADWIN for the unsupervised setting, we present details of this method in Section 3.3 to support our proposal using simple statistics from data.

3.3 ADWIN

ADWIN (ADaptive-WINdowing) [5] is one of the most popular and accurate supervised drift detectors from the literature. The algorithm idea is to maintain and monitor a variable-length window W with recent information obtained from the stream, such as the error rate of a classifier or its binary output in terms of hits and misses. In both cases, obtaining the labels of all examples from the stream is an essential step to maintain W.

The window W is partitioned into every possible two sub-windows with varying lengths, W_1 and W_2, that are used to determine if a change has happened. A drift is detected if any two sub-windows have distinct averages according to a confidence statistical bound ϵ derived from Hoeffding's theorem [26]. Upon detecting a drift, W_1 is replaced by W_2, and a new W_2 is initialized from scratch with the following stream examples.

To better understand the statistical bound employed by ADWIN to detect drifts, consider n_1 and n_2 as the lengths of windows W_1 and W_2, n as the sum $n_1 + n_2$, m as the harmonic mean of $\{n_1, n_2\}$, σ^2_W as the variance of the observed values in the window W, and a level of confidence δ. Equation 4 defines such a bound given the sub-windows W_1 and W_2. ADWIN pseudocode is presented in Algorithm 1.

$$\epsilon = \sqrt{\frac{2}{m}\sigma^2_W \ln(\frac{2n}{\delta})} + \frac{2}{3m}\ln(\frac{2n}{\delta}) \tag{4}$$

Algorithm 1 ADWIN

Input: A data stream S, a confidence level ϵ
1: $W \leftarrow \emptyset$
2: **for** $x_i \in S$ **do**
3: $W \leftarrow W \cup \{x_i\}$ (i.e., add x_i to the head of W)
4: **for** each split of W into $W = W_1 \cdot W_2$ **do**
5: **while** $|\hat{\mu}_{W_1} - \hat{\mu}_{W_2}| \geq \epsilon$ **do**
6: Drop elements from the tail of W
7: **end while**
8: **end for**
9: **end for**

The version presented in Algorithm 1 is computationally costly since the algorithm must verify every two subsets of a window W. To overcome this limitation, the authors proposed an efficient version by maintaining buckets of data via an exponential histogram technique [5]. The exponential histogram technique reduces the computational and space complexity of ADWIN by grouping data into buckets of varying sizes, where each bucket stores a summary of the data (e.g., counts or statistics) instead of individual data points. These buckets are organized so that

their sizes grow exponentially (e.g., $1, 2, 4, 8, \ldots$), allowing the algorithm to maintain a compact representation of the window W. When a new data point arrives, it is added to the smallest bucket, and buckets are merged as needed to maintain the exponential structure. This approach significantly reduces the number of subsets that need to be checked for concept drift, as the algorithm only examines the boundaries between buckets rather than all possible pairs of subsets in W.

Regarding computational cost, the exponential histogram ensures that operations such as adding a new data point or checking for drift are performed in constant time $O(1)$. Additionally, the space complexity is reduced to logarithmic $O(log(W))$ as the number of buckets grows logarithmically with the size of the window W. In this work, we consider such optimized version.

Although computationally efficient, the algorithm may be considered impractical in scenarios with a high cost for accessing labeled data, requiring unsupervised detectors as proposed in this paper. The access to class labels is an optimistic assumption made by methods such as DDM, EDDM, ECDD, and ADWIN.

4 ADWIN-U: Adaptive Windowing for Unsupervised Drift Detection

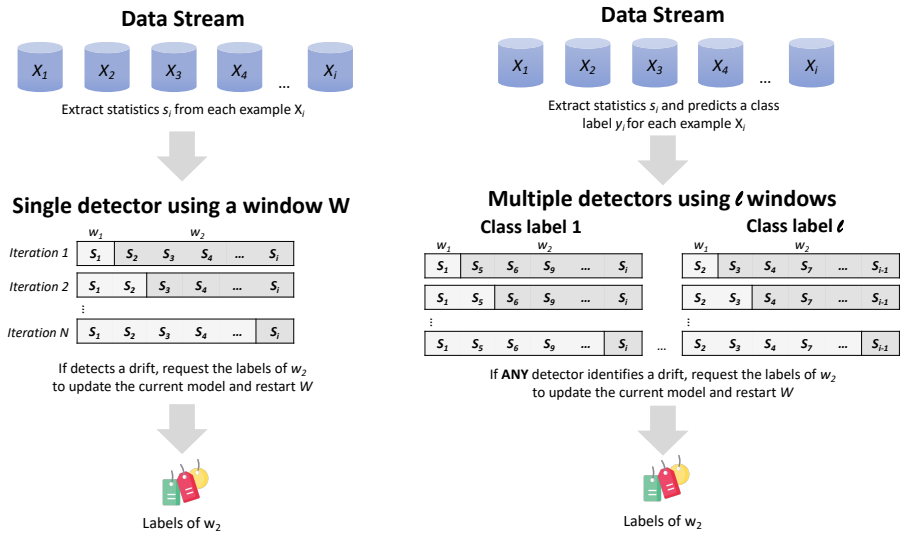
We propose ADWIN-U (Adaptive Windowing for Unsupervised Drift Detection), a drift detector based on the windowing strategy adopted by ADWIN to detect changes without requesting any labeled data for window monitoring. Instead, ADWIN-U monitors simple descriptive statistics from the examples (e.g., skewness, kurtosis) to detect changes in the stream following a fully unsupervised approach. As well as the original ADWIN, our detector can work coupled with a model. In this case, only a few labels (last seen examples) are requested for the model update after a drift is detected. However, no labels are requested during the drift detection/monitoring task.

4.1 Monitoring approaches

ADWIN-U extracts statistics from the data and stores it in single or multiple windows for drift monitoring. We do not maintain the windows explicitly, but a compressed bucket representation obtained by an exponential histogram technique [13] that allows using multiple windows efficiently. We propose two approaches for ADWIN-U according to the number of detectors and, consequently, the number of windows monitored:

- **Single detector:** we extract a statistic s_i from each stream example X_i and store it into a single window W that is monitored over time by the ADWIN detector. Thus, instead of monitoring changes in the error rate of a classifier, we monitor changes in a descriptive measure of the stream examples;
- **Multiple detectors:** we extract a statistic s_i from each stream example X_i while a supervised model predicts a label $y_i \in Y$ for the example, in which $Y = \{c_1, c_2, \ldots, c_l\}$ is a set of l class labels for the classification task. Then, the statistic s_i is stored in a corresponding window W_y according to the predicted class label y. Thus, in a multi-class problem with l class labels, ADWIN-U maintains and monitors l independent windows. Besides, if any detector identifies a concept drift, all of them are restarted.

Figure 3 illustrates the single and multiple detectors approaches. While the single detector approach is model-independent, using multiple detectors requires a classification model to maintain the statistics separated by class into l windows according to the predictions. However, it is not a limitation since rarely a drift detector is not coupled with a classification model. It is also important to note that the assignment of the statistic s_i to a window W_y is solely based on the predicted labels provided by a model, not requiring the actual labels.



(a) Single detector (b) Multiple detectors

Fig. 3 Proposed approaches for ADWIN-U. The method can monitor drifts using a single or multiple detectors. When multiple detectors are considered, we need a classification model to separate the statistics s_i extracted from the examples according to the predicted class y.

4.2 Overview of statistical measures for monitoring

Our unsupervised approach can monitor any statistics extracted from the data following the two approaches previously discussed. This paper evaluates the performance of ten simple statistical measures. The primary criterion guiding this selection was computational efficiency and the ability to update the statistics incrementally in an online fashion, which is critical for streaming applications. In this regard, we prioritized statistics that could be updated incrementally with each new data point, allowing for continuous monitoring without storing large amounts of historical data [49].

The statistics are shown in Table 1. In these equations, we consider that x is the sample we want to extract a statistical measure, and n is the number of observations in the sample. Since we extract these values of each example from the stream, n will be the dimensionality of the example. Besides these statistics,

we also monitor the highest probability outputted by the classification model given an example. This information is denoted as *probability* in the table results.

Table 1 Statistical measures considered by ADWIN-U in the experimental evaluation.

Statistical measure	Equation
Mean	$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$
Median	$\widetilde{x} = \begin{cases} x_{(n+1)/2}, & \text{if } n \text{ is odd} \\ \frac{1}{2}\left(x_{n/2} + x_{n/2+1}\right), & \text{if } n \text{ is even} \end{cases}$
Variance	$\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n}$
Standard deviation	$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n}}$
Harmonic mean	$\mathrm{HM} = \frac{n}{\sum_{i=1}^{n} x_i^{-1}}$
Geometric mean	$\mathrm{GM} = \sqrt[n]{\prod_{i=1}^{n} x_i}$
Skewness	$g_1 = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^3}{n\sigma^3}$
Kurtosis	$\kappa = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^4}{n\sigma^4}$
Coefficient of Variation	$\mathrm{CV} = \frac{\sigma}{\overline{x}}$
Median absolute deviation	$\mathrm{MAD} = \mathrm{median}(\lvert x_i - \widetilde{x}\rvert)$

Although mean, standard deviation and variance are popular measures to monitor changes in data distribution, mainly for supervised methods, we note through our experiments that skewness and kurtosis [32] presented the best results in the unsupervised setting, as we will further discuss in Section 6.2. These statistics measure higher-order information from data, capturing aspects of the distribution beyond central tendency and dispersion. In the following, we present a brief discussion of these measures [9].

Skewness is a measure of the lack of symmetry of a distribution. A distribution is symmetric (or zero skewness) if it looks the same to the left and right of the center point, while the mean, median, and mode are the same. Conversely, a distribution has positive skewness when the tail on the right side is longer (i.e., the mean and median will be greater than the mode) and negative when the tail of the left side of the distribution is longer or than the tail on the right side (i.e., the mean and median will be lower than the mode). We illustrate these cases in Figure 4.

Kurtosis is a measure that defines how heavy-tailed or light-tailed a distribution is when compared to a Gaussian distribution. For example, in Figure 5, we illustrated a Gaussian distribution that has zero kurtosis (left), a *sharper* distribu-

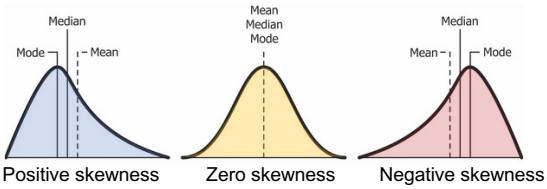


Fig. 4 Example of asymmetric (with positive and negative skewness) and symmetric distributions.

tion that has positive kurtosis (center), and a *flatter* distribution that has negative kurtosis (right).

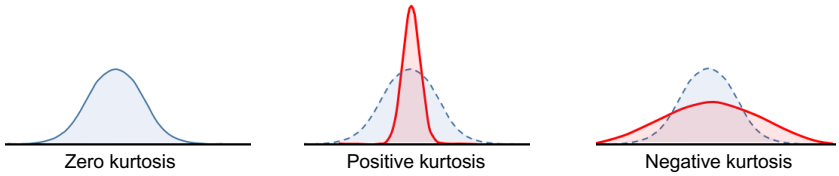


Fig. 5 Example of data distributions with positive and negative kurtosis.

Variance measures the spread of data points around the mean. Higher variance indicates that the data points are more dispersed, while lower variance indicates that they are more tightly clustered around the mean. In Figure 6, we illustrate two distributions around the mean in zero with low and high variances of approximately 0.2 and 4, respectively.

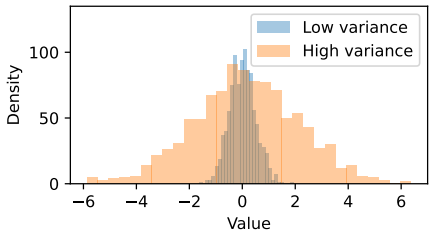
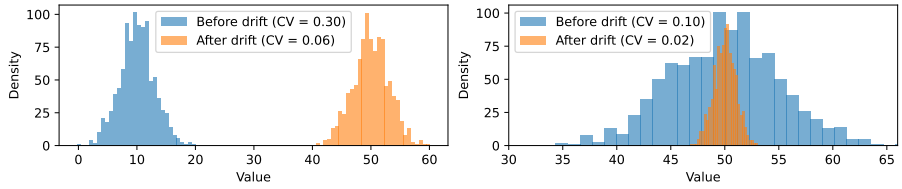


Fig. 6 Example of data distributions with low and high variance.

The Coefficient of Variation (CV) measures the relative spread of data by dividing the standard deviation by the mean. This measure captures variations in the data's spread and central tendency. In Figure 7-*left*, we illustrate a distribution before and after drift where the standard deviation is the same, but the means differ, resulting in different CV values. In Figure 7-*right*, we present another scenario where the distributions have the same mean but different standard deviations before drift, also leading to distinct CV values that can be used to detect concept drift.



(a) Change in mean (b) Change in standard deviation

Fig. 7 Example of Coefficient of Variation (CV) in a data distribution before and after concept drift with mean and standard deviation changes. In both cases, the values computed by CV reflect the drift.

The Harmonic Mean (HM) is a type of average particularly useful when dealing with rates, ratios, or scenarios where lower values should have a greater influence. Unlike the arithmetic mean, the harmonic mean assigns greater weight to smaller values, making it less sensitive to outliers with extremely high magnitudes. While it is not typically used for comparing data distributions, the harmonic mean is effective when data contains features in which smaller values carry more significance, or feature magnitude fluctuations are inversely proportional to their importance. On the other hand, the Geometric Mean (GM) calculates the central tendency and is particularly appropriate for data involving growth rates or multiplicative relationships, as it captures the compounding effects inherent in such data. Both statistics (HM and GM) are alternatives to the traditional arithmetic mean.

The Median Absolute Deviation (MAD) calculates the median of the absolute deviations of data points from the distribution's median. Unlike the standard deviation, which is sensitive to outliers, MAD focuses on the central portion of the data. Such characteristic makes it particularly useful in scenarios where extreme values may distort traditional measures of dispersion such as standard deviation or variance.

4.3 Computational complexity

The ADWIN-U complexity is analogous to its supervised version. For the single detector approach, the worst-case time complexity is $\mathcal{O}(\log(W))$, and for the multiple detectors approach, it is $\mathcal{O}(l \log(W))$, where l is the number of detectors (i.e., the same number of classes in a problem), and W is the window length.

However, it is important to consider the cost of extracting the statistical measure that populates the monitored windows. For efficient processing, we need to consider incremental computation of the measure instead of recomputing the values from scratch as new examples arrive [49]. For example, to compute skewness incrementally, we must store information such as the current mean, variance, and third central moment. As new examples arrive, the mean is updated, and the second and third central moments are adjusted accordingly. This allows the efficient calculation of skewness without iterating through all previous examples, reducing both time and space complexity.

5 Balanced Accuracy by the Amount of Requested Labeled Data (BAR): an evaluation measure for realistic stream settings

A concept drift detector is an algorithm that takes a data stream as input and outputs an alarm if a change in the distribution of the examples is detected, indicating the position in the stream where the changes occur [4]. Evaluating the drift detector's performance is challenging due to the stream dynamics and lack of ground-truth information about the changes. In an ideal scenario, we know the exact positions of all changes in a stream, allowing us to evaluate the detector by measures that consider the compromise between detecting actual changes and avoiding false alarms, such as Mean Time between False Alarms (MTFA), Mean Time to Detection (MTD), and Missed Detection Rate (MDR) [4]. However, this ground truth is completely observable only in simulated and artificial data, limiting the use of such measures in practice.

Since the primary role of a concept drift detector is to trigger updates in a predictive model at the right times to maintain stable performance in a non-stationary environment, it is usual to consider the accuracy of a model coupled with a detector as an indirect measure of the detector performance.

The main limitation of this approach is that it does not consider the number of changes detected. For example, a detector that identifies a change in each new instance will keep the classifier updated with the most recent data for the entire stream. In this case, the detector can lead to an accurate model, which requires 100% of labeled data for continuous updates. However, in realistic stream settings, data arrives at high-frequency rates, and requesting labels is costly or unfeasible.

Due to the costs of obtaining labeled data, unsupervised drift detectors are more suitable in stream settings. In this direction, evaluating a detector coupled with a classifier also needs to consider the trade-off between accuracy and the amount of labeled data requested for the model update, favoring accurate detectors that do not make the classifier highly dependent on labeled data.

Inspired by the traditional f-measure, which considers the harmonic mean of precision and recall, we propose an evaluation measure for unsupervised concept drift detectors named *Balanced Accuracy by the Amount of Requested Labeled Data* (BAR). The BAR measure considers the harmonic mean of the model accuracy and the proportion of labeled data requested for its updates over the stream evaluation. It must be noted that accuracy is the primary metric used to compose our measure. However, it can be replaced by other measures adequate to the evaluation settings or problem, such as Kappa statistic [7]. Equation 5 presents the BAR measure.

$$BAR = 2 \times \frac{accuracy \times (1 - requested\ labels)}{accuracy + (1 - requested\ labels)} \tag{5}$$

To illustrate the trade-off between accuracy and requested labels captured by the BAR measure, we show six scenarios in Table 2. In these scenarios, we have strong, average, and weak classifiers that can depend on or be independent (or less dependent) of requesting labels. Ideally, the best scenario is an accurate classifier (strong) that requests labels a few times (label requesting independent). In this case, we consider the detector precise in identifying the most relevant changes only at the right times.

In Table 2, we note the average and label independent classifier placed as the second-best scenario according to our measure. A strong but label dependent

Table 2 BAR values obtained in varying scenarios considering strong, average, and weak classifiers that can depend on labels or be independent of such requests.

Scenario	Accuracy (%)	Requested labels (%)	**BAR**
Strong classifier Label requesting dependent	90	90	0.18
Strong classifier Label requesting independent	90	20	0.85
Average classifier Label requesting dependent	70	90	0.18
Average classifier Label requesting independent	70	20	0.75
Weak classifier Label requesting dependent	30	90	0.15
Weak classifier Label requesting independent	30	20	0.44

classifier is ranked near the bottom of the list with BAR of 0.18. We find the weak and label dependent classifier at the bottom of the ranking with BAR of 0.15.

It should be noted that lower BAR values are not an unquestionable indicator of a poor drift detector. For some complex datasets (e.g., with a large number of underrepresented classes), it is challenging to have an accurate classifier, which will inevitably reduce the measure values. Thus, it is essential to compare the results of a detector with an upper bound that could be the BAR values obtained by baseline methods that request 100% of labels.

6 Experimental evaluation

The algorithms were implemented in Python using the ADWIN implementation provided by scikit-multiflow [38]. All our experiments are reproducible, the source codes and additional results are publicly available on the paper's website [1].

6.1 Settings

We performed all our experiments on ten real-world stream datasets from the USP Data Stream Repository [46]. In Table 3, we describe the number of examples used to build an initial model when needed (e.g., for the multiple detectors approach of ADWIN-U, and Blind/Persistent baselines), the number of test examples, features, and classes of each dataset.

Table 3 Description of the evaluated non-stationary stream datasets.

Dataset	# Initial train	# Test	# Features	# Classes
Electricity	1,000	44,312	8	2
GasSensor	3,910	10,000	128	6
Insects-Abrupt	1,000	51,848	33	6
Insects-Gradual	1,000	23,150	33	6
Insects-Incremental	1,000	56,018	33	6
LADPU	300	22,650	96	10
Outdoor	2,000	2,000	21	40
Rialto	5,000	77,250	27	10
StarLightCurves	1,000	8,236	1,024	3
Yoga	300	3,000	426	2

We based our analysis on three evaluation measures: *i*) accuracy, *ii*) percentage of requested labeled data for the model update after drift detections, and *iii*) our proposed measure BAR, which combines both measures.

To compute the evaluation metrics, we employ a rolling holdout validation approach in which the training and test sets are defined based on drift positions detected over the stream. Initially, we train a model considering the first examples N from the stream, as defined in Table 3, and then we evaluate the model performance on the remaining examples. Since the detector is coupled with a model, we used the last-seen examples for training (model update) whenever a change is detected, and then we tested on the following examples, repeating the process until the end of the stream. Thus, we can evaluate if the model updates are performed at the best times. All results consider a Random Forest [8] as the base learner and a confidence level $\delta = 0.002$ for the ADWIN.

We compare our proposal against the following baselines that simulate two extreme and opposite situations of a drift detector coupled with a classifier:

- **Blind:** a static classifier trained a single time with the initial examples from the stream and never updated. This baseline simulates a blind drift detector coupled with a classifier that does not identify any drift. This scenario does not require any labeled data in the test phase;
- **Persistent:** a classifier that updates its model with every arrival example from the stream. This baseline simulates a persistent detector that identifies a drift for every new example processed, triggering continuous model updates and requiring 100% of labeled data in the test phase.

In addition to the baselines, we compare the results of ADWIN-U against an unsupervised drift detector based on the use of the Kolmogorov-Smirnov statistical test (KS-TEST) [33, 40] that monitors changes in the distribution of each feature of the examples. We also compare our unsupervised detector against the supervised version of ADWIN. Specifically, we monitor two supervised information from the stream, leading to the following versions: *i*) ADWIN (error), which monitors changes in the classifier´s error rate, i.e., real-values between 0 and 1; and *ii*) ADWIN (output), which monitors the classifier´s hits and misses, i.e., binary values where 0 means the prediction was wrong and 1 means the prediction was correct.

6.2 Analyzing and choosing the best setting for ADWIN-U

As discussed in Section 4, ADWIN-U can follow two approaches i) using a single detector and ii) multiple detectors. In addition, different statistical measures can be extracted from each stream example for the drift monitoring in both cases. In this section, we analyze the performance of both approaches using different statistical measures.

In order to find the best setting, we can initially identify the best statistical measure for each approach and then compare them. Table 4 shows the accuracy achieved by ADWIN-U with multiple detectors using ten different statistical measures and using the probability outputted by the classifier. In this table, the last column indicates the rank position of each algorithm's setting according to the Friedman test [14]. The rank position of a setting represents its mean ranks across

all outcomes, where low ranks indicate that a setting wins more often than its competitors with higher ranks.

Table 4 Accuracy of ADWIN-U with multiple detectors and different statistical measures.

	Elec	GasSensor	Insects-A	Insects-G	Insects-I	LADPU	Outdoor	Rialto	SLCurves	Yoga	Rank
Mean	77.62	66.00	46.09	37.28	38.81	52.60	48.80	49.92	23.22	56.23	6.70
Median	74.29	65.89	47.52	50.28	16.96	51.32	48.80	49.92	89.27	56.23	6.65
Variance	77.28	69.33	54.27	36.44	39.81	48.28	48.80	51.07	23.22	56.23	6.10
HM	73.31	55.28	46.70	36.04	12.54	49.02	48.80	49.92	25.45	56.23	8.80
GM	75.52	61.61	46.77	48.13	12.54	50.72	48.80	49.92	85.45	56.23	7.60
Std.	76.71	64.00	45.97	60.47	53.06	53.44	48.80	32.49	23.22	56.23	6.60
Skewness	76.59	68.87	57.14	52.87	60.87	58.80	48.80	58.12	89.30	63.43	2.95
Kurtosis	76.55	72.00	60.75	60.01	59.75	54.47	18.75	65.59	89.63	74.47	3.00
CV	76.20	58.76	62.18	64.06	59.31	53.49	48.80	56.27	23.22	56.23	5.00
MAD	75.08	62.87	53.88	46.09	16.92	45.49	48.80	49.92	88.60	69.87	6.75
Probability	76.01	26.26	49.45	52.20	57.10	56.08	48.80	40.53	85.81	67.33	5.85

In Table 4, we note that the most accurate statistical measure was skewness, followed by kurtosis, while Harmonic Mean (HM) and Geometric Mean (GM) showed the lowest accuracies. Table 5 shows the BAR results.

Table 5 BAR of ADWIN-U with multiple detectors and different statistical measures.

	Elec	GasSensor	Insects-A	Insects-G	Insects-I	LADPU	Outdoor	Rialto	SLCurves	Yoga	Rank
Mean	41.58	42.00	53.24	47.74	46.41	49.13	65.59	66.59	37.68	71.99	7.00
Median	35.17	49.96	55.73	46.41	26.23	49.62	65.59	66.59	88.69	71.99	6.05
Variance	44.34	46.31	66.60	48.89	47.22	62.98	65.59	62.48	37.68	71.99	5.20
HM	55.43	43.36	61.97	45.88	22.29	50.17	65.59	66.59	38.01	71.99	5.90
GM	35.90	43.45	55.55	49.04	22.29	48.02	65.59	66.59	89.88	71.99	6.40
Std.	42.83	43.63	43.05	60.45	60.55	57.34	65.59	36.07	37.68	71.99	6.10
Skewness	47.46	52.49	47.47	47.91	59.9	49.03	65.59	42.71	90.49	68.54	5.75
Kurtosis	48.65	52.37	48.59	49.35	58.84	49.98	30.30	49.23	86.70	78.55	4.90
CV	46.44	53.05	47.12	62.28	46.07	45.93	65.59	43.04	37.68	71.99	6.20
MAD	43.77	45.47	70.03	52.73	27.28	55.24	65.59	66.59	65.73	69.58	5.25
Probability	45.60	40.95	51.13	60.42	56.48	47.62	65.59	35.70	82.38	64.68	7.25

For BAR, which considers the trade-off between accuracy and the number of requested labels for model retraining, we note in Table 5 that Kurtosis showed the best results followed by Variance when ADWIN-U employs multiple detectors. On the other hand, the class probability outputted by the model obtained the worst results. We provide a detailed discussion regarding the number of requested labels by ADWIN-U to interpret the BAR results better in Section 6.3.

In Table 6 and Table 7, we show the accuracy and BAR results, respectively, obtained by ADWIN-U using a single detector and different statistical measures. In this case, Geometric Mean (GM) obtained the best accuracies and BAR results over the ten datasets evaluated.

In summary, the best statistical measures for the multiple detectors approach were skewness and kurtosis, while geometric mean (GM) shows the best general results for the single detector approach. In order to compare both approaches, we show a paired comparison in Figure 8, considering the best statistical measure for each approach. In this figure, each point represents the result obtained for a dataset. The points are placed on the x-axis according to the results obtained by the ADWIN-U with multiple detectors and on the y-axis according to the result obtained by the ADWIN-U with a single detector. Thus, the points below the diagonal represent datasets in which the multiple detectors approach outperforms the single detector approach.

Table 6 Accuracy of ADWIN-U with a single detector and different statistical measures.

	Elec	GasSensor	Insects-A	Insects-G	Insects-I	LADPU	Outdoor	Rialto	SLCurves	Yoga	Rank
Mean	74.35	62.54	50.74	28.86	44.36	57.08	48.80	49.92	23.22	56.23	5.70
Median	73.87	57.85	56.55	38.13	12.96	56.9	48.80	49.92	73.24	56.23	5.60
Variance	77.84	66.62	51.36	28.86	46.66	46.55	48.80	38.3	23.22	56.23	6.00
HM	74.04	54.63	53.74	32.47	30.43	54.68	48.80	49.92	23.22	56.23	6.75
GM	76.6	60.4	56.13	41.76	34.75	56.10	48.80	49.92	23.22	56.23	4.75
Std.	77.94	60.04	46.39	51.67	46.18	46.71	48.80	34.91	23.22	56.23	5.75
Skewness	74.62	56.66	52.57	55.00	43.76	57.15	17.2	41.45	84.49	56.23	5.60
Kurtosis	75.62	68.98	48.64	53.05	43.93	55.22	15.05	44.56	23.22	56.23	5.85
CV	75.36	46.6	45.83	50.8	52.15	53.78	18.55	44.17	23.22	56.23	6.95
MAD	76.49	58.57	54.83	39.03	12.54	46.70	48.80	49.92	23.22	56.23	6.05
Probability	75.79	27.26	44.65	43.63	58.53	43.51	46.85	34.45	86.17	56.23	7.00

Table 7 BAR of ADWIN-U with a single detector and different statistical measures.

	Elec	GasSensor	Insects-A	Insects-G	Insects-I	LADPU	Outdoor	Rialto	SLCurves	Yoga	Rank
Mean	30.28	40.55	44.39	36.95	34.13	44.78	65.59	66.59	37.68	71.99	5.60
Median	39.69	38.46	36.71	23.18	14.10	35.53	65.59	66.59	69.33	71.99	6.40
Variance	39.08	44.53	32.39	36.95	40.73	22.95	65.59	26.43	37.68	71.99	6.60
HM	51.13	40.86	29.75	27.02	35.17	25.69	65.59	66.59	37.68	71.99	6.45
GM	45.62	41.49	38.15	45.44	40.81	28.75	65.59	66.59	37.68	71.99	4.85
Std.	28.38	47.55	35.99	23.57	40.9	37.51	65.59	29.46	37.68	71.99	6.05
Skewness	37.88	52.53	47.92	34.19	34.67	29.51	23.02	33.05	79.76	71.99	5.40
Kurtosis	59.39	48.13	50.08	23.37	32.82	27.85	22.56	33.25	37.68	71.99	6.15
CV	30.8	40.72	34.6	30.2	46.15	33.85	27.61	33.2	37.68	71.99	6.85
MAD	48.84	46.06	30.24	30.33	22.29	49.50	65.59	66.59	37.68	71.99	5.35
Probability	35.82	42.07	40.75	21.22	40.72	37.00	33.77	27.76	81.37	71.99	6.30

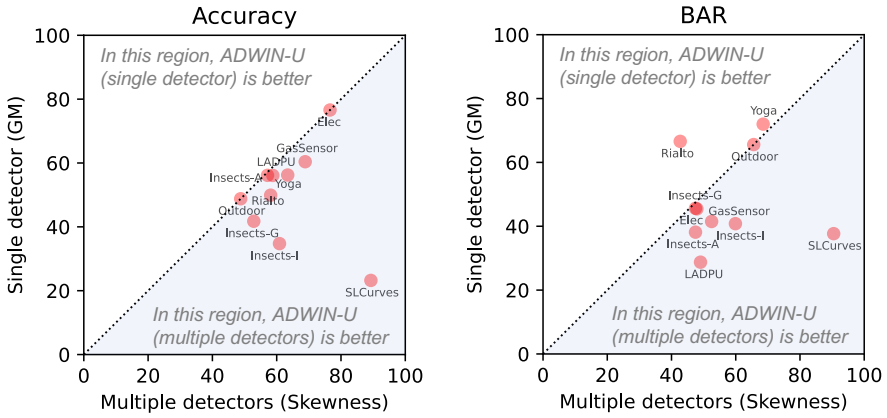


Fig. 8 Comparison of ADWIN-U using multiple detectors with skewness as statistical measure for monitoring and a single detector using geometric mean for monitoring. Each point represents the result of a dataset using both approaches.

In Figure 8 (*left*), we note that ADWIN-U with multiple detectors outperforms the single detector approach for all datasets, considering the accuracy. For BAR, we note in Figure 8 (*right*) that the ADWIN-U with multiple detectors outperforms the single detector approach for most of the datasets, except for Rialto and Yoga.

From the analysis conducted in this section, we realized that the best setting for ADWIN-U is to consider the multiple detectors approach monitoring the skewness or kurtosis. Thus, we consider this setting as the default for the following analyses.

6.3 Comparison of ADWIN-U against rivals

We compare the results of the best settings identified for ADWIN-U, i.e., with multiple detectors and using skewness and kurtosis as statistical measures for drift monitoring, against the two supervised versions of ADWIN, KS-TEST, and the two baselines (Blind and Persistent). In Table 8, we show the comparison considering the accuracy as a evaluation measure.

Table 8 Accuracy of classifiers with supervised and unsupervised drift detectors for updates.

	Elec	GasSensor	Insects-A	Insects-G	Insects-I	LADPU	Outdoor	Rialto	SLCurves	Yoga
ADWIN-U (skew)	76.59	68.87	57.14	52.87	60.87	58.80	48.80	58.12	89.30	63.43
ADWIN-U (kurt)	76.55	72.00	60.75	60.01	59.75	54.47	18.75	65.59	89.63	74.47
KS-TEST	77.71	56.17	72.34	74.35	64.37	71.33	36.95	46.83	91.62	77.30
ADWIN (output)	75.26	59.41	46.96	46.01	58.63	60.75	34.10	37.48	86.61	74.13
ADWIN (error)	67.04	58.93	48.61	37.30	39.09	44.67	46.75	54.64	42.18	70.07
Persistent	87.27	92.36	74.60	76.73	65.48	80.11	63.55	65.46	92.16	78.30
Blind	64.85	56.16	53.88	25.56	12.54	31.79	48.80	49.92	23.22	56.23

Table 8 shows that the most accurate classifier is the Persistent baseline since this classifier is updated over the entire stream requiring 100% of labels. For the unsupervised approaches, we note that ADWIN-U (kurtosis) shows competitive results with KS-TEST, and both algorithms outperform the performance of supervised versions of ADWIN. The poor performance of the Blind baseline evidences the importance of a drift detector to trigger model updates to classify the examples of these non-stationary datasets.

Considering the best results of ADWIN-U (i.e., with kurtosis) and ADWIN (i.e., with output), we note that the supervised version slightly outperforms our proposal in the LADPU and Outdoor datasets. A key characteristic shared by these datasets is their high number of classes. Specifically, the Outdoor dataset contains 40 classes, most of which are not represented in ADWIN-U's initial training set. This class imbalance poses a challenge for ADWIN-U, as our approach assigns a detector per predicted class. Consequently, the emergence of new classes may cause examples to be incorrectly associated with existing detectors.

In order to statistically compare the results obtained by the unsupervised drift detectors over the ten datasets evaluated, we performed the Friedman test with a significance level of 5% (p-value < 0.05) and the Nemenyi posthoc test [14]. We visually represent the results of this test using a critical difference diagram. In this diagram, we sort the methods according to their average ranking outputted by the statistical test. The methods connected by a line do not present statistically significant differences. Figure 9 exhibits the diagram considering the results previously presented in Table 8. In this diagram, the unsupervised detectors KS-TEST and ADWIN-U (kurtosis) show results without statistical difference compared to the supervised Persistent baseline.

As discussed in Section 5, evaluating drift detectors is challenging, and analyzing the accuracy of a classifier updated according to the detector is a simplified alternative. One of the main reasons is the lack of ground-truth information about the drift position for most datasets. For datasets such as Insects-Abrupt, StarLightCurves, and Yoga, we have this information from its donors [44, 46]. Thus, we can use this information to compare the actual drift positions against the positions returned by the detectors.

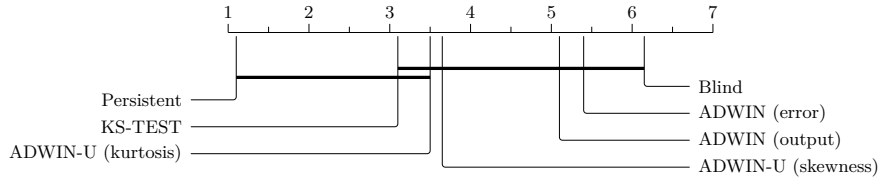
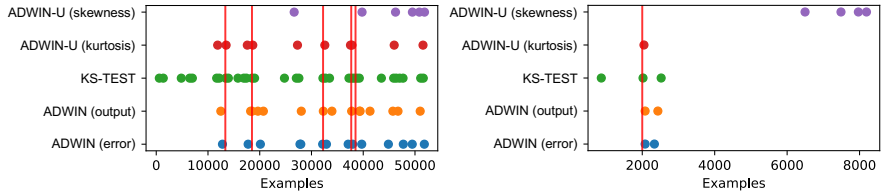


Fig. 9 Critical difference diagram regarding the accuracy results.

In Figure 10, we compare the drift positions for Insects-Abrupt and StarLight Curves datasets. In this figure, we indicate the actual drift positions for each dataset with vertical red lines. For example, while the Insects-Abrupt dataset has five actual drifts, StartLightCurves has a single abrupt drift point at the example 2,000 from the stream. The aligned points in a row illustrate the drifts identified by a detector. Thus, the most efficient detectors have points close to the red lines and a low rate of false alarms.



(a) Insects-Abrupt (b) StarLightCurves

Fig. 10 Drift positions identified by each detector. The vertical lines are the actual drift positions, and the points in a row indicate the drifts detected by a method.

It should be noted that the classifier updated according to the KS-TEST achieved 72.34% accuracy for the Insects-Abrupt dataset, while ADWIN-U (kurtosis) achieved 60.75%. However, we note in Figure 10 that KS-TEST shows many false alarms, while ADWIN-U (kurtosis) shows the lower false alarm rate between the detectors. For the StartLightCurves dataset, only ADWIN-U is absent from false alarms with a single detection very close to the actual drifts. For this dataset, KS-TEST shows 91.62% accuracy while ADWIN-U (kurtosis) obtained 89.63%.

These results show that more sensitive detectors can lead to an accurate classifier due to the constant updates. However, it does not guarantee precise detection of changes in data distribution and makes evident the need for an evaluation measure that considers the costs associated with label requests such as BAR. In Figure 11, we show the number of drifts detected by each method, in which KS-TEST is a more sensitive method for most datasets (e.g., Insects-Abrupt, Insects-Gradual, Insects-Incremental, LADPU, and Yoga) identifying a higher number of drifts.

Table 9 shows a comparison considering the BAR measure. For this metric, we note that ADWIN-U (kurtosis) shows the best results for most datasets, and the Blind classifier is a competitive baseline. Since both ADWIN versions and the

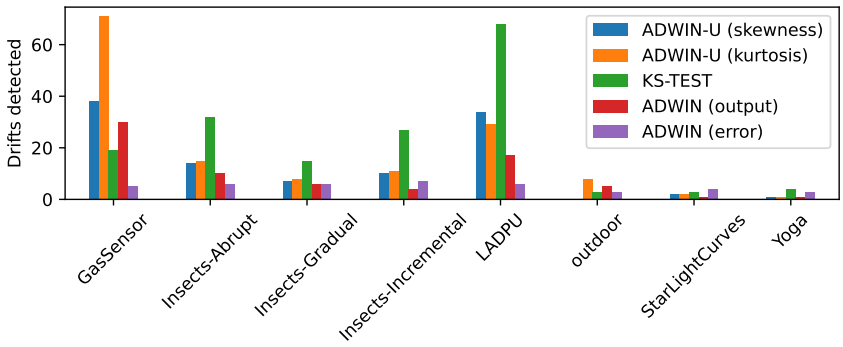


Fig. 11 Number of drifts detected by each method.

Persistent classifier are fully supervised approaches, they show the minimal BAR result for all datasets.

Table 9 BAR results of classifiers with supervised and unsupervised drift detectors.

	Elec	GasSensor	Insects-A	Insects-G	Insects-I	LADPU	Outdoor	Rialto	SLCurves	Yoga
ADWIN-U (skew)	47.46	52.49	47.47	47.91	59.90	49.03	65.59	42.71	90.49	68.54
ADWIN-U (kurt)	48.65	52.37	48.59	49.35	58.84	49.98	30.30	49.23	86.70	78.55
KS-TEST	1.40	9.18	70.70	70.81	69.66	17.44	29.82	0.64	86.43	67.56
ADWIN (output)	0	0	0	0	0	0	0	0	0	0
ADWIN (error)	0	0	0	0	0	0	0	0	0	0
Persistent	0	0	0	0	0	0	0	0	0	0
Blind	78.68	71.93	70.03	40.71	22.29	48.24	65.59	66.59	37.68	71.99

The critical difference diagram considering the BAR measure is presented in Figure 12. ADWIN-U (kurtosis) is ranked first, with no statistical difference to Blind, ADWIN-U (skewness), and KS-TEST. Conversely, Persistent baseline and supervised ADWIN are in the last positions.

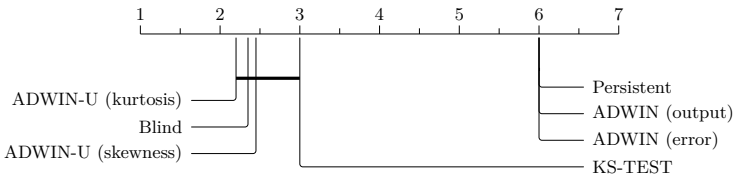


Fig. 12 Critical difference diagram regarding the BAR results.

Previously, we discussed the average BAR results for all datasets considering the entire stream in Table 9. However, it is possible to analyze BAR over time. Thus, we show the BAR results for Electricity and GasSensor datasets in Figure 13. Since both versions of ADWIN request 100% of labels for drift monitoring, their results are constant over the stream. For the three datasets presented, both versions of ADWIN-U outperform KS-TEST over time in the entire stream.

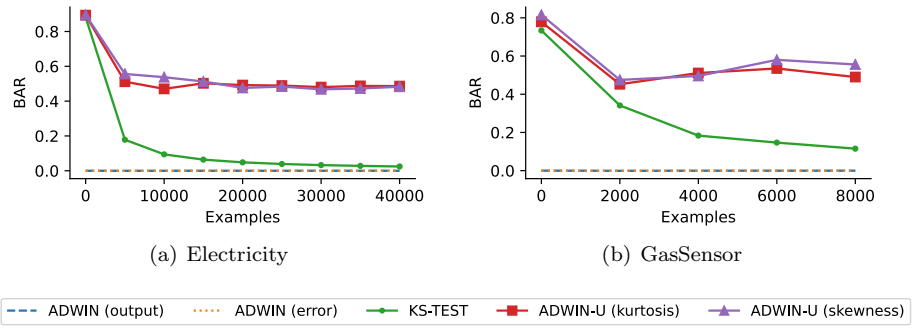


Fig. 13 BAR results over time.

To better understand the BAR results, Figure 14 shows the number of labels requested by each drift detector over the stream for four datasets. The main advantage of ADWIN-U is to show competitive accuracy results while requesting a small amount of data. This characteristic is evidenced by the BAR results and the number of labeled data requested, as illustrated in Figure 14. We note in this figure that for most cases, both versions of ADWIN-U are the methods that request fewer data over time. For some datasets, such as Electricity, GasSensor, and Rialto, KS-TEST requires as much data as the fully supervised ADWIN version.

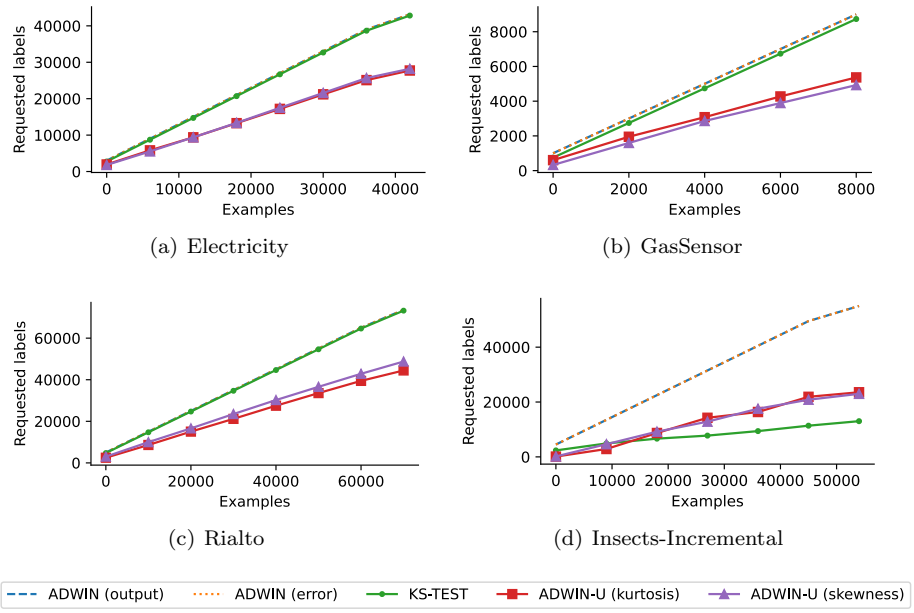


Fig. 14 Number of requested labels by each drift detector over the stream.

6.4 Discussion

The superior results obtained by ADWIN-U compared with its supervised version raise an important question regarding the factors influencing such performance differences. While the evident differences in the approach using multiple detectors are significant, we hypothesize that two key factors may contribute to this outcome, though other factors might also play a role:

- Base learner behavior: The supervised version of ADWIN is inherently model-dependent, as it relies on detecting concept drift based on increases in the base learner's error rate. This dependency can lead to false drift detections driven by factors unrelated to actual changes in the data distribution. For instance, in the case of tree-based models, errors may temporarily rise due to suboptimal splits or overfitting to transient patterns rather than actual changes in the underlying data. Furthermore, the emergence or disappearance of classes could also increase the error rate temporally, even in the absence of a true distributional change;
- Data characteristics: The nature of the dataset plays a pivotal role in the comparative performance of ADWIN-U and its supervised counterpart. Concept drifts resulting from subtle shifts in feature values that do not immediately impact the predictive performance of a model can be more effectively detected by ADWIN-U. Its unsupervised approach focuses on tracking variations in feature distributions rather than solely relying on performance metrics. In contrast, the supervised version may fail to promptly detect such shifts, particularly if they do not immediately degrade the model's accuracy.

By accounting for these factors, it becomes evident that the observed performance improvements of ADWIN-U are not solely attributable to algorithmic differences but also reflect intrinsic dependencies on model behavior and dataset characteristics.

7 Conclusions

Stream data measured over time by modern devices are generated in dynamic environments with frequent changes in their distribution. This non-stationary environment increasingly imposes the need for adaptive machine learning algorithms. Thus, concept drift detectors are at the core of many of these algorithms to decide the best times for model updates.

ADWIN is considered state-of-the-art for drift detection in data streams. Although ADWIN can monitor any information from the stream, virtually all work from the literature employs it in a supervised way to monitor the performance of a classifier, which is costly or even infeasible in many real-world problems.

We propose an approach to use ADWIN in an unsupervised manner for drift monitoring. Our approach is based on monitoring simple statistical measures from individual stream examples (e.g., kurtosis or skewness). The drift monitoring is performed by class using multiple detectors built based on model predictions. We experimentally show for different problems that our approach achieved competitive results for different problems, outperforming the fully supervised approach based on error monitoring while requesting a few labeled data.

ADWIN-U is an alternative solution that complements its supervised counterpart, which is suitable depending on the actual labels' availability. We recognize that each version has its advantages under different circumstances. For instance, in many stream classification problems derived from regression tasks, such as the Electricity dataset, the goal is to predict whether the energy price will increase or decrease the next day. In this scenario, the actual label of the previous prediction becomes available daily, making supervised drift detection methods like ADWIN appropriate and a safe choice. Conversely, in applications such as classifying insect species using an optical sensor in the field (Insects datasets), there is no confirmation of the correctness of predictions for performance monitoring. In such cases, unsupervised drift detectors become not only suitable but often the only viable option for detecting changes in data distribution. This distinction highlights the importance of context when selecting drift detection techniques.

Besides the unsupervised drift detector, we introduce a new evaluation measure that takes into account the trade-off between accuracy and proportion of requested labeled data. Such a measure prioritizes precise detectors with a low false alarm rate, being a relevant contribution to evaluating novel drift detectors. In future works, we plan to adapt ADWIN-U to monitor multiple statistical measures simultaneously using a voting scheme or dynamic selection strategy.

References

1. Assis, D.N., Souza, V.M.A.: Supporting website (2025). `https://sites.google.com/view/adwin-u/`
2. Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., Morales-Bueno, R.: Early drift detection method. In: Proceedings of the international workshop on knowledge discovery from data streams, pp. 77–86 (2006)
3. Bayram, F., Ahmed, B.S., Kassler, A.: From concept drift to model degradation: An overview on performance-aware drift detectors. Knowledge-Based Systems p. 108632 (2022)
4. Bifet, A.: Classifier concept drift detection and the illusion of progress. In: Proceedings of the International Conference on Artificial Intelligence and Soft Computing, pp. 715–725. Springer (2017)
5. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the SIAM international conference on data mining, pp. 443–448. SIAM (2007)
6. Bifet, A., Holmes, G., Pfahringer, B.: Leveraging bagging for evolving data streams. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 135–150. Springer (2010)
7. Bifet, A., Read, J., Žliobaitė, I., Pfahringer, B., Holmes, G.: Pitfalls in benchmarking data stream classification and how to avoid them. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 465–479. Springer (2013)
8. Breiman, L.: Random forests. Machine learning **45**, 5–32 (2001)
9. Bruce, P., Bruce, A., Gedeck, P.: Practical statistics for data scientists: 50+ essential concepts using R and Python. O'Reilly Media (2020)

10. Cerqueira, V., Gomes, H.M., Bifet, A., Torgo, L.: Studd: a student–teacher method for unsupervised concept drift detection. Machine Learning pp. 1–28 (2022)
11. Cesario, E.: Big data analytics and smart cities: applications, challenges, and opportunities. Frontiers in big data **6**, 1149402 (2023)
12. Cesario, E., Folino, F., Manco, G., Pontieri, L.: An incremental clustering scheme for duplicate detection in large databases. In: Proceedings of the International Database Engineering and Application Symposium, pp. 89–95 (2005)
13. Datar, M., Gionis, A., Indyk, P., Motwani, R.: Maintaining stream statistics over sliding windows. SIAM journal on computing **31**(6), 1794–1813 (2002)
14. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine learning research **7**(1), 1–30 (2006)
15. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Proceedings of the Brazilian Symposium on Artificial Intelligence, pp. 286–295. Springer (2004)
16. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM computing surveys **46**(4), 1–37 (2014)
17. Gemaque, R.N., Costa, A.F.J., Giusti, R., Dos Santos, E.M.: An overview of unsupervised drift detection methods. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **10**(6), e1381 (2020)
18. Ghislain, V., Nadine, H., Jean-Pierre, V.: Adaptive threshold computation for cusum-type procedures in change detection and isolation problems. Computational Statistics & Data Analysis **52**(9), 4161–4174 (2008)
19. Gomes, H.M., Bifet, A., Read, J., Barddal, J.P., Enembreck, F., Pfharinger, B., Holmes, G., Abdessalem, T.: Adaptive random forests for evolving data stream classification. Machine Learning **106**, 1469–1495 (2017)
20. Gomes, H.M., Grzenda, M., Mello, R., Read, J., Le Nguyen, M.H., Bifet, A.: A survey on semi-supervised learning for delayed partially labelled data streams. ACM Computing Surveys **55**(4), 1–42 (2022)
21. Gonçalves Jr, P.M., de Carvalho Santos, S.G., Barros, R.S., Vieira, D.C.: A comparative study on concept drift detectors. Expert Systems with Applications **41**(18), 8144–8156 (2014)
22. Gözüaçık, Ö., Büyükçakır, A., Bonab, H., Can, F.: Unsupervised concept drift detection with a discriminative classifier. In: Proceedings of the ACM international conference on information and knowledge management, pp. 2365–2368 (2019)
23. Gözüaçık, Ö., Can, F.: Concept learning using one-class classifiers for implicit drift detection in evolving data streams. Artificial Intelligence Review **54**, 3725–3747 (2021)
24. Hassani, M.: Concept drift detection of event streams using an adaptive window. In: Proceedings of the International Conference on Modelling and Simulation, pp. 230–239 (2019)
25. Hinder, F., Artelt, A., Hammer, B.: Towards non-parametric drift detection via dynamic adapting window independence drift detection (dawidd). In: Proceedings of the International Conference on Machine Learning, pp. 4249–4259 (2020)
26. Hoeffding, W.: Probability inequalities for sums of bounded random variables. The collected works of Wassily Hoeffding pp. 409–426 (1963)

27. Hu, H., Kantardzic, M., Sethi, T.S.: No free lunch theorem for concept drift detection in streaming data classification: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **10**(2), e1327 (2020)
28. Huang, D.T.J., Koh, Y.S., Dobbie, G., Bifet, A.: Drift detection using stream volatility. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 417–432. Springer (2015)
29. Huang, D.T.J., Koh, Y.S., Dobbie, G., Pears, R.: Detecting volatility shift in data streams. In: Proceedings of the IEEE International Conference on Data Mining, pp. 863–868. IEEE (2014)
30. Iwashita, A.S., Papa, J.P.: An overview on concept drift learning. IEEE access **7**, 1532–1547 (2018)
31. Jiao, B., Guo, Y., Gong, D., Chen, Q.: Dynamic ensemble selection for imbalanced data streams with concept drift. IEEE Transactions on Neural Networks and Learning Systems (2022)
32. Joanes, D.N., Gill, C.A.: Comparing measures of sample skewness and kurtosis. Journal of the Royal Statistical Society: Series D (The Statistician) **47**(1), 183–189 (1998)
33. Kifer, D., Ben-David, S., Gehrke, J.: Detecting change in data streams. In: Proceedings of the International Conference on Very Large Data Bases, vol. 4, pp. 180–191. Toronto, Canada (2004)
34. Krawczyk, B., Cano, A.: Online ensemble learning with abstaining classifiers for drifting and noisy data streams. Applied Soft Computing **68**, 677–692 (2018)
35. Liu, W., Zhang, H., Ding, Z., Liu, Q., Zhu, C.: A comprehensive active learning method for multiclass imbalanced data streams with concept drift. Knowledge-Based Systems **215**, 106778 (2021)
36. Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, G.: Learning under concept drift: A review. IEEE transactions on knowledge and data engineering **31**(12), 2346–2363 (2018)
37. Maciel, B.I.F., Santos, S.G.T.C., Barros, R.S.M.: A lightweight concept drift detection ensemble. In: Proceedings of the IEEE International Conference on Tools with Artificial Intelligence, pp. 1061–1068 (2015)
38. Montiel, J., Read, J., Bifet, A., Abdessalem, T.: Scikit-multiflow: A multi-output streaming framework. Journal of Machine Learning Research **19**(72), 1–5 (2018)
39. Pesaranghader, A., Viktor, H.L.: Fast hoeffding drift detection method for evolving data streams. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 96–111. Springer (2016)
40. Reis, D.M., Flach, P., Matwin, S., Batista, G.: Fast unsupervised online drift detection using incremental kolmogorov-smirnov test. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1545–1554 (2016)
41. Roberts, S.W.: Control chart tests based on geometric moving averages. Technometrics **42**(1), 97–101 (2000)
42. Ross, G.J., Adams, N.M., Tasoulis, D.K., Hand, D.J.: Exponentially weighted moving average charts for detecting concept drift. Pattern recognition letters **33**(2), 191–198 (2012)

43. Souza, V.M.A., Chowdhury, F.A., Mueen, A.: Unsupervised drift detection on high-speed data streams. In: Proceedings of the IEEE International Conference on Big Data, pp. 102–111. IEEE (2020)
44. Souza, V.M.A., Parmezan, A.R.S., Chowdhury, F.A., Mueen, A.: Efficient unsupervised drift detector for fast and high-dimensional data streams. Knowledge and Information Systems **63**, 1497–1527 (2021)
45. Souza, V.M.A., Pinho, T., Batista, G.: Evaluating stream classifiers with delayed labels information. In: Brazilian conference on intelligent systems, pp. 408–413. IEEE (2018)
46. Souza, V.M.A., Reis, D.M., Maletzke, A.G., Batista, G.: Challenges in benchmarking stream learning algorithms with real-world data. Data Mining and Knowledge Discovery **34**, 1805–1858 (2020)
47. Souza, V.M.A., Silva, D.F., Batista, G.E., Gama, J.: Classification of evolving data streams with infinitely delayed labels. In: IEEE 14th International Conference on Machine Learning and Applications, pp. 214–219. IEEE (2015)
48. Souza, V.M.A., Silva, D.F., Gama, J., Batista, G.: Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In: Proceedings of the SIAM international conference on data mining, pp. 873–881. SIAM (2015)
49. Tschumitschew, K., Klawonn, F.: Incremental statistical measures. In: Learning in non-stationary environments: Methods and Applications, pp. 21–55. Springer (2012)
50. Yang, L., Manias, D.M., Shami, A.: Pwpae: An ensemble framework for concept drift adaptation in iot data streams. In: Proceedings of the IEEE Global Communications Conference, pp. 01–06. IEEE (2021)
51. Yeh, A.B., Mcgrath, R.N., Sembower, M.A., Shen, Q.: Ewma control charts for monitoring high-yield processes based on non-transformed observations. International Journal of Production Research **46**(20), 5679–5699 (2008)
52. Zhong, S., Souza, V.M.A., Mueen, A.: Combining filtering and cross-correlation efficiently for streaming time series. ACM Transactions on Knowledge Discovery from Data **16**(5), 1–24 (2022)
53. Žliobaite, I.: Change with delayed labeling: When is it detectable? In: Proceedings of the IEEE international conference on data mining workshops, pp. 843–850. IEEE (2010)