Manuscript Number: ESWA-D-19-04180

Title: USDD: Ultimately Simple Concept Drift Detection Method

Article Type: Full length article

Keywords: Concept drift; drift detection; data stream; online learning.

USDD: Ultimately Simple Concept Drift Detection Method

Bruno Iran Ferreira Maciel[a], Juan Isidro González Hidalgo[a], Roberto Souto Maior de Barros[a,*]

[a]*Centro de Informática, Universidade Federal de Pernambuco, Cidade Universitária, 50.740-560, Recife-PE, Brazil.*

Abstract

In this research we present a novel and simple approach to the concept change detection problem. Drift detection is a fundamental issue in data stream mining because the generated models need to be updated when significant changes in the underlying data distribution occur. A number of change detection approaches have been proposed but most of them have some limitation such as limited effect on the accuracy results, high computational complexity, poor sensitivity to gradual change, or the opposite problem of high false positive rate. The proposed approach, named Ultimately Simple Drift Detector (USDD), has low false positive and false negative rates as well as low computational complexity as it avoids multiple scans on its memory buffer by sequentially processing the stream. Extensive experimentation against six of the very best current detectors, and even an ensemble of detectors used as an upper bound for accuracy, on a wide variety of datasets reveals that, using limited resources, USDD delivers high accuracy in classification and at the same time maintains a competitive true detection rate when compared to the other methods.

Keywords: Concept drift, drift detection, data stream, online learning.

1. Introduction

A Data stream environment can be described as a continuous and potentially unlimited sequence of patterns that arrive in high speed, and whose probability distribution may vary over time. These aspects make it impractical to load the complete stream into main memory and then process it, which is common to most traditional data mining approaches, where the data set is available for loading into main memory and then be read as many times as necessary.

Concerning the problem of classification, especially in supervised learning, the objective is to classify fully-labeled instances (also referred as examples, patterns, tuples, or objects) into a set of common properties named classes or labels. The learning algorithm classifies the instances based on the prediction model induced from the previously processed instances.

*Corresponding author
Email addresses: bifm@cin.ufpe.br (Bruno Iran Ferreira Maciel), jigh@cin.ufpe.br (Juan Isidro González Hidalgo), roberto@cin.ufpe.br (Roberto Souto Maior de Barros)

However, the current prediction model may become inefficient or detrimental to classification after the occurrence of concept changes or drifts (Gama et al., 2014). This problem of concept change is inherent in dynamic environments because they are often non-stationary. In the data stream area, a concept drift is usually characterized as an unexpected replacement of the probability distribution of the data by another. A concept drift detection algorithm (detector) (Barros & Santos, 2018; Gonçalves Jr. et al., 2014) is a piece of software responsible for analyzing the results of the data stream classification and its objective is the detection of such concept drifts to permit adjustments in the classifier in real-time.

In dynamic environments, it is difficult to perform automatic learning, since there are hidden contexts, i.e. all the characteristics of the concepts are not explicitly provided. In many real-world domains, concepts depend on context and can change whenever context changes occur. Examples of such applications are those of weather forecasting, that can vary radically with the seasons, as well as the analysis of patterns of purchases of customers, which may change over time depending on days of the week, inflation, other purchases, etc. Often the source of the change is unknown, making learning a more complex task. In such environments, online learning is usually used to classify the data.

There are several definitions of online learning in the literature. This work adopts the understanding that online learning algorithms work by processing instances by only reading them, training their model without the need for storage or reprocessing of the instances (Oza & Russell, 2001). It also assumes that each example has pseudo-randomness, being unique, and its mapping function is updated with each new training (Fern & Givan, 2003). In other words, online learning is a particular case of incremental learning, where the learning process needs some mechanism to update concepts so that learning is continuous with each new arrived instance. In such scenarios, constraints regarding run-time and memory consumption are common. To be effective, the learning algorithm needs to be able to evolve its predictive model so that it follows the changes and quickly adapts to them.

According to Polikar et al. (Polikar et al., 2001), an online learning algorithm should fulfill the following requirements: (i) it can access data only once and sequentially; (ii) the time and space complexity of the algorithm must not scale with the number of instances; and (iii) the algorithm should be able to self-adapt.

In online learning scenarios, the use of ensembles of classifiers can present significant results in improving the prequential accuracy when compared to a single-classifier-based approach (Oza & Russell, 2001; Fern & Givan, 2003; Polikar et al., 2001). However, in environments where limited hardware resource constraints are imposed, using a single classifier approach may not be a matter of choice. In this situation, identifying changes in context is essential for adapting the prediction model. The use of concept drift detectors is a common practice employed in several data stream works (Gama et al., 2004; Barros et al., 2017; Cabral & Barros, 2018).

This paper proposes Ultimately Simple Drift Detector (USDD). It is inspired on the common boosting strategy of using individual classifiers with error rates below 1/2 (Freund & Schapire, 1996; Barros et al., 2016; Santos & Barros, 2019). The experiments executed to test USDD against state-of-the-art detectors and an ensemble of detectors suggest it is statistically better than or equivalent to the other detectors, as it delivers higher or equal global classification accuracy in most situations, detects most drifts earlier, and, in absolute values, presents the best results of the Matthews Correlation Coefficient (MCC) (Matthews, 1975).

The rest of the paper is organized as follows: Section 2 presents related work regarding concept drift detection; Section 3 describes the USDD algorithm and its pseudo-code; Section 4 shows the experiments configuration, also including brief descriptions of the synthetics and real-world datasets used in the tests; Section 5 shows the results of the experiments, including the means of the prequential accuracy of the classifiers and the statistical evaluation of the drift detections; and, finally, Section 6 provides conclusions and discusses future work.

2. Related work

Online learning has been studied in several research fields including medicine, meteorology, data mining, and machine learning. The concept of online learning may be interpreted differently because the terms incremental and online learning have been used with different meanings in the literature (Minku, 2010).

Concept drift detection methods are lightweight software that monitor the predictions of a base classifier (Barros & Santos, 2018; Gonçalves Jr. et al., 2014). Several concept drift detection methods have been proposed in the literature and some of the best performing ones are Drift Detection Method (DDM) (Gama et al., 2004), Reactive Drift Detection Method (RDDM) (Barros et al., 2017), Hoeffding-based Drift Detection Method A-Test (HDDM$_A$) (Frías-Blanco et al., 2015), Stacking Fast Hoeffding Drift Detection Method (FHDDMS) (Pesaranghader et al., 2018), Fisher Test Drift Detector (FTDD) (Cabral & Barros, 2018), and Wilcoxon Rank Sum Test Drift Detector (WSTD) (Barros et al., 2018). Some ensembles of detectors have also been proposed, e.g. ADDM (Du et al., 2014) and Drift Detection Ensemble (DDE) (Maciel et al., 2015).

2.1. *Drift Detection Methods*

Drift Detection Method (DDM) (Gama et al., 2004) detects concept drifts in data streams by analyzing the error rate of the classifier and its standard deviation. For each position i in the stream, DDM defines the error rate p_i as the probability of making an incorrect prediction and its standard deviation is given by $s_i = \sqrt{p_i(1-p_i)/i}$. Based on the Probably Approximately Correct (PAC) learning model (Mitchell, 1997), the authors of DDM argue that, provided that the distribution of the examples remains stationary, the error rate p_i should decrease as the number of examples (i) increases (for an infinite number of examples). A significant increase in the error of the algorithm suggests that the class distribution changed and, hence, the actual decision model became inappropriate.

Reactive Drift Detection Method (RDDM) (Barros et al., 2017) periodically shortens the number of instances of very long stable concepts to tackle a known performance loss problem of DDM. With this purpose, it discards old instances of very long concepts aiming to detect drifts earlier, improving the precision of its detections and especially the final accuracy. Moreover, using its recommended default configuration, RDDM presents especially strong performance in datasets with gradual concept drifts and when the sizes of the concepts are many thousand instances long.

Hoeffding-based Drift Detection Method (HDDM) (Frías-Blanco et al., 2015) monitors the performance of a base learner by applying "some probability inequalities that assume only independent, uni-variate and bounded random variables to obtain theoretical guarantees for the detection of such distributional changes". It provides bounds on both false positive and false negative rates and does *not* assume the measured values satisfy a Bernoulli distribution. Hoeffding-based Drift Detection Method A-Test (HDDM$_A$) is one of the proposed versions of HDDM and its authors claim (Frías-Blanco et al., 2015) it "involves moving averages and is more suitable to detect abrupt changes".

Fast Hoeffding Drift Detection Method (FHDDM) (Pesaranghader & Viktor, 2016) performs drift detection based on Hoeffding's inequality. More specifically, FHDDM uses a sliding window of size n (default 200) and detects a drift when a significant difference between the maximum and the most recent probabilities of correct predictions is observed. This method calculates the difference between those probabilities (ΔP) and a threshold (ϵ), which is found using the probability of error in Hoeffding's inequality (δ, default 10^{-7}). A drift is detected when $\Delta\text{P} \geq \epsilon$.

Stacking Fast Hoeffding Drift Detection Method (FHDDMS) (Pesaranghader et al., 2018) extends FHDDM by maintaining windows of different sizes, a short and a long sliding window. The rationale behind this approach is to reduce the detection delay and false negative rate. Intuitively, a short window should detect abrupt drifts faster, while a long window should detect gradual drifts with a lower false negative rate.

Fisher Test Drift Detector (FTDD) (Cabral & Barros, 2018) is one of three concept drift detectors based on an efficient implementation of Fisher's Exact test. It draws on Statistical Test of Equal Proportions (STEPD) (Nishida & Yamauchi, 2007) and on the deficiency of its statistical test of equal proportions in situations where the data samples are small or imbalanced. FTDD detects drifts using Fisher's Exact test instead of the test of equal proportions in all situations.

Wilcoxon Rank Sum Test Drift Detector (WSTD) (Barros et al., 2018) was proposed with the goal of identifying less False Positives (PF) than STEPD (Nishida & Yamauchi, 2007) and also being statistically more accurate. It uses an efficient implementation of the Wilcoxon rank-sum statistical test, which is used statistically to determine if two independent samples come from populations with the same distribution in the data (Wilcoxon, 1945). Similarly to STEPD, WSTD monitors the predictions of the base classifier using two windows (recent and old). Despite presenting strong all-round performance, its accuracy improvements

Pontua os pontos fortes dos algoritmos desenvolvidos. Qual o dif

are usually larger in datasets with abrupt concept drifts and its main strength is the precision of its detections of concept drifts.

Drift Detection Ensemble (DDE) (Maciel et al., 2015) is a simple ensemble of detectors that aggregates the warnings and drift detections of three concept drift detection methods which monitor the results of a single base classifier aiming to improve the results of the individual methods using different strategies and configurations. DDE can be programmed to use different default combinations of detectors depending on the chosen sensitivity of the ensemble, the number of members necessary to detect warnings and drifts. As different drift detectors are better in different scenarios, DDE can also be parameterized to use different combinations of individual methods.

3. Ultimately Simple Concept Drift Detection Method

This section provides a detailed description of USDD, the proposed drift detection algorithm. It is based on causal inference and on the assumption that, within a single concept, the error rate of the classifier tends to decrease until it reaches a certain level where it becomes stable. Thus, any perturbation that causes a degradation in the error rate of the classification is considered a concept drift.

To detect the drifts, USDD maintains a sliding window of size w with the most recent instances of the data, to calculate the proportion of classification errors within this window, and then compares this result with threshold limits for warnings (α_w) and drifts (α_d). In addition, warnings and drifts are only monitored after a certain number of instances (n) are processed within a single concept, in order to guarantee the maturity of the constructed decision model. Consequently, to some extent, USDD works similarly to DDM, because it also monitors the predictions of the base learner using one window and relies on some measurement of the performance of the base classifier to decide when to signal warnings and drifts.

USDD does not depend on the probability distribution of the data and we assume the predictions made by the classifier are independent of each other. It is also understood that the distribution of the sum (or mean) of a large number of independent and identically distributed (i.i.d.) random variables is approximately normal, regardless of the underlying distribution of those variables (Sejdic & Falk, 2018). This distribution is expected to converge from any distribution to a standard normal distribution when the number of examples tends to infinity.

It is important to mention that most other window-based concept drift detection methods (e.g. STEPD, FTDD, WSTD, etc.) compare the distributions of two windows of data, one with the most recent examples and the other with historical information, possibly leading to more memory usage.

The main strength of USDD lies in the simplicity of its strategy to identify concept changes. It monitors the performance of the classifier in real-time and assumes that concept drifts only occur when the accuracy of the classifier in the sliding window drops below 50%, i.e. the errors exceed the correct answers.

This strategy is similar to the one adopted in several boosting algorithms, e.g. AdBoost.M1 (Freund & Schapire, 1996), Boosting-like Online Learning Ensemble (BOLE) (Barros et al., 2016), Online AdaBoost-based M1 (OABM1) (Santos & Barros, 2019), etc., where the weak classifiers with errors greater than 1/2 are discarded, because they are considered detrimental to the committee.

The parameters of USDD and their respective default values are the size of the sliding window ($w = 80$), the minimum number of instances before drifts can be detected ($n = 300$), and error levels (percentage) for the detection of warnings ($\alpha_w = 0.35$) and drifts ($\alpha_d = 0.46$). These specific values where chosen empirically to maximize the accuracy of the classifier based on a large number of datasets. To optimize the detections of the drifts, the suggested values are $w = 80$, $n = 1000$, $\alpha_w = 0.4$, and $\alpha_d = 0.5$. It is also worth observing that larger concepts contribute to induce better generalizations, minimizing the errors, and that such scenarios make the choice of parameter values less relevant to the classification and detection results.

Despite its simplicity, its efficiency is supported by theoretical properties of statistical inference. Based on the PAC learning model (Mitchell, 1997), the number of errors of the classifier tends to be stable or decrease with more examples within a stationary concept, and a significant increase in the number of errors suggests a change in the distribution of the data, i.e. concept drift.

3.1. The USDD implementation

This subsection gives additional, more concrete, details of the USDD implementation. Algorithm 1 shows a fairly detailed abstract pseudo-code corresponding to the Java code that implements USDD in the Massive Online Analysis (MOA) framework (Bifet et al., 2010), release 2014.11.

The inputs to USDD are a data stream (*stream*), and parameter values defining the sliding window size (w), the error percentage levels used for the detection of warnings (α_w) and concept drifts (α_d), and the minimum number of instances (n) before monitoring the results of the classifier to detect concept drifts.

The initialization of the algorithm includes the calculation of the warning and drifts limits in number of instances, by multiplying the error percentage levels α_w and α_d by the size of the of sliding window (w) (lines 1–2 in the pseudo-code); as well as of the number of ignored instances in the start of a new concept (*numIgnoredInst*) to avoid maintaining the sliding window with results that cannot be used to detect drifts (line 3). This value is compared to the number of processed instances of the current concept (*numInst*) in line 13 to ignore the first instances of each concept, allowing time for the classifier to learn the concept.

Lines 4–9 and 26–28 are responsible for resetting the variables related to the current concept in the initialization and whenever a concept drift is detected, respectively. Note that, in Java, array allocations are dynamic and, thus, the array used to implement the sliding window with the last prediction results (*stPred*) will have the exact size (w) required (Line 7). It is also important to state that, for memory and run-time efficiency, the storage strategy adopted in this array is that of a circular queue and the type chosen for its elements was Java's standard numeric type *int*.

Algorithm 1: USDD - Ultimately Simple Drift Detector

```
Input:
    data stream $stream$, window size $w$, warning level $\alpha_w$, drift level $\alpha_d$,
    minimum number of predictions to suppose the decision model is stable $n$
 1 warnLimit $\leftarrow w \times \alpha_w$   Number of errors in the sliding window needed to signal warnings
 2 driftLimit $\leftarrow w \times \alpha_d$   Number of errors in the sliding window needed to signal drifts
 3 numIgnoredInst $\leftarrow n - w$   Number of ignored predictions in the start of a new concept
 4 numInst $\leftarrow 0$   Number of processed instances in current concept
 5 numErr $\leftarrow 0$   Number of errors in the predictions stored in the sliding window
 6 pos $\leftarrow 0$   Last/next used position in the sliding window (stPred)
 7 stPred $[*] \leftarrow 0$
 8 foreach $instance$ in $stream$ do
 9     pred $\leftarrow$ prediction $(instance)$   Prediction result of the classifier is received
10     numInst $\leftarrow$ numInst $+ 1$
11     if numInst $>$ numIgnoredInst then
12         numErr $\leftarrow$ numErr $-$ stPred [pos ] $+$ pred   Updates number of errors in the sliding window
13         stPred [pos ] $\leftarrow$ pred   The last prediction result received is stored in the sliding window
14         pos $\leftarrow$ pos $+ 1$   Updates the position in the sliding window
15         if pos $= w$ then
16             pos $\leftarrow 0$
17         end
18         if numInst $\geq n$ then
19             if numErr $<$ warnLimit then
20                 raise $stable$
21             end
22             else
23                 if numErr $<$ driftLimit then
24                     raise $warning$
25                 end
26                 else
27                     numInst $\leftarrow$ numErr $\leftarrow$ pos $\leftarrow 0$
28                     stPred $[*] \leftarrow 0$
29                     raise $drift$
30                 end
31             end
32         end
33     end
34 end
```

The loop starting in line 10 is responsible for processing the data stream, abstracting the details related to the classification because, in the MOA framework, these details are implemented in other classes. In addition, line 11 captures the result of the classification of each instance and its possible values are 0, for correct predictions, and 1, for the incorrect ones.

When the condition of Line 13 is met, USDD starts to store these results ($pred$) in the sliding window ($stPred$) and to maintain the number of errors ($numErr$) in these stored predictions (lines 14–18). It is important to emphasize that, because the results are only 0's or 1's, the number of errors coincides with the sum of the values stored in $stPred$. Thus, for each new result, USDD simply subtracts the oldest value in the window (position pos) and adds the new result, before its substitution, and then updates the position. It is also worth highlighting the simplicity, elegance, and efficiency of our implementation of the sliding window as a circular queue using an array and a couple of variables.

The condition of line 19 is used to decide when concept drifts start do be monitored, and this decision depends on the value of parameter n. Note that, in this point, the sliding window will be completely filled and USDD decides the current state of the classifier based on the number of errors $numErr$. The condition in line 20 captures the situations where the classifier is considered stable; the one in line 23 refers to the warning state, when an alternative classifier is trained in parallel to substitute the original classifier should a concept drift be detected later in line 25.

4. Experimental Setup

This section describes all the relevant information on the experiments designed to test and evaluate USDD against DDM, FHDDMS, FTDD, HDDM$_A$, RDDM, WSTD, and DDE. All the drift detection methods were tested with their respective default parameter values, as proposed by their authors, and were tested using both Hoeffding Tree (HT) and Naïve Bayes (NB) as bases learners because they are simple, fast, efficient, and commonly used in experiments in the data stream area, and there are freely available implementations in MOA.

DDE was included in the experiments with the intention of serving as an upper bound for the accuracy results of current state-of-the-art drift detection. As such, considering the results of Barros and Santos (Barros & Santos, 2018) and including FHDDMS as an additional candidate, we tested in advance several combinations of components and sensitivities for DDE and selected the configuration that delivered the best results in each of the two base learners.

Based on these previous results, for the experiments using HT, DDE was configured with HDDM$_A$, RDDM, and FHDDMS as components and sensitivity 2. For NB, the chosen configuration has sensitivity 1 and the components are HDDM$_A$, RDDM, and FTDD. In its original default configuration proposed in 2015, DDE used sensitivity 1 and its components were DDM, HDDM$_A$, and Hoeffding-based Drift Detection Method W-Test (HDDM$_W$).

The experiments included both artificial and real-world datasets. In the artificial datasets, the tests were executed 30 times and the mean accuracies of the methods were calculated with 95% confidence intervals.

The accuracy evaluation adopted the Prequential methodology (Dawid, 1984) with a sliding window of size 1000 as its forgetting mechanism (Gama et al., 2013; Hidalgo et al., 2019). In this methodology, each incoming instance is used initially for testing and subsequently for training.

To verify the existence or non-existence of statistical differences between the tested detectors, a non-parametric evaluation was applied to rank the detectors according to the prequential accuracy of the classifiers, as prescribed in (Demšar, 2006). To test if the observed differences in their mean ranks correspond to significant differences in the performance of the classifier, the Friedman statistic (Friedman, 1937) was computed and compared to the associated Critical Difference (CD) for Chi-squared distribution with significance level $\alpha = 0.05$. Moreover, as the rejection of the null hypothesis of the method indicates the existence of statistical differences between some of the methods but does not define which of them are statistically different (Demšar, 2006), the Bonferroni Dunn post-hoc test (Dunn, 1961) was applied.

4.1. Datasets

This subsection describes all the datasets chosen for the experiments described in Section 5. All of them have been previously used in the area, and most of them are available in the MOA framework.

Artificial datasets have the advantage that any desired drift behavior can be explicitly specified. Based on four dataset generators available in MOA, we built *abrupt* and *gradual* drift configurations, with three different versions of each of them, 30 artificial datasets in total. Each generator was configured with four concept drifts distributed at regular intervals and three sizes: 20k (with concept drifts in instances 4k, 8k, 12k, and 16k), 130k (with concept drifts in instances 26k, 52k, 78k, and 104k), and 890k (with concept drifts in instances 178k, 356k, 534k, and 712k).

To simulate the abrupt drifts, different concepts were simply joined. In the case of gradual changes, a probability function available in MOA was used to increase the chances of selecting instances of the new concept, instead of instances of the previous one, and the two concepts coexist for 500 instances.

The selected artificial dataset generators are described below and Table 1 shows their main characteristics.

Table 1: Descriptive summary of artificial datasets configuration.

Dataset	#Attributes	#Classes
Agrawal	9	2
Led	24	10
Random RBF	10	2
Sine	2	2

Agrawal generator (Agrawal et al., 1993; Santos et al., 2014) stores information from people aiming to receive a loan, and these people are classified as group A or B. The attributes are: salary, commission, age, education level, zip code, etc. The authors proposed ten functions, each with different forms of evaluation, and it is possible to add noise. The last five functions (F6–F10) were used to generate the datasets.

LED generator (Gonçalves & Barros, 2013; Frías-Blanco et al., 2015) represents the problem of predicting the digit of a seven segment LED display. It has 24 categorical attributes (17 of them are irrelevant), and a categorical class. Each attribute has a 10% probability of being inverted (noise). Concept drifts are simulated by changing the position of the seven relevant attributes.

Sine generator (Gama et al., 2004; Barros & Santos, 2019) uses two numeric attributes (x, y), two classes ("negative" and "positive") and two functions to generate the contexts. In Sine$_1$, each instance is positive if the point (x, y) is below the curve $y = sin(x)$, whereas Sine$_2$ uses the function $y = 0.5 + 0.3 \times \sin(3\pi x)$. Concept drifts can be simulated either by alternating between Sine$_1$ and Sine$_2$ or by reversing the results, i.e. points below the curves become negative.

Random RBF generator (Bifet et al., 2009; Santos et al., 2015) uses n centroids with their centers, labels, and weights randomly defined, and a Gaussian distribution to determine the values of m attributes. Concept drifts are simulated by changing the centroids' positions. The datasets were generated with two classes, five attributes, 20 centroids in the model and 20 centroids with drift.

Regarding real-world datasets, normally there is no firm indication whether they have concept drifts or not. Thus, we decided to select them using as criterion the difference in the performance of the classification with and without the use of concept drift detectors, assuming that the use of a detector would only improve the classification in those datasets where concept drifts indeed exist. The seven chosen real-world datasets are described below.

Airlines (Ikonomovska et al., 2011; Santos et al., 2019) is a binary dataset with 539,383 instances, inspired in the regression dataset proposed by Elena Ikonomovska (Elena, 2008; Ikonomovska et al., 2011). The dataset contains flight arrival and departure details of commercial flights within the USA. There are 7 attributes: name of the company (with 18 possibilities), flight number, airports of origin and destination (with 293 possibilities), day of the week, flight time, and arrival time. The goal is to predict whether flights are delayed or not, given the information of the scheduled departure.

Connect-4 (Tromp, 1995) contains all legal 8-ply positions in the game of connect-4 in which neither player has won yet, and in which the next move is not forced. It is composed of 42 attributes and 67,557 instances. The outcome class is the game theoretical value for the first player: win, loss, or draw.

Outdoor (Losing et al., 2016; Barros et al., 2018) was obtained from images recorded by a mobile phone in a garden environment. The task is to classify 40 different objects; each of them was approached five times in sunny and five times in cloudy conditions. The approaches were conducted in different directions. Each approach represents 10 images in temporal order within the dataset, for a total of 4000 instances.

Rialto (Losing et al., 2016) is a dataset containing 82,250 instances, with ten classes, of the colorful buildings near the famous Rialto Bridge in Venice. The colors were encoded in a normalized histogram of 27 RGB dimensions (attributes). The images were obtained from time space in fixed-video Webcam shots. The recordings cover 20 consecutive days during May and June 2016 and continued change in climate and lighting conditions affect performance.

Spam Data (Katakis et al., 2010, 2008) is a high dimensional dataset based on the e-mail messages from the original Spam Assassin Collection (Foundation, 2006): the boolean bag-of-words approach was used for representing e-mails. It consists of 9,324 instances and 500 attributes (words derived after feature selection).

The authors claim the characteristics of these spam messages gradually change with time, i.e. this dataset contains gradual concept drift.

Wine-Quality-Red and Wine-Quality-White are two datasets presented in (Cortez et al., 2009), based on wine data (red and white). Due to privacy and logistical issues, they only contain the physical-chemical (inputs) and sensory (output) variables; data on types of grapes, wine brands, prices, etc. are not included. The Wine-Quality-Red and Wine-Quality-White databases have 1,599 and 4,898 instances, respectively, and they both have 11 attributes and the output, which is based on sensory data (median of at least 3 evaluations made by wine specialists), where each specialist rated wine quality between 0 (very bad) and 10 (excellent). The objective is to model wine quality based on physical-chemical tests.

5. Experimental results and analysis

This section presents the experimental results of the methods, including analyses of their accuracies and concept drift identifications on the selected datasets with the two tested base learners.

5.1. Accuracy results and analysis

Tables 2 and 3 present the accuracy results of the tested methods in all selected datasets as well as their ranks using HT and NB, respectively. Note that, in each dataset and in the ranks, the best result is shown in **bold**.

Table 2: Mean accuracies in percentage (%) using HT, with 95% confidence intervals in the artificial datasets.

Type-Size	Dataset	DDM	FHDDMS	FTDD	HDDM$_A$	RDDM	WSTD	DDE	USDD
Abr-20k	Agrawal	86.02 (+-0.35)	87.17 (+-0.14)	86.62 (+-0.24)	87.07 (+-0.17)	86.87 (+-0.18)	**87.28 (+-0.18)**	87.13 (+-0.15)	86.85 (+-0.20)
	LED	70.39 (+-0.74)	71.14 (+-0.33)	66.45 (+-0.68)	71.51 (+-0.19)	**71.94 (+-0.18)**	68.05 (+-0.66)	70.77 (+-0.44)	71.57 (+-0.23)
	RRBF	74.02 (+-0.39)	74.85 (+-0.19)	75.02 (+-0.19)	74.88 (+-0.19)	74.70 (+-0.16)	75.03 (+-0.25)	74.91 (+-0.16)	**75.06 (+-0.23)**
	Sine	90.36 (+-0.61)	91.34 (+-0.11)	91.36 (+-0.11)	91.36 (+-0.13)	90.83 (+-0.17)	**91.41 (+-0.11)**	91.32 (+-0.12)	90.99 (+-0.11)
Abr-130k	Agrawal	92.55 (+-0.08)	92.21 (+-0.26)	92.68 (+-0.13)	92.55 (+-0.16)	92.62 (+-0.08)	92.56 (+-0.21)	**92.72 (+-0.10)**	92.44 (+-0.21)
	LED	71.62 (+-0.58)	72.53 (+-0.16)	71.96 (+-0.32)	73.49 (+-0.10)	73.51 (+-0.11)	72.23 (+-0.31)	72.67 (+-0.29)	**73.55 (+-0.12)**
	RRBF	79.57 (+-0.24)	77.83 (+-0.28)	**80.02 (+-0.14)**	79.47 (+-0.21)	79.66 (+-0.20)	79.34 (+-0.29)	79.83 (+-0.19)	79.47 (+-0.24)
	Sine	94.34 (+-0.08)	94.61 (+-0.09)	**94.64 (+-0.09)**	94.60 (+-0.10)	94.34 (+-0.11)	94.61 (+-0.09)	94.61 (+-0.09)	94.53 (+-0.09)
Abr-890k	Agrawal	95.45 (+-0.04)	95.39 (+-0.08)	**95.49 (+-0.05)**	95.37 (+-0.09)	94.60 (+-0.08)	95.42 (+-0.08)	95.39 (+-0.15)	95.39 (+-0.15)
	LED	70.32 (+-0.59)	72.89 (+-0.07)	73.72 (+-0.04)	73.63 (+-0.04)	73.68 (+-0.04)	73.31 (+-0.08)	73.58 (+-0.06)	**73.75 (+-0.04)**
	RRBF	85.24 (+-0.07)	82.06 (+-0.35)	**85.32 (+-0.08)**	84.71 (+-0.16)	83.41 (+-0.10)	84.01 (+-0.18)	85.16 (+-0.08)	85.12 (+-0.14)
	Sine	97.44 (+-0.05)	97.53 (+-0.06)	97.53 (+-0.07)	97.53 (+-0.06)	96.43 (+-0.12)	**97.54 (+-0.06)**	97.53 (+-0.06)	97.52 (+-0.05)
Grad-20k	Agrawal	84.89 (+-0.23)	85.58 (+-0.12)	84.13 (+-0.78)	85.46 (+-0.18)	**85.59 (+-0.14)**	82.94 (+-1.40)	85.45 (+-0.17)	85.10 (+-0.17)
	LED	69.33 (+-0.87)	70.62 (+-0.38)	65.10 (+-0.76)	71.19 (+-0.19)	**71.74 (+-0.19)**	65.26 (+-1.07)	69.65 (+-0.67)	70.96 (+-0.24)
	RRBF	73.77 (+-0.27)	73.81 (+-0.18)	74.25 (+-0.32)	74.18 (+-0.21)	73.94 (+-0.17)	73.61 (+-0.41)	74.11 (+-0.21)	**74.28 (+-0.21)**
	Sine	88.24 (+-0.15)	88.28 (+-0.14)	88.12 (+-0.12)	88.29 (+-0.15)	88.16 (+-0.13)	88.21 (+-0.13)	88.23 (+-0.16)	**88.57 (+-0.12)**
Grad-130k	Agrawal	92.38 (+-0.08)	91.91 (+-0.28)	92.35 (+-0.14)	92.28 (+-0.17)	92.45 (+-0.08)	91.87 (+-0.40)	**92.46 (+-0.09)**	92.13 (+-0.22)
	LED	71.36 (+-0.55)	72.45 (+-0.17)	71.51 (+-0.26)	73.45 (+-0.11)	**73.51 (+-0.11)**	71.65 (+-0.35)	72.27 (+-0.32)	73.42 (+-0.12)
	RRBF	**79.66 (+-0.25)**	77.55 (+-0.31)	79.61 (+-0.14)	79.25 (+-0.23)	79.58 (+-0.14)	79.11 (+-0.30)	79.50 (+-0.25)	79.40 (+-0.20)
	Sine	94.07 (+-0.07)	94.10 (+-0.06)	93.98 (+-0.08)	94.03 (+-0.07)	94.02 (+-0.07)	94.08 (+-0.08)	94.05 (+-0.08)	**94.14 (+-0.07)**
Grad-890k	Agrawal	95.43 (+-0.04)	95.34 (+-0.08)	**95.45 (+-0.06)**	95.33 (+-0.09)	94.53 (+-0.08)	95.15 (+-0.19)	95.44 (+-0.05)	95.39 (+-0.04)
	LED	70.49 (+-0.56)	72.88 (+-0.07)	73.70 (+-0.04)	73.63 (+-0.03)	73.68 (+-0.04)	73.22 (+-0.08)	73.57 (+-0.06)	**73.73 (+-0.04)**
	RRBF	85.20 (+-0.10)	82.22 (+-0.42)	**85.29 (+-0.09)**	84.79 (+-0.15)	83.38 (+-0.10)	84.00 (+-0.22)	85.06 (+-0.09)	85.11 (+-0.13)
	Sine	97.39 (+-0.05)	97.40 (+-0.05)	97.41 (+-0.05)	97.39 (+-0.06)	96.43 (+-0.11)	97.42 (+-0.05)	97.41 (+-0.04)	**97.44 (+-0.05)**
Real	Airlines	65.30	65.38	64.75	65.00	**66.01**	65.40	65.14	64.95
	Connect4	74.12	75.20	74.35	75.04	**75.43**	75.21	75.20	75.20
	Outdoor	59.27	60.25	59.21	59.96	58.63	60.12	59.64	**60.82**
	Rialto	36.88	48.87	30.83	45.70	48.17	33.65	46.96	**55.65**
	SpamData	92.11	**92.60**	92.30	91.98	92.27	91.99	92.03	91.91
	WineRed	52.97	52.07	53.17	52.82	52.76	51.34	52.71	**53.53**
	WineWhite	43.33	46.10	43.32	43.31	43.84	45.02	44.04	**47.02**
Rank	–	5.47	4.69	4.31	4.63	4.53	4.90	4.00	**3.47**

Table 3: Average accuracies in percentage (%) using NB, with 95% confidence intervals in the artificial datasets.

Type-Size	Dataset	DDM	FHDDMS	FTDD	HDDM$_A$	RDDM	WSTD	DDE	USDD
Abr-20k	Agrawal	85.96 (+-0.47)	87.00 (+-0.14)	85.54 (+-1.47)	86.86 (+-0.19)	86.90 (+-0.11)	86.96 (+-0.30)	**87.08 (+-0.10)**	86.77 (+-0.14)
	LED	70.41 (+-0.74)	71.16 (+-0.33)	66.31 (+-0.80)	71.51 (+-0.19)	**71.95 (+-0.18)**	67.95 (+-0.70)	70.83 (+-0.29)	71.58 (+-0.23)
	RRBF	72.68 (+-0.55)	**74.89 (+-0.17)**	74.79 (+-0.17)	74.74 (+-0.18)	74.68 (+-0.19)	74.77 (+-0.28)	74.73 (+-0.17)	74.83 (+-0.15)
	Sine	84.80 (+-1.95)	89.14 (+-0.25)	89.17 (+-0.25)	89.07 (+-0.24)	88.99 (+-0.25)	**89.18 (+-0.24)**	89.12 (+-0.26)	88.86 (+-0.24)
Abr-130k	Agrawal	88.25 (+-0.25)	88.69 (+-0.06)	**88.89 (+-0.06)**	88.71 (+-0.19)	88.77 (+-0.07)	**88.89 (+-0.07)**	88.87 (+-0.06)	88.83 (+-0.06)
	LED	70.68 (+-1.00)	72.54 (+-0.16)	71.11 (+-0.49)	73.49 (+-0.10)	73.52 (+-0.11)	72.10 (+-0.41)	73.31 (+-0.12)	**73.55 (+-0.12)**
	RRBF	73.14 (+-0.64)	75.23 (+-0.06)	75.28 (+-0.06)	75.24 (+-0.06)	75.22 (+-0.07)	75.21 (+-0.12)	**75.29 (+-0.06)**	**75.29 (+-0.07)**
	Sine	84.50 (+-2.11)	89.47 (+-0.12)	89.48 (+-0.12)	89.37 (+-0.11)	89.38 (+-0.11)	89.49 (+-0.12)	**89.55 (+-0.10)**	89.44 (+-0.12)
Abr-890k	Agrawal	88.35 (+-0.74)	89.07 (+-0.02)	**89.37 (+-0.02)**	89.30 (+-0.02)	89.24 (+-0.02)	89.28 (+-0.03)	89.23 (+-0.02)	89.30 (+-0.02)
	LED	70.30 (+-0.82)	72.90 (+-0.07)	73.72 (+-0.09)	**73.89 (+-0.04)**	73.85 (+-0.04)	73.35 (+-0.09)	73.72 (+-0.05)	73.88 (+-0.04)
	RRBF	72.43 (+-0.74)	75.32 (+-0.02)	**75.46 (+-0.03)**	75.43 (+-0.02)	75.39 (+-0.03)	75.41 (+-0.04)	75.42 (+-0.02)	75.44 (+-0.02)
	Sine	84.12 (+-2.35)	89.49 (+-0.05)	89.50 (+-0.05)	89.44 (+-0.05)	89.54 (+-0.04)	89.49 (+-0.05)	**89.57 (+-0.04)**	89.49 (+-0.05)
Grad-20k	Agrawal	84.90 (+-0.19)	84.98 (+-0.23)	81.01 (+-1.97)	84.83 (+-0.45)	**85.19 (+-0.18)**	78.87 (+-1.66)	85.15 (+-0.19)	84.31 (+-0.33)
	LED	69.42 (+-0.88)	70.64 (+-0.38)	64.76 (+-0.93)	71.18 (+-0.18)	**71.76 (+-0.18)**	65.03 (+-1.18)	71.04 (+-0.33)	70.97 (+-0.24)
	RRBF	73.14 (+-0.43)	73.79 (+-0.19)	73.05 (+-0.63)	73.88 (+-0.18)	73.88 (+-0.19)	69.32 (+-0.98)	**73.95 (+-0.18)**	**73.95 (+-0.17)**
	Sine	86.95 (+-0.22)	86.88 (+-0.21)	86.54 (+-0.21)	86.96 (+-0.20)	86.99 (+-0.21)	86.73 (+-0.21)	86.81 (+-0.21)	**87.12 (+-0.22)**
Grad-130k	Agrawal	87.94 (+-0.43)	88.30 (+-0.08)	88.31 (+-0.19)	88.41 (+-0.14)	88.48 (+-0.08)	86.32 (+-1.22)	**88.50 (+-0.06)**	88.42 (+-0.08)
	LED	70.92 (+-0.92)	72.46 (+-0.17)	70.40 (+-0.34)	73.46 (+-0.10)	**73.51 (+-0.11)**	71.18 (+-0.44)	73.29 (+-0.13)	73.43 (+-0.12)
	RRBF	72.82 (+-0.71)	75.02 (+-0.06)	75.06 (+-0.06)	75.10 (+-0.06)	75.12 (+-0.08)	73.85 (+-0.53)	**75.14 (+-0.07)**	75.11 (+-0.07)
	Sine	88.60 (+-0.94)	89.30 (+-0.11)	89.17 (+-0.10)	89.28 (+-0.11)	89.32 (+-0.10)	89.23 (+-0.11)	89.25 (+-0.10)	**89.33 (+-0.11)**
Grad-890k	Agrawal	88.59 (+-0.56)	88.96 (+-0.02)	**89.27 (+-0.03)**	89.22 (+-0.06)	89.17 (+-0.02)	88.91 (+-0.19)	89.16 (+-0.02)	89.21 (+-0.02)
	LED	70.90 (+-0.91)	72.89 (+-0.07)	73.64 (+-0.09)	**73.89 (+-0.04)**	73.85 (+-0.03)	73.19 (+-0.09)	73.70 (+-0.05)	73.87 (+-0.04)
	RRBF	72.43 (+-0.70)	75.29 (+-0.02)	**75.42 (+-0.03)**	**75.42 (+-0.02)**	75.37 (+-0.03)	75.17 (+-0.12)	75.40 (+-0.02)	75.41 (+-0.02)
	Sine	83.92 (+-2.67)	89.50 (+-0.05)	89.47 (+-0.06)	89.44 (+-0.05)	**89.53 (+-0.04)**	89.47 (+-0.05)	**89.53 (+-0.03)**	89.49 (+-0.05)
Real	Airlines	65.35	65.83	66.76	67.23	**67.50**	66.84	66.85	65.63
	Connect4	74.47	75.22	74.23	74.95	75.23	75.12	**75.46**	74.84
	Outdoor	59.49	60.63	59.80	60.11	59.85	61.03	**61.17**	61.09
	Rialto	36.63	47.47	27.52	43.30	44.88	27.26	47.42	**55.65**
	SpamData	89.34	91.73	91.35	90.96	91.39	**91.80**	91.08	91.16
	WineRed	47.70	47.46	47.70	47.63	46.92	47.85	46.92	**48.48**
	WineWhite	42.82	46.90	44.88	43.16	43.41	45.40	45.63	**47.45**
Rank	–	7.21	4.58	4.92	4.08	3.56	5.16	3.34	**3.15**

In absolute terms, USDD delivers competitive accuracies across all concept sizes and dataset generators, with both abrupt and gradual concept changes, as well as in real-world datasets, with both HT and NB. Nevertheless, it is easy to note that the very best results in the individual datasets were split among several methods in the tests with both base learners, i.e., there was no absolute dominance of any method, and this was to be expected because the selected drift detectors were the very best among those available at the moment. Despite this, it is worthwhile highlighting that the performance of USDD was solid in most situations and, even though the experiments also included an ensemble (DDE) configured to deliver its best possible accuracy results and to serve as upper-bound, USDD was the top-ranked method in the tests with both classifiers. These results are evidence of its efficiency, in spite of the simplicity of its decision strategy.

Another point worth noting is that USDD was very effective in the real-world datasets as well. However, it is also important to mention that some real-world datasets may not have any concept drift, while others may have very gradual concept changes, temporal dependencies on the class label, and/or imbalanced data. Such scenarios might make several methods deliver similar results because none of them were designed for such situations.

Complementing the analysis of the reported results, we applied the Bonferroni-Dunn post-hoc test (Demšar, 2006) using USDD as the base method. They were applied twice, once for the results using each base learner. The results are presented using graphics, where the CD is represented by a red bar: methods that are connected to USDD by the bar are statistically similar and the methods on the right hand side which are not connected by the bar are statistically inferior to USDD. The results of the tests with the data at Tables 2 and 3 are summarized in Figures 1 and 2, respectively. Note that, with both HT and NB, USDD followed by DDE were the best methods in the evaluation, whereas DDM and WSTD were the lowest-ranked ones.

Figure 1 graphically represents the results using HT as base classifier. The numeric values of the ranks can be seen below the graphics as well as in the last line of table 2. Although there is a difference between values in absolute terms, in statistical terms it is only possible to state that USDD is significantly better than DDM with respect to the accuracy of the classification. On the other hand, as illustrated in Figure 2, using NB, USDD was statistically superior to FTDD, WSTD, and DDM.

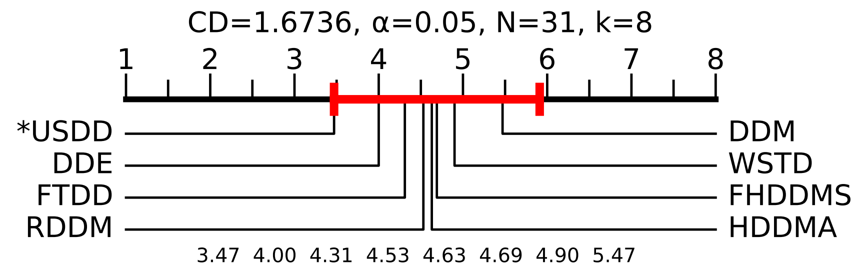

Figure 1: Accuracy statistical comparison of the methods with HT using the Bonferroni-Dunn post-hoc test on all tested datasets.

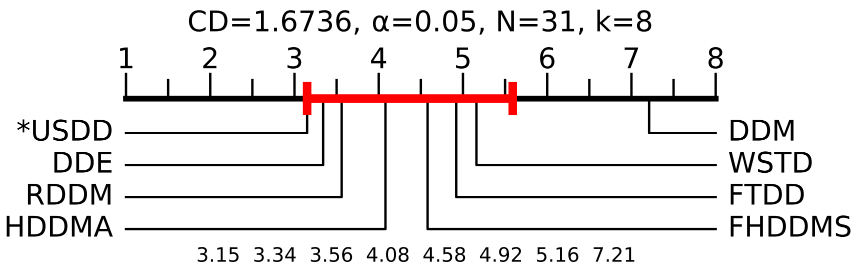

Figure 2: Accuracy statistical comparison of the methods with NB using the Bonferroni-Dunn post-hoc test on all tested datasets.

5.2. Drift identification analysis

The identifications of concept drifts in the selected datasets is another dimension that can be used to evaluate the performance of drift detectors. To this end, a number of different metrics are usually computed in the whole process of identifying the drifts. These metrics are Mean Distance (μD) of the identifications to the exact points of the drifts, False Negatives (FN), False Positives (FP), Precision and Recall (Fawcett, 2006), and MCC (Matthews, 1975).

Statistically, a false negative occurs when the drift exists but it is not detected: this is known as error type 1. On the other hand, a false positive happens when a drift is detected but in fact it does not exist (error type 2). Precision is defined as $TP\ /\ (TP+FP)$ and returns the proportion of the detected drifts that are existing drifts. Recall is computed by $TP\ /\ (TP + FN)$ and its result is the proportion of the existing concept drifts that are correctly detected by each method (Barros et al., 2018). These two metrics are usually computed together. Finally, MCC is a correlation coefficient between the observed and estimated binary classifications of the detections (drift or not drift) and is considered one of the most adequate measures of the detections of the methods. It returns values between -1 and $+1$: the value $+1$ represents a perfect forecast, 0 is no better than the random forecast, and -1 indicates total disagreement between forecasts and observations (Matthews, 1975). The value of MCC is determined using the following equation:

$$\mathrm{MCC} = \frac{(TP \times TN - FP \times FN)}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}$$

Tables 4 and 5 present the results of the experiments executed with the metrics described above to evaluate the performance of the methods concerning the identifications of the concept drifts in the abrupt datasets with classifiers HT and NB respectively. The best results for each metric evaluated are written in **bold**. Note that in the measures of μD, FN, and FP, the lowest values are those that determine the best performance; in the case of precision, recall and MCC, the highest values obtained are the best. It is also important to point out that to categorize the detections as True Positives (TP) or FP, a tolerance interval of 100 instances from the correct points of drift was considered.

The results of the experiments using HT, presented in Table 4, indicate that USDD (and also FHDDMS) delivered its best performances in all sizes of the Sine dataset, obtaining most of the best values of the calculated metrics. In the LED datasets, USDD also delivered several of the best results in the datasets of 130K and 890K instances. In addition, observe that, on several datasets, USDD achieved good results of MCC, the most general of the calculated metrics, and this made it the best-ranked method by this criterion.

Using NB as base learner (Table 5), the detections of USDD were not that different and this was reflected in its MCC and precision results: USDD achieved the best rank in both metrics. And, again, its best results were obtained in the Sine datasets, though it also delivered good results in some of the LED and Agrawal datasets.

Finally, it is worth noting that the detection results of USDD in the RRBF datasets were not good, especially the larger ones, despite its competitive accuracy results in these datasets.

6. Conclusions

This paper presents USDD, an innovative, efficient, and ultimately simple concept drift detection algorithm. Its main strength lies on the simplicity of the strategy adopted to identify concept drifts based on the classification errors of the base learner. The initial motivation came from the common boosting strategy of discarding members of the ensemble with error rate greater than 1/2.

The proposed method was tested with two different base learners, Naïve Bayes (NB) and Hoeffding Tree (HT), using 24 artificial and seven real-world datasets, against six of the very-best concept drift detectors available, as well as an ensemble of detectors set with its best configuration for maximizing the classification

Table 4: Concept drift identifications of the methods in the abrupt datasets using Hoeffding Tree as base classifier.

Detector	μD	FN	FP	Precision	Recall	MCC	Dataset	Detector	μD	FN	FP	Precision	Recall	MCC
DDM	49.06000	2.43000	10.50000	0.14450	0.39170	0.23390		RDDM	51.11000	0.96670	1.20000	0.72110	0.75830	0.73700
FHDDMS	33.86000	**0.06670**	0.50000	0.89890	**0.98330**	0.93850	Agraw.	WSTD	**25.48000**	0.23330	0.33330	0.92830	0.94170	0.93300
FTDD	28.00000	0.86670	0.90000	0.79290	0.78330	0.78480	20k	DDE	37.22000	0.26670	**0.16670**	**0.96170**	0.93330	**0.94560**
HDDMA	33.07000	0.46670	0.40000	0.90500	0.88330	0.89210		USDD	63.75000	0.50000	1.23000	0.76960	0.87500	0.81640
DDM	N/A	4.00000	4.17000	0.00000	0.00000	-0.00610		RDDM	91.50000	3.93000	4.03000	0.01670	0.01670	0.01650
FHDDMS	48.87000	**1.43000**	3.10000	**0.51780**	**0.64170**	**0.56940**	LED	WSTD	**43.57000**	2.77000	8.43000	0.16370	0.30830	0.21500
FTDD	63.13000	3.73000	**2.17000**	0.07330	0.06670	0.06860	20k	DDE	61.44000	3.47000	2.83000	0.16670	0.13330	0.14740
HDDMA	70.12000	3.43000	3.40000	0.14170	0.14170	0.14150		USDD	69.83000	2.43000	2.63000	0.37260	0.39170	0.37970
DDM	78.11000	3.70000	3.50000	0.08000	0.07500	0.07690		RDDM	62.63000	2.30000	4.03000	0.31370	0.42500	0.36250
FHDDMS	**35.79000**	**0.93330**	2.90000	0.53720	**0.76670**	0.63850	RRBF	WSTD	36.42000	1.20000	1.20000	**0.72600**	0.70000	**0.70720**
FTDD	42.45000	1.83000	**1.03000**	0.68000	0.54170	0.60480	20k	DDE	48.35000	1.40000	1.43000	0.66940	0.65000	0.65630
HDDMA	49.71000	1.93000	2.13000	0.50500	0.51670	0.50840		USDD	57.07000	1.13000	1.67000	0.69210	0.71670	0.69710
DDM	43.41000	0.16670	3.73000	0.58810	0.95830	0.73670		RDDM	25.78000	0.06670	4.47000	0.58040	0.98330	0.73570
FHDDMS	14.88000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**	Sine	WSTD	**12.59000**	**0.00000**	0.03330	0.99330	**1.00000**	0.99650
FTDD	13.54000	**0.00000**	0.26670	0.94890	**1.00000**	0.97280	20k	DDE	17.01000	**0.00000**	0.16670	0.96670	**1.00000**	0.98240
HDDMA	13.88000	**0.00000**	0.13330	0.97560	**1.00000**	0.98680		USDD	40.64000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**
DDM	70.90000	3.03000	21.00000	0.07090	0.24170	0.12370		RDDM	48.31000	2.93000	3.30000	0.24780	0.26670	0.25650
FHDDMS	30.18000	**0.00000**	1.53000	0.78910	**1.00000**	0.87990	Agraw.	WSTD	**19.87000**	**0.00000**	0.53330	0.90780	**1.00000**	**0.94980**
FTDD	22.16000	0.06670	0.60000	0.88330	0.98330	0.92980	130k	DDE	41.65000	0.23330	**0.30000**	**0.93000**	0.94170	0.93550
HDDMA	32.97000	0.16670	0.50000	0.89560	0.95830	0.92490		USDD	65.10000	0.56670	1.77000	0.72000	0.85830	0.77900
DDM	N/A	4.00000	**3.70000**	0.00000	0.00000	-0.00090		RDDM	N/A	4.00000	4.90000	0.00000	0.00000	-0.00100
FHDDMS	**56.31000**	**1.73000**	23.63000	0.09920	**0.56670**	0.23530	LED	WSTD	62.90000	2.67000	23.73000	0.07470	0.33330	0.15180
FTDD	62.92000	3.60000	6.07000	0.06670	0.10000	0.08080	130k	DDE	65.82000	3.43000	4.07000	0.13200	0.14170	0.13510
HDDMA	70.38000	3.57000	4.27000	0.09470	0.10830	0.10040		USDD	66.87000	2.17000	4.33000	**0.37530**	0.45830	**0.40670**
DDM	N/A	4.00000	4.20000	0.00000	0.00000	-0.00090		RDDM	N/A	4.00000	6.03000	0.00000	0.00000	-0.00110
FHDDMS	42.42000	**0.20000**	13.37000	0.24270	**0.95000**	0.47430	RRBF	WSTD	**30.68000**	0.70000	3.27000	0.54010	0.82500	0.66270
FTDD	38.98000	1.13000	**1.27000**	**0.71220**	0.71670	**0.71150**	130k	DDE	49.84000	1.33000	2.07000	0.58860	0.66670	0.62100
HDDMA	44.66000	1.53000	3.63000	0.41450	0.61670	0.50330		USDD	63.01000	1.13000	3.63000	0.52480	0.71670	0.59830
DDM	56.20000	2.00000	5.57000	0.28630	0.50000	0.37450		RDDM	56.15000	0.10000	4.97000	0.56820	0.97500	0.72270
FHDDMS	14.77000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**	Sine	WSTD	11.64000	**0.00000**	0.20000	0.96220	1.00000	0.97980
FTDD	**11.55000**	**0.00000**	0.16670	0.96890	**1.00000**	0.98330	130k	DDE	19.32000	**0.00000**	0.06670	0.98670	**1.00000**	0.99300
HDDMA	14.72000	**0.00000**	0.13330	0.97560	**1.00000**	0.98680		USDD	41.22000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**
DDM	N/A	4.00000	20.90000	0.00000	0.00000	-0.00030		RDDM	53.00000	1.57000	40.40000	0.06030	0.60830	0.19020
FHDDMS	27.78000	**0.00000**	1.77000	0.77350	**1.00000**	0.86930	Agraw.	WSTD	**18.96000**	**0.00000**	0.76670	0.87330	**1.00000**	0.93070
FTDD	19.27000	**0.00000**	0.40000	0.92440	**1.00000**	0.95960	890k	DDE	39.09000	0.10000	**0.20000**	**0.95500**	0.97500	**0.96440**
HDDMA	46.56000	1.13000	1.80000	0.64510	0.71670	0.67670		USDD	66.47000	0.16670	1.33000	0.80490	0.95830	0.87100
DDM	N/A	4.00000	**4.23000**	0.00000	0.00000	-0.00010		RDDM	67.40000	3.83000	34.47000	0.00390	0.04170	0.01270
FHDDMS	50.60000	**1.43000**	152.00000	0.01710	**0.64170**	0.10440	LED	WSTD	**49.63000**	2.63000	103.87000	0.01340	0.34170	0.06730
FTDD	57.06000	3.40000	33.70000	0.01830	0.15000	0.05240	890k	DDE	64.78000	3.10000	25.30000	0.03300	0.22500	0.08580
HDDMA	67.50000	3.40000	29.87000	0.01880	0.15000	0.05290		USDD	65.55000	**1.43000**	30.30000	**0.08430**	**0.64170**	**0.23140**
DDM	N/A	4.00000	5.63000	0.00000	0.00000	-0.00020		RDDM	74.22000	3.70000	20.97000	0.01290	0.07500	0.03110
FHDDMS	38.76000	**0.26670**	58.40000	0.06620	**0.93330**	0.24550	RRBF	WSTD	**28.49000**	0.63330	10.00000	0.26970	0.84170	0.47100
FTDD	35.48000	0.83330	**1.30000**	**0.71960**	0.79170	**0.75260**	890k	DDE	52.06000	1.03000	2.43000	0.58980	0.74170	0.65560
HDDMA	46.79000	1.90000	6.67000	0.25940	0.52500	0.36460		USDD	61.62000	1.20000	4.80000	0.49310	0.70000	0.57040
DDM	83.38000	3.03000	7.43000	0.12610	0.24170	0.17280		RDDM	37.06000	0.06670	29.33000	0.12880	0.98330	0.35190
FHDDMS	15.19000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**	Sine	WSTD	11.14000	**0.00000**	0.16670	0.96890	**1.00000**	0.98330
FTDD	**10.52000**	**0.00000**	0.26670	0.95240	**1.00000**	0.97430	890k	DDE	15.13000	**0.00000**	0.03330	0.99330	**1.00000**	0.99650
HDDMA	18.53000	0.03330	0.16670	0.96500	0.99170	0.97760		USDD	42.87000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**
DDM	1.25000	1.08333	3.08333	7.83333	7.91667	7.75000		RDDM	2.33333	1.91667	2.08333	7.16667	7.08333	7.25000
FHDDMS	6.16667	**7.25000**	4.04167	3.70833	**1.75000**	3.12500	Rank	WSTD	**7.66667**	6.37500	4.37500	3.58333	2.62500	3.08333
FTDD	6.75000	4.83333	5.41667	4.08333	4.16667	4.41667		DDE	4.66667	5.00000	**6.66667**	**2.41667**	4.00000	2.83333
HDDMA	4.41667	4.04167	4.91667	4.41667	4.95833	4.83333		USDD	2.75000	5.50000	5.41667	2.79167	3.50000	**2.70833**

accuracy of the base learners in the chosen datasets, serving as upper-bound. The accuracy results of the experiments were also statistically evaluated using the Friedman test along with the Bonferroni-Dunn post-hoc test.

In addition to comparing accuracy, the experiments also evaluated the detections of concept drifts based on mean distance to drifts (μD), FN, FP, Precision, Recall, and the Matthews Correlation Coefficient (MCC). The results showed USDD as the best ranked method in accuracy as well as in MCC with both base classifiers, though not statistically different to some other methods.

As future work, we propose investigating theoretical guarantees to the results depending on the chosen sliding window size as well as to the stability of the learned concepts. Empirically, we observed that longer concepts lead to more precise models and, in these situations, USDD could deliver competent results with smaller window sizes.

Table 5: Concept drift identifications of the methods in the abrupt datasets using Naive Bayes as base classifier.

Detector	μD	FN	FP	Precision	Recall	MCC	Dataset	Detector	μD	FN	FP	Precision	Recall	MCC
DDM	53.74000	2.10000	12.40000	0.16680	0.47500	0.27350		RDDM	47.86000	0.76670	2.20000	0.64370	0.80830	0.71580
FHDDMS	31.55000	**0.03330**	0.46670	0.90720	**0.99170**	0.94680	Agraw.	WSTD	**27.00000**	0.10000	**0.13330**	**0.97000**	0.97500	**0.97180**
FTDD	27.87000	0.90000	0.93330	0.75290	0.77500	0.76130	20k	DDE	27.19000	0.20000	8.97000	0.32220	0.95000	0.54640
HDDMA	34.48000	0.56670	0.73330	0.82890	0.85830	0.84250		USDD	62.61000	**0.03330**	0.66670	0.88170	**0.99170**	0.93190
DDM	N/A	4.00000	4.20000	0.00000	0.00000	-0.00610		RDDM	91.50000	3.93000	4.03000	0.01670	0.01670	0.01650
FHDDMS	48.87000	**1.43000**	3.10000	**0.51780**	**0.64170**	**0.56940**	LED	WSTD	**41.69000**	2.83000	7.13000	0.18650	0.29170	0.22390
FTDD	63.13000	3.73000	**2.00000**	0.08170	0.06670	0.07210	20k	DDE	69.72000	3.40000	10.07000	0.05100	0.15000	0.08680
HDDMA	70.35000	3.43000	3.40000	0.14170	0.14170	0.14150		USDD	69.83000	2.43000	2.63000	0.37260	0.39170	0.37970
DDM	75.67000	3.80000	4.33000	0.04590	0.05000	0.04730		RDDM	70.53000	2.23000	3.30000	0.35740	0.44170	0.39520
FHDDMS	**34.81000**	**0.90000**	2.67000	0.56930	**0.77500**	0.66020	RRBF	WSTD	38.84000	1.13000	1.13000	**0.74720**	0.71670	**0.72790**
FTDD	45.25000	1.57000	**0.96670**	0.74000	0.60830	0.66870	20k	DDE	44.03000	1.37000	7.43000	0.27380	0.65830	0.42190
HDDMA	52.35000	1.73000	2.03000	0.54000	0.56670	0.55200		USDD	55.47000	1.13000	1.67000	0.68600	0.71670	0.69460
DDM	48.13000	0.93330	2.40000	0.59450	0.76670	0.66960		RDDM	30.04000	**0.00000**	1.13000	0.83730	**1.00000**	0.90780
FHDDMS	14.50000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**	Sine	WSTD	13.53000	**0.00000**	0.13330	0.97780	**1.00000**	0.98780
FTDD	14.20000	**0.00000**	0.06670	0.98670	**1.00000**	0.99300	20k	DDE	**11.41000**	**0.00000**	5.57000	0.43590	**1.00000**	0.65700
HDDMA	20.52000	**0.00000**	0.10000	0.98220	**1.00000**	0.99040		USDD	38.21000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**
DDM	62.10000	3.03000	27.70000	0.06090	0.24170	0.11230		RDDM	50.25000	2.67000	6.37000	0.21560	0.33330	0.26190
FHDDMS	31.42000	0.13330	5.97000	0.40710	0.96670	0.62460	Agraw.	WSTD	**25.96000**	**0.10000**	1.07000	0.81560	**0.97500**	**0.88820**
FTDD	29.08000	0.50000	**0.63330**	**0.85170**	0.87500	0.86270	130k	DDE	28.70000	0.30000	20.10000	0.17990	0.92500	0.40060
HDDMA	41.65000	1.50000	2.03000	0.55710	0.62500	0.58840		USDD	62.98000	**0.10000**	3.03000	0.61150	**0.97500**	0.76430
DDM	N/A	4.00000	4.07000	0.00000	0.00000	-0.00090		RDDM	76.67000	3.90000	4.83000	0.01110	0.02500	0.01660
FHDDMS	**56.31000**	**1.73000**	23.63000	0.09920	**0.56670**	0.23530	LED	WSTD	58.40000	2.67000	19.00000	0.08730	0.33330	0.16590
FTDD	62.91000	3.63000	**4.20000**	0.06960	0.09170	0.07910	130k	DDE	68.46000	3.07000	12.90000	0.08050	0.23330	0.13350
HDDMA	71.08000	3.57000	4.27000	0.09470	0.10830	0.10040		USDD	66.87000	2.17000	4.33000	**0.37530**	0.45830	**0.40670**
DDM	62.67000	3.90000	4.93000	0.02890	0.02500	0.02670		RDDM	51.00000	3.57000	8.03000	0.04370	0.10830	0.06830
FHDDMS	38.31000	**0.70000**	19.23000	0.14790	**0.82500**	0.34860	RRBF	WSTD	**36.97000**	1.10000	4.50000	0.43460	0.72500	0.55170
FTDD	41.52000	1.50000	**1.60000**	**0.63060**	0.62500	**0.62550**	130k	DDE	40.11000	1.37000	14.57000	0.16110	0.65830	0.32270
HDDMA	53.16000	2.77000	3.73000	0.25790	0.30830	0.28100		USDD	54.63000	1.03000	9.50000	0.25510	0.74170	0.43080
DDM	71.15000	3.10000	6.83000	0.12560	0.22500	0.16420		RDDM	51.82000	0.90000	7.10000	0.38120	0.77500	0.52810
FHDDMS	13.91000	**0.00000**	0.33330	0.95000	**1.00000**	0.97190	Sine	WSTD	12.59000	**0.00000**	0.33330	0.94440	**1.00000**	0.96940
FTDD	13.35000	**0.00000**	0.13330	0.97560	**1.00000**	0.98680	130k	DDE	**11.43000**	**0.00000**	15.13000	0.22750	**1.00000**	0.47090
HDDMA	41.72000	1.00000	1.20000	0.73060	0.75000	0.73950		USDD	36.89000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**
DDM	N/A	4.00000	35.80000	0.00000	0.00000	-0.00040		RDDM	52.68000	1.70000	58.87000	0.04040	0.57500	0.15130
FHDDMS	29.83000	0.16670	43.40000	0.08190	0.95830	0.27990	Agraw.	WSTD	**25.65000**	0.06670	8.83000	0.34520	0.98330	0.57510
FTDD	29.35000	0.46670	**0.70000**	**0.84330**	0.88330	**0.86220**	890k	DDE	27.54000	0.26670	74.67000	0.04990	0.93330	0.21440
HDDMA	47.00000	2.77000	4.83000	0.21510	0.30830	0.25510		USDD	62.70000	**0.03330**	15.40000	0.21980	**0.99170**	0.46260
DDM	N/A	4.00000	**4.07000**	0.00000	0.00000	-0.00010		RDDM	57.67000	3.80000	31.33000	0.00580	0.05000	0.01700
FHDDMS	**50.60000**	**1.43000**	152.00000	0.01710	**0.64170**	0.10440	LED	WSTD	52.26000	2.73000	78.83000	0.01870	0.31670	0.07500
FTDD	58.88000	3.73000	6.87000	0.03720	0.06670	0.04900	890k	DDE	61.76000	2.87000	60.53000	0.01910	0.28330	0.07310
HDDMA	70.40000	3.67000	6.70000	0.04600	0.08330	0.06160		USDD	68.49000	2.03000	14.10000	**0.17220**	0.49170	**0.27810**
DDM	N/A	4.00000	5.30000	0.00000	0.00000	-0.00020		RDDM	64.93000	3.53000	44.83000	0.01010	0.11670	0.03430
FHDDMS	**35.74000**	**0.93330**	135.57000	0.02220	**0.76670**	0.13030	RRBF	WSTD	36.81000	1.33000	28.40000	0.08740	0.66670	0.24030
FTDD	37.90000	1.73000	**4.30000**	**0.36130**	0.56670	**0.44960**	890k	DDE	38.10000	1.40000	73.47000	0.03460	0.65000	0.14950
HDDMA	57.08000	3.20000	7.57000	0.09730	0.20000	0.13810		USDD	55.34000	1.10000	62.23000	0.04500	0.72500	0.18020
DDM	67.00000	3.93000	6.97000	0.00720	0.01670	0.01050		RDDM	40.42000	0.80000	58.87000	0.05340	0.80000	0.20570
FHDDMS	14.18000	**0.00000**	4.87000	0.47220	**1.00000**	0.68270	Sine	WSTD	13.13000	**0.00000**	2.93000	0.63110	**1.00000**	0.78540
FTDD	13.53000	**0.00000**	0.86670	0.86920	**1.00000**	0.92660	890k	DDE	**12.54000**	**0.00000**	69.93000	0.05610	**1.00000**	0.23570
HDDMA	43.82000	2.33000	2.97000	0.37830	0.41670	0.39590		USDD	36.85000	**0.00000**	**0.00000**	**1.00000**	**1.00000**	**1.00000**
DDM	1.16667	1.00000	3.91667	7.91667	8.00000	7.91667		RDDM	2.91667	2.58333	3.16667	6.58333	6.41667	6.75000
FHDDMS	6.33333	**7.04167**	3.75000	3.79167	**1.95833**	3.12500	Rank	WSTD	**7.41667**	6.16667	5.04167	2.91667	2.83333	2.50000
FTDD	5.75000	4.00000	**7.08333**	2.66667	5.00000	3.16667		DDE	6.25000	5.16667	1.66667	6.08333	3.83333	5.66667
HDDMA	3.25000	3.25000	5.66667	3.75000	5.75000	4.83333		USDD	2.91667	6.79167	5.70833	**2.29167**	2.20833	**2.04167**

Acknowledgements

Bruno Maciel and Juan Hidalgo are PhD students supported by postgraduate grants from CAPES.

References

Agrawal, R., Imielinski, T., & Swami, A. (1993). Database mining: a performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, *5*, 914–925.

Barros, R. S. M., Cabral, D. R. L., Gonçalves, P. M., Jr., & Santos, S. G. T. C. (2017). RDDM: Reactive drift detection method. *Expert Systems with Applications*, *90*, 344–355.

Barros, R. S. M., Hidalgo, J. I. G., & Cabral, D. R. L. (2018). Wilcoxon rank sum test drift detector. *Neurocomputing*, *275*, 1954–1963.

Barros, R. S. M., & Santos, S. G. T. C. (2018). A large-scale comparison of concept drift detectors. *Information Sciences*, *451-452*, 348–370.

Barros, R. S. M., & Santos, S. G. T. C. (2019). An overview and comprehensive comparison of ensembles for concept drift. *Information Fusion*, *52*, 213–244.

Barros, R. S. M., Santos, S. G. T. C., & Gonçalves Jr., P. M. (2016). A boosting-like online learning ensemble. In *Proceedings of IEEE International Joint Conference on Neural Networks (IJCNN)* (pp. 1871–1878). Vancouver, Canada.

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: massive online analysis. *Journal of Machine Learning Research*, *11*, 1601–1604.

Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., & Gavaldà, R. (2009). New ensemble methods for evolving data streams. In *Proceedings of 15th ACM International Conference on Knowledge Discovery and Data Mining (KDD'09)* (pp. 139–148). Paris, France.

Cabral, D. R. L., & Barros, R. S. M. (2018). Concept drift detection based on Fisher's Exact test. *Information Sciences*, *442-443*, 220–234.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, *47*, 547–553.

Dawid, A. P. (1984). Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society. Series A (General)*, (pp. 278–292).

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journ. of Machine Learning Research*, *7*, 1–30.

Du, L., Song, Q., Zhu, L., & Zhu, X. (2014). A selective detector ensemble for concept drift detection. *The Computer Journal*, *58*, 457–471.

Dunn, O. J. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, *56*, 52–64.

Elena, I. (2008). Airline dataset. URL: `http://kt.ijs.si/elena_ikonomovska/data.html`.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*, 861–874.

Fern, A., & Givan, R. (2003). Online ensemble learning: An empirical study. *Machine Learning*, *53*, 71–109.

Foundation, T. A. S. (2006). Apache SpamAssassin Project. `http://spamassassin.apache.org`.

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In *Proceedings of International Conference on Machine Learning (ICML)* (pp. 148–156).

Frías-Blanco, I., del Campo-Ávila, J., Ramos-Jiménez, G., Morales-Bueno, R., Ortiz-Díaz, A., & Caballero-Mota, Y. (2015). Online and non-parametric drift detection methods based on hoeffding's bounds. *IEEE Transactions on Knowledge and Data Engineering*, *27*, 810–823.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, *32*, 675–701.

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In *Advances in Artificial Intelligence: SBIA 2004* (pp. 286–295). Springer volume 3171 of *LNCS*.

Gama, J., Sebastião, R., & Rodrigues, P. (2013). On evaluating stream learning algorithms. *Machine Learning*, *90*, 317–346.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, *46*, 44:1–37.

Gonçalves, P. M., Jr., & Barros, R. S. M. (2013). Speeding up statistical tests to detect recurring concept drifts. In R. Lee (Ed.), *Computer and Information Science* (pp. 129–142). Springer volume 493 of *Studies in Computational Intelligence*.

Gonçalves Jr., P. M., Santos, S. G. T. C., Barros, R. S. M., & Vieira, D. C. L. (2014). A comparative study on concept drift detectors. *Expert Systems with Applications*, *41*, 8144–8156.

Hidalgo, J. I. G., Maciel, B. I. F., & Barros, R. S. M. (2019). Experimenting with prequential variations for data stream learning evaluation. *Computational Intelligence*, . URL: `https://doi.org/10.1111/coin.12208`.

Ikonomovska, E., Gama, J., & Džeroski, S. (2011). Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery*, *23*, 128–168.

Katakis, I., Tsoumakas, G., & Vlahavas, I. (2008). Machine learning & knowledge discovery group. URL: `http://mlkd.csd.auth.gr/concept_drift.html`.

Katakis, I., Tsoumakas, G., & Vlahavas, I. (2010). Tracking recurring contexts using ensemble classifiers: an application to email filtering. *Knowledge and Information Systems*, *22*, 371–391.

Losing, V., Hammer, B., & Wersing, H. (2016). Knn classifier with self adjusting memory for heterogeneous concept drift. In *2016 IEEE 16th International Conference on Data Mining (ICDM)* (pp. 291–300).

Maciel, B. I. F., Santos, S. G. T. C., & Barros, R. S. M. (2015). A lightweight concept drift detection ensemble. In *Proceedings of 27th IEEE Internat. Conference on Tools with Artificial Intelligence (ICTAI'15)* (pp. 1061–1068). Vietri sul Mare, Italy.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, *405*, 442–451.

Minku, L. L. (2010). *Online Ensemble Learning in the Presence of Concept Drift*. Ph.D. thesis School of Computer Science, The University of Birmingham.

Mitchell, T. (1997). *Machine Learning*. New York, NY, USA: McGraw-Hill.

Nishida, K., & Yamauchi, K. (2007). Detecting concept drift using statistical testing. In *Proceedings of 10th International Conference on Discovery Science (DS'07)* (pp. 264–269). Springer volume 4755 of *LNCS*.

Oza, N. C., & Russell, S. (2001). Experimental comparisons of online and batch versions of bagging and boosting. In *Proceedings of International Conference on Knowledge Discovery and Data Mining* (pp. 359–364).

Pesaranghader, A., Viktor, H., & Paquet, E. (2018). Reservoir of diverse adaptive learners and stacking fast hoeffding drift detection methods for evolving data streams. *Machine Learning*, *107*, 1711–1743.

Pesaranghader, A., & Viktor, H. L. (2016). Fast hoeffding drift detection method for evolving data streams. In P. Frasconi, N. Landwehr, G. Manco, & J. Vreeken (Eds.), *Machine Learning and Knowledge Discovery in Databases* (pp. 96–111). Springer.

Polikar, R., Upda, L., Upda, S. S., & Honavar, V. (2001). Learn++: an incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *31*, 497–508.

Santos, S. G. T. C., & Barros, R. S. M. (2019). Online adaboost-based methods for multiclass problems. *Artificial Intelligence Review*, . URL: `https://doi.org/10.1007/s10462-019-09696-6`.

Santos, S. G. T. C., Barros, R. S. M., & Gonçalves Jr., P. M. (2015). Optimizing the parameters of drift detection methods using a genetic algorithm. In *Proceedings of 27th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'15)* (pp. 1077–1084). Vietri sul Mare, Italy.

Santos, S. G. T. C., Barros, R. S. M., & Gonçalves Jr., P. M. (2019). A differential evolution based method for tuning concept drift detectors in data streams. *Information Science*, *485*, 376–393.

Santos, S. G. T. C., Gonçalves Jr., P. M., Silva, G. D. S., & Barros, R. S. M. (2014). Speeding up recovery from concept drifts. In *Machine Learning and Knowledge Discovery in Databases* (pp. 179–194). Springer volume 8726 of *LNCS*.

Sejdic, E., & Falk, T. H. (2018). *Signal Processing and Machine Learning for Biomedical Big Data*. CRC Press.

Tromp, J. (1995). UCI machine learning repository. URL: `http://archive.ics.uci.edu/ml`.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics bulletin*, *1*, 80–83.

ESWA-2019-USDD-Highlights.txt

- USDD: a novel and very simple concept drift detector
- USDD detects drifts when errors are above 50% on a reference sliding window.
- The detection strategy is inspired on boosting methods.
- USDD was tested against state of the art detectors using two base learners.
- USDD was superior to the other six methods in both accuracy and detections.

***Conflict of Interest**

Declaration of interests

☓ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: