Class Distribution Monitoring for Concept Drift Detection

Diego Stucchi
Politecnico di Milano
Milan, Italy
diego.stucchi@polimi.it

Luca Frittoli
Politecnico di Milano
Milan, Italy
luca.frittoli@polimi.it

Giacomo Boracchi
Politecnico di Milano
Milan, Italy
giacomo.boracchi@polimi.it

***Abstract*—We introduce Class Distribution Monitoring (CDM), an effective concept-drift detection scheme that monitors the class-conditional distributions of a datastream. In particular, our solution leverages multiple instances of an online and nonparametric change-detection algorithm based on QuantTree. CDM reports a concept drift after detecting a distribution change in any class, thus identifying which classes are affected by the concept drift. This can be precious information for diagnostics and adaptation. Our experiments on synthetic and real-world datastreams show that when the concept drift affects a few classes, CDM outperforms algorithms monitoring the overall data distribution, while achieving similar detection delays when the drift affects all the classes. Moreover, CDM outperforms comparable approaches that monitor the classification error, particularly when the change is not very apparent. Finally, we demonstrate that CDM inherits the properties of the underlying change detector, yielding an effective control over the expected time before a false alarm, or Average Run Length (ARL_0).**

***Index Terms*—concept drift detection, online change detection, supervised learning, multivariate datastreams**

I. Introduction

Datastreams represent a challenging scenario for machine learning models [1] since their distribution might change over time, resulting in a *concept drift* [2]. This phenomenon has been widely studied in settings where the drift worsens the performance of a classifier, which must be adapted to the new data distribution. To this purpose, most solutions monitor the classification error, ignoring drifts that have little impact on the error rate, which are called *virtual drifts*. However, in practical situations such as in industrial monitoring, any distribution change in streaming data should be promptly detected for diagnostic purposes. Moreover, in the emerging field of *open-set recognition* [3], a classifier is required to recognize the occurrence of known classes and also to detect samples that do not belong to any known class, thus it is crucial to update the decision boundary of the classifier even when the accuracy on known classes does not decrease. This enables updating also the regions in which the classifier predicts with low confidence, where unknown samples might appear.

Most concept drifts can be detected by monitoring the data distribution by *online change-detection tests* [4], which is another common approach in concept-drift detection [2]. However, none of these methods can exploit supervised information since they overlook class labels. Our intuition is that class labels can be included in statistically sound change-detection tests to monitor the class-conditional distributions instead of the overall data distribution. To the best of our knowledge, this approach has never been investigated before.

We fill this gap by proposing *Class Distribution Monitoring* (CDM)[1], in which we employ separate instances of *QuantTree Exponentially Weighted Moving Average* (QT-EWMA) [5] to monitor the class-conditional distributions. QT-EWMA is a nonparametric online change-detection test based on QuantTree histograms [6], and is designed to monitor multivariate datastreams. We report a concept drift after detecting a change in the class-conditional distribution of at least one class. The main advantages of CDM are: *i)* it can detect any relevant drift, including virtual ones that have little impact on the classification error and are by design ignored by methods that monitor the error rate of a classifier; *ii)* it can detect concept drifts affecting only a subset of classes more promptly than methods that monitor the overall data distribution, since the other class-conditional distributions do not change; *iii)* it provides insights on which classes have been affected by concept drift, which might be crucial for diagnostics and adaptation; *iv)* it effectively controls false alarms by maintaining a target *Average Run Length* (ARL_0), i.e., the expected time before a false alarm [4], which can be set before monitoring.

To summarize, our main contributions are:

- We introduce Class Distribution Monitoring (CDM), a novel online and nonparametric monitoring scheme for concept-drift detection leveraging supervised samples.
- Our CDM can, by design, detect drifts affecting only a subset of classes and, contrarily to most concept drift detectors, identify the drifted classes.
- We theoretically and empirically demonstrate that CDM can be configured to yield the desired ARL_0, thus effectively controlling false alarms, even though it employs several change-detection tests simultaneously.

Our experiments on synthetic and real-world datastreams show that CDM outperforms algorithms monitoring the overall distribution when the concept drift affects only a subset of classes, while achieving comparable detection delays when the change affects all classes. CDM can also effectively detect

[1] Our code is available at https://boracchi.faculty.polimi.it/Projects. This paper is in the Proceedings of the International Joint Conference on Neural Networks ©2022 IEEE, DOI: 10.1109/IJCNN55064.2022.9892772.

virtual drifts, which are ignored by methods that monitor the classification error but might be relevant in practice.

II. Problem Formulation

We address the problem of detecting a concept drift in a virtually unlimited datastream $\{(x_t, y_t)\}$, where each sample $x_t \in \mathbb{R}^d$ is associated to a class label $y_t \in \{1, \dots, M\}$. We assume that the observations x_t are independent realizations of a random vector that follows an initial distribution ϕ_0. We denote by ϕ_0^m the *class-conditional* distribution, i.e., the distribution of instances belonging to class m, defined by

$$\mathbb{P}_{\phi_0^m}(x_t) = \mathbb{P}_{\phi_0}(x_t \mid y_t = m), \tag{1}$$

for each $m \in \{1, \dots, M\}$. In other words, we say that $x_t \sim \phi_0$ if and only if $x_t \sim \phi_0^m$, where $y_t = m$. We assume that a concept drift affects at least one class-conditional distribution, resulting in a change $\phi_0^m \to \phi_1^m$ occurring at an unknown time τ for some $m \in \{1, \dots, M\}$. We assume that an annotated dataset TR sampled from the initial distribution ϕ_0 is provided before monitoring, to configure the concept-drift detector.

In the concept-drift detection literature [7]–[10] it is usually assumed that, during monitoring, the true labels y_t are revealed after the prediction made by a classifier $\mathcal{K}$, to provide immediate feedback on whether the classification was correct. We operate in the same settings, even though in practical situations the labels are typically provided only for a few samples of the datastream. In the latter case, methods that require the true labels can take as input only those samples x_t for which the label y_t is provided.

The goal of a concept-drift detection algorithm is detecting any distribution change as soon as possible by analyzing the incoming samples. We indicate by t^* the detection time, and we measure the detection performance by the detection delay $t^* - \tau$. A crucial challenge in change detection is controlling false alarms, which in online settings means maintaining a target *Average Run Length* (ARL_0), defined as

$$\text{ARL}_0 = \mathbb{E}_{\phi_0}[t^*], \tag{2}$$

which is the expected time before having a false alarm, namely a detection that does not correspond to any distribution change [4]. The ARL_0 represents the online counterpart of the false positive rate in statistical hypothesis testing. Operating at a controlled ARL_0 allows to limit the frequency of false alarms, which typically trigger costly adaptation procedures such as re-training a classifier. This is particularly important in industrial monitoring, and in experimental test-bed to enable a fair comparison between different concept-drift detectors. Unfortunately, the vast majority of concept-drift detection methods fail to control the ARL_0 effectively.

III. Related Work

Concept-drift detection [2] is a challenging problem in datastream learning, and has been addressed in different settings and by different approaches, which we summarize here. Since we focus only on concept-drift detection, we do not review the literature on *concept-drift adaptation*. We refer to [11] for a survey on this subject.

The most popular concept-drift detection methods analyze the binary stream $\{e_t\}$ defined by the errors of a classifier $\mathcal{K}$:

$$e_t = \mathbb{1}(\mathcal{K}(x_t) \neq y_t), \tag{3}$$

and report a concept drift when the error rate increases. In particular, *Drift Detection Method* (DDM) [7] and its variants [8], [9] apply statistical tests on recent windows of $\{e_t\}$ to assess whether the error rate has increased significantly. Moreover, it has recently been proposed to monitor the classification performance on individual classes to handle imbalanced datastreams and drifts affecting only some classes [12], [13]. However, none of these solutions can be configured to maintain the ARL_0. In contrast, *EWMA for Concept Drift Detection* (ECDD) [14] analyzes $\{e_t\}$ online by an *Exponentially Weighted Moving Average* (EWMA) chart [15], which enables controlling false alarms by setting the ARL_0. Thanks to this property, in our experiments we can fairly compare ECDD and our solution by configuring them to maintain the same ARL_0.

Another relevant class of concept drift detection methods monitors the distribution of the input data, overlooking the information possibly coming from class labels, and therefore can operate also when few or no labels are available. These methods leverage online change-detection tests [4] to analyze the data distribution over time. A popular approach consists in monitoring the likelihood of the streaming data with respect to a density model such as a Gaussian [16] or a Gaussian Mixture [17], [18]. The main limitation of these approaches is the assumption that ϕ_0 can be approximated by a distribution from a known family, which might not be the case when dealing with real-world data.

A very flexible nonparametric approach consists in modelling the initial data distribution by a histogram [5], [6], [19], and then monitoring the proportion of incoming samples that falls in each bin of the histogram. For instance, QuantTree [6] computes the Pearson test statistic [20] over fixed-size batches, while its extension QT-EWMA [5] enables online monitoring controlling the ARL_0. Other nonparametric online change detectors are either based on PCA [21], [22], permutation tests [23], [24], or the Maximum Mean Discrepancy statistic (MMD) [25] computed over sliding windows [26], [27]. However, among these, the only one that can control the ARL_0 regardless of the initial data distribution is Scan-B [26].

IV. Proposed Solution

Here we briefly introduce QT-EWMA [5] (Section IV-A), which is the change-detection algorithm we use to define our solution. Then, we present Class Distribution Monitoring (CDM) (Section IV-B). Finally, we demonstrate that CDM inherits the properties of QT-EWMA and analyze its computational complexity (Section IV-C). In particular, we show that CDM can control false alarms by yielding the desired ARL_0.

A. Concept Drift Detection by Distribution Monitoring

Most concept-drift detection methods that monitor the distribution of the datastream $\{x_t\}$ compute at each time t a test statistic T_t, and report a drift after detecting a distribution change [2]. Typically, a change is detected when $T_t > h_t$, where h_t is a threshold defined to control the probability of having a false alarm. The detection time t^* is defined as the first time t in which the statistic exceeds the threshold. We adopt QuantTree Exponentially Weighted Moving Average (QT-EWMA) [5], which effectively controls the ARL_0 and is also completely nonparametric, i.e., it does not require any assumption on the initial data distribution ϕ_0.

QT-EWMA models ϕ_0 by a QuantTree histogram [6] built on the training set TR. The histogram is defined by $Q = \{(S_k, \pi_k)\}_{k=1}^K$, where S_k are the histogram bins, π_k the corresponding target bin probabilities, and K is the number of bins to be set a priori. Then, QT-EWMA monitors the proportion of samples falling in each bin of the histogram by K EWMA statistics [15]:

$$Z_{k,t} = (1-\lambda)Z_{k,t-1} + \lambda b_{k,t}, \quad Z_{k,0} = \pi_k, \tag{4}$$

where the binary statistics $b_{k,t} = \mathbb{1}(x_t \in S_k)$ indicate the bin of the histogram in which x_t falls, for $k \in \{1, \dots, K\}$. Then, the statistic T_t is defined by

$$T_t = \sum_{k=1}^{K} \frac{(Z_{k,t} - \pi_k)^2}{\pi_k}. \tag{5}$$

Each statistic $Z_{k,t}$ is an incremental measure of the proportion of samples acquired until time t falling in each bin S_k. The statistic T_t assesses how much the $Z_{k,t}$ deviate from the target bin probabilities π_k, thus it is similar to the Pearson statistic [20]. The main advantage of this solution is that the distribution of T_t (5), like any other statistic based exclusively on the number of points falling in the bins of a QuantTree histogram, is independent from ϕ_0, as demonstrated in [6]. This property enables nonparametric monitoring, and allows to define thresholds $\{h_t\}$ for QT-EWMA such that:

$$\mathbb{P}_{\phi_0}(T_t > h_t \mid T_k \leq h_k \forall k < t) = \alpha, \tag{6}$$

which have been shown to guarantee a desired ARL_0 when $\alpha = 1/\text{ARL}_0$ [28]. These thresholds are computed by Monte Carlo simulations that are described in detail in [5].

B. Class Distribution Monitoring

QT-EWMA, like other concept-drift detectors that monitor the data distribution, is designed to operate in unsupervised settings, and therefore ignores the labels y_t, which we assume to be regularly provided during monitoring. As a result, concept drifts affecting only a subset of classes can be hard to detect following this approach.

To exploit class labels, we propose Class Distribution Monitoring (CDM), which is illustrated in Algorithm 1. First, we divide the training set TR into M subsets TR^m and use these to construct M QuantTree histograms $Q^m = \{(S_k^m, \pi_k)\}$ [6], corresponding to the classes $m \in \{1, \dots, M\}$ (lines 4–5). When an input sample x_t is provided with its label y_t, we find the histogram bin such that $x_t \in S_k^m$ in the QuantTree Q^m corresponding to its label $m = y_t$ (line 11). Then, we compute the QT-EWMA statistic $T_{t_m}^m$ (5) (lines 12–13), where t_m is the number of samples of class m observed until time t. We report a concept drift as the first time t when $T_{t_m}^m > h_{t_m}$, where h_{t_m} is the QT-EWMA threshold defined by (6) (lines 14–18). We remark that, contrarily to the other concept-drift detectors, our algorithm returns, on top of the detection time t^*, the class m^* that triggered the detection (line 21).

Algorithm 1 Class Distribution Monitoring (CDM)

```
Input: datastream $\{(x_t, y_t)\}_t$, target probabilities $\{\pi_k\}_{k=1}^K$,
    thresholds $\{h_t\}_t$, $TR = \{(x, y)\}$
Output: detection flag ChangeDetected, detection time
    $t^*$, drifted class $m^*$
 1: // Configuration:
 2: ChangeDetected ← False, $t^* \leftarrow \infty, m^* \leftarrow 0$
 3: for $m = 1, \dots, M$ do
 4:     $TR^m \leftarrow \{x : (x, y) \in TR, y = m\}$
 5:     build QuantTree $Q^m = \{(S_k^m, \pi_k)\}$ [6] from $TR^m$
 6:     initialize $t_m \leftarrow 0$, $Z_{k,0}^m \leftarrow \pi_k, k \in \{1, \dots, K\}$
 7: end for
 8: // Monitoring:
 9: for $t = 1, \dots$ do
10:     if the label $y_t$ is provided then
11:         $m \leftarrow y_t,\ t_m \leftarrow t_m + 1,\ b_{k,t_m} \leftarrow \mathbb{1}(x_t \in S_k^m)$
12:         compute $Z_{k,t_m}^m$ (4) for $k \in \{1, \dots, K\}$
13:         compute QT-EWMA statistic $T_{t_m}^m$ (5)
14:         if $T_{t_m}^m > h_{t_m}$ then
15:             ChangeDetected ← True
16:             $t^* \leftarrow t,\ m^* \leftarrow m$
17:             break
18:         end if
19:     end if
20: end for
21: return ChangeDetected, $t^*, m^*$
```

C. Properties of CDM

Here we illustrate the most important properties of CDM, in particular the control of the ARL_0, and analyze its computational complexity.

Online and Nonparametric Monitoring. Consistently with the notation introduced in Section IV-A, we can see CDM as an online change-detection test with statistic $\tilde{T}_t$ defined as

$$\tilde{T}_t = T_{t_m}^m, \quad m = y_t, \tag{7}$$

and thresholds $\tilde{h}_t = h_{t_m}$. The online nature of CDM is evident from Algorithm 1, where the datastream is processed one sample (x_t, y_t) at a time. The nonparametric nature of CDM derives from the fact that the distribution of the test statistic $\tilde{T}_t$, like any other statistic based on QuantTree, does not depend on the initial distribution ϕ_0, as shown in [6].

Control of the ARL_0. CDM inherits from QT-EWMA the control of false alarms by maintaining a target ARL_0. In particular, we demonstrate that, since (6) holds for QT-EWMA, CDM yields the same ARL_0 as the QT-EWMA monitoring each class-conditional distribution.

Proposition 1. *Let $\tilde{T}$ be the test statistic of CDM defined in (7), and let $\{h_t\}$ be the QT-EWMA thresholds yielding the target ARL$_0$. Then, the change-detection test defined by $\tilde{T}$ yields the same ARL$_0$.*

Proof. To prove the Proposition, we need to show that (6) holds for $\tilde{T}$. By the definition of $\tilde{T}$ in (7) and the law of total probability we have that

$$\begin{aligned}\mathbb{P}_{\phi_0}(\tilde{T}_t > \tilde{h}_t \mid \tilde{T}_k \leq \tilde{h}_k \ \forall k < t) = \\ = \sum_{m=1}^{M} \mathbb{P}_{\phi_0}(T^m_{t_m} > h_{t_m} \mid T^m_k \leq h_k \ \forall k < t_m, y_t = m) \cdot \\ \cdot \mathbb{P}_{\phi_0}(y_t = m \mid T^m_k \leq h_k \ \forall k < t_m). \quad (8)\end{aligned}$$

Since all the samples (x_t, y_t) in the datastream are assumed to be independent, the label y_t associated with x_t is independent from the values of the statistic T^m_k for $k < t_m$, so we can drop the conditioning in the second factor of the second term of (8). Moreover, the probability under ϕ_0 in the first factor is conditioned on the event "$y_t = m$", so it coincides with the probability under the class-conditional distribution ϕ^m_0 defined in (1). Hence, (8) becomes:

$$\begin{aligned}\sum_{m=1}^{M} \mathbb{P}_{\phi^m_0}(T^m_{t_m} > h_{t_m} \mid T^m_k \leq h_k \ \forall k < t_m) \mathbb{P}_{\phi_0}(y_t = m) = \\ = \sum_{m=1}^{M} \alpha \cdot \mathbb{P}_{\phi_0}(y_t = m) = \alpha, \quad (9)\end{aligned}$$

where the penultimate equality derives from the fact that (6) holds for the QT-EWMA test statistics T^m monitoring each class-conditional distribution. The last equality in (9) derives from the assumption that, under ϕ_0, each sample x_t has a label $y_t = m \in \{1, \ldots, M\}$, so the events $\{y_t = m\}_{m=1}^M$ represent a partition of the probability space, thus their probabilities sum to 1. The fact that (8) = (9) proves that (6) holds for $\tilde{T}$, showing that CDM yields ARL$_0 = 1/\alpha$ [28]. □

We remark that Proposition 1 holds for any CDM defined by an online change-detection algorithm that can be configured to yield the desired ARL$_0$ by setting a constant false alarm probability over time as in (6). This means that, in principle, we can define CDM using other change-detection tests. However, to the best of our knowledge, QT-EWMA is the only nonparametric and online change-detection test for multivariate datastreams whose thresholds can be set to satisfy (6), which is not guaranteed by other methods controlling the ARL$_0$, such as Scan-B [26] and ECDD [14].

Computational Complexity. Similarly to QT-EWMA [5], CDM is extremely efficient in both computational and memory overhead. It places each sample x_t in its bin in the QuantTree histogram Q^m corresponding to its label $m = y_t$, resulting in $\mathcal{O}(K)$ operations [6]. Then, CDM updates the corresponding statistics Z^m_{k,t_m} (4) for $k \in \{1, \ldots, K\}$, thus requiring to store in memory $M \cdot K$ values, namely K statistics per class.

V. Experiments

Here we illustrate our experiments, which we designed to demonstrate that CDM outperforms mainstream concept-drift detection methods that monitor either the error rate of a classifier or the overall data distribution. First, we present the real-world and synthetic datasets on which we test our solution (Section V-A), then we formally define the figures of merit we use (Section V-B) and the reference methods from the literature (Section V-C). Finally, we present and discuss our experiments and their results (Sections V-D,V-E).

A. Considered Datasets

Real-world data. The INSECTS dataset [29] is a well-known benchmark for classification and concept-drift detection. It contains feature vectors ($d = 33$) extracted from sensor measurements describing the wing-beat frequency of six (annotated) species of flying insects. The dataset contains six concepts, each representing measurements acquired at a different temperature, which influences the flying behavior of the insects. This allows us to introduce realistic concept drifts by sampling the datastream from different concepts before and after the change point τ. In our experiments, the stationary condition ϕ_0 is characterized by the class-conditional distributions $\{\phi^m_0\}_{m=1}^M$ describing the features of $M = 4$ different insect species from each of the six concepts. We consider multiple drifts $\phi_0 \to \phi_1$ that consist in a temperature change affecting one or more classes, namely $\phi^m_0 \to \phi^m_1 \neq \phi^m_0$. In these settings, for each stationary distribution ϕ_0, the change $\phi_0 \to \phi_1$ is defined among 5 potential temperature changes affecting one of $2^M - 1 = 15$ different subsets of the M classes, for a total of 75 distribution changes per initial concept ϕ_0. In our experiments we consider training sets containing 256 instances of each class, sampled without replacement from each class-conditional distribution ϕ^m_0.

Synthetic data. To interpret the results obtained on real-world data, we synthetically generate various distribution changes $\phi_0 \to \phi_1$ and assess their impact on the classification error. In particular, we define the stationary distribution ϕ_0 as a mixture of $M = 2$ Gaussians (one per class) $\phi^1_0 = \mathcal{N}(\mu^1_0, I)$ and $\phi^2_0 = \mathcal{N}(\mu^2_0, I)$ in $\mathbb{R}^2$, where I denotes the identity matrix, $\mu^1_0 = [0, 0]^T$, and $\mu^2_0 = [\delta, 0]^T$ for some $\delta > 0$. Post-change distribution ϕ_1 is defined by shifting $\phi^2_0 \to \phi^2_1 = \mathcal{N}(\mu^2_1, I)$, while keeping ϕ^1_0 fixed. Changes are thus regulated by μ^2_1, which we move over a grid around μ^1_0 (see Figure 1). Also in this case, we consider training sets containing 256 samples drawn from each ϕ^m_0.

This setup was designed to assess when CDM is a better option than ECDD. The classification error varies when μ^2_1 moves along the horizontal direction, which is the line connecting μ^1_0 and μ^2_0: these changes can be promptly detected by ECDD when they increase the error rate. In contrast, changes translating μ^2_1 vertically (thus orthogonal to the line joining μ^1_0 and μ^2_0), do not change the error rate but only the input distribution. These changes cannot be detected by ECDD, but are perceivable by CDM, whose performance only depends by the change magnitude. Here we measure the

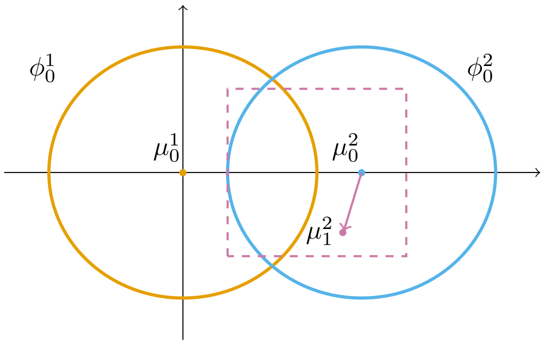


Fig. 1: Illustration of Gaussian class-conditional distributions generating synthetic data. The distributions are represented by the mean and the 3σ ellipsoid. We consider changes $\phi_0 \rightarrow \phi_1$ defined by translating the mean of ϕ_0^2 inside the dashed rectangle, as in this example.

change magnitude by the symmetric Kullback-Leibler distance $sKL(\phi_0^2, \phi_1^2)$ [30], which in this case is equal to $\frac{1}{2}\|\mu_1^2 - \mu_0^2\|_2$.

B. Figures of Merit

We consider two common figures of merit in the change-detection literature. First, we assess the control of false alarms by computing the *empirical* ARL_0, i.e., the average detection time in datastreams distributed as ϕ_0. Thanks to Proposition 1, we expect the empirical ARL_0 of CDM to approach the target ARL_0 set before monitoring. Then, we measure the detection power by the *average detection delay* (or ARL_1), namely the average difference between the detection time t^* and the actual change point τ. The detection delay is computed considering only datastreams where no false alarms were reported before the change, thus $t^* > \tau$.

C. Considered Methods

To ensure a fair comparison, we only consider methods that *i)* are nonparametric and *ii)* control the false alarms by setting a target ARL_0 before monitoring. In particular, we consider ECDD [14], which monitors the error rate of a classifier, Scan-B [26] and QT-EWMA [5], which are nonparametric and online change-detection tests monitoring the data distribution.

ECDD [14] employs an EWMA control chart [15] to monitor the sequence $\{e_t\}$ (3) defined by the errors of a classifier $\mathcal{K}$. In particular, ECDD computes a statistic

$$U_t = (1-r)U_{t-1} + re_t, \quad U_0 = \hat{p}_{0,0}, \tag{10}$$

where $\hat{p}_{0,t}$ indicates the average error rate of $\mathcal{K}$ up to time t, and r is the EWMA parameter, which we set to $r = 0.2$ as in [14]. A concept drift is detected when $U_t > \hat{p}_{0,t} + L\sigma_t$, where σ_t is the estimated standard deviation of U_t:

$$\sigma_t = \sqrt{\hat{p}_{0,t}(1-\hat{p}_{0,t})\frac{r}{2-r}(1-(1-r)^{2t})}. \tag{11}$$

Since U_t is an incremental estimate of the error rate of $\mathcal{K}$, which gives exponentially larger weights to the latest elements of e_t compared to older elements, and since a one-sided decision rule is applied, ECDD can only detect drifts that increase the classification error. The control limit L can be tuned to yield a target ARL_0, and [14] provides polynomial approximations to compute L for different values of the target ARL_0 as a function of $\hat{p}_{0,t}$.

TABLE I: Computational complexity for processing a new sample (x_t, y_t) and memory requirement of CDM and the other considered methods. The computational complexity of ECDD [14] is that of the classifier $\mathcal{K}$, indicated by $\mathcal{O}(\mathcal{K})$.

Method	ECDD [14]	Scan-B [26]	QT-EWMA [5]	CDM (ours)
Complexity	$\mathcal{O}(\mathcal{K})$	$\mathcal{O}(nBd)$	$\mathcal{O}(MK)$	$\mathcal{O}(K)$
Memory	2	$(n+1)Bd$	MK	MK

In our experiments on INSECTS, we train $\mathcal{K}$ as a k-Nearest Neighbors (k-NN) classifier ($k = 9$), and we never update it during monitoring. On the synthetic dataset we employ a Linear Discriminant Analysis (LDA) classifier, which is faster and yields excellent performance over Gaussian classes.

In terms of computational complexity, computing and updating U_t (10) and $\hat{p}_{0,t}$ are extremely cheap operations, which require storing in memory only 2 scalar values, namely U_{t-1} and $\hat{p}_{0,t-1}$. The computational complexity of ECDD therefore depends on that of the classifier, which we indicate by $\mathcal{O}(\mathcal{K})$, which has to be applied on each x_t.

Scan-B [26] is a nonparametric change-detection algorithm that monitors the input distribution by computing at each time t, the average Maximum Mean Discrepancy (MMD) [25] between a sliding window of a fixed size B and n reference windows of the same size sampled from the training set TR. This requires updating n Gram matrices for each sample x_t by computing B times the MMD statistic, resulting in $\mathcal{O}(nBd)$ operations [27]. Therefore, Scan-B stores in memory n reference windows of B d-dimensional inputs, on top of the current window, resulting in $(n+1)Bd$ memory footprint. Thresholds are set by analyzing the asymptotic behavior of ARL_0 when the threshold tends to infinity, while in CDM and QT-EWMA the thresholds are defined by (6), providing more accurate control of the ARL_0 [5]. As in [26], we set the window size $B = 50$ and $n = 5$.

QT-EWMA [5], is a nonparametric change-detection algorithm which we have described in Section IV-A. To enable a fair comparison, since CDM leverages M instances of QT-EWMA each one based on a QuantTree histogram with $K = 16$ bins, we set the number of bins of QT-EWMA to MK and, according to [5], we set $\lambda = 0.03$ in (4). As shown in [5], QT-EWMA is very efficient since it performs $\mathcal{O}(MK)$ operations to place each sample x_t in the corresponding bin of the QuantTree histogram [6], and requires storing only the MK scalar values of the statistic $Z_{k,t-1}$ (4) to be updated at time t.

In Table I we compare the computational complexity and memory requirements of CDM (discussed in Section IV-C) to those of the other considered methods. This analysis shows that CDM and QT-EWMA are extremely efficient from both

TABLE II: Empirical ARL_0 of the considered methods on the 6 concepts of the INSECTS dataset [29].

Concept	Method (target ARL_0)			
	ECDD [14]	Scan-B [26]	QT-EWMA [5]	CDM (ours)
	(400)	(300)	(375)	(375)
A	376.51	382.08	379.10	375.44
B	371.07	384.56	361.78	374.47
C	373.16	381.65	371.66	365.32
D	374.14	387.17	367.18	369.94
E	371.82	376.28	375.10	374.64
F	377.67	374.22	375.58	371.87

the computational and memory points of view. In contrast, Scan-B performs more operations and stores more data, and these requirements increase with the data dimension d, contrarily to CDM and QT-EWMA. ECDD has negligible memory requirements, but its computational complexity depends on the classifier $\mathcal{K}$, which is applied to each sample x_t to form $\{e_t\}$.

D. Concept Drift Detection on INSECTS Data

In this Section, we discuss the empirical ARL_0 and the detection delay achieved on the INSECTS dataset by the considered models in the settings described in Section V-A.

$\mathbf{ARL_0}$. We compute the empirical ARL_0 of the considered methods on the six concepts of the INSECTS dataset [29], which we denote by A, B, C, D, E, F. We consider each concept as a stationary distribution ϕ_0, and we sample without replacement 5000 training sets and 5000 datastreams of length 8000 from each ϕ_0. Then, we configure the considered methods on the training sets, and compute the empirical ARL_0 as the average detection time over these stationary datastreams.

We report the results of this experiment in Table II, which shows that ECDD fails at accurately controlling the target $\mathrm{ARL}_0 = 400$. In contrast, the empirical ARL_0 of CDM and QT-EWMA approaches their target, which we set to $\mathrm{ARL}_0 = 375$ to match the empirical ARL_0 of ECDD. Similarly to ECDD, Scan-B does not accurately control the ARL_0, and this is consistent with the experiments in [5]. For this reason, we set the target $\mathrm{ARL}_0 = 300$ in Scan-B to yield approximately the same empirical ARL_0 as the other methods. Table II indicates that in these settings it is possible to fairly compare the detection delays of the considered methods, since these all yield approximately the same empirical ARL_0.

Detection delay. For each of the 450 changes $\phi_0 \to \phi_1$ (75 for each of the 6 initial concepts) described in Section V-A, we sample without replacement 1000 training sets and 1000 datastreams to be monitored. Each datastream is the concatenation of $\tau = 160$ points drawn from ϕ_0 and 7000 points drawn from ϕ_1. Table III reports the average detection delays of the considered methods depending on the drifted classes. As suggested in [31], we rank the considered methods according to their average detection delay obtained on each of the 450 changes (rank = 1 for the method with the lowest detection delay, etc.), and report their average rank. We also report the p-values of the Nemenyi [32] and Dunn [33] post-hoc tests, to assess whether the differences between the best-ranking method and the others are statistically significant.

TABLE III: Average detection delays on the 15 subsets of classes affected by change of the INSECTS dataset [29].

Drifted classes	ECDD [14]	Scan-B [26]	QT-EWMA [5]	CDM (ours)
1	207.98	212.53	267.73	**195.45**
2	245.85	162.58	195.44	**124.92**
3	264.27	224.99	278.57	**204.00**
4	224.91	235.87	265.96	**196.74**
1,2	198.17	131.71	174.80	**114.44**
1,3	172.62	169.87	223.50	**160.98**
1,4	165.77	163.63	221.66	**145.82**
2,3	163.66	126.56	167.55	**112.18**
2,4	176.53	119.41	154.95	**106.49**
3,4	210.04	169.88	218.90	**153.51**
1,2,3	139.29	115.01	152.91	**103.60**
1,2,4	148.03	103.24	141.09	**98.89**
1,3,4	144.81	134.83	183.41	**131.38**
2,3,4	132.36	**96.92**	136.90	98.57
1,2,3,4	122.38	**88.86**	128.04	91.44
Avg. rank	2.416	2.356	3.524	**1.704**
Nemenyi-p	$6.95 \cdot 10^{-6}$	$1.44 \cdot 10^{-1}$	$5.18 \cdot 10^{-17}$	–
Dunn-p	$2.43 \cdot 10^{-7}$	$2.01 \cdot 10^{-2}$	$6.40 \cdot 10^{-19}$	–

We observe that CDM turns out to be the best method in 13 out of the 15 considered changes, and the best in terms of average rank. The Nemenyi and Dunn tests show that the gap of CDM over ECDD and QT-EWMA is statistically significant (p-value < 0.05). The gap between CDM and Scan-B is less remarkable, but still significant according to the Dunn test.

As expected, all the methods tend to yield lower detection delays when the change affects more classes. In particular, the difference between the detection delays of CDM and QT-EWMA is larger when the change affects only one class rather than when it affects all of them, showing that monitoring the class-conditional distributions can indeed improve the detection performance in these cases. This effect is even more apparent in the comparison between CDM and Scan-B.

Most remarkably, CDM substantially outperforms ECDD in terms of average detection delay in all the considered settings. This is due to the fact that ECDD can only detect concept drifts that increase the classification error, while the considered drifts in the INSECTS dataset might have little impact on the error rate of a classifier. To further analyze the relation between detection delay and classification error, we plot in Figure 2 the average detection delays of CDM and ECDD against the difference between the classification error after (p_1) and before the change (p_0). Each plot reports the results obtained on the 75 drifts we consider for each initial concept A, B, C, D, E, F. We highlight the relation between $p_1 - p_0$ and the detection delay by plotting the moving average (weighted by a Gaussian kernel) of the detection delay as a function of $p_1 - p_0$. These results qualitatively show that the performance of ECDD only depends on $p_1 - p_0$, which is often small and sometimes even negative. In contrast, CDM can detect any change in the class-conditional distributions, thus yielding a lower detection delay in most cases.

E. Concept Drift Detection on Gaussian Data

Concept drifts might not always heavily impact the classification performance, as we have shown in Figure 2 on the INSECTS dataset. Here we further analyze the fundamental difference between monitoring the classification error (ECDD)

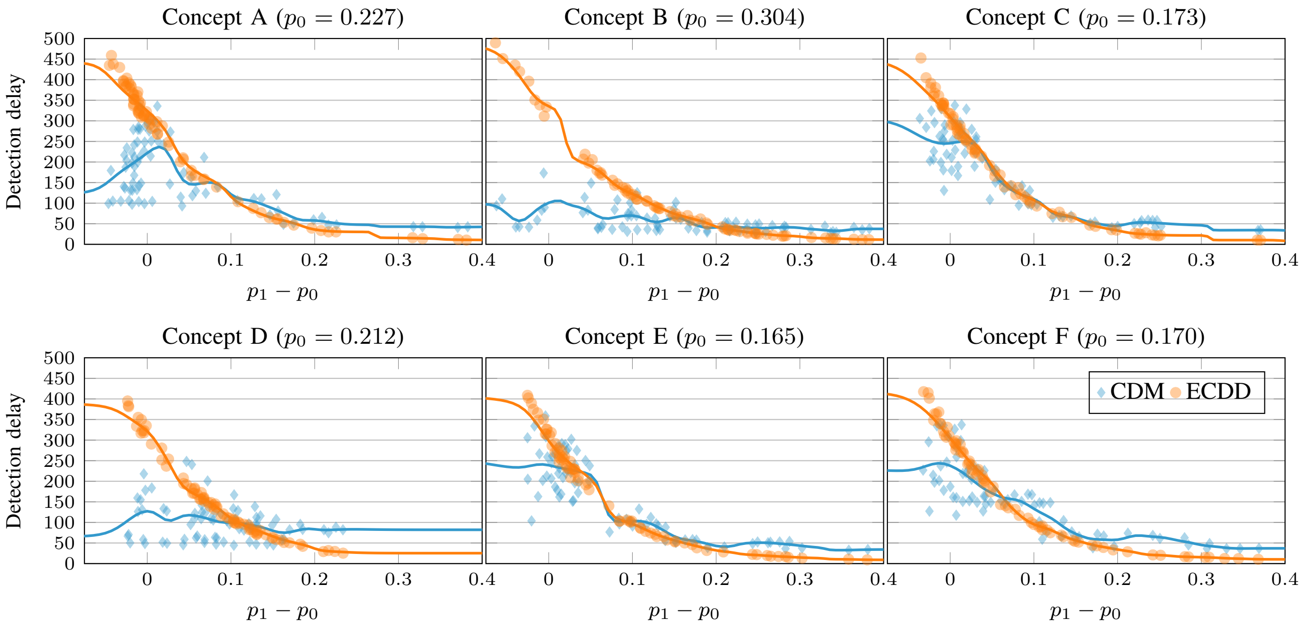


Fig. 2: Detection delay achieved by ECDD and CDM on the INSECTS dataset [29] for each of the 6 stationary concepts, plotted against the difference $p_1 - p_0$, where p_0, p_1 are the error rates of $\mathcal{K}$ before and after the drift. Each dot is the average of 1000 realizations of the same change $\phi_0 \to \phi_1$ i.e., thus with the same affected classes.

and the input distribution (CDM), by considering the synthetic scenario described in Section V-A, where we can control both $p_1 - p_0$ and the change magnitude $sKL(\phi_0^2, \phi_1^2)$. We configure ECDD and CDM to maintain the same ARL_0 as in Section V-D.

The results of this experiment are illustrated in Figure 3. Figures 3(a)-(b) report the detection delays respectively achieved by ECDD and CDM as a heatmap. The color coded value at a coordinate $\overline{\mu} \in \mathbb{R}^2$ represents the detection delay achieved by the model when $\mu_1^2 = \overline{\mu}$, averaged over 5000 experiments. Moreover, in the same figures we report, respectively, $p_1 - p_0$ and $sKL(\phi_0^2, \phi_1^2)$ as contour plots.

As expected, ECDD cannot detect virtual drifts, as can be seen by the large detection delays on the right side of Figure 3(a), but it achieves excellent detection performance when the translation reduces the distance between the two class-conditional distributions, increasing the classification error ($p_1 - p_0 > 0$). In contrast, the detection delay of our CDM only depends on the distance $sKL(\phi_0^2, \phi_1^2)$, as it can be appreciated in Figure 3(b), where the level curves of the detection delays are circular and follow $\frac{1}{2}\|\mu_1^2 - \mu_0^2\|_2$. Figure 3(c) reports the difference between the detection delays of ECDD and CDM. ECDD outperforms CDM when μ_1^2 falls inside a relatively small triangular portion of $\mathbb{R}^2$, corresponding to drifts that significantly increase the error rate while keeping the distance between ϕ_0^2 and ϕ_1^2 low. However, the difference is substantial only in a small region close to μ_0^2, where the change is nearly negligible and the performance of both methods is rather poor. CDM yields lower detection delays than ECDD in all the other cases, and the performance difference is quite large, especially when the drift reduces the classification error.

VI. Conclusions and Future Work

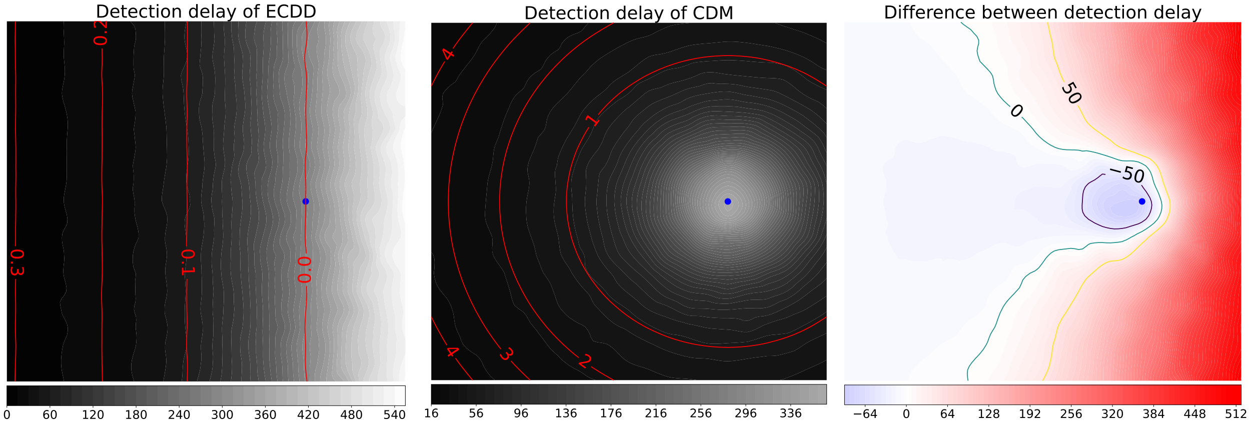
We have introduced CDM, a novel concept-drift detection method that monitors the class-conditional distributions using QT-EWMA [5]. Our experiments on real-world datastreams and synthetic data show that our solution can effectively detect virtual drifts that are ignored by methods that monitor the error rate of a classifier. In many circumstances, CDM yields lower detection delays than methods that monitor the overall data distribution, especially when the drift affects only a subset of classes. Moreover, our CDM is built upon solid theoretical guarantees on false alarms, enabling to set the ARL_0 before monitoring. Another important advantage of CDM compared to other concept-drift detection methods is that our solution returns information on which class triggered the detection, and this can be crucial for adaptation and diagnostics in general.

Future work will extend CDM by applying other detectors controlling the ARL_0 in parametric settings or in combination with QT-EWMA, to improve the detection performance when parametric assumptions can be made on some class-conditional distributions. We will also investigate the application of CDM when class labels are not available during monitoring, using the predictions of a classifier instead.

References

[1] M. Bahri, A. Bifet, J. Gama, H. M. Gomes, and S. Maniu, “Data stream analysis: Foundations, major tasks and tools,” *WIREs: Data Mining and Knowledge Discovery*, vol. 11, no. 3, p. e1405, 2021.

[2] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, “Learning under concept drift: A review,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2018.



(a) (b) (c)

Fig. 3: Results of the experiment on synthetic data. The blue dot represents the pre-change mean μ_0^2. (a,b) report, at each coordinate $\overline{\mu}$, the average detection delay achieved by ECDD and CDM, respectively, when $\mu_0^2 \to \overline{\mu}$. In (a) the contour lines indicate the difference in classification error $p_1 - p_0$ before and after the change, and in (b) the change magnitude $sKL(\phi_0^2, \phi_1^2)$. (c) report the difference between the detection delay achieved over synthetic data by ECDD and CDM, with contour lines.

[3] C. Geng, S.-J. Huang, and S. Chen, "Recent advances in open set recognition: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3614–3631, 2021.

[4] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application*. Prentice Hall Englewood Cliffs, 1993, vol. 104.

[5] L. Frittoli, D. Carrera, and G. Boracchi, "Change detection in multivariate datastreams controlling false alarms," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 421–436.

[6] G. Boracchi, D. Carrera, C. Cervellera, and D. Macciò, "QuantTree: histograms for change detection in multivariate data streams," in *International Conference on Machine Learning*. PMLR, 2018, pp. 639–648.

[7] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Brazilian Symposium on Artificial Intelligence*. Springer, 2004, pp. 286–295.

[8] I. Frias-Blanco, J. del Campo-Ávila, G. Ramos-Jimenez, R. Morales-Bueno, A. Ortiz-Diaz, and Y. Caballero-Mota, "Online and non-parametric drift detection methods based on Hoeffding's bounds," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 810–823, 2014.

[9] R. S. M. de Barros, J. I. G. Hidalgo, and D. R. de Lima Cabral, "Wilcoxon rank sum test drift detector," *Neurocomputing*, vol. 275, pp. 1954–1963, 2018.

[10] G. J. Ross, D. K. Tasoulis, and N. M. Adams, "Nonparametric monitoring of data streams for changes in location and scale," *Technometrics*, vol. 53, no. 4, pp. 379–389, 2011.

[11] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.

[12] S. Wang and L. L. Minku, "AUC estimation and concept drift detection for imbalanced data streams with multiple classes," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[13] Ł. Korycki and B. Krawczyk, "Concept drift detection from multi-class imbalanced data streams," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 1068–1079.

[14] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191–198, 2012.

[15] S. Roberts, "Control chart tests based on geometric moving averages," *Technometrics*, vol. 1, no. 3, pp. 239–250, 1959.

[16] V. Guralnik and J. Srivastava, "Event detection from time series data," in *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999, pp. 33–42.

[17] L. I. Kuncheva, "Change detection in streaming multivariate data using likelihood detectors," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, pp. 1175–1180, 2011.

[18] C. Alippi, G. Boracchi, D. Carrera, and M. Roveri, "Change detection in multivariate datastreams: Likelihood and detectability loss," *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 2, pp. 1368–1374, 2016.

[19] T. S. Lau, W. P. Tay, and V. V. Veeravalli, "A binning approach to quickest change detection with unknown post-change distribution," *IEEE Transactions on Signal Processing*, vol. 67, no. 3, pp. 609–621, 2018.

[20] E. L. Lehmann and J. P. Romano, *Testing statistical hypotheses*. Springer, 2006.

[21] L. I. Kuncheva and W. J. Faithfull, "PCA feature extraction for change detection in multidimensional unlabeled data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 69–80, 2013.

[22] A. A. Qahtan, B. Alharbi, S. Wang, and X. Zhang, "A PCA-based change detection framework for multidimensional data streams," in *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 935–944.

[23] S.-S. Ho, "A Martingale framework for concept change detection in time-varying data streams," in *Proceedings of the 22nd International Conference on Machine Learning*, 2005, pp. 321–327.

[24] N. Mozafari, S. Hashemi, and A. Hamzeh, "A precise statistical approach for concept change detection in unlabeled data streams," *Computers & Mathematics with Applications*, vol. 62, no. 4, pp. 1655–1669, 2011.

[25] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[26] S. Li, Y. Xie, H. Dai, and L. Song, "M-statistic for kernel change-point detection," *Advances in Neural Information Processing Systems*, vol. 28, pp. 3366–3374, 2015.

[27] N. Keriven, D. Garreau, and I. Poli, "NEWMA: a new method for scalable model-free online change-point detection," *IEEE Transactions on Signal Processing*, vol. 68, pp. 3515–3528, 2020.

[28] T. M. Margavio, M. D. Conerly, W. H. Woodall, and L. G. Drake, "Alarm rates for quality control charts," *Statistics & Probability Letters*, vol. 24, no. 3, pp. 219–224, 1995.

[29] V. Souza, D. M. dos Reis, A. G. Maletzke, and G. E. Batista, "Challenges in benchmarking stream learning algorithms with real-world data," *Data Mining and Knowledge Discovery*, vol. 34, no. 6, pp. 1805–1858, 2020.

[30] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[31] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[32] P. B. Nemenyi, *Distribution-free multiple comparisons*. PhD Thesis, Princeton University, 1963.

[33] O. J. Dunn, "Multiple comparisons among means," *Journal of the American Statistical Association*, vol. 56, no. 293, pp. 52–64, 1961.