Data Mining and Knowledge Discovery (2025) 39:85
https://doi.org/10.1007/s10618-025-01155-x

RMIDDM: an unsupervised and interpretable concept drift detection method for data streams

Ruivaldo Neto[1] · Brenno Alencar[1] · Heitor Murilo Gomes[2] · Albert Bifet[3] · João Gama[4] · Guilherme Cassales[3] · Ricardo Rios[1]

Received: 2 August 2024 / Accepted: 4 August 2025 / Published online: 18 September 2025

Abstract
Traditional machine learning techniques assume that data is drawn from a stationary source. This assumption is challenged in contexts with data streams for presenting constant and potentially infinite sequences whose distribution is prone to change over time. Based on these settings, detecting changes (a.k.a. concept drifts) is necessary to keep learning models up-to-date. Although state-of-the-art detection methods were designed to monitor the loss of predictive models, such monitoring falls short in many real-world scenarios where the true labels are not readily available. Therefore, there is increasing attention to unsupervised concept drift detection methods as approached in this paper. In this work, we present an unsupervised and interpretable method based on Radial Basis Function Networks (RBFN) and Markov Chains (MC), referred to as RMIDDM (Radial Markov Interpretable Drift Detection Method). In our method, RBF performs, in the intermediate layer, an activation process that implicitly produces groups of observations collected over time. Simultaneously, MC models the transitions between groups to support the detection of concept drifts, which happens when the active group changes and its probability exceeds a given threshold. A set of experiments with synthetic datasets and comparisons with state-of-the-art algorithms demonstrated that the proposed method can detect drifts at runtime in an efficient, interpretable, and independent way of labels, presenting competitive results and behavior. Additionally, to show its applicability in a real-world scenario, we analyzed new COVID-19 cases, deaths, and vaccinations to identify new waves as concept drifts and generate Markov models that allow understanding of their interaction.

Keywords Data stream · Concept drift · Radial basis function network · Markov chain

Brenno Alencar, Heitor Murilo Gomes, Albert Bifet, João Gama, Guilherme Cassales and Ricardo Rios have contributed equally to this work.

Extended author information available on the last page of the article

1 Introduction

The volume of data produced by computer systems has increased considerably over time, specially driven by recent technological advances, such as the emergence of mobile devices, the popularization of social networks, and the expansion of the Internet of Things (Cohen et al. 2009). Although Machine learning (ML) techniques have been successfully considered to extract useful information from such data (Mitchell 1997), their effectiveness is often limited in scenarios where data is produced as continuous and potentially infinite sequences, known as data streams (Zhou et al. 2015).

In such a scenario, the execution time and computational resource are restricted making impossible to collect all data before being modeled (Bifet and Kirkby 2009). In addition to these limitations, they also have to deal with changes in data distribution, usually called concept drifts (Gama 2010; Lu et al. 2018), that degrade the performance of underlying models. To prevent this situation, researchers proposed supervised concept drift detection methods based on loss monitoring (Gama et al. 2014). However, these methods require true labels to be readily available, which is not feasible in some real-world scenarios (Aggarwal 2006). In this case, unsupervised solutions support the detection based on the identification of examples that do not fit the known data structure (Spinosa et al. 2007). Usually, this analysis is based on three main approaches: clustering, outlier detection, or dissimilarity measures (Ryu et al. 2012). According to the literature (Gama et al. 2014), there are classical methods that can be applied in unsupervised scenarios, such as Cumulative Sum (CUSUM) and PageHinkley (PH) (Page 1954). However, their main limitations are the linear approach to detect drifts on traditional statistics calculated from observations as mean and variance.

To address these limitations, this work proposes a novel drift detection method based on Radial Basis Function Networks (RBFNs) and Markov Chains (MCs), named RMIDDM (Radial Markov Interpretable Drift Detection Method). In the proposed method, RBFNs are used to cluster observations that exhibit similar concepts or behaviors as new data is continuously collected. Complementarily, MCs are employed to model the probabilistic transitions between these concepts, enabling the identification of alerts and drifts. By integrating these components, RMIDDM enables runtime detection of changes in a computationally efficient, label-independent, and interpretable manner. Furthermore, experiments on both synthetic and real-world datasets demonstrate that RMIDDM achieves competitive performance and desirable interpretability. The primary contributions of this work can be summarized as follows:

- A model-based method without requiring observation windows;
- Non-linear strategy to find concept drifts;
- Label-independent method to model data streams and detect changes;
- Visual interpretation of changes based on graph transition.

The remainder of this paper is organized as follows: Sect. 2 introduces the essential background concepts. Section 3 reviews related work. Section 4 presents the main contribution of this study—the RMIDDM method. Sections 5 and 6 describe the experimental evaluations conducted on synthetic and real-world datasets, respectively. Finally, Sect. 7 discusses the results, draws conclusions, and outlines directions for future research.

2 Background and literature review

In this section, we briefly describe the theoretical background necessary to understand better our proposal.

2.1 Data streams and concept drift

A data stream can be defined as an uninterrupted and potentially infinite sequence of events (Gama et al. 2014; Aggarwal 2006; Read et al. 2020), which usually occurs at high frequencies and requires to be processed as the observation values are collected from the analyzed system. For these reasons, algorithms designed to deal with data streams might meet the following requirements (Bifet and Kirkby 2009): (i) data are potentially unlimited in size, thus making it unfeasible to store them in memory permanently; and (ii) the short time interval between the monitoring needs for online modeling, i.e., offline approaches tends to produce outdated models once the data distribution may change over time. The detection of such changes has been widely studied by concept drift detectors, whose main definition can be formalized based on Bayesian Decision Theory (Duda et al. 2000; Bifet and Kirkby 2009; Gama et al. 2014). According to the authors, let $X = \{x_1, x_2, \dots\}$ be a data stream such that $x_t \in \mathbb{R}^D$ is an instance in a *D*-space of attributes collected at the time instant *t* and $\forall x_t$, $\exists y_i$, where $Y = \{y_1, y_2, \dots, y_k\}$ is the set of classes. The optimal classifier to model $f : x_t \to y_i$ is determined from the *a priori* probabilities $p(y_i)$ and the probability density function conditioned to the classes $p(x_t|y_i)$. Therefore, a concept can be defined as a set *S* of *a priori* and conditional probabilities for the classes, as shown in Eq. 1:

$$S = \{(p(y_1), p(x_t|y_1)), (p(y_2), p(x_t|y_2)), ..., (p(y_k), p(x_t|y_k))\} \tag{1}$$

Moreover, the classification of an x_t instance based on the maximum a posteriori probability can be obtained by Eq. 2:

$$p(y_i|x_t) = \frac{p(y_i)p(x_t|y_i)}{p(x_t)} \tag{2}$$

Since $p(y_i)$ and $p(x_t|y_i)$ uniquely determine the joint distribution $p(x_t, y_i)$, concepts can be defined as the joint distribution $p(x_t, y_i)$ Gama et al. (2014). Thus, it is possible to affirm that there is a concept drift between t_0 e t_1 if:

$$\exists x_t : p_{t_0}(x_t, y_i) \neq p_{t_1}(x_t, y_i) \tag{3}$$

In summary, a dataset has legitimate expected results in t_0, however, in t_1, it presents different expected results, also legitimate (Kolter and Maloof 2007). Concept drifts can be categorized as virtual or real (Gama et al. 2014; Bifet and Kirkby 2009). Virtual changes affect the a priori probability of the classes, $p(y_i)$, and do not modify target concepts. On the other hand, real concept drifts arise from changes in the a posteriori probability, $p(y_i|x_t)$, and modify the expected results.

According to Gama et al. (2014), concept drifts can exhibit one of four main behaviors: (a) Abrupt, (b) Gradual, (c) Incremental, or (d) Recurrent. Figure 1 illustrates these distinct behaviors for two concepts: A and B, represented by blue and yellow circles, respectively. Generally, abrupt patterns (a) correspond to fast switches from one state to another. Incremental (b) and gradual (c) patterns, in turn, exhibit slow changes, noticeable only after a long observation period. Finally, recurrent patterns (d) are characterized by the re-occurrence of previous concepts after variable intervals. Although concept drift is well-defined and widely studied in the literature, most definitions rely on the availability of labeled instances. In contrast, this work focuses on scenarios where labels are unavailable, as discussed in the following section.

In practical terms, there are several tools that implement algorithms to ananlyze data streams and detect concept drifts. The most well-known is the framework MOA (Massive Online Analysis) [15][1] which contains the state-of-the-art algorithms combining machine learning and concept drift algorithms to model data streams (Bifet et al. 2010). Moreover, MOA also presents a set of standardized evaluation routines, thus enabling reproducible experiments and comparisons. The framework architecture is modular and new algorithms can be implemented by extending MOA's abstract classes. Experiments can use data from ARFF (Attribute-Relation File Format) files or created by one of the available synthetic data stream generators. By considering such advantages, we decided to use MOA to implement our work, create our synthetic datasets from its stream generators, and assess results along with the main available algorithms by using its evaluation tests.

2.2 Radial basis function networks

Radial Basis Function Networks (RBFNs) are multi-layered models designed to analyze complex patterns and address non-linearly separable problems through function approximation. Their key distinguishing feature lies in the activation mechanism,

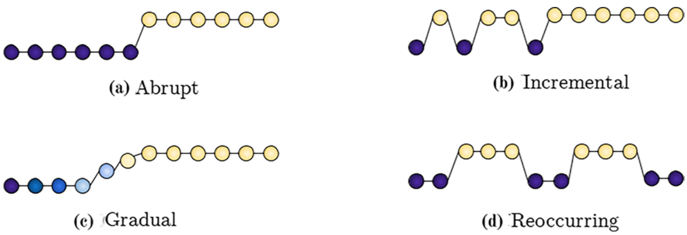


Fig. 1 Concept Drift main patterns (Gama et al. 2014)

[1] For more details, access https://moa.cms.waikato.ac.nz/,

which is driven by the distance between the input data and predefined centers (Du and Swamy 2019).

In summary, the architecture of a Radial Basis Function Network (RBFN) is structured around radial basis activation functions that serve as the neurons of the hidden layer. Among the various types of radial functions, the Gaussian function (Eq. 4) is the most widely adopted, often referred to as the kernel function in the context of RBFNs. This function is parameterized by a center $\mathbf{c_i}$ and a width (or spread) σ, which together define the receptive field of each neuron. The activation of a neuron is determined by the distance *r*, typically computed using a norm such as the Euclidean distance $\|\mathbf{v_i} - \mathbf{c_i}\|$, where $\mathbf{v_i}$ is the input vector. The parameter σ controls the radius of influence of the center, such that smaller values lead to more localized responses, while larger values allow broader activation. This localized and distance-based activation mechanism enables RBFNs to approximate complex nonlinear functions with high flexibility and interpretability.

$$\varphi(v_i) = e^{-(\sigma r)^2} \tag{4}$$

The detection method proposed in this paper employs a modified RBF network, consisting only of the input and hidden (intermediate) layers. This simplification is feasible because the primary role of the hidden layer is to perform a distance-based activation that naturally groups similar observations. Each neuron in the hidden layer functions as a cluster center, and an observation activates the center to which it is closest, based on a radial (Gaussian) function. As new data arrives, a change in the active center (i.e., a transition from one group to another) may indicate a shift in the underlying data distribution. Therefore, concept drifts are identified by monitoring these group transitions over time, without the need for a fully connected output layer or explicit supervision.

2.3 Markov chains

A Markov Chain is a stochastic process characterized by the memoryless property, meaning that the probability of transitioning to the next state depends solely on the current state and not on the sequence of preceding states. This fundamental property is formalized in Eq. 5, which expresses that the conditional probability of the next state is determined only by the immediately preceding state.

$$p\left(x_t = a_j | x_{t-1} = a_i, \ldots, x_0 = a_0\right) = p\left(x_t = a_j | x_{t-1} = a_i\right) = p_{ij} \tag{5}$$

In summary, a Markov process can occupy a finite set of states $a_1, a_2, \ldots, a_r$, where the probability of transitioning from state a_i to state a_j is denoted by p_{ij}. These transition probabilities depend solely on the current state *i* and the next state *j*, reflecting the memoryless nature of the process. A common way to visualize a Markov Chain is through a state diagram, where each node represents a state and directed edges between nodes are labeled with their corresponding transition probabilities.

The concept drift detection algorithm proposed in this paper uses a Markov Chain to model the cluster created by the RBF network. Clusters represent the states of the

Markovian model, and the transitions are modeled by changes (activations) between pairs of groups. The model reflects these changes by increasing the corresponding probability and proportionally decreasing other transitions.

3 Related work

Algorithms to detect concept drifts characterize and quantify changes in concepts by delimiting the moments or time intervals when they occur (Basseville and Nikiforov 1993). This section summarizes the most prominent algorithms described in the literature to deal with concept drift. These algorithms typically detect concept drift by monitoring changes in data characteristics or through indicators generated by underlying machine learning models. Among the vast literature on concept drift, we have selected algorithms based on two specific criteria: (i) widespread adoption in closely related works focusing on new applications and theoretical advancements; and (ii) validation by the scientific community through MOA.

Such algorithms focus on alerts yielded when the learner performance degrades and reaches previously-defined limits. As discussed in Gama et al. (2014), they are typically organized into three groups: *sequential*, *statistical*, and *window-based* methods.

Sequential methods continuously evaluate performance indicators of the underlying machine learning model, detecting drifts when indicators degrade and reach a pre-defined limit. *Cumulative Sum (CUSUM)*, *PageHinkley (PH)* (Page 1954), and *Geometric Moving Average (GMA)* (Roberts 2000) are well-known methods belonging to this group.

Statistical methods detect concept drifts through the analysis of statistical parameters such as mean and standard deviation associated to the model prediction results. In this category, the most well-known methods are: *Drift Detection Method (DDM)* (Gama et al. 2004), *Early Drift Detection Method (EDDM)* (Baena-García et al. 2006), *Exponentially Weighted Moving Average (EWMA)* (Ross et al. 2012), and *Reactive Drift Detection Method (RDDM)* (Barros et al. 2017).

Window-based methods use intervals (windows) to summarize previous and current data. A significant difference between the distributions between these windows suggests the occurrence of a concept drift. This difference is verified through statistical tests or mathematical inequalities as noticed in the *Adaptive Windowing (ADWIN)* (Bifet and Gavalda 2007), *SeqDrift* (Pears et al. 2014), and *HDDMA*/*HDDMW* (Blanco et al. 2015) methods.

In summary, the primary aim of the methods presented is to monitor learner performance by evaluating their prediction errors. However, some of these methods can be used to monitor concept drifts without using the label. In this case, the concept drift is detected by monitoring attributes from data streams. In Cerqueira et al. (2020) and Cerqueira et al. (2022), the authors introduce a label-independent drift detector called *Student–Teacher approach for Unsupervised Drift Detection (Studd)*. This detector leverages a student-teacher (ST) learning paradigm based on predictive models, in which the teacher is the primary one. The key component of this paradigm is the mimicking loss, which is determined by the discrepancy between the teacher's and student's predictions for the same instance. The loss of the student model serves as a proxy for the behavior of the primary model.

The authors in Žliobaite (2010) propose the use of hypothesis tests to compare two samples using a univariate statistical test, such as the Two-sample Kolmogorov-Smirnov test, the Wilcoxon rank-sum test, and the Two-sample t-test. The Two-Sample Kolmogorov-Smirnov Test determines whether two samples originate from the same continuous distribution, contrasting with the alternative hypothesis that they come from different distributions. The Ranksum Wilcoxon Rank Sum Test evaluates whether two independent samples are derived from identical continuous distributions with equal medians. The Two-Sample t-test assesses whether two independent samples are drawn from normal distributions with unknown but equal (or optionally unequal) variances and the same mean.

In Faithfull et al. (2019), the authors introduce a label-independent ensemble strategy in which each ensemble member monitors a distinct feature of the input space. The approach is classified as a multivariate detector for combining decisions in a straightforward voting scheme and a variable threshold. In this sense, we highlight that concept drift detectors can also be used to analyze univariate and multivariate data. Detectors for univariate data streams monitor a single observation over a time period. On the other hand, multivariate detectors monitor a combination of several attributes collected over time.

We compare our concept drift detection method, detailed in the following section, with the following baselines: CUSUM, DDM, EDDM, EWMA, HDDMA, PageHinkley, SeqDrift1, Studd, Kolmogorov-Smirnov test, Wilcoxon rank-sum test, T-test, and Multivariate detector.

4 Proposed method: RMIDDM

Modelling evolving graph-structured data and detecting changes is a very important tool for understanding the dynamics of the represented domain and identifying changes in data streams in an unsupervised manner (Loglisci et al. 2024). As depicted in Fig. 2, our method relies on an architecture constructed on top of two main components: Radial Basis Function Networks (RBFN) and Markov Chains (MC).

The process starts, in Step 1, with the acquisition of a new data stream sample. In Step 2, the data is sent to an RBFN, which assigns it to an existing group (containing previous samples with similar characteristics), or, alternatively, creates a new group representing a new concept. A Markov Chain is, then, built in Step 3 to represent the RBFN output, by modeling the transition probability between groups as new data arises. Finally, in Step 4, the transition elements and their associated probabilities are analyzed to determine whether the transition reflects expected behavior, a potentially imminent concept drift (alert zone), or a confirmed concept drift. This classification constitutes the final output of our architecture, as illustrated in Step 5.

All steps used in our method are detailed in Algorithm 1. When a sample x_t is received at time *t*, it is processed by the RBF network activation process (Lines 5–14), which clusters the events into groups that represent the concepts of the system. This process computes the activation value φ of the sample for each group, using a non-linear Gaussian function (Line 6), previously shown in Eq. 4. In terms of computational complexity, the activation function has linear complexity with respect to the data dimensionality *D*. Considering that the dimension is typically is limited (univariate, in our experiments), as in most time series analyses, the activation step can be reasonably approximated as having constant complexity *O*(1). The search for an active group is based on assessing the new data and existing groups by using the activation function. Therefore, the complexity of this process is linear defined as $O(|\mathrm{C}|)$, such that $|\mathrm{C}| \ll |X|$. The group with the highest activation value becomes the current active group $\hat{c}_t$. If no group reaches the minimum activation value (λ), the current sample becomes the new active group (Lines 10–12). In terms of computational complexity, creating a new center involves simply appending the new observation to the list of centers. Therefore, this operation runs in constant time *O*(1).

Next, the algorithm verifies whether a group has been previously activated ($\hat{c}_{t-1}$). If not, it sets the $\hat{c}_{t-1}$ to the current one, $\hat{c}_t$. Then, the transition between the *concept group* and the *active group* is updated in the Markov Chain (Line 15). This process increases the probability of the transition by a factor α and proportionally reduces the probabilities of the other transitions. The probability update is performed only for transitions originating from the previously active group. Consequently, the computational complexity of this step is $O(|\mathrm{N}_u|)$, where N_u denotes the set of neighbors of node *u*, including the self-loop.

In its final step, if the current group $\hat{c}_t$ is not the same as the last activated concept group $\hat{c}_{t-1}$, the algorithm verifies whether the resulting probability is lower than a threshold δ (Line 16). If so, the sample is classified as an alert. Otherwise, the sample is classified as a concept drift, and the last activated group is updated to $\hat{c}_t$, for the future samples to be received. In concept drift area, one of the big challenges is not to consider outlier or noise as true drift (Gama et al. 2014), which can strongly affect detectors by abruptly and significantly altering observation patterns. However, in our algorithm, all changes are interpreted through the probability distribution, $p(\hat{c}_t|x_t)$. As a result, observations that differ significantly or that seldom resemble a previous concept are not immediately treated as concept drifts, due to their low likelihood under the current distribution. If the previous and current groups are the same, no change is detected and the return is *null*.

Algorithm 1 RBFN and MC method for concept drift detection.

Data:
x_t - data stream sample
C - all radial functions (groups)
$\hat{c}_{t-1}$ - group activated by the previous sample
Constants:
σ - gaussian radius
λ - activation threshold
α - moment transition
δ - concept drift detection threshold
Result: *null* or *alert* or *conceptDrift*

begin
    /* $\hat{c}_t$ is the current active group */
    $\hat{c}_t \longleftarrow null$;
    $result \longleftarrow null$;
    $\varphi_{\min} \longleftarrow \lambda$;
    foreach c_i *in* ***C*** **do**
        $\varphi_i \longleftarrow e^{-(\sigma\|x_t-c_i\|)^2}$
        if $\varphi_i >= \varphi_{\min}$ **then**
            $\hat{c}_t \longleftarrow c_i$;
            $\varphi_{\min} \longleftarrow \varphi_i$;
        end
    end
    if $\hat{c}_t == null$ **then**
        C.append(x_t);
        $\hat{c}_t \longleftarrow x_t$;
    end
    if $\hat{c}_{t-1} == null$ **then**
        $\hat{c}_{t-1} \longleftarrow \hat{c}_t$;
    end
    $p(\hat{c}_t|x_t) \longleftarrow MC(\hat{c}_{t-1}, \hat{c}_t, \alpha)$;
    if $\hat{c}_{t-1} \neq \hat{c}_t$ **then**
        if $p(\hat{c}_t|x_t) \leq \delta$ **then**
            $result \longleftarrow alert$;
        end
        else
            $result \longleftarrow conceptDrift$;
        end
    end
    return $result$;
end

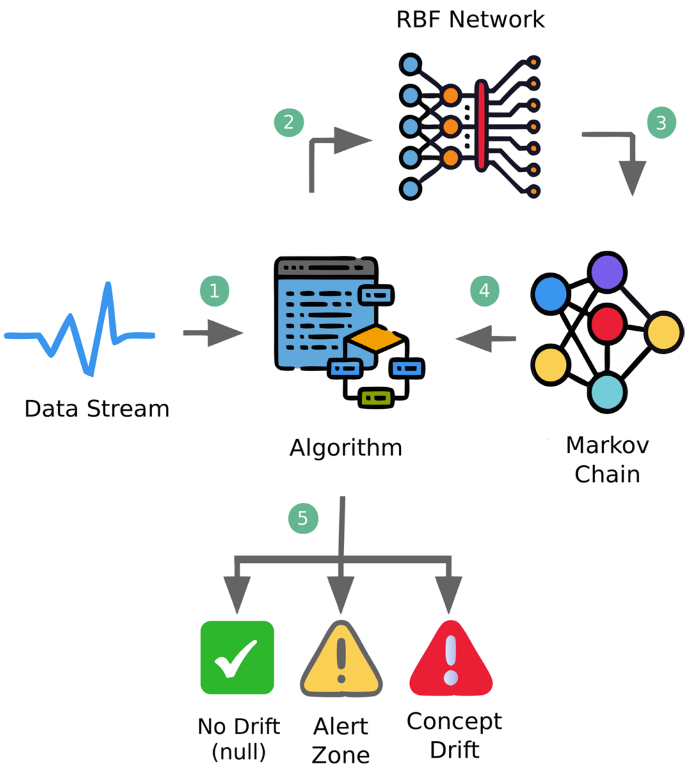


Fig. 2 Proposed architecture based on a Radial Basis Function Network and a Markov Chain. The numbered circles illustrate the sequential processing steps of our method. Steps 3 and 4 constitute the core components, involving the implementation of the RBF Network and the Markov Chain, respectively. Step 5 represents the final output stage, which includes three possible outcomes: (i) no drift detected (null), (ii) alert zone, and (iii) drift detected

It is worth emphasizing that the final model yielded by our proposed method is fully interpretable, i.e., the decision process to determine whether a sample x_t represents a concept drift is visually understood by the resulting MC graph and its transition probabilities. Furthermore, by using an RBFN, our contribution looks for nonlinear relationships within the data stream as soon as a new observation is collected, neither requiring historical windows nor labels to train a model. Finally, by analyzing our algorithm, we observe that its computational complexity can be expressed as $O(DC) + O(\mathrm{N}_u)$. In the case of univariate data, as used in our experiments, the complexity simplifies to $O(\mathrm{C}) + O(\mathrm{N}_u)$. Considering the worst-case scenario in the Markov chain, where each cluster may transition to any other (including itself), we have $\mathrm{N}_u = \mathrm{C}$. Thus, the overall complexity becomes $O(2\mathrm{C})$, which is linear in the number of clusters.

In summary, the proposed method comprises two essential components: the construction of the transition graph and the computation of probabilities for drift detection. The first component relies on a RBF network, which is driven by two parameters:

σ, responsible for computing the activation of centers, and λ, which selects the center with the highest influence on the current observation.

As shown in Eq. 4, the smaller the value of σ, the wider the Gaussian function becomes. In general, as commonly observed in Artificial Neural Network (ANN) architectures, smaller σ values tend to activate multiple centers, even those relatively distant from the input. Conversely, larger σ values result in narrower Gaussians, leading to activation only when the new observation is very close to a center, thus encouraging more selective and localized responses.

Complementarily, λ defines the minimum activation threshold required for a center to be considered responsive to a given input. Each input generates varying activation levels across existing centers, and the center with the highest activation is selected, as long as its activation value is greater than the threshold λ. If no center surpasses this threshold, it indicates that none of the existing centers can adequately represent the new observation. In such cases, a new center is created, which consequently triggers an alert or signals a potential concept drift.

To better understand the influence of σ and λ on center creation, consider Fig. 3, which illustrates how different combinations of these parameters affect the resulting number of centers C. In the figure, lighter regions of the surface correspond to a higher number of centers. This visualization highlights how the choice of σ and λ directly impacts the model's sensitivity to new observations.

In practical terms, only a small number of observations are needed to define the RBF initial parameters. For instance, if an initial sample of observations exhibits nonstationary behavior, a Gaussian function can be used to cluster them and establish the first concept. Drift detection is then handled by the second core component of our method, which is based on a Markovian process. In this stage, a drift is triggered only when the transition probability exceeds a predefined threshold δ, which can be adjusted according to the specific requirements of the application. For example, in

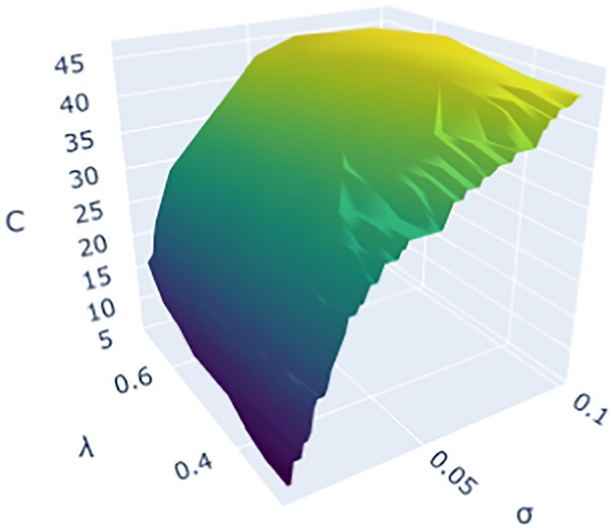


Fig. 3 Relationship between σ and λ in relation the creation of groups

real-time applications where rapid decision-making is critical, a lower δ value may be preferable to capture even minor transitions. Additionally, the optional parameter α controls the rate at which transition probabilities increase or decay over time, offering further flexibility in adapting the sensitivity of the method.

4.1 Illustrative example

To better illustrate our method, let $X = \{0.11, 0.12, 0.13, 0.33, 0.34, 0.45, 0.6, 0.33, 0.25, 0.14, 0.11, 0.15\}$, where $x_i \in \mathbb{R}$, be an initial sequence of observations collected from a data stream. Consider the following parameters for RMIDDM: $\sigma = 3$, $\lambda = 0.8$, $\alpha = 0.25$, $\delta = 0.5$. Figures 4 and 5 illustrate how both components, RBF and MC, operate in our approach.

In a given time instant $t = 1$, a new observation $x_1 = 0.11$ is analyzed by our proposal. Once there is not previous group, RBF considers x_1 as a new group, Fig. 4a, and MC creates a self-loop with a transition probability equal to 1.00, Fig. 5(T1).

After receiving two new observations $x_2 = 0.12$ and $x_3 = 0.13$, Fig. 4b and c, RMIDDM assigns both to the first cluster, as their activation values exceed the threshold $\lambda = 0.8$. For example, considering x_2, the activation in this first group is $\varphi(x_2) = e^{-(3\cdot|0.12-0.11|)^2} = 0.99$. A similar activation value is obtained for x_3, and thus, no update is required in the Markov Chain, as shown in Fig. 5(T2) and (T3).

Next, in Fig. 4d, a new observation $x_4 = 0.33$ is analyzed by RBF, producing the activation value $\varphi(x_4) = e^{-(3\cdot|0.33-0.11|)^2} = 0.65$, which is lower than $\lambda = 0.8$. Therefore, no center is actually activated, requiring the definition of a new center. In this case, MC updates the transition probabilities using $\alpha = 0.25$ as illustrated in Fig. 5(T4). Once the new probability is still lower than $\delta = 0.5$, an alert is just triggered.

By receiving a new observation $x_5 = 0.34$, Fig. 4(e), both existing groups are evaluated: $\varphi(x_1) = e^{-(3\cdot|0.34-0.11|)^2} = 0.62$ and $\varphi(x_4) = e^{-(3\cdot|0.34-0.33|)^2} = 0.99$. Considering the same center was activated, MC updates the probabilities using $\alpha = 0.25$, as show in Fig. 5(T5), and, considering $\delta = 0.5$, a concept drift is detected. At the next time instant $x_6 = 0.45$, Fig. 4f, the same center remains activated, leading to the transition in Fig. 5(T6).

To illustrate how an outlier is modeled by RMIDDM, consider $x_7 = 0.6$, shown in Fig. 4g, whose activations in both current centers are $\varphi(x_1) = e^{-(3\cdot|0.6-0.11|)^2} = 0.12$ and $\varphi(x_4) = e^{-(3\cdot|0.6-0.33|)^2} = 0.52$. Such results require the creation of a new center, in which the transitions are represented in Fig. 5(T7); thus triggering an alert.

Finally, $x_8 = 0.33$ and $x_9 = 0.25$, Fig. 4h and i, keep activated the center on observation x_4—Fig. 5(T8), and (T9)—and reinforce the current concept. Later, when $x_{10} = 0.14$, $x_{11} = 0.11$, and $x_{12} = 0.15$ are analyzed—Fig. 4j, l, and m—the first center is again activated (x_1) and the new probability is $\delta = 0.5$, as illustrated in Fig. 5 (T11).

In the following sections, we investigated the performance of our method by detecting drifts in synthetic and real-world scenarios.

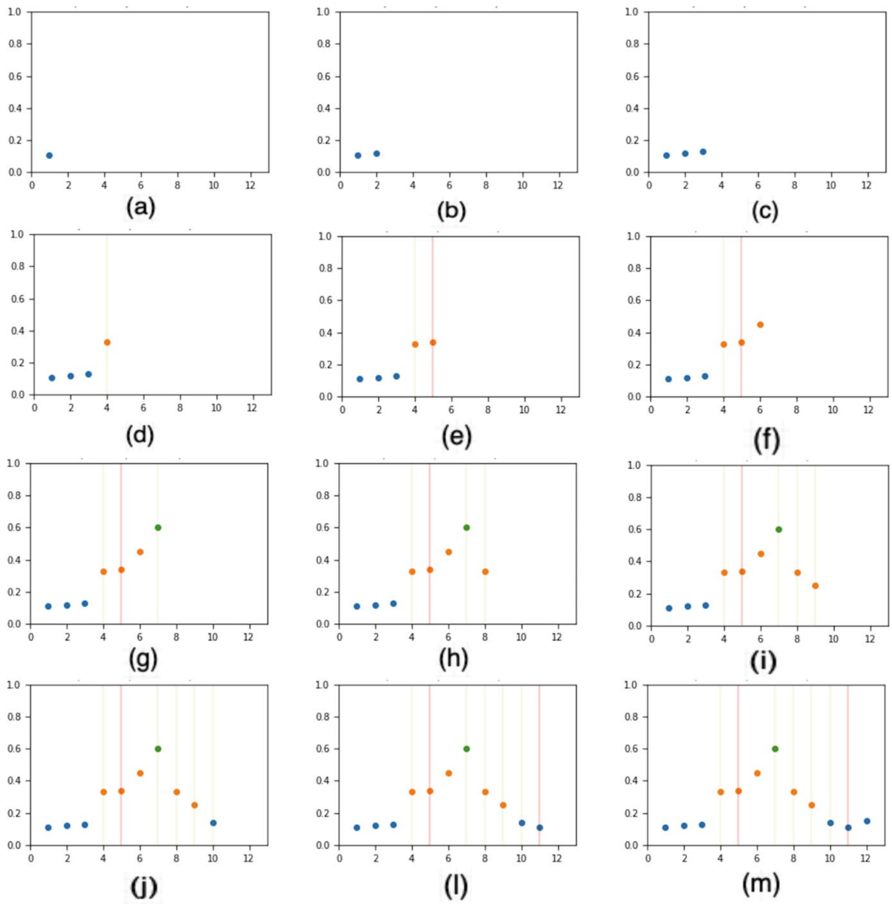


Fig. 4 Example of a data stream. Dots with different colors represent concepts varying over time. Red vertical lines illustrate the concept drift detections

5 Detecting drifts in synthetic datasets

This section discusses a set of experiments conducted on synthetic datasets, which allowed measuring the performance of our method in controlled scenarios by simulating different concept drift situations. Before discussing the obtained results in Sect. 5.4, we present the process used to create the datasets (Sect. 5.1), the evaluation metrics (Sect. 5.2), and the algorithms used as baselines (Sect. 5.3).

5.1 Datasets

Aiming at creating the datasets used to assess our method, we have modified the data generators provided by MOA to produce synthetic behaviors with different noise influences, i.e., besides presenting deterministic patterns, we also included sto-

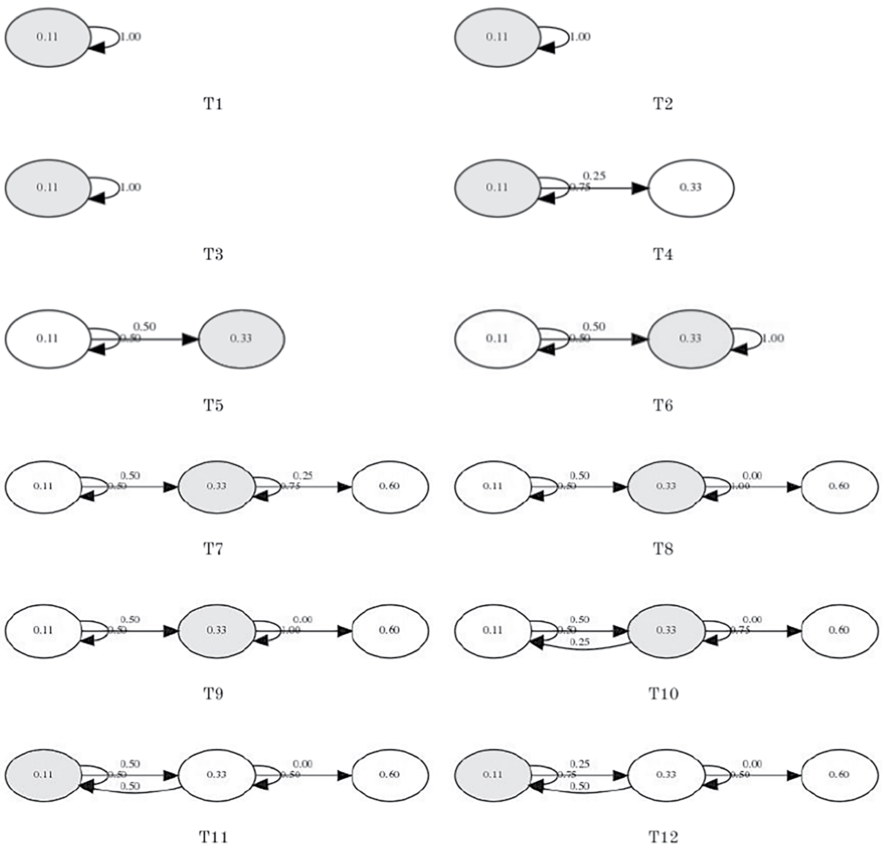


Fig. 5 Online learning based on a Markov chain

chasticity simulating noises normally present in real-world scenarios. Formally, we included additive noise to the data streams as $\hat{X} = X + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$.

All datasets were exported to files in the ARFF format, containing data streams with 2,500 observations, normalized within the interval [0, 1]. Such streams followed the four patterns depicted in Fig. 1. The *AbruptChangeGenerator* class was used to produce sets with abrupt changes, while *GradualChangeGenerator* was applied in the generation of sets with gradual and incremental changes. Similarly, the *NoChangeGenerator* class was used to produce sets without changes. All classes were parameterized to generate non-binary observations, with concepts composed of 400 instances and noise limited to the range $[-0.1, 0.2]$. It was not necessary to produce a specific dataset for the *reoccurring* pattern, as the *abrupt* dataset exhibits this behavior as well. Figure 6 shows the data streams used in our experimental evaluation, in which the vertical green lines represent the concept drifts used as ground truth in our analyses.

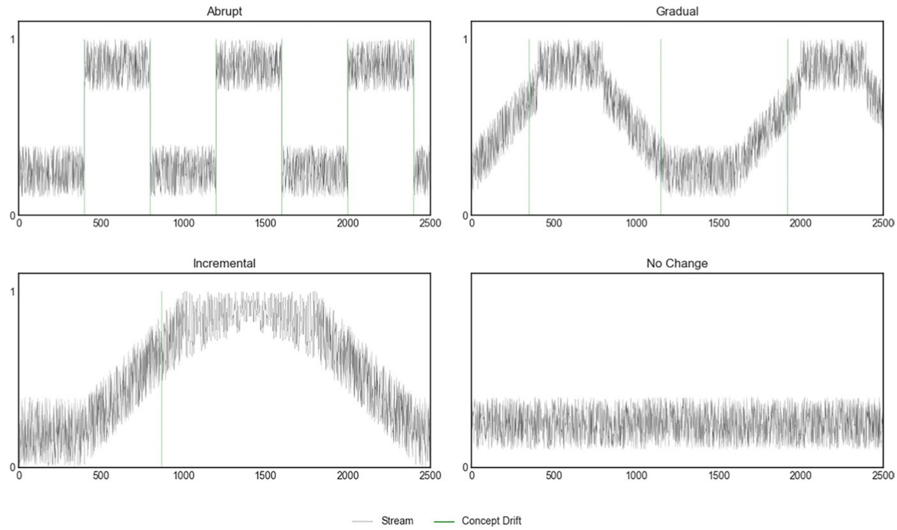


Fig. 6 Graphical representation of the datasets used in the experiment

Table 1 Evaluation Metrics implemented by the MOA's *BasicConceptDriftPerformanceEvaluator* class and used to assess our results

Metric	Observation
PT	Processing time – Average per instance in seconds
TP	True positive – Number of correct detections
FP	False positive – Number of erroneous detections
DD	Detection delay – Average number of instances between concept drift and detection

5.2 Evaluation process

The performance of our proposal was compared to the state-of-the-art methods by considering evaluation measures typically considered in data streams and concept drift areas [15], as shown in Table 1. Furthermore, information on the Real Drifts (RD) in the data stream was added to the results tables, allowing the comparison of results obtained from ground truth.

In our experiments, the training process employed the Prequential Cross-Validation (PrequentialCV) evaluation strategy (Bifet et al. 2015). In data stream scenarios, prequential evaluation is commonly used and follows a test-then-train protocol: each incoming data instance is first used to evaluate the model and then used for training. The prequential error is calculated as the cumulative loss between predicted and true labels (Lu et al. 2018). PrequentialCV enhances this approach by integrating the robustness of cross-validation with the adaptability of prequential evaluation, effectively bridging traditional machine learning and data stream learning. The PrequentialCV method adopted in our study, configured with 10 folds, is designed for large-scale data streams and addresses critical challenges such as class imbalance, concept drift at multiple time scales, and the partitioning of data for training and testing across multiple models (Bifet et al. 2015).

5.3 Baselines

The results obtained from the proposed method (RMIDDM) were compared to the most well-known algorithms discussed in Sect. 3: CUSUM, DDM, EDDM, EWMA, HDDMA, PageHinkley, SeqDrift1, Studd, Kolmogorov-Smirnov test (HypothesisTest_ks), Wilcoxon rank-sum test (HypothesisTest_wrs), T-test (HypothesisTest_tt), and Multivariate detector. We selected such algorithms due to their traditional use in comparative studies in the literature (Gama et al. 2014; Ross et al. 2012; Barros et al. 2017; Blanco et al. 2015) and availability in the MOA framework. In Table 2, we listed the parameters considered by the algorithms. Concerning the parameters, we have considered the hyperparameter optimization routine suggested by Akiba et al. (2019), which takes into account the maximum number of true positives as the objective function.

5.4 Overall results

The first analysis was conducted using a synthetic dataset without concept drifts to evaluate the tendency of each method to produce false positives (FP). As shown in Table 3, only the algorithms based on hypothesis testing generated FP alerts. Regarding processing time, RMIDDM and EDDM achieved the best performance. In this scenario, RMIDDM correctly grouped all observations into a single cluster, as expected, forming one stable group and indicating the absence of drift.

Next, we evaluated the algorithms using a dataset containing abrupt changes, with the results presented in Table 4. In this experiment, RMIDDM and SeqDrift1 achieved the best performance in terms of true positives (TP) and false positives (FP), each correctly identifying 5 out of the 6 existing drifts without generating any false positives. In contrast, the hypothesis test-based methods successfully detected all true drifts but also produced the highest number of false positives. Additionally, the proposed method achieved the third-best processing time (PT), with only a slight difference from the fastest solution. The remaining algorithms showed inferior per-

Table 2 Parameters used by every concept drift method

Algorithm	Parameters
RMIDDM	$\sigma = 2$; $\lambda = 0.5$; $\alpha = 0.005$; $\delta = 1.0$
ADWIN	$\delta = 0.002$
CUSUM	MinNumInstances = 30; $\delta = 0.005$; $\lambda = 50$
DDM	MinNumInstances = 30; WarningLevel = 2; OutControlLevel = 3
EDDM	–
EWMA	MinNumInstances = 30; $\lambda = 0.2$
HDDMA	DriftConfidence = 0.001; WarningConfidence = 0.005
PageHinkley	MinNumInstances = 30; $\delta = 0.005$; $\lambda = 50$; $\alpha = 1$
SeqDrift1	delta = 0.01; deltaWarning = 0.1; block = 200
Sudd	sizeBatchTrain = 500
Hypothesis Tests	windowSize = 10; threshold = 0.05
Multivariate	agreement = 20%

Table 3 Results of the algorithms for the dataset without concept drifts

Algorithm	PT	RD	TP	FP	DD
RMIDDM	0.010	0	0	0	–
ADWIN	0.011	0	0	0	–
CUSUM	0.012	0	0	0	–
DDM	0.012	0	0	0	–
EDDM	0.011	0	0	0	–
EWMA	0.011	0	0	0	–
HDDMA	0.012	0	0	0	–
PageHinkley	0.011	0	0	0	–
SeqDrift1	0.012	0	0	0	–
HypothesisTest_ks	0.013	0	0	16	–
HypothesisTest_tt	0.014	0	0	14	–
HypothesisTest_wrs	0.014	0	0	12	–
Studd	0.056	0	0	0	–
MultivariateDetector	0.012	0	0	0	–

Table 4 Results of the algorithms for the dataset with abrupt concept drifts

Algorithm	PT	RD	TP	FP	DD
RMIDDM	0.010	6	5	0	166.67
ADWIN	0.011	6	3	3	24.50
CUSUM	0.011	6	3	0	68.83
DDM	0.012	6	3	0	38.83
EDDM	0.011	6	0	0	–
EWMA	0.012	6	1	0	1.00
HDDMA	0.012	6	3	0	–
PageHinkley	0.011	6	1	0	16.17
SeqDrift1	0.013	6	5	0	167.50
HypothesisTest_ks	0.013	6	6	6	72.00
HypothesisTest_tt	0.013	6	6	10	8.67
HypothesisTest_wrs	0.013	6	6	6	61.83
Studd	0.056	6	0	0	–
MultivariateDetector	0.012	6	1	0	16.00

formance, with EDDM failing to detect any drifts. In this scenario, RMIDDM created two centers, effectively distinguishing between observations from the two underlying concepts.

The outputs produced by the algorithms are graphically illustrated in Fig. 7. The solid green lines represent actual concept drifts, the blue-dashed lines represent drifts correctly identified, and the red-dashed ones represent false-positive drifts.

The following experiment was conducted on the dataset with gradual concept drifts, and the results are summarized in Table 5. The table highlights that RMIDDM achieved the best overall performance, correctly identifying all three actual drifts without producing any false positives. The SeqDrift1 algorithm showed the second-best performance, successfully detecting all three drifts but generating one false positive and exhibiting a substantially higher detection delay (DD). Both CUSUM and HDDMA accurately detected two drifts without any false positives. In contrast, the remaining algorithms demonstrated lower overall performance. Notably, the proposed method and ADWIN achieved the best processing time (PT). In this scenario, our proposed method also formed two centers, effectively representing the two dominant behavioral patterns in the data.

Figure 8 presents the dataset alongside the outputs produced by all evaluated algorithms. As shown in the figure, the detection delay observed in RMIDDM was primarily caused by the second concept drift. This delay can be attributed to the gradual nature of the drift, which led to a slow and smooth transition between concepts. Since RMIDDM relies on noticeable shifts in data distribution to trigger a detection, it took longer to accumulate sufficient evidence to signal the change. Despite this delay, a visual inspection confirms that RMIDDM outperforms the other methods, demonstrating its superior detection capabilities.

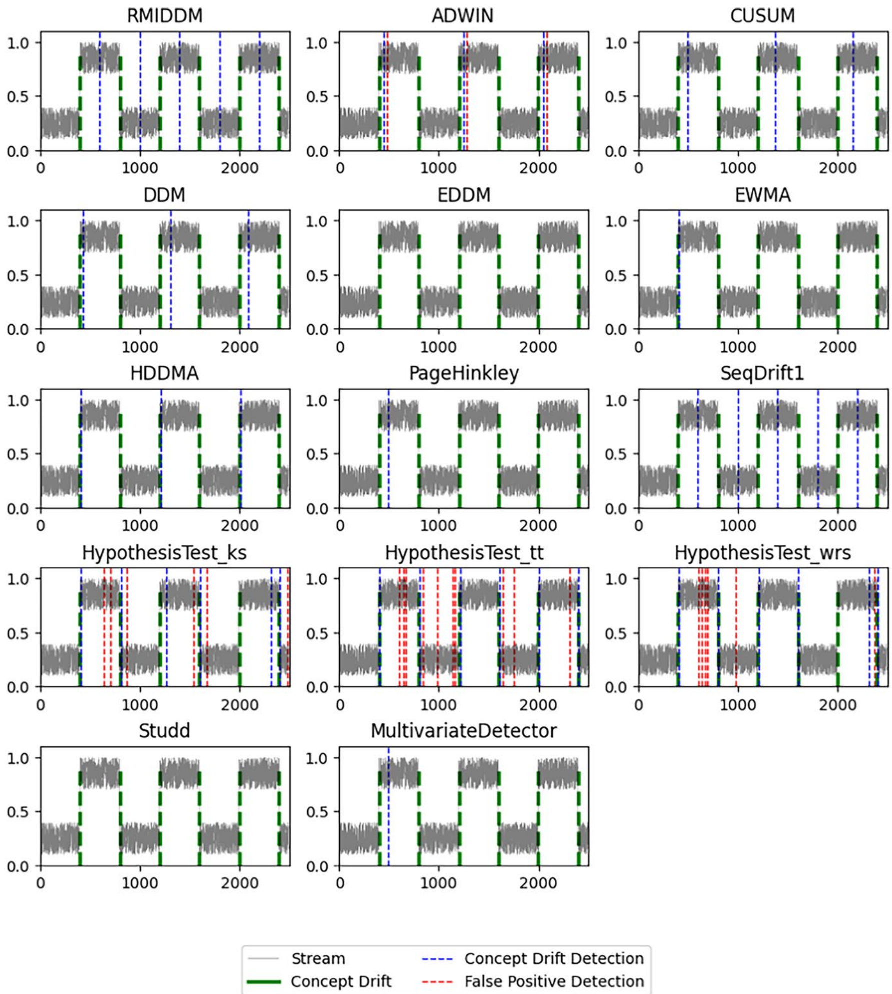


Fig. 7 Results produced by the algorithms after analyzing the dataset with abrupt concept drifts. In all plots, the x-axis represents time, and the y-axis represents the observed values

Table 5 Results of the algorithms for the dataset with gradual concept drifts

Algorithm	PT	RD	TP	FP	DD
RMIDDM	0.010	3	3	0	101.00
ADWIN	0.011	3	3	17	215.67
CUSUM	0.011	3	2	0	83.33
DDM	0.011	3	1	1	58.33
EDDM	0.011	3	0	0	–
EWMA	0.012	3	0	0	–
HDDMA	0.011	3	2	0	100.67
PageHinkley	0.011	3	1	0	22.33
SeqDrift1	0.012	3	3	1	194.33
HypothesisTest_ks	0.013	3	2	13	32.33
HypothesisTest_tt	0.014	3	3	12	221.00
HypothesisTest_wrs	0.013	3	3	13	122.00
Studd	0.056	3	0	0	–
MultivariateDetector	0.012	3	1	0	22.00

The final experiment using synthetic datasets focused on incremental concept drifts, which are inherently more challenging to detect. As shown in Table 6, all algorithms that successfully detected the actual drift also produced false positives. Consistent with previous results, the detectors based on hypothesis testing exhibited the poorest overall performance. Interestingly, some algorithms that performed poorly in earlier experiments produced only a single false positive in this scenario. For the incremental drift setting, our proposed method generated three centers, trying to capture the gradual transition between concepts.

Indeed, the performances of all methods are considered inconclusive, thus remaining an essential open problem that requires deepest investigations in future work. Figure 9 illustrates the outputs produced by all methods.

It is worth emphasizing that, even considering different metrics, the results demonstrate an outstanding performance of the proposed method, being competitive with the main detectors available in the state-of-the-art. In the next section, we analyze RMIDDM in a real-world scenario. Besides assessing its detection performance, we also investigated its capability to provide interpretable information from the decision process. As previously discussed, several real-world scenarios are looking for solutions based on XAI, i.e., solutions capable of delivering not only good performances but also mechanisms to explain every execution step.

6 Detecting drifts in real-world datasets: case study with COVID-19

To evaluate the applicability of the proposed method in a real-world scenario, we performed an experiment on the COVID-19 dataset (Mathieu et al. 2021) provided by *Our World in Data - OWID* organization. Briefly, the COVID-19 disease is a severe acute respiratory syndrome caused by the coronavirus 2 (SARS-CoV-2), which has deeply affected not only the health systems but also the global economy and politics. Due to its rapid spread, the World Health Organization (WHO) has declared a pandemic outbreak on March 11th, 2020 [35].

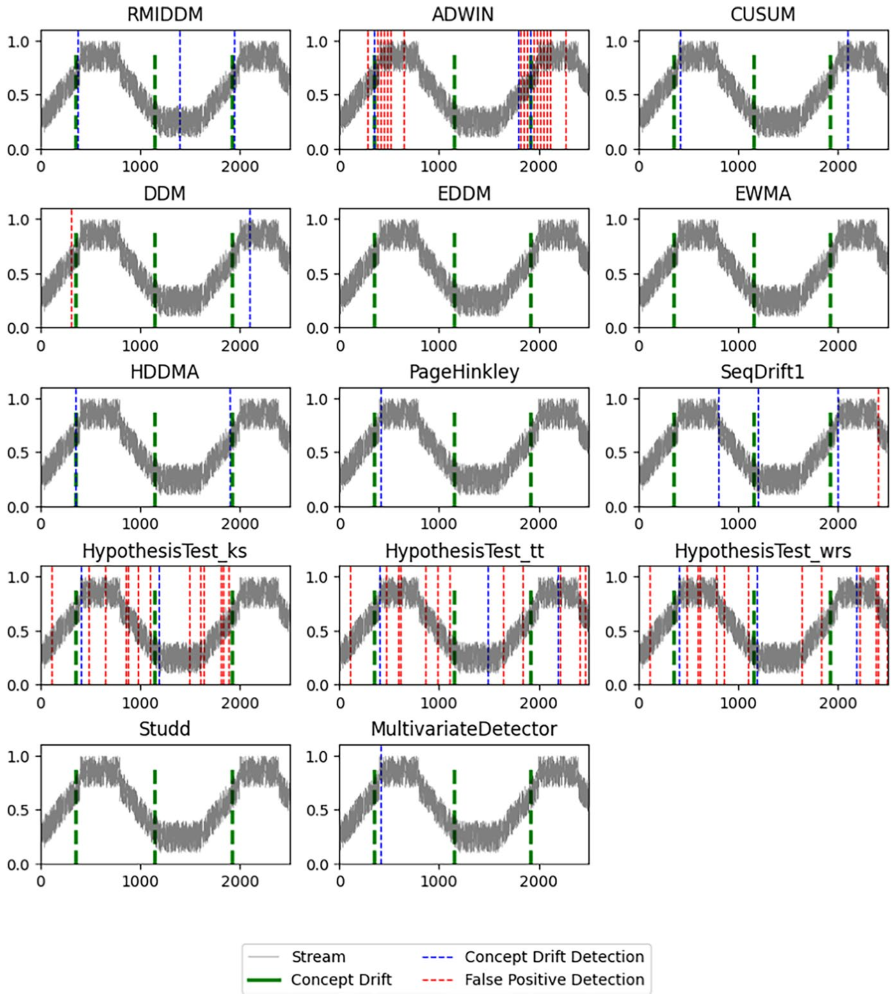


Fig. 8 Results produced by the algorithms after analyzing the dataset with gradual concept drifts. In all plots, the x-axis represents time, and the y-axis represents the observed values

From the full dataset available in Our World in Data (OWID), we analyzed three key variables: new COVID-19 cases, new deaths, and the total number of people who received at least one dose of a vaccine. The only preprocessing step applied was the use of a seven-day moving average to smooth the time series data. Subsequently, we selected three drift detection methods that showed the best performance in the synthetic dataset experiments: CUSUM, HDDMA, and our proposal.

An important point is the challenge of using the evaluation metrics in real-world scenarios. In general, they are considered in synthetic data streams due to the need for ground truth information (Webb et al. 2016). To overcome this limitation, our contribution was assessed in terms of visual inspection as further discussed.

The graphical result of this experiment is presented in Fig. 10, in which the blue line outlines the total number of people who received at least one dose of the vaccine,

Table 6 Results of the algorithms for the dataset with incremental concept drifts

Algorithm	PT	RD	TP	FP	DD
RMIDDM	0.010	1	1	2	501.00
ADWIN	0.012	1	1	14	27.00
CUSUM	0.012	1	1	1	434.00
DDM	0.012	1	1	1	349.00
EDDM	0.011	1	0	0	–
EWMA	0.011	1	0	1	–
HDDMA	0.012	1	1	1	–
PageHinkley	0.012	1	1	1	449.00
SeqDrift1	0.012	1	1	2	331.00
HypothesisTest_ks	0.013	1	1	23	182.00
HypothesisTest_tt	0.014	1	1	24	24.00
HypothesisTest_wrs	0.014	1	1	12	381.00
Studd	0.056	1	0	0	–
MultivariateDetector	0.012	1	1	1	388.00

and the green and red lines represent the seven-day average of new cases and deaths, respectively. Vertical-dashed lines in orange denote detected concept drifts.

The first observation concerns the inherent complexity of analyzing non-stationary and noisy data streams. Even after applying smoothing techniques, such as a moving average, residual variability remains and can influence model performance. Additionally, changes in statistical moments, particularly mean and variance, further impact performance, especially for methods that rely on fixed time windows.

Although CUSUM detected only two drifts, Fig. 10a, its performance is noteworthy for identifying changes during periods when the number of new cases began to decline. This highlights an important application of the method, particularly in capturing reductions following the initial wave of transmission and during the period influenced by the increase in vaccination coverage.

Another relevant behavior is observed in Fig. 10b, which presents the results produced by HDDMA. In contrast to CUSUM, HDDMA was able to detect three regions associated with increases in the number of cases. The first two correspond to the onset of the first and second waves of transmission, while the final region includes a few drifts that preceded a brief rise in cases following the second wave.

The final results, presented in Fig. 10c, correspond to our proposed method, RMIDDM. Overall, the method demonstrates a balanced performance compared to the two previous approaches in terms of the number of detected drifts. Importantly, the detections captured both increasing and decreasing trends in the data. Notably, only 20 centers were created to cluster all observations. Additionally, although 12 changes were detected, most transitions between centers were not flagged as drifts, but rather as normal fluctuations–indicating the robustness of the method in the presence of isolated or minor variations.

For brevity, we focus on the first three detected concept drifts: 22/04/2020, 03/06/2020, and 16/12/2020. The Markovian model prior to the first drift is shown in Fig. 11a, where the active group is highlighted as node **8**. On the day of the first drift, a significant change occurred–marking the onset of the pandemic–and node **131** became the new active group, as shown in Fig. 11b.

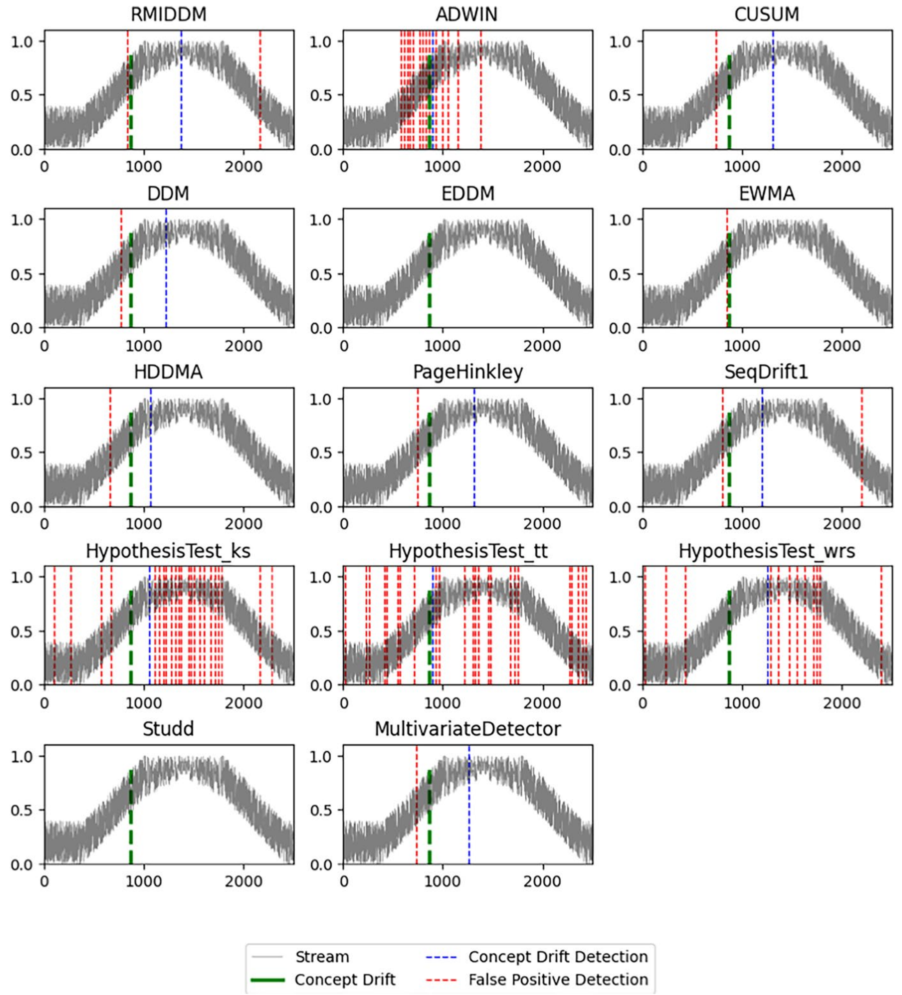
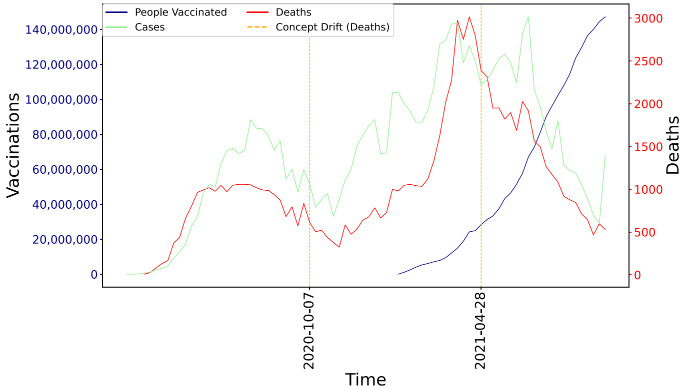
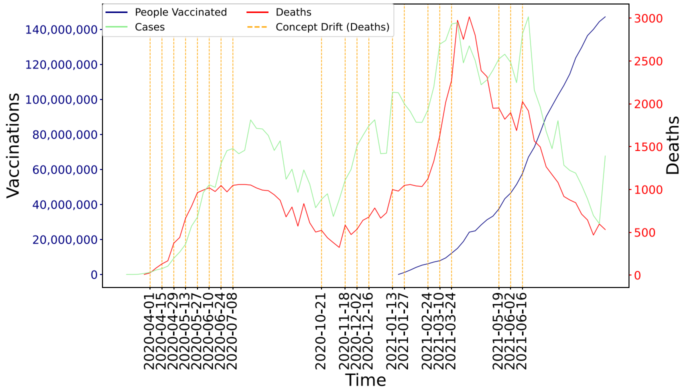


Fig. 9 Results produced by the algorithms after analyzing the dataset with incremental concept drifts. In all plots, the x-axis represents time, and the y-axis represents the observed values

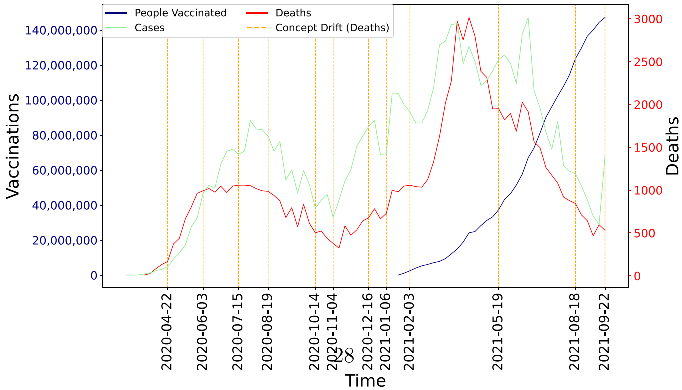
The second concept drift occurred during a stable period, when the number of new deaths had reached a plateau. The Markovian models before and after this drift are presented in Fig. 11c and d, respectively. The third drift preceded a new COVID-19 wave in December 2020, with the corresponding models shown in Fig. 11e and f. In addition to these previously mentioned events, a significant drift was detected on 2021-05-19, coinciding with a reduction in new cases and deaths as vaccination rates began to increase. This experiment using COVID-19 data demonstrates the practical applicability of the proposed method in a real and impactful scenario, while also highlighting the value of the interpretability provided by the Markov models.



(a) CUSUM



(b) HDDMA

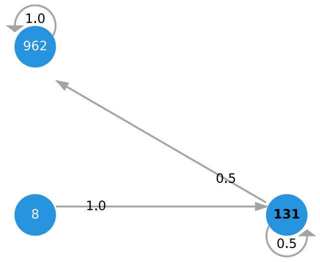


(c) RMIDDM

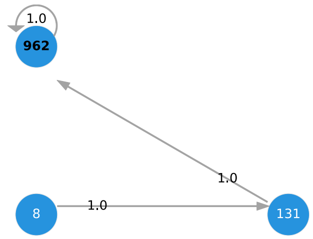
Fig. 10 Experiments conducted on data streams collected from "Our World in Data". **a** Results obtained using CUSUM detected two concept drifts, primarily associated with periods of declining new deaths. **b** HDDMA identified drifts during periods when new deaths were increasing. **c** Our proposed method demonstrated a balanced behavior between the two previous approaches, detecting drifts during both increases and decreases in new deaths over time

(a) Markovian model before the concept drift on 2020-04-22.

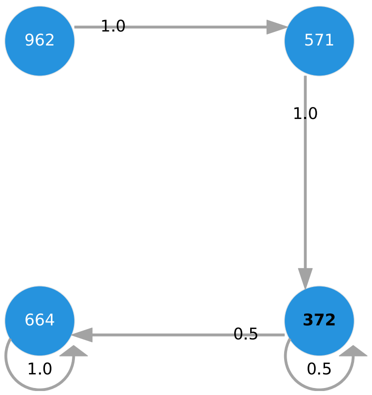
(b) Markovian model after the concept drift on 2020-04-22.



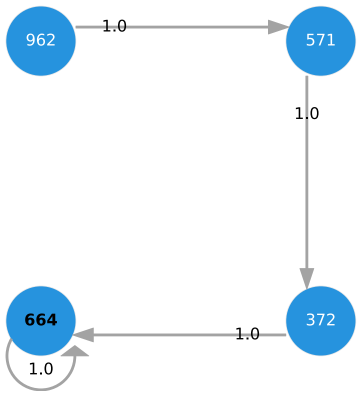
(c) Markovian model before the concept drift on 2020-06-03.



(d) Markovian model after the concept drift on 2020-06-03.



(e) Markovian model before the concept drift on 2020-12-16.



(f) Markovian model after the concept drift on 2020-12-16.

Fig. 11 Markovian models from experiment with COVID-19 real-world datasets

7 Final remarks

This paper introduces a novel and well-structured framework for concept drift detection, referred to as RMIDDM, which combines Radial Basis Function Networks (RBFN) with Markov Chains. Our method advances the state of the art by detecting changes at runtime in a computationally efficient manner while also offering interpretable results. A key advantage is that it does not require the definition of time windows, significantly enhancing usability and reducing the need for expert tuning. These features make the methodology not only theoretically relevant but also practical and well-suited for a wide range of real-world data stream scenarios.

According to our results, one may notice that our proposal presents outstanding results, detecting drifts with several patterns. Experiments with synthetic and real-world data were performed to evaluate the proposed method in addition to comparing the results with the main algorithms available in the literature.

Moreover, the time spent by RMIDDM analyzing new observations is reduced because the only information needed is whether or not the current state is suitable, and the probability modifications only involve both states associated with the transition.

Another significant contribution is related to spatial complexity. Once we save the network along with its transition probabilities, all observations previously analyzed can be disregarded, thus significantly saving memory spaces. This property, along with the low reduced processing time, makes our method suitable for devices with limited storage and processing capabilities, normally considered in smart devices used in the Internet of Things.

We emphasize that RMIDDM can also be applied when eXplainable Artificial Intelligence (XAI) is an important requirement. In summary, by producing a visual metaphor summarizing how the method detects drifts, one is able to interpret and understand each step of the decision-making process.

The main contribution of our proposal is its ability to detect concept drifts directly from the data stream in an unsupervised manner, i.e., without relying on labeled data. Additionally, our experiments address a class of non-i.i.d. problems in which data observations exhibit temporal dependencies. This setting poses the challenge of preserving the order in which data is produced and analyzed, as is typical in time series, which prevents the use of parallel batch processing. Consequently, in scenarios involving high-volume data streams, our method faces practical limitations in handling sequential observations, thus highlighting an important limitation and opening avenues for future research.

Finally, based on the evaluation metrics presented in our results, we identify two additional directions for future work: (i) reducing detection delay (DD), and (ii) improving the detection of incremental concept drifts.

Author contributions R.N. and R.A.R. conceptualized the study. R.N. and B.A. organized the data. R.N., B.A., H.G., A.B., J.G., G.C., R.R. planned the experiments. R.N., and B.A. implemented the algorithms and performed the experiments. R.N., B.A., H.G., A.B., J.G., G.C., R.R. interpreted the results, and wrote the manuscript. All authors reviewed the manuscript.

Funding This paper is based upon projects sponsored by CNPq, Brazil, under grants [404771/2024-6, 406354/2023-5, and 312755/2023-6], Maria Emilia Foundation under grant [01/2023], and FAPESB/CAPES under grant [88887.568498/2020-00]. Furthermore, this material is partially based upon work supported by the FAPESB INCITE grant PIE0002/2022 and FAPESB grant 1589/2021.

Data availability The datasets, source code, notebooks, and models used in this study are available at https://github.com/rnetonet/rmiddm_covid. The repository also includes a video explaining how to use our approach, along with an interactive web-based system that allows users to experiment with our method in a real-world application.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

Aggarwal CC (2006) Data streams: models and algorithms (Advances in database systems). Springer, Berlin, Heidelberg

Akiba T, Sano S, Yanase T, Ohta T, Koyama M (2019) Optuna: a next-generation hyperparameter optimization framework. In: Proc. of XXV international conference on knowledge discovery and data mining

Baena-García M, Campo-Ávila J, Fidalgo R, Bifet A, Gavaldá R, Morales-Bueno R (2006) Early drift detection method. In: In 4th international workshop on knowledge discovery from data streams

Barros RSM, Lima Cabral DR, Gonçalves PM Jr, CarvalhoSantos SGT (2017) RDDM: reactive drift detection method. Expert Syst Appl 90:344–355

Basseville M, Nikiforov IV (1993) Detection of abrupt changes: theory and application. Prentice-Hall Inc, Upper Saddle River, NJ, USA

Bifet A, Holmes G, Kirkby R, Pfahringer B (2010) Moa: massive online analysis. J Mach Learn Res 11:1601–1604

Bifet A, Francisci Morales G, Read J, Holmes G, Pfahringer B (2015) Efficient online evaluation of big data stream classifiers. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp 59–68

Bifet A, Gavalda R (2007) Learning from time-changing data with adaptive windowing. In: Proceedings of the 2007 SIAM international conference on data mining, pp. 443–448. SIAM

Bifet A, Kirkby R (2009) Data stream mining a practical approach

Blanco IIF, Campo-Ávila J, Ramos-Jiménez G, Bueno RM, Díaz AAO, Mota YC (2015) Online and non-parametric drift detection methods based on Hoeffding's bounds. IEEE Trans Knowl Data Eng 27(3):810–823

Cerqueira V, Gomes HM, Bifet A, Torgo L (2022) Studd: a student-teacher method for unsupervised concept drift detection. Mach Learn 112(11):1–28

Cerqueira V, Gomes HM, Bifet A (2020) Unsupervised concept drift detection using a student–teacher approach. In: Discovery Science: 23rd International Conference, DS 2020, Thessaloniki, Greece, October 19–21, 2020, Proceedings 23, Springer, pp 190–204

Cohen J, Dolan B, Dunlap M, Hellerstein JM, Welton C (2009) Mad skills: new analysis practices for big data. Proc VLDB Endow 2(2):1481–1492. https://doi.org/10.14778/1687553.1687576

Disease Control C (CDC) P (2020) Symptoms of coronavirus disease 2019 (COVID-19). https://www.cdc.gov. Online; Accessed 03 Nov 2020

Du KL, Swamy MNS (2019) Neural networks and statistical learning. Springer, London

Duda RO, Hart PE, Stork DG (2000) Pattern classification, 2Nd edn. Wiley-Interscience, New York, NY, USA

Faithfull WJ, Rodríguez JJ, Kuncheva LI (2019) Combining univariate approaches for ensemble change detection in multivariate data. Inf Fusion 45:202–214

Gama J (2010) Knowledge discovery from data streams, 1st edn. Chapman & Hall/CRC, USA

Gama J, Žliobaitė IE, Bifet A, Pechenizkiy M, Bouchachia A (2014) A survey on concept drift adaptation. ACM Comput Surv 46(4):44–14437. https://doi.org/10.1145/2523813

Gama J, Medas P, Castillo G, Rodrigues PP (2004) Learning with drift detection 3171:286–295

Kolter JZ, Maloof MA (2007) Dynamic weighted majority: an ensemble method for drifting concepts. J Mach Learn Res 8:2755–2790

Loglisci C, Impedovo A, Calders T, Ceci M (2024) Heuristic approaches for non-exhaustive pattern-based change detection in dynamic networks. J Intell Inf Syst 62(5):1455–1492

Lu J, Liu A, Dong F, Gu F, Gama J, Zhang G (2018) Learning under concept drift: a review. IEEE Trans Knowl Data Eng 31(12):2346–2363

Machine Learning for Data Streams: with Practical Examples in MOA

Mathieu E, Ritchie H, Ortiz-Ospina E, Roser M, Hasell J, Appel C, Giattino C (2021) A global database of covid-19 vaccinations. medRxiv https://doi.org/10.1101/2021.03.22.21254100, https://arxiv.org/abs/https://www.medrxiv.org/content/early/2021/03/26/2021.03.22.21254100.full.pdf

Mitchell TM (1997) Machine learning, 1st edn. McGraw-Hill Inc, New York, NY, USA

Page ES (1954) Continuous inspection schemes. Biometrika 41(1/2):100–115. https://doi.org/10.2307/2333009

Pears R, Sakthithasan S, Koh YS (2014) Detecting concept change in dynamic data streams —a sequential approach based on reservoir sampling. Mach Learn 97(3):259–293

Read J, Rios RA, Nogueira T, Mello RFD (2020) Data streams are time series: challenging assumptions. In: Brazilian Conference on Intelligent Systems, Springer, pp 529–543

Roberts SW (2000) Control chart tests based on geometric moving averages. Technometrics 42(1):97–101. https://doi.org/10.2307/1271439

Ross GJ, Adams NM, Tasoulis DK, Hand DJ (2012) Exponentially weighted moving average charts for detecting concept drift. Pattern Recogn Lett 33(2):191–198. https://doi.org/10.1016/j.patrec.2011.08.019

Ryu JW, Kantardzic MM, Kim M-W, Ra Khil A (2012) An efficient method of building an ensemble of classifiers in streaming data. In: Srinivasa S, Bhatnagar V (eds) Big data analytics. Springer, Berlin, Heidelberg, pp 122–133

Spinosa EJ, Carvalho AP, Gama JA (2007) Olindda: A cluster-based approach for detecting novelty and concept drift in data streams. In: Proceedings of the 2007 ACM Symposium on Applied Computing. SAC '07, pp. 448–452. ACM, New York, NY, USA. https://doi.org/10.1145/1244002.1244107

Webb GI, Hyde R, Cao H, Nguyen HL, Petitjean F (2016) Characterizing concept drift. Data Min Knowl Disc 30(4):964–994

Zhou L, Jiang J, Liao R, Yang T, Wang C (2015) FPGA based low-latency market data feed handler. In: Xu W, Xiao L, Li J, Zhang C, Zhu Z (eds) Computer engineering and technology. Springer, Berlin, Heidelberg, pp 69–77

Žliobaite I (2010) Change with delayed labeling: When is it detectable? In: 2010 IEEE international conference on data mining workshops, IEEE, pp 843–850

Authors and Affiliations

Ruivaldo Neto[1] · Brenno Alencar[1] · Heitor Murilo Gomes[2] · Albert Bifet[3] · João Gama[4] · Guilherme Cassales[3] · Ricardo Rios[1]

✉ Ricardo Rios
ricardoar@ufba.br

Ruivaldo Neto
rnetonet@gmail.com

Brenno Alencar
brenno.mello@ufba.br

Heitor Murilo Gomes
heitor.gomes@vuw.ac.nz

Albert Bifet
abifet@waikato.ac.nz

João Gama
jgama@fep.up.pt

Guilherme Cassales
guilherme.cassales@waikato.ac.nz

[1] Institute of Computing, Federal University of Bahia, Salvador, BA 40170-110, Brazil

[2] School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6140, New Zealand

[3] Department of Computer Science, University of Waikato, Hamilton 3240, New Zealand

[4] INESC TEC, University of Porto, 4200-465 Porto, Portugal