Forward and Backward Inertial Anomaly Detector: A Novel Time Series Event Detection Method

Janio Lima, Pedro Alpis, Rebecca Salles, Luciana Escobar, Fabio Porto, Esther Pacitti, Rafaelli Coutinho, Eduardo Ogasawara

Forward and Backward Inertial Anomaly Detector: A Novel Time Series Event Detection Method

Janio Lima
CEFET/RJ
janio.lima@aluno.cefet-rj.br

Pedro Alpis
CEFET/RJ
pedro.alpis@eic.cefet-rj.br

Rebecca Salles
CEFET/RJ
rebecca.salles@eic.cefet-rj.br

Luciana Escobar
CEFET/RJ
luciana.escobar@eic.cefet-rj.br

Fabio Porto
LNCC
fporto@lncc.br

Esther Pacitti
INRIA & University of Montpellier
esther.pacitti@inria.fr

Rafaelli Coutinho
CEFET/RJ
rafaelli.coutinho@cefet-rj.br

Eduardo Ogasawara
CEFET/RJ
eogasawara@ieee.org

Abstract

Time series event detection is related to studying methods for detecting observations in a series with special meaning. These observations differ from the expected behavior of the data set. In data streaming scenarios, it is possible to observe an increase in the speed of data generation in time series. Therefore, adapting to time series changes becomes crucial. Thus, identifying events associated with these changes is essential for timely and correct decision-making. Although there are many methods to detect events, it is still possible to have difficulties detecting them correctly, particularly those associated with concept drift. In order to fill the gap in the literature, this work proposes a new method, named Forward and Backward Inertial Anomaly Detector (FBIAD), for detecting events in time series. It contributes by analyzing surrounding inertia around observations. FBIAD outperformed other methods both in accuracy and elapsed time.

***K*eywords** event detection · inertia · time series · anomalies

1 Introduction

In analyzing time series, it is often possible to observe a significant change in the behavior of a time series at a certain point or time interval. Such a behavior change usually characterizes the occurrence of an event [18, 37]. An event detected in time series can often represent the occurrence of a phenomenon with specific and defined meaning in a certain domain of knowledge [13].

Thus, the event detection problem is particularly relevant for applications based on sensor data analysis. Examples of such applications include water quality analysis [27], reflection seismic analysis [40] and oil drilling and exploration [3]. Often, the capture of data by sensors is prone to errors, generating spurious data, which should not be confused with relevant events about the monitored phenomenon. Furthermore, an event can be observed by a set of sensors. In contrast, each sensor can report information related to different events [10].

Events detected in time series commonly present as anomalies [7, 32] or change points (*change points*) [2]. Anomalies are observations that do not conform to the expected pattern of behavior within the dataset [1, 17]. In turn, change points separate different states that generate the time series. The change point detection problem is related to concept drifts in time series. In this case, the detection of change points aims to find the specific instant (or interval) in time that marks the occurrence of the concept deviation.

The literature presents many methods developed to detect events in time series. However, choosing and applying a suitable method for a given time series is not a simple task. The choice and parameterization of a detection method are directly related to the initial assumptions regarding the behavior of the time series and the statistical properties intrinsic to the data. It is a complex task, particularly considering that the nature of events observed in a time series is often unknown. As methods tend to specialize in detecting a single type of event, the adoption of one method may override the detection of the other [30]. The incorrect usage may lose relevant events that could alert experts to make timely decisions or produce multiple false positives. There is a loss of credibility in using these techniques to control applications in both cases.

Furthermore, all these challenges become even more critical when in online monitoring systems (commonly associated with data in *streaming*). The demand for detection of these events is pressured by the need for speed for computational processing [19]. However, these detected events may force systems to adapt to the changes in provided *streaming* data [34]. It is an analogous scenario of adaptability of machine learning models [11, 14, 39, 38, 22].

In this scenario, machine learning models are expected to adjust themselves in real-time to achieve greater robustness and stability. However, adaptability may not lead to robustness. An adaptive model that reacts to short-lived phenomena changes rapidly and responds to spurious disturbances. Adaptations must occur when the phenomena have sufficient duration to characterize them as significant changes. Such a relationship between robustness and adaptability is not trivial, going back to the plasticity-stability dilemma [21].

Therefore, the problem addressed in this work is the detection of anomalies and, when possible, identifying them as a change point. In this sense, this work provides a new anomaly detection method called Forward and Backward Inertial Anomaly Detector (FBIAD). The method is inspired by adaptive normalization [25] and is based on forward and backward inertia that can recognize both anomalies and abrupt change points. This method's intuition is associated with the fact that if an observation of time series is non-anomalous, its surrounding inertia [16] should be close. The method is evaluated with other anomaly detector methods and outperforms other methods both in speed and accuracy.

In addition to this introduction, Section 2 presents a general background on event detection. Section 3 presents the related work. Section 4 presents FBIAD. Section 5 presents the experimental evaluation and its discussion, while Section 6 makes final remarks.

2 Background

2.1 Time series

A time series is a set of data organized chronologically. Formally a time series Y is expressed as a sequence of observations $< y_1, y_2, ..., y_n >$, where $y_i \in R$ and n is the number of observations, with y_1 being the oldest observation and y_n the most recent observation. This representation refers to a series with only one variable, called univariate time series [12, 31].

Some characteristics of time series make it difficult to apply traditional statistical methods or require specific analysis techniques. For example, the autocorrelation between the observations of a series goes against the assumption made by statistical methods of independence and identical distribution between adjacent observations. Due to that, it is common to study time series observations as a function of their past data [31].

The notion of stationarity refers to characteristics of a series that maintain the regularity of some statistical measures over time [29]. It includes the mean, variance, and autocovariance [31]. Many methods applied to time series analysis assume their stationarity. However, it is very common for time series to be non-stationary. When they violate any of the constraints of a stationary series, they are considered non-stationary. The occurrence of non-stationarity is usually due to the change in the mean and variance of the series. They are commonly associated with events [29].

Time series decomposition aims to identify latent components of the series that are non-observable and related to different types of temporal variations. The main components of a time series are trends, cycles, seasonality, and residuals. The trend indicates rising or falling values, and cycles are occurrences of peaks or valleys repetitively over time. Seasonality is similar to cycles. However, it refers to their repetitive occurrence in each year, and the residuals are the remaining components, relating to unpredictable variations [4].

2.2 Events

Events in time series are points with special meaning. Although identified as specific observations, they might be related to intervals around them. Events can refer, for example, to anomalies, change points, peaks, or variations in the data components [15, 7].

A common type of event is an anomaly, which refers to observations that do not match the expected behavior of the dataset [7]. In the most general concept, anomalies are observations that stand out because they seem not to have been generated by the same process as the others. In this case, anomalies can be modeled as isolated observations of the remaining data based on similarity or distance functions [17]. Given a set of observations X, the index positions of anomalies can be identified as $A(X)$ through Eq. 1, where $q_1(X)$ and $q_3(X)$ are respectively the first and third quartile and *iqr* is the interquartile distance [17].

$$A(X) = \{i\}, \forall i \in \{1, \cdots, n\}, |X| = n \mid x_i \not\subseteq [q_1(X) - 1.5 \cdot iqr(X), q_3(X) + 1.5 \cdot iqr(X)] \tag{1}$$

As for time series, the anomalies can be of trend or volatility. Trend anomalies occur when an event deviates from the expected trend of the time series. In contrast, volatility anomalies occur when there is a large change in the variance of time series samples [17, 11, 2].

Event detection refers to detecting points of special meaning in a time series. One way to divide the series analysis for event detection is to access the data online and offline. When access is in real-time, as new observations are generated, the analysis is called online detection. The offline detection is when directly accessing the complete dataset [35].

2.3 Trend anomalies detection

In the context of time series, there is a particular interest in detecting anomalies that may represent an event that deviates from the trend of the process that generates the time series Y. They are called trend anomalies (TA). Let $\hat{Y}$ be an estimate of the process generating Y, produced by fitting a trend model α, such that $\hat{Y} = \alpha(Y)$. Since ϵ is a time series of residuals (*white noise*) obtained after removing $\hat{Y}$, it follows that the trend anomalies (TA) of Y are usually identified as $TA(Y, \hat{Y})$ through Eq. 2.

$$TA(Y, \hat{Y}) = A(\epsilon) \mid \epsilon_i = y_i - \hat{Y}_i \tag{2}$$

2.4 Volatility anomalies detection

Most financial time series exhibit nonlinear properties that cannot be captured by existing linear models, as the volatility varies greatly over time. Thus, a demand arises for studying the volatility of the time series. Econometric models appear to deal with the non-linearity of data, including stochastic volatility. It includes autoregressive conditional heteroscedasticity (ARCH) and generalized autoregressive conditional heteroscedasticity (GARCH). The latter being the best known and applied [6]. In the financial area, volatility is associated with risk, which can indicate an event in the context of time series.

GARCH-type models involve estimating volatility based on previous observations. GARCH is a non-linear time series model, where a series Y is explained according to Eq. 3, μ_i being the average component. The noise sequence w_i is i.i.d. N(0.1), so the conditional distribution of $\tilde{y}_i = y_i - \mu_i$, given $\tilde{y}_{i-1}, \tilde{y}_{i-2}, \ldots$ is $\mathrm{N}(0, \sigma_i^2)$ [6]. Such a model can be used as α for detecting anomalies similarly to Eq. 2, or even, its estimates of instantaneous volatilities can be subject to an anomaly detection (Eq. 3).

$$y_i = \mu_i + \sigma_i w_i \tag{3}$$

2.5 Change point detection

The change point detection methods aim to find the points or intervals that represent a transition between different states in a process that generates the time series [33]. Change point detection can be defined as a hypothesis testing problem, where the null hypothesis H_0 characterizes the absence of change points and the alternative hypothesis H_A negates H_0. Let $seq_{i,p}(y)$ be a subsequence of observations of a time series and $t, k \in \{i, \ldots, i + p\}$, where $t < k$. It is formally assumed that $H_0 : \forall\, t, k\ (t < k) \mid \mathbb{P}_{y_t} = \mathbb{P}_{y_k}$ and $H_A : \forall\, t\ \exists\, k\ (t \neq k) \mid \mathbb{P}_{y_t} \neq \mathbb{P}_{y_k}$, where $\mathbb{P}_{y_i}$ is the probability density function of the subsequence and k is a change point [8]. The seminal method of change point detection (SCP) became a reference in the literature for change point [18].

3 Related Work

Event detection aims to identify points with special meaning in a time series. Although identified as specific points, they may be related to series intervals around the detected event with a significant change in the behavior of the data. This change refers, for example, to spikes, change points, or anomalies [18].

In the literature, there are several methods and *frameworks* applicable to the detection of *online* events, with data generated in real-time. An overview of the state of the art in this area is presented by Habeeb et al. [19], whose study presents two main divisions in the detection of online events: (i) static models and (ii) dynamic models. Static models are trained on large datasets and applied to online data. In contrast, in dynamic models, training is started on a subset of data, and learning continues as new observations are received [19].

Four general strategies are used as a base for many methods found in literature: (i) model deviation analysis, (ii) classification-based analysis, (iii) clustering-based analysis, or (iv) statistical techniques [7]. In model deviation, a model is fitted to the observations in a series. Then events are identified when observations deviate from the fitted model. Classification-based and clustering-based abnormalities are identified by comparing them to samples previously learned of classes or clustered. Domain-based strategies compare new data samples against what is expected based on expert knowledge. Finally, statistical techniques are used to identify deviations from the data distribution [26].

The literature presents several methods for detecting trend anomalies, particularly in defining different trend models. Among them are decomposition-based and KNN-CAD [5]. The decomposition method adopts an approach that consists of decomposing the time series into trend, seasonality, and residuals, on which the search for anomalies is carried out [17].

The anomalies method is focused on the analysis of time series according to trends and seasonality [9]. Anomalize is a decomposition-based method implemented in *anomalize* R package. This method removes trend and seasonal components and searches for anomalies in the residuals. Nevertheless, when dealing with volatility, one of the commonly applied approaches is GARCH, which works by estimating volatility based on previous observations [6].

Change Finder (CF) is a method that searches for anomalies and change points, developed based on seminal change point (SCP), a reference in the literature related to change point detection [33]. Due to that, CF is the most related work. In the first phase of CF, a α model is fitted to the time series X resulting in $\hat{X}$. From the residuals of the series s, defined in Equation 4, the anomalies are marked. The second phase consists of defining a new series $\overline{s}_p$, which is defined from the moving averages of s with p terms. Anomalies found in this new series $\overline{s}_p$ result in the detection of change points.

$$s_i = (\hat{x}_i - x_i)^2\ ,\ \hat{x}_i = \alpha(x)_i \tag{4}$$

Some anomaly detectors are specialized for certain types of data. It includes, for example, the work of Ullah et al. [36], which is a combination of CNN and LSTM to find anomalies in surveillance scenes. It is a related work since it uses bidirectional analysis, but it is not based on inertia and is not focused on time series.

An anomaly detection method was introduced by Ren et al. [28] as a service on the *Microsoft Azure* cloud platform. This service, called Microsoft Anomaly Detector (MAD), uses the SR-CNN algorithm, based on *spectral residual (SR)* algorithms that evaluate the time series in an unsupervised way to create a salience map. The salience map is used as input to convolutional neural networks (CNN) that search for anomalies [28].

A *framework* for *online* detection is proposed by Talagala et al. [34], based on computing boundaries of normal data behavior to identify significant changes in new observations. Another example of a framework is *Harbinger*, which provides functions for event detection, evaluation, visualization, the combination of methods, and comparison of detections performed [30]. Instead of a specific method, Harbinger allows different event detection methods to be implemented and integrated.

From the previous comments, it is possible to observe that no other work studies the forward and backward inertia to analyze the divergence of observations. It opens room to study the proposed method.

4 Forward and Backward Inertial Anomaly Detector

The Forward and Backward Inertial Anomaly Detector (FBIAD) is a novel unified change point and trend anomaly detector. It can identify both punctual and interval anomalies and classify them as (i) punctual trend anomalies, (ii) abrupt change points, and (iii) generic anomalies. The event detection process of FBIAD comprises four phases:

(1) processing forward and backward time series sliding windows; (2) computing forward and backward inertial differentiation (3) registering forward and backward anomalies; (4) classifying anomalies. It is depicted in Figure 1 and described in the following subsections.

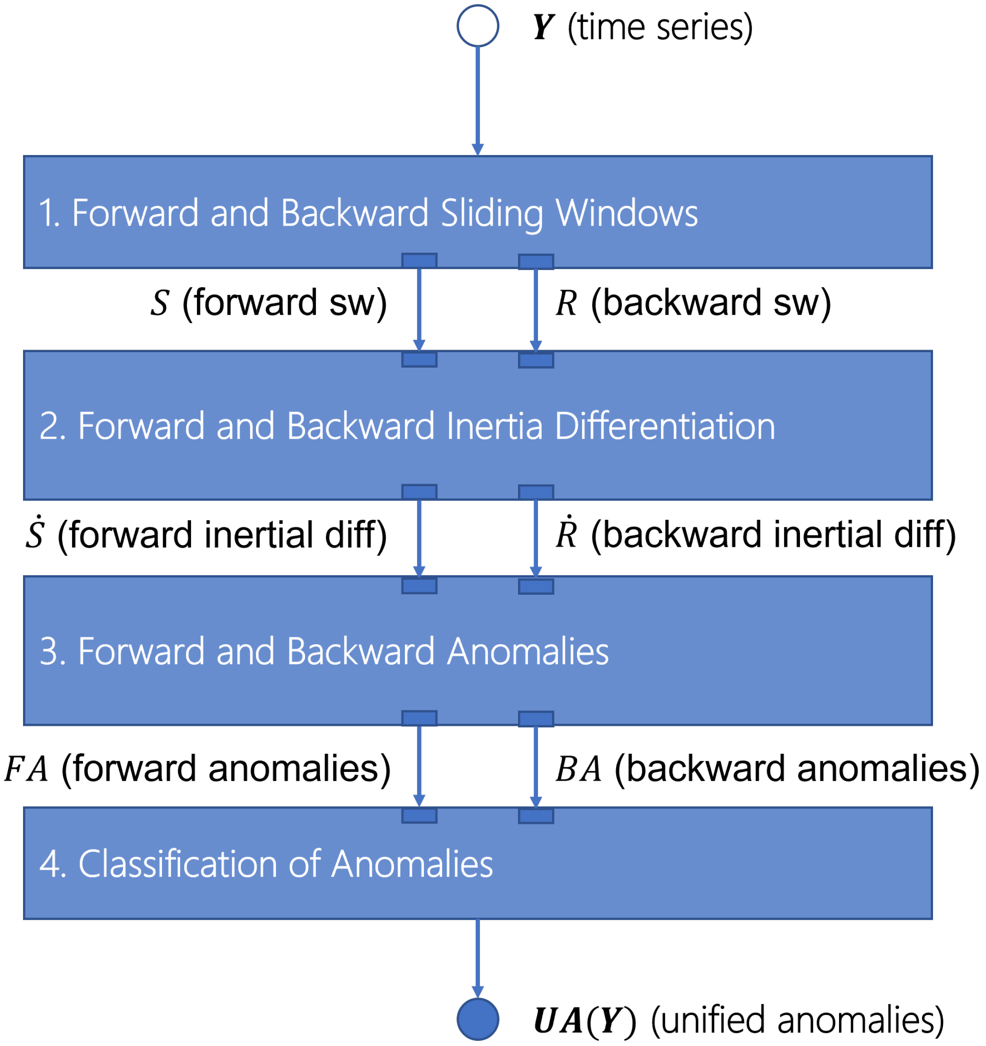
$\boldsymbol{Y}$ (time series)
1. Forward and Backward Sliding Windows
S (forward sw)
R (backward sw)
2. Forward and Backward Inertia Differentiation
$\dot{S}$ (forward inertial diff)
$\dot{R}$ (backward inertial diff)
3. Forward and Backward Anomalies
FA (forward anomalies)
BA (backward anomalies)
4. Classification of Anomalies
$\boldsymbol{UA}(\boldsymbol{Y})$ (unified anomalies)

Figure 1: Workflow for FBIAD

4.1 Forward and Backward Time Series Sliding Windows

A subsequence is a continuous sample of a series. The i-th subsequence of length p in a series Y, represented by $seq_{p,i}(Y)$, is an ordered sequence of values $(y_i, y_{i+1}, \cdots, y_{i+p-1})$, where $|seq_{p,i}(Y)| = p$ and $1 \leq i \leq |Y| - p$.

Sliding windows of size p consist of exploring all subsequences of size p of a series. Formally, they can be represented by $sw_p(Y)$, which corresponds to a matrix S of size $(|Y| - p + 1) \times p$. Each row vector s_i in S is the i-th subsequence of length p in y. Given $S = sw_p(Y)$, $\forall s_i \in S$, $S_i = seq_{p,i}(Y)$. It is worth noting that the sliding windows organize the columns of the matrix S so that the j-th column corresponds to a lag of the original series y by $(p - j)$ preceding values. In this way, S is a forward time series sliding windows for Y.

Let Y be a time series, such that $|Y| = n$ and $Y =< y_1, \cdots, y_n >$. The reverse of the time series $rev(Y)$ is defined as the reverse sequence of Y. Thus, $rev(Y) =< y_n, \cdots, y_1 >$. Let R be an array for the sliding windows of $rev(Y)$, such that $R = sw_p(rev(Y))$. Then, R is a backward time series sliding windows for Y.

4.2 Forward and Backward Inertial Differentiation

Given S as the matrix formed by sliding windows of size p over a time series Y of size n. Let $s_i = (s_{i_1}, \cdots, s_{i_p})$ the i-th row vector for S, such that $1 \leq i \leq (|Y| - p)$. For each $s_i \in S$, the inertia for all s_i is expressed as vector μ_i [16]. In this paper, the inertia is computed as the the average of that window s_i, and is expressed in Equation 5.

$$\mu_i = \frac{\sum_{j=1}^{p}(s_{i_j})}{p} \tag{5}$$

Given a inertial vector μ_i and sliding windows S, the inertial differentiation for S is expressed as $\dot{S}$ and defined by Equation 6. The inertial differentiation is applied over all rows s_i. At the end of this process, the matrix $\dot{S}$ has mean 0 and variance σ^2 [23]. Besides, $\dot{S}$ is the forward inertial differentiation with respect to Y.

$$\dot{S}_i = s_i - \mu_i, \forall i \in [1, |\mu|] \tag{6}$$

Given R as the array corresponding to the backward time series sliding windows for Y. Then, $b\mu_i$ is the backward inertia for all row vectors r_i in R computed using Equation 5, by changing μ_i with $b\mu_i$ and s_{i_j} with r_{i_j}. The inertial differentiation for R is expressed as $\dot{R}$ by applying Equation 6, by respectively changing $\dot{S}_i$, s_i, μ_i to $\dot{R}_i$, r_i, $b\mu_i$. In this case, the $\dot{R}$ is the backward inertial differentiation with respect to Y.

4.3 Forward and Backward Anomalies

Given $\dot{S}$ as the forward inertial differentiation for Y. Let $\dot{S}_p$ be the p column vector of $\dot{S}$. It is associated to observations p to n of time series Y. The $i-th$ observation of $\dot{S}$ (for short, $\dot{S}_{p_i}$) can be interpreted as the error of observation for Y_{i+p} to the model established by the forward inertia μ_i ($1 \leq i \leq n-p$). In this case, $FA(Y)$ is the set indexes for forward inertial anomalies for Y, *i.e.*, $FA(Y) = A(\dot{S}_p)$.

Conversely, given $\dot{R}$ as the backward inertial differentiation for Y. Let $\dot{R}_p$ be the p column vector of $\dot{R}$. It is associated to reverse observations $n-p$ to 1 of time series Y. The observations of $\dot{R}_{p_i}$ can be interpreted as the error of observations Y_{n-p-i} to the model established by the backward inertia $b\mu_i$ ($1 \leq i \leq n-p$). In this case, $BA(Y)$ is the set indexes for backward inertial anomalies for Y, *i.e.*, $BA(Y) = n - p - A(\dot{R}_p)$. It is worth mentioning that the term $n - p$ is used to adjust indexes for the reverse time series ($rev(Y)$) with respect to the original indexes positions for observations of Y.

4.4 Classification of anomalies

Given $FA(Y)$ and $BA(Y)$ as forward and backward anomalies for Y, the unified anomalies (UA) for Y is the union of both forward and backward anomalies. Formally, $UA(Y)$ is computed as $UA(Y) = FA(Y) \cup BA(Y)$. According to the FBIAD method, all index positions established by $UA(Y)$ are anomalies for Y.

Let $IA(Y)$ be the intersection of $FA(Y)$ with $BA(Y)$, *i.e.*, $IA(Y) = FA(Y) \cap BA(Y)$. Then, all non-consecutive indexes in $IA(Y)$ are punctual trend anomalies: $PTA(Y)$. Formally, $PTA(Y) = \{p\} \mid p \in IA(Y) \land p-1 \notin UA(Y) \land p+1 \notin UA(Y)$. The reasoning for it, is that each position p in $PTA(Y)$ is found as anomaly both in forward and backward anomalies. It means the observation y_i differs significantly from the forward μ_i and backward $b\mu_{n-p-i}$ inertia. In other words, it does not fit in both nearby previous and following observations.

Consider consecutive indexes (p, $p+1$) in $UA(Y)$. This pair is an abrupt change point if $p+1$ is only present in $FA(Y)$ and p is only present in $BA(Y)$. Observation at p fits forward inertia well but is an anomaly with backward inertia. Conversely, $p+1$ fits backward inertia well but is an anomaly concerning the forward inertia. All points that satisfy these criteria are punctual change points: $PCP(Y)$.

Other anomalies which are not classified as punctual trend anomalies and punctual change points are classified as generic anomalies. Formally, a index p is a generic anomaly when $p \in UA(Y) \land p \notin PTA(Y) \land p \notin PCP(Y)$.

5 Experimental Evaluation

This section presents an experimental evaluation comparing FBIAD against representative methods for different event detection methods: (i) trend anomaly (Anomalize, ANM); (ii) change point detection (Change Finder, CF); (iii) volatility anomaly (GARCH), (iv) cloud-based (Microsoft Anomaly Detector, MAD).

5.1 Experimental setup

The experiments were performed in a general-purpose virtual machine type Standard_DS2_v2, available at Microsoft Azure, with a dual-core CPU and 7 GB of RAM. Except for MAD, which was run through its API, all methods were implemented directly in R, including FBIAD, CF [33], GARCH [6], and ANM. The parameters used for each method are described in Table 1.

Table 1: Parameters by method

Method	Parameter	Value
FBIAD	*w* (sliding windows size)	90
FBIAD	*alpha*	3.0
ANM	*alpha*	3.0
CF	*mdl* (model)	linear regression
CF	*m* (moving average size)	90
GARCH	*variance.model*	model = sGARCH
GARCH	*variance.model*	garchOrder = (1, 1)
GARCH	*mean.model*	armaOrder = (1, 1)
GARCH	*mean.model*	include.mean = TRUE
GARCH	*distribution.model*	normal distribution
GARCH	*alpha*	3.0
MAD	*sensitivity*	0.9

Many parameters are specific to each method, whose works cited in Section 3 provide additional information to those presented in Table 1. On the other hand, when different methods use similar parameters, their values were kept the same in the experiments to preserve comparability. For FBIAD and CF methods that work with sliding windows (parameters *w* and *m*, respectively), a value of 90 was used. Another parameter common to several methods is alpha, set to 3.0 in all cases.

Data were submitted for analysis using batch detection mode. The methods were selected because of their ability to deal with different events in time series. It includes trend anomalies, change points, volatility, and machine learning.

The *GECCO Challenge 2018* and *Yahoo Labs* datasets [30] were used. The *Numenta Anomaly Benchmark*[1] was added due to the wide use of these datasets in the literature and because they provide series with different properties such as volatility, trend, as well as series with and without seasonality.

The *GECCO Challenge 2018* [24] presents nine variables related to drinking water composition[2]. This dataset, with observations of water components collected every minute, is important for identifying anomalies in water quality. An extract with 1500 observations of about one day of the given time series was selected to perform the experiments. The selected data contains 72 labeled events, including additional events imputed by [24] into the data to simulate difficulties.

The *Yahoo Labs* dataset has synthetic and real series about data traffic in *Yahoo* services[3]. The Yahoo dataset features different series patterns. In addition, there are labeled data of the anomalies present in the series, allowing the proper evaluation of the detections. Finally, the nature of the time series (internet data traffic) allows evaluating the applicability of the evaluated methods in a scenario similar to cloud data traffic.

In turn, *Numenta* presents several series with appropriate labels to evaluate event detection methods. From Numenta, two series were selected with synthetic and real data. The synthetic series simulates the occurrence of anomalies and a real one with data on monitoring cloud services from *Amazon Web Services* (AWS), such as CPU utilization, network

[1] Available in the R language OTSAD package

[2] Available on http://www.spotseven.de/gecco-challenge/gecco-challenge-2018

[3] Available on https://yahooresearch.tumblr.com/post/114590420346/

data traffic, and disk reading. This second series was selected as another opportunity to evaluate the methods applied to the analysis of cloud data traffic.

It is possible to evaluate the result of the detections using different metrics, both concerning the assertiveness of the detections and their performance. Hit analysis is similar to machine learning classification metrics, considering the scenario of matching detected events with labeled data. Common ways of analyzing classification are the calculation of the confusion matrix, indicating true positives (TP) (event hits), true negatives (TN) (no event hits), false positives (FP), and false negatives (FN) [20].

In addition, different metrics are broken down from this matrix, such as accuracy, which shows the recognition rate. Three metrics are often used together to deepen the interpretation of results: (i) precision: exactness measure; (ii) recall: completeness measure or true positive rate; and (iii) F_1: harmonic mean of precision and recall [20]. The detections were compared with the labeled data of the series. To analyze different aspects of the detection, precision, recall, F_1, and accuracy was used. Detailed analysis and comparison between FBIAD and other methods are presented in Subsection 5.2. The result of each method by metric and dataset is presented in 5.3 and Subsection 5.4 shows an analysis of performance.

5.2 Detail analysis of pH Serie - Gecco Challenge Dataset

This subsection details the analysis of the pH series from the Gecco Challenge dataset to explore the results of the detections and behavior of methods. The pH is the first series of the Gecco Challenge. It was chosen for illustration and exploration of results. Nevertheless, the satisfactory results of the FBIAD in this series were similar to the other series of the dataset. It maintains high levels of accuracy, a balance between precision, recall, and F_1. To this extent, it led the FBIAD method to have the best-consolidated results for this dataset as presented in Table 3 and discussed in Section 5.3.

Figures 2 to 6 show the same time series (pH), submitted for event detection to the FBIAD, ANM, CF, GARCH, and MAD methods, respectively. In each figure, points marked green represent true positives, points marked red represent false positives, and in blue, points are alluding to false negatives. In turn, Table 2 presents the data obtained from the confusion matrix of each method for the pH series, where it is possible to evaluate the numbers TP, TN, FP, and FN.

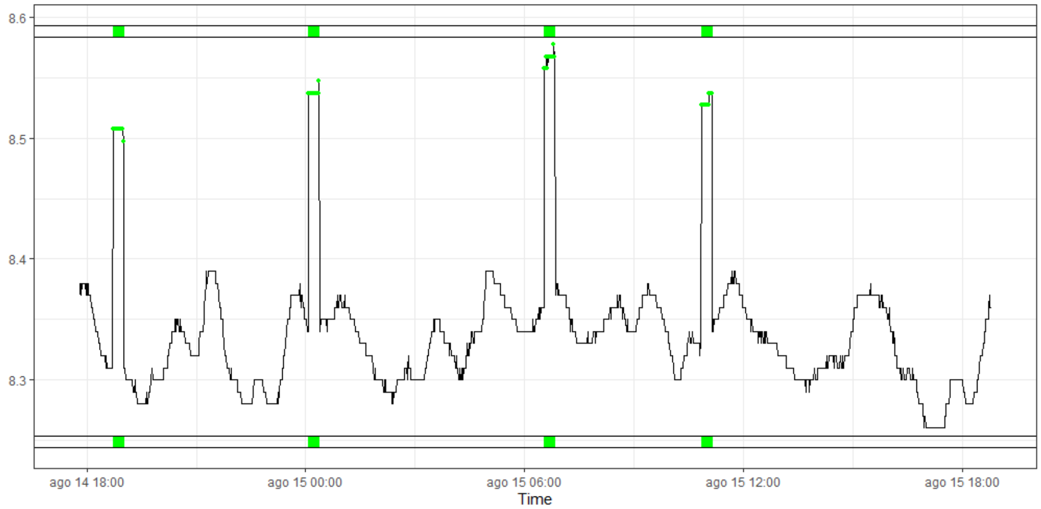

Figure 2: Detections with FBIAD: pH serie

As can be seen in Figure 2 and the confusion matrix (Table 2), for the pH series, the FBIAD method was able to correctly identify all events with no false positives or false negatives. All other methods showed lower results in the pH series, with a negative emphasis on the ANM method, which scored false positives throughout almost the entire series. However, it also marked all true events, Figure 4. The MAD method, on the other hand, does not seem to have any major problems visually, as there are not many false positives in Figure 6. However, when analyzing the Figure 6 and the confusion matrix (Table 2), it can be seen that the degree of true positive of MAD is low, reaching only 9 out of 72 real events.

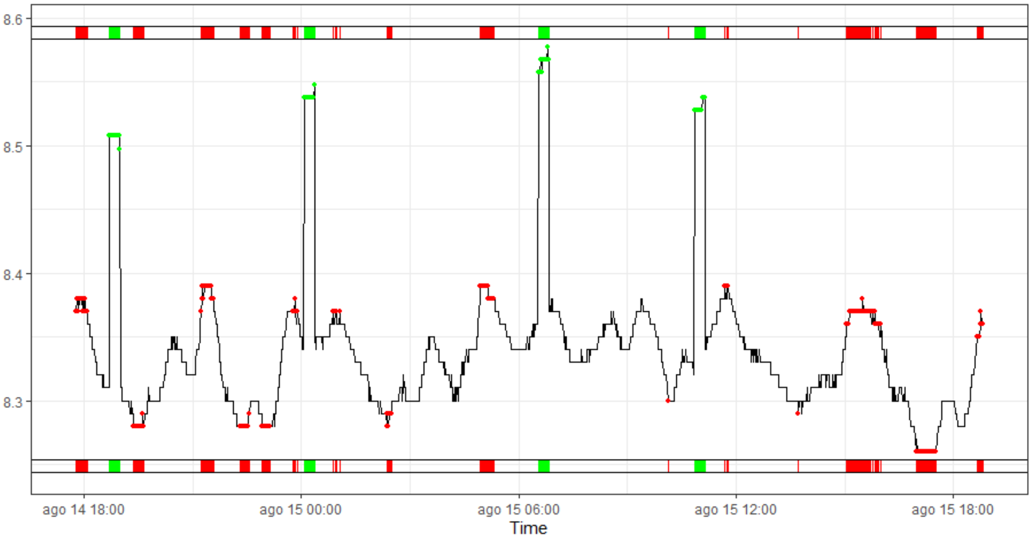

Figure 3: Detections with ANM: pH series

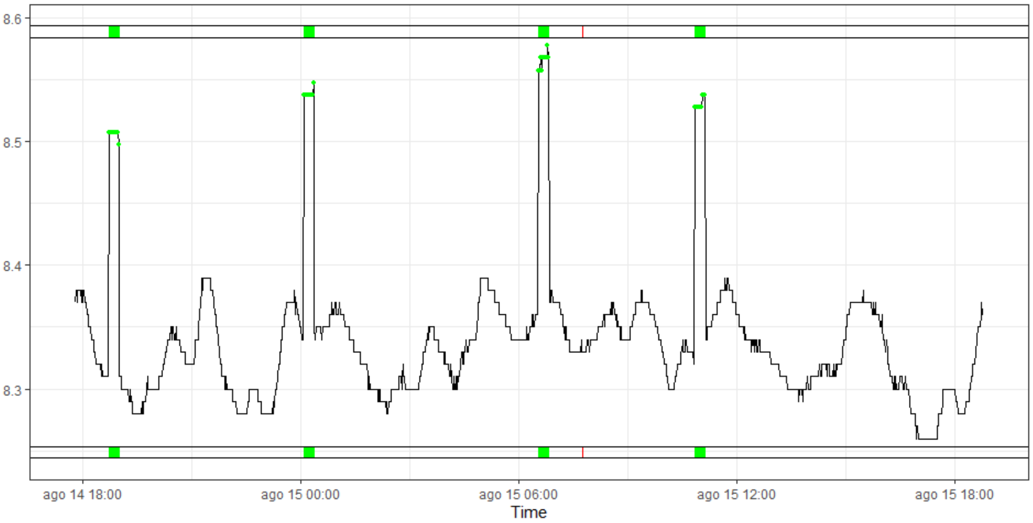

Figure 4: Detections with CF: pH series

Table 2: Confusion Matrix for pH series

Method	TP	TN	FP	FN
FBIAD	72	1429	0	0
ANM	72	1201	228	0
CF	72	1428	1	0
GARCH	68	1266	163	4
MAD	9	1416	13	63

5.3 Overall Detection Analysis

The results for each metric and dataset is presented in Tables 3, 4, and 5. The values were computed by the mean of the metric obtained by each method in the series. Finally, Table 6 shows the overall results of all series in all datasets evaluated.

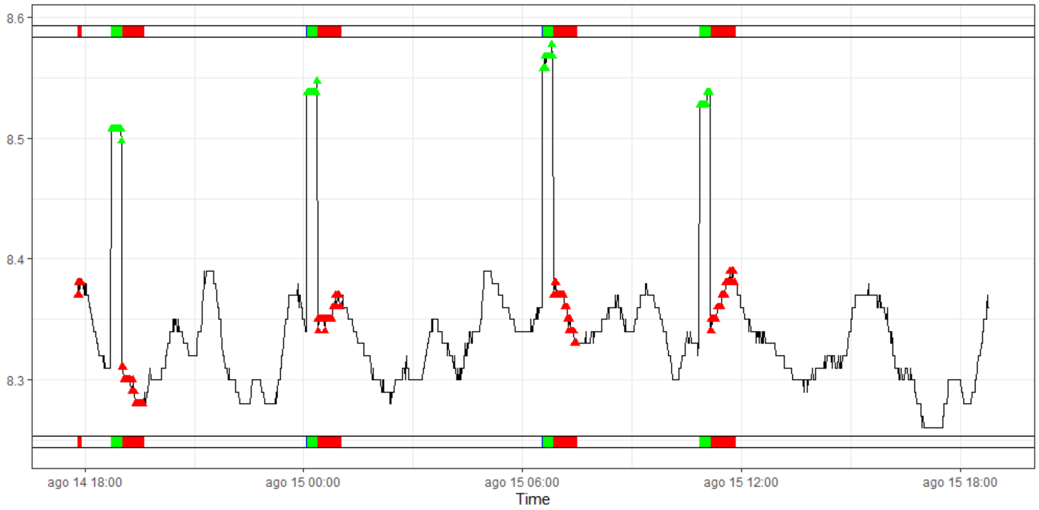

Figure 5: Detections with GARCH: pH series

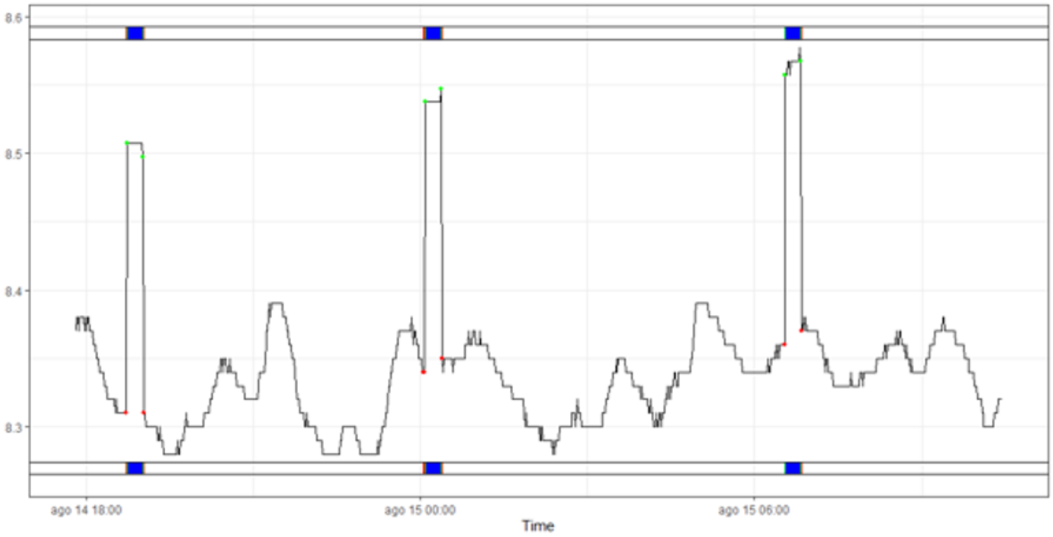

Figure 6: Detections with MAD: pH series

In all tables with metrics analysis, the best results for each metric are marked in bold in the respective column. When there is no difference in metric values between the best methods, with a statistical significance level of 5%, the respective columns have only underlined values.

Analyzing the summarized results in the detections for the *Gecco Challenge* series, it is observed that the FBIAD presents superior results for almost all metrics. It is possible to observe these results in Table 3. At least the FBIAD presented the same result as ANM in the recall metric.

As mentioned in the detailed analysis of the pH series, Section 5.2, and confirmed in Table 3, the FBIAD method obtained consistently superior results in the analysis of water quality. In this same dataset, another method with good results was the CF. In addition to being close to the FBIAD accuracy, it also had balanced results in the metrics precision, recall, and F_1. Finally, the worst accuracy was presented by ANM with 0.80, and in F_1, which represents the balance between precision and recall, MAD presented the worst result with 0.13.

For the *Numenta* and *Yahoo Labs* datasets, there is a balance in the detection results comparison between FBIAD and other methods evaluated. Each method performs better on one metric, loses on another, and draws on two. In the *Numenta* dataset, ANM performed best for recall with 0.90, but FBIAD obtained satisfactory results with 0.74.

Table 3: Summary Results for Gecco Challenge Dataset

Method	**Precision**	**Recall**	F_1	**Accuracy**
FBIAD	**0.40**	0.50	**0.60**	**0.95**
ANM	0.12	0.50	0.19	0.80
CF	0.28	0.36	0.46	0.93
GARCH	0.19	0.33	0.27	0.91
MAD	0.23	0.07	0.13	0.91

Table 4: Summary Results for Numenta Dataset

Method	**Precision**	**Recall**	F_1	**Accuracy**
FBIAD	0.03	0.74	0.04	1.00
ANM	0.00	**0.90**	0.01	0.80
CF	0.05	0.65	0.08	0.96
GARCH	0.01	0.36	0.03	0.92
MAD	0.04	0.65	0.10	1.00

It is relevant to note that the precision values of all methods were low for the Numenta series, as can be seen in Table 4. The low results in precision affected the F_1 metric, whose results were also generally low. Although none of the methods stood out, the FBIAD results were high for recall and accuracy. These results are significant for the FBIAD due to the type of series and data traffic in the cloud, representing a relevant domain area for time series research.

Deepening the comparison in the Yahoo dataset, the FBIAD and MAD may have a relative balance between precision and recall, leading them to obtain the best F_1 result for this dataset. The other methods, in turn, have very low precision values, except for CF. Low precision indicates the presence of many false positives, impairing the quality of detections. In addition to the balance between metrics, the FBIAD presented a higher value for precision and an accuracy of 100%. It is shown in Table 5.

Table 5: Summary Results for Yahoo Labs Dataset

Method	**Precision**	**Recall**	F_1	**Accuracy**
FBIAD	**0.78**	0.56	0.57	1.00
ANM	0.03	0.92	0.06	0.81
CF	0.44	0.61	0.48	0.98
GARCH	0.04	0.25	0.11	0.94
MAD	0.54	0.91	0.60	0.98

Table 6 consolidates the results of all series. It confirms the best result achieved by the FBIAD method. The recall was the only metric that the FBIAD did not obtain the best global result. However, in the joint analysis of the metrics precision, recall, and F_1, the method balanced and surpassed the others. Finally, again the FBIAD is among the best methods for metric accuracy.

Table 6: Overall Detection Analysis: all series

Method	**Precision**	**Recall**	F_1	**Accuracy**
FBIAD	**0.40**	0.60	**0.40**	0.98
ANM	0.05	**0.77**	0.06	0.80
CF	0.26	0.54	0.34	0.96
GARCH	0.08	0.31	0.14	0.92
MAD	0.27	0.54	0.28	0.96

5.4 Overall Performance Analysis

Table 7 shows the execution time in seconds for each method in different datasets. Faster methods have lower execution times. It is possible to observe a predominance of better performance of the CF, FBIAD, and ANM and worse performance of the MAD. On the other hand, the FBIAD method presented low execution times in all datasets, always among the three fastest methods.

Table 7: Execution Time (s) per Method and Dataset

Method	**Gecco Challenge**	**Nu-menta**	**Yahoo Labs**
FBIAD	0.10	0.21	0.08
ANM	0.15	0.13	0.09
CF	**0.04**	**0.07**	**0.02**
GARCH	0.63	0.47	0.33
MAD	1.86	2.77	2.04

As can be seen in Table 7, the FBIAD method had the second shortest execution time in two of the three datasets evaluated. For example, in the Gecco Challenge dataset, despite the best CF method, the FBIAD method outperformed ANM, GARCH, and MAD by 50%, 630%, and $1,860\%$, respectively.

These results of FBIAD performance are significant to reinforce its applicability, especially when considering that the execution time is essential for rapidly identifying anomalies. Finally, good execution times help the scalability of the method to larger datasets.

6 Conclusion

The results achieved found that the FBIAD performed satisfactorily in the evaluated scenarios, surpassing other methods in many scenarios and, at least, being equal in others. The FBIAD also presented a good performance at execution time, a fundamental aspect of decision-making response time based on data.

As discussed in Section 3, many methods available in the literature have specialized behavior in certain time series. On the other hand, the FBIAD can deal with series with both trending and change points, demonstrating its versatility and diversity of applications. The results of the experimental evaluations corroborated this intuition. The presented method (FBIAD) was always among the best results in the analyzed series that contained relevant domain areas for time series research, such as cloud data traffic, equipment sensor monitoring, and water quality monitoring.

Furthermore, the relevance of the FBIAD is reinforced when the methods are compared: from decomposition-based methods (ANM) to machine learning methods using convolutional neural networks (MAD). Furthermore, the quality of detections does not affect the computational performance of the method. FBIAD has consistently presented the best execution times.

In future work, the application of the FBIAD in the online detection of events in streaming data can be evaluated. In addition, its combination with other methods can be explored to expand its versatility further.

Acknowledgment

The authors thank CNPq, CAPES, and FAPERJ for partially sponsoring this research.

References

[1] Aggarwal, C. C. *Outlier Analysis*. Springer Science & Business Media, 2013.

[2] Aminikhanghahi, S. and Cook, D. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, 2017.

[3] Bach, K., Gundersen, O., Knappskog, C., and Öztürk, P. Automatic case capturing for problematic drilling situations. In *Lecture Notes in Computer Science*, volume 8765, pages 48–62, 2014.

[4] Bell, W. R., Holan, S. H., and McElroy, T. S. *Economic Time Series: Modeling and Seasonality*. CRC Press, 2018.

[5] Blázquez-García, A., Conde, A., Mori, U., and Lozano, J. A Review on Outlier/Anomaly Detection in Time Series Data. *ACM Computing Surveys*, 54(3), 2021.

[6] Carmona, R. *Statistical Analysis of Financial Data in R*. Springer Science & Business Media, 2013.

[7] Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 2009.

[8] Chen, H. and Zhang, N. Graph-based change-point detection. *Annals of Statistics*, 43(1):139–176, 2015.

[9] Dancho, M. and Vaughan, D. anomalize: Tidy Anomaly Detection. Technical report, https://CRAN.R-project.org/package=anomalize, 2020.

[10] Dao, M.-S., Zettsu, K., Pongpaichet, S., Jalali, L., and Jain, R. Exploring spatio-temporal-theme correlation between physical and social streaming data for event detection and pattern interpretation from heterogeneous sensors. In *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, pages 2690–2699, 2015.

[11] Ditzler, G., Roveri, M., Alippi, C., and Polikar, R. Learning in Nonstationary Environments: A Survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25, 2015.

[12] Esling, P. and Agon, C. Time-series data mining. *ACM Computing Surveys*, 45(1), 2012.

[13] Gabarda, S. and Cristóbal, G. Detection of events in seismic time series by time-frequency methods. *IET Signal Processing*, 4(4):413–420, 2010.

[14] Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), 2014.

[15] Gensler, A. and Sick, B. Performing event detection in time series with SwiftEvent: an algorithm with supervised learning of detection criteria. *Pattern Analysis and Applications*, 21(2):543–562, 2018.

[16] Gujarati, D. N. and Porter, D. C. *Basic Econometrics*. McGraw-Hill Publishing, 2008.

[17] Gupta, M., Gao, J., Aggarwal, C., and Han, J. Outlier Detection for Temporal Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 26(9):2250–2267, 2014.

[18] Guralnik, V. and Srivastava, J. Event Detection from Time Series Data. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, pages 33–42, New York, NY, USA. ACM, 1999.

[19] Habeeb, R. A., Nasaruddin, F., Gani, A., Targio Hashem, I., Ahmed, E., and Imran, M. Real-time big data processing for anomaly detection: A Survey. *International Journal of Information Management*, 45:289–307, 2019.

[20] Han, J., Pei, J., and Kamber, M. *Data Mining: Concepts and Techniques*. Elsevier, 2011.

[21] Haykin, S. O. *Neural Networks and Learning Machines*. Pearson Education, 2011.

[22] Khamassi, I., Sayed-Mouchaweh, M., Hammami, M., and Ghédira, K. Discussion and review on evolving data streams and concept drift adapting. *Evolving Systems*, 9(1), 2018.

[23] Larsen, R. J. and Marx, M. L. *An Introduction to Mathematical Statistics and Its Applications*. Pearson Education, 2017.

[24] Moritz, S., Rehbach, F., Chandrasekaran, S., Rebolledo, M., and Bartz-Beielstein, T. GECCO Industrial Challenge 2018 Dataset: A water quality dataset for the 'Internet of Things: Online Anomaly Detection for Drinking Water Quality'. Technical report, https://zenodo.org/record/3884398, 2018.

[25] Ogasawara, E., Martinez, L., De Oliveira, D., Zimbrão, G., Pappa, G., and Mattoso, M. Adaptive Normalization: A novel data normalization approach for non-stationary time series. In *Proceedings of the International Joint Conference on Neural Networks*, 2010.

[26] Ogasawara, E., Salles, R., Escobar, L., Baroni, L., Lima, J., and Porto, F. Online event detection for sensor data. In *Ibero-Latin American Congress on Computational Methods in Engineering*, Rio de Janeiro, RJ, 2021.

[27] Perelman, L., Arad, J., Housh, M., and Ostfeld, A. Event detection in water distribution systems from multivariate water quality time series. *Environmental Science and Technology*, 46(15):8212–8219, 2012.

[28] Ren, H., Xu, B., Wang, Y., Yi, C., Huang, C., Kou, X., Xing, T., Yang, M., Tong, J., and Zhang, Q. Time-series anomaly detection service at Microsoft. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3009–3017, 2019.

[29] Salles, R., Belloze, K., Porto, F., Gonzalez, P., and Ogasawara, E. Nonstationary time series transformation methods: An experimental review. *Knowledge-Based Systems*, 164:274–291, 2019.

[30] Salles, R., Escobar, L., Baroni, L., Zorrilla, R., Ziviani, A., Kreischer, V., Delicato, F., Pires, P. F., Maia, L., Coutinho, R., Assis, L., and Ogasawara, E. Harbinger: Um framework para integração e análise de métodos de detecção de eventos em séries temporais. In *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 73–84. SBC, 2020.

[31] Shumway, R. H. and Stoffer, D. S. *Time Series Analysis and Its Applications: With R Examples*. Springer, 2017.

[32] Taha, A. and Hadi, A. Anomaly detection methods for categorical data: A review. *ACM Computing Surveys*, 52(2), 2019.

[33] Takeuchi, J.-I. and Yamanishi, K. A unifying framework for detecting outliers and change points from time series. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):482–492, 2006.

[34] Talagala, P., Hyndman, R., Smith-Miles, K., Kandanaarachchi, S., and Muñoz, M. Anomaly Detection in Streaming Nonstationary Temporal Data. *Journal of Computational and Graphical Statistics*, 29(1):13–27, 2020.

[35] Truong, C., Oudre, L., and Vayatis, N. Selective review of offline change point detection methods. *Signal Processing*, 167, 2020.

[36] Ullah, W., Ullah, A., Haq, I., Muhammad, K., Sajjad, M., and Baik, S. CNN features with bi-directional LSTM for real-time anomaly detection in surveillance networks. *Multimedia Tools and Applications*, 80(11):16979–16995, 2021.

[37] Wang, H., Bah, M., and Hammad, M. Progress in Outlier Detection Techniques: A Survey. *IEEE Access*, 7:107964–108000, 2019.

[38] Wang, S., Minku, L., and Yao, X. A Systematic Study of Online Class Imbalance Learning with Concept Drift. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10):4802–4821, 2018.

[39] Webb, G., Hyde, R., Cao, H., Nguyen, H., and Petitjean, F. Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994, 2016.

[40] Wu, Y., Lin, Y., Zhou, Z., Bolton, D., Liu, J., and Johnson, P. DeepDetect: A Cascaded Region-Based Densely Connected Network for Seismic Event Detection. *IEEE Transactions on Geoscience and Remote Sensing*, 57(1):62–75, 2019.