Evaluating Temporal Bias in Time Series Event Detection Methods

Luciana Escobar[1], Rebecca Salles[1], Janio Lima[1], Cristiane Gea[1], Lais Baroni[1], Artur Ziviani[2], Paulo Pires[3], Flavia Delicato[3], Rafaelli Coutinho[1], Laura Assis[1], Eduardo Ogasawara[1]

Federal Center of Technological Education of Rio de Janeiro - CEFET/RJ
LNCC - Laboratório Nacional de Computação Científica, UFF - Universidade Federal Fluminense
{luciana.vignoli,rebecca.salles,lais.baroni}@eic.cefet-rj.br,
{janio.lima,cristiane.gea}@aluno.cefet-rj.br,
ziviani@lncc.br,{paulo.pires,fdelicato}@ic.uff.br,
{refaelli.coutinho,laura.assis}@cefet-rj.br, eogasawara@ieee.org

Abstract. The detection of events in time series is an important task in several areas of knowledge where operations monitoring is essential. Experts often have to choose the most appropriate event detection method for a time series, which can be complex. There is a demand for benchmarking different methods in order to guide this choice. For this, standard classification accuracy metrics are usually adopted. However, they are insufficient for a qualitative analysis of the tendency of a method to anticipate or delay event detections. Such analysis is interesting for applications in which tolerance for *close* detections is important rather than focusing only on accurate ones. In this context, this paper proposes a more comprehensive benchmark of event detection methods by including the analysis of temporal bias. For that, metrics based on the time distance between event detections and identified events are adopted. Computational experiments were conducted using real-world and synthetic datasets from *Yahoo Labs* and resources from the *Harbinger* framework for event detection. Adopting the proposed temporal bias metrics helped obtain a complete overview of the performance and general behavior of detection methods.

Categories and Subject Descriptors: H.1 [**Models and Principles**]: Miscellaneous; I.2 [**Artificial Intelligence**]: Miscellaneous; I.6 [**Simulation and Modeling**]: Miscellaneous

Keywords: Event Detection, Time Series, Benchmarking, Temporal Bias

1. INTRODUCTION

In time series analysis, it is often possible to observe a significant change in behavior at a certain point or time interval. Such behavior change generally characterizes the occurrence of an event [Guralnik and Srivastava, 1999]. An event can represent a phenomenon with defined meaning in a domain of knowledge. In this context, the event detection problem becomes particularly relevant, especially for applications based on sensor data analysis. Examples of such applications can be observed in chemistry, reflection seismic, and oil drilling and exploration, where monitoring of operations is essential.

Experts often have to choose the most appropriate event detection method for a time series and application. This choice can be a complex task as there are several detection methods in the literature. Each one presents different characteristics or assumptions about the analyzed time series. In addition, the nature of the events contained in the time series is often not known. Commonly, events detected in time series refer to anomalies and change points. In this case, events of certain types may be neglected or misidentified due to the wrong choice of a detection method. Failures in event identification can affect the decision-making process or lead to false positives. As a result, there is a credibility loss in control techniques and possible damage to applications. It indicates a demand for benchmarking the results of different methods for event detection. Such a process aims to guide the choice of suitable methods for detecting events of a time series in a particular application.

For benchmarking event detection methods, common classification quality metrics, such as balanced accuracy, F1, precision, and recall, are usually adopted as a base for comparison [Han et al., 2011; Lavin and Ahmad, 2016]. Such metrics focus mainly on a quantitative analysis of detection accuracy. On the other hand, inaccuracy in event detection does not always indicate a poor result. Inaccuracy in detecting an event can often result from its anticipated or delayed detection. In this context, the metrics usually adopted are insufficient for a qualitative analysis of the tendency of a detection method to anticipate or delay the detection of events. Such an analysis can be interesting for several applications that are tolerant of close detections rather than focusing on accurate ones [Singh and Olinsky, 2017].

This paper proposes a more comprehensive benchmark process for comparing methods for detecting events in time series. Such a process seeks to support the decision-making of the most appropriate method for a given application with a basis not only on the usual quantitative analysis of accuracy but also on a qualitative analysis of the temporal bias methods. In this context, we propose new metrics based on the time distance of event detections to the actual events previously identified in the time series for enabling the latter. These metrics help guide the choice of appropriate methods, considering their temporal bias. It enables the analysis of how close their detections come to the events, not just when they precisely detect them.

Computational experiments were conducted in order to benchmark different methods for event detection in time series. For that, the *Harbinger* framework was adopted. It integrates and enables the benchmarking of different state-of-the-art event detection methods. During the evaluation, eleven detection methods were selected among the available in *Harbinger*. These methods encompass the search for anomalies and change points through different statistical, volatility, proximity, and machine learning methods. For means of comparison using ground truth data, we have adopted real-world and synthetic datasets from *Yahoo Labs*. A total of 367 time series were analyzed. For each method, four metrics were computed to structure the comparative analysis and discussion of the results. Adopting the proposed detection temporal bias metrics helped obtain a more comprehensive benchmark of detection methods and general behavior.

The paper is organized as follows. Section 2 presents the theoretical background. Section 3 addresses the related works and highlights the contributions of the proposed benchmarking method against the existing literature. Section 4 describes the adopted methodology. Section 5 presents the experimental evaluation and its discussion. Conclusions and final considerations are presented in Section 6.

2. TIME SERIES AND EVENT DETECTION

Data that is recorded over time leads to a representation called a time series. When a series is examined, it is expected to find a temporal relationship, which has influenced the past data and may continue to influence the future. This behavior of a time series can change due to the occurrence of an event, so the detection of events becomes an important aspect in the mining of temporal data. This work addresses the detection of two distinct types of events: anomalies and change points.

2.1 Time Series

A time series is a sequence of observations collected over time. Generally, a time series y is considered a stochastic process, i.e., a sequence of n random variables $<y_1, y_2, \cdots, y_n>$[Carmona, 2013]. A time series observation is referred as y_i, indexed in time by $i = 1, \ldots, n$, such that y_1 represents the first observation and y_n portrays the most recent one. A subsequence of size p obtained from y

that ends at the i position can be represented by $seq_{i,p}(y)$, which is a continuous sequence of values $<y_{i-(p-1)}, y_{i-(p-2)}, \ldots, y_i>$, where $|seq_{i,p}(y)| = p$ and $p \leq i \leq |y|$. The moving average $\overline{y}_{i,p}$, at i position, of p terms in a time series y is calculated by the average of t_k observations from the subsequence $seq_{i,p}(y)$, as defined in Equation 1.

$$\overline{y}_{i,p} = p^{-1} \cdot \sum_{k=1}^{p} t_k \mid t_k \in seq_{i,p}(y), 1 \leq k \leq p \tag{1}$$

2.2 Trend Anomalies Detection

Anomalies are observations that highlight because they appear not to be generated by the same process as the other observations in the time series. Thus, anomalies can be modeled as isolated observations of the remaining data based on similarity or distance functions. They can be identified as $a(y)$ using the Eq. 2, where $Q_1(y)$ and $Q_3(y)$ are the first and third quartiles, respectively, and IQR is the interquartile distance [Gupta et al., 2014].

$$a(y) = \{i, \forall\ i \mid y_i \notin [Q_1(y) - 1.5 \cdot IQR(y), Q_3(y) + 1.5 \cdot IQR(y)]\} \tag{2}$$

In the time series context, there is a particular interest in detecting anomalies that may represent the occurrence of an event that escapes the trend inherent in the y generating process. Let $\hat{y}$ be an estimate of the y generating process, produced by adjusting a α model, with $\hat{y}_i = \alpha(y)_i$. Since ϵ is a time series of residues (*white noise*) got after removing the trend $\hat{y}$, trend anomalies of y are identified as $at(y)$ through Equation 3.

$$at(y) = a(\epsilon),\ \epsilon_i = y_i - \hat{y}_i \tag{3}$$

The literature presents several methods for detecting trend anomalies. Among them are those based on decomposition, adaptive normalization (AN), and KNN-CAD. The decomposition method adopts an approach that comprises decomposing the time series into three components: trend, seasonality, and the rest, on which the search for anomalies occurs [Gupta et al., 2014]. In AN, inertia is used to address non-stationary series by calculating the moving average [Salles et al., 2019]. KNN-CAD is an anomaly detection method based on the k-NN algorithm that adapts to non-stationarity in the data flow. According to [Gammerman and Vovk, 2007], KNN-CAD uses an approach involving distance from k nearest neighbors. The greater the distance between a point p and its kth-neighbor, the greater is the chance of being an anomaly.

2.3 Volatility anomalies detection

Most financial time series exhibit non-linear properties, which existing linear models can not capture, as the volatility of these series varies widely over time. Thus, there is a demand for the study of the volatility of time series. Econometric models appear to address the data's non-linearity, including stochastic volatility, such as ARCH and GARCH, the latter being the most well-known and applied [Carmona, 2013]. The works applied to the financial area associate volatility with risk, showing an event in time series.

GARCH-type models involve estimating volatility based on previous observations. GARCH is a nonlinear time series model, where a time series y_i is defined from the average component μ_i according to Equation 4. The noise sequence w_i is i.i.d. $N(0, 1)$, so that the conditional distribution of $\tilde{y}_i = y_i - \mu_i$, given $\tilde{y}_{i-1}, \tilde{y}_{i-2}, \ldots$ is $N(0, \sigma_i^2)$ [Carmona, 2013]. Such a model can be used as α for anomaly detection (Equation 3), or even its instantaneous volatility estimates can be subject to anomaly detection (Equation 2).

$$y_i = \mu_i + \sigma_i w_i \tag{4}$$

2.4 Change points detection

The methods of detecting change points aim to find in the time series the points or intervals in time that represent a transition between different states in a process that generates the time series data [Takeuchi and Yamanishi, 2006]. It is possible to define the change point detection as a hypothesis test 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 from a time series, $t, k \in \{i, \ldots, i+p\}$, and $t \leq k$, formally $H_0 : \forall\ t, k\ (t \neq k) \mid P_{y_t} = P_{y_k}$ e $H_A : \forall\ t\ \exists\ k\ (t \neq k) \mid P_{y_t} \neq P_{y_k}$, where P_{y_i} is the probability density function of the subsequence and k is a change point [Chen and Zhang, 2015].

One proposal developed for detecting change points in time series is based on the Exponentially Weighted Moving Average (EWMA). According to [Raza et al., 2015], EWMA is a method used to detect small changes in the moving average of a time series. The EWMA method uses a weighting constant (λ) that decides the importance of current and historical observations. Considering y_t as the observation value in time t, the EWMA model can be defined by Equation 5.

$$z_t = \lambda y_t + (1 - \lambda)\, z_{t-1}, \tag{5}$$

Where λ is a smoothing constant ($0 \leq \lambda \leq 1$), and z is the exponentially weighted moving average (EWMA). Analyzing Equation 5, z_0 is equal to the average of the initial data [Atashgar et al., 2020], and the smoothing constant λ is determined considering the change size to be detected [Assareh et al., 2015]. Therefore, EWMA assigns higher weights to recent data and lower weights to older data.

However, the seminal method of detecting change points (SCP) has become a reference in the literature [Guralnik and Srivastava, 1999]. Proposed to identify the moment of change in the time series behavior, the SCP is an iterative algorithm for adjusting a model to a time segment. In other words, this method utilizes a univariate approach. It follows a subsequent strategy, where models are fitted to data segments before and after the point, for any point in time. In addition, a likelihood criterion is used to determine if there is a new change point. There will be a new change point if the total of fit errors is smaller than when there is no change point.

Besides the advantages as mentioned earlier, the SCP method promoted the development of many approaches, such as the *ChangeFinder* (CF) [Takeuchi and Yamanishi, 2006], for example, which is composed of two phases. In the first phase, given a time series y, a model α is adjusted, resulting in $\hat{y}_i$ and identifying its anomalies from the residuals in the series s defined in the Equation 6. In the second stage, a new series $\overline{s}_p$ is produced, which is composed of the moving averages of s with p terms. The change points detection is reduced to anomalies detection in $\overline{s}_p$.

$$s_i = (\hat{y}_i - y_i)^2\ ,\ \hat{y}_i = \alpha(y)_i \tag{6}$$

2.5 Machine learning-based event detection methods

In contrast to the methods presented previously, the methods based on machine learning are not necessarily restricted to certain kinds of applications/problems. In the same way, they are not restricted to detecting a specific type of events, such as only anomalies, change points. The machine learning methods used to compare with the method proposed in this work are Feed-Forward Neural Network (NNET), Convolutional Neural Networks (CNN), Support Vector Machine (SVM), and Extreme Learning Machine (ELM), K-MEANS, and Long Short Term Memory (LSTM).

According to Haykin [2011], NNET can be seen as a set of mathematical tools used to learning the relationship between input and output variables. One of the advantages of its use resides in the flexibility of the distributed model defined by the weights of the network. Thus, the linear and non-linear decisions can be defined by adjusting the neural network configuration. Each node in a layer is an artificial neuron. Its input is modified by weight and added to all other inputs. The resulting value is passed through a transfer function to one or more neurons in the next layer [Riese and Keller, 2020].

One of the most well-known deep learning methods is CNN. It is a network formed basically by modules superimposed on convulsion and grouping layers. CNN considers the identification of the local relations between the analyzed data [Lim and Zohren, 2021]. A CNN comprises applying the convolution layers, composed of several neurons, to the input data. Combining the inputs of a neuron with the respective weights of each connection produces an output for the next layer. A matrix that contains the weights assigned to a neuron's connections represents the convolution filter.

The idea behind deep learning is to discover multiple levels of representation expecting high-level resources may represent a more abstract semantics of the data [Guo et al., 2017]. A CNN is an architecture composed of three distinct layers: an input layer, a convolutional layer, and a pooling layer, which reduces the size of the input data. The convolution layers in the datasets can be applied as an extractor of characteristics implicit in the data. The Equation 7 presents a convolution process, where g is the input layer, h is one of the k filters that a CNN. It is used to optimize the learning process at time t, $*$ is the convolution operator, and n is a hidden layer in the neural network.

$$(g * h)[n] \equiv \sum_{t=0}^{k} g[k-t]h[t] \tag{7}$$

LSTM networks have the same properties as conventional recurring networks. However, they can store information for long periods when processing a time sequence. The memory points of an LSTM network are called cells. Cells can carry information until the end of a sequence or identify information that the network should forget after some processing step[Greff et al., 2017].

Support Vector Machines (SVM) are supervised learning models used in classification tasks to analyze data and recognize patterns. This technique consists of mapping input data points to a high-dimensional resource space using kernel transformations and aiming for pattern recognition in the data. SVM seeks to locate classifiers with a greater distance between the support vectors from an infinite number of classifiers [Chauhan et al., 2019].

SVM-based classifier is suitable for handling large sets of attributes during the event detection and for quadratic optimization seeking to obtain a stable solution [Rahul and Choudhary, 2021]. SVM algorithms generally perform well on classification problems. Consider $\{a_1, b_1\}, \cdots, \{a_i, b_i\}$, such that $a \in R^n$ and $b \in \{-1, 1\}$, $i = 1, \cdots, N$, where N is the number of training instances, a is the input vector and b is the desired classification. The objective is to estimate a function $F : R^n \to \{-1 \text{ or } 1\}$, using the training examples and applying it in the test examples, in order to classify them correctly.

Extreme Learning Machine (ELM) is a learning algorithm for hidden layer feed-forward neural networks. ELM has a lower training error and a lower weight standard compared to other machine learning models. In this structure, parameters of the hidden nodes are generated randomly, and the weights of the output are calculated analytically [Tang et al., 2016]. Among the advantages compared to conventional gradient-based learning methods, [Ismaeel et al., 2015] highlights that ELM is suitable for most nonlinear activation functions. It can achieve a better-generalized performance compared to backpropagation. Huang et al. [2006] stand out some points about the performance of ELM: *(i)* extremely fast learning speed; *(ii)* better generalization performance, compared to gradient-based learning algorithms; and *(iii)* ability to work with differentiable and non-differentiable activation functions.

K-Means is applied to identify clusters in a dataset. The k clusters are identified based on the similarity. It is based on the Euclidean distance of the observations relative to the centroids of each cluster. The value k is the arbitrary number of clusters to be identified. The centroid of each cluster is defined by the algorithm iteratively. They correspond to the mean values of the observations within each cluster [Muniyandi et al., 2012].

3. RELATED WORK

Comparing different methods of detecting anomalies in time series has been extensively explored in the scientific community. In Chandola et al. [2009] presented a grouping of anomaly detection techniques in different categories, covering classification based on closest neighbors, grouping, and statistics.

Braei and Wagner [2020] reported a comparison of twenty anomaly detection methods in univariate time series. They were divided into three categories: statistical methods, classical machine learning methods, and methods using neural networks. The results gathered showed better performance in statistical methods. They evaluated that the properties of the time series affect the performance of the algorithms. To better understand the several data mining techniques for detecting anomalies, some hybrid approaches (a combination of two methods) were also analyzed. The literature indicates that such techniques provide better results and commonly overcome an isolated approach [Agrawal and Agrawal, 2015]. The survey described by Aminikhanghahi and Cook [2017] enumerates, categorizes, and compares several methods to detect change points in time series. The methods investigated include supervised and unsupervised algorithms.

Using distinct databases, Huang et al. [2012] compared the following machine learning methods: SVM, Least Square Support Vector Machine (LS-SVM), Proximal Support Vector Machine (PSVM), and ELM. The results show that ELM offers a unified learning platform with widespread attribute mapping, which can be directly used in regression and multiclass classification applications. Besides, ELM tends to present fewer optimization constraints, greater scalability, superior results for generalization performance, and lower learning speed concerning the others models. Gupta et al. [2021] analyzed different cases of COVID-19 in India (confirmed, killed, and recovery), using machine learning models: Random Forest, Linear Model, SVM, Decision Tree e Neural Network. The results revealed that the Random Forest presented a superior performance compared to the other evaluated models.

Based on data collected from diverse regions of Nanjing city, China, in 1999, Dong et al. [2003] used regression and machine learning techniques to predict the heating value of solid waste. The results show that the three-layer feed-forward neural network with backpropagation surpasses the multiple linear regression model. In Huang et al. [2006], ELM is proposed as a new learning algorithm for feed-forward neural networks of one hidden layer. The authors compare its performance with learning algorithms considered benchmarks, such as NNET and SVM. The ELM randomly chooses hidden nodes and determines the output weights. The results achieved report that the ELM can present superior generalization performance and greater learning speed. Aiming at the prediction of geological disasters. Zhang et al. [2019] proposed the combination of *(i)* Ensemble Empirical Mode Decomposition (EEMD) for decomposing micro-seismic signals, *(ii)* Singular Value Decomposition (SVD) for extracting values, and *(iii)* ELM for establishing a classification model. The results show that ELM performs better than other machine learning models (neural networks and support vector machines).

Delays related to event detections are found in studies involving Wireless Sensory Networks (WSNs). The main objective is to calculate the delay until the event is detected by an individual node and the delivery delay in a transmission network. In this context, Wang et al. [2011] presented a framework for capturing delays in detecting events in large-scale WSN networks. The average delay is obtained by averaging several hops in the network. In contrast, the soft delay threshold is defined as the delay when an event is detected with a probability p. On the other hand, no references were found that used the delay measure as a metric to evaluate an event detection concerning a real event in the series.

The works available in the literature generally specialize in detecting events of specific semantic and may neglect or misinterpret events of different types contained in a time series. These works commonly implement a limited number of detection methods, which may be unsuitable for specific applications. There is a demand for a study with different detection methods, thus identifying different types of events, enabling a comparative analysis of their detections. The presented study made a different comparison. In addition to covering different types of methods, it reported classic and new metrics, seeking to measure when the detection occurred in the neighborhood of the event.

4. PROPOSED METHODOLOGY

The proposed methodology aims to compare different event detection methods in times series through metrics that attempt to assess the quality and quantity of anomaly detection, change points, and both. The methodology seeks to make it possible to parameterize the methods to assess individual performance fair and consistent. Five steps were adopted in the proposed methodology: *(i)* data acquisition, *(ii)* methods choice, *(iii)* parameters definition, *(iv)* methods execution (framework), and *v)* metrics choice for evaluating results.

Data acquisition includes choosing and defining the datasets used in the experiments. In this process, we use synthetic and real-world datasets containing a labeled reference. This reference comprises a variable containing event label (*i.e.*, occurrence of events) to compare the performance of the methods. The next step involves choosing the methods used to assess the detection of events in time series. For the experimental evaluation, eleven methods were chosen, five based on machine learning, one based on proximity, and the other five on statistical techniques.

An optimal parameters setting reflects on the success of the event search. Consequently, the complexity of this step depends on each method and the size of its set of parameters. The adjustment of the moving average size must occur in the CF method. Also, it is possible to choose a model such as linear regression, ARIMA, AR, or ETS. ETS consists of an exponential smoothing model. The GARCH method should configure a more extensive set of parameters than the other methods. Table I provides all parameters, the respective values used in the experiments, their description, and to which methods they are associated. Two different window sizes were tested in the first two valid series of each dataset: $w = \{50, 100\}$. According to the results obtained (Table II), $w = 50$ presented the best results, and it was chosen as the window size in the computational experiments. For the other parameters, default values were used as shown in Table I.

The Harbinger framework [Salles et al., 2020] was used to run the methods. This framework can include methods aiming at a unified detection of different types of events in time series and the comparative performance analysis of different detection methods applied. Harbinger implements and combines the results of some of the main event detection methods available in the literature. Besides, this framework, beyond allowing the inclusion of new methods, also makes it possible to optimize its respective parameters. The detections can be evaluated through graphical visualization of the results and several quality metrics computations. Such characteristics allow the proper conduct of comparative analyzes between the different detection methods addressed.

To analyze the event detections in the approached time series, and perform the evaluation and comparison of the methods, four metrics were used: *(i)* F1, *(ii)* Balanced Accuracy, *(iii)* a priori, *(iv)* a posteriori. The metrics Accuracy Balanced and F1 were adopted in the evaluation of the results. Both metrics are widely employed in the literature and provide a good estimate of precision in detecting different methods. Detection bias was computed from distance-based metrics, which have been proposed in this work. With these metrics, detections are measured before (negative values - a priori distance) and after (positive values - a posteriori distance) of the reference event in the series.

Table I: Variables and their respective acronyms.

Variables and values	**Description**	**Method**
$w\ = 50$	window size	NA, SCP
$input\_size\ = 5$	input window size	SVM, ELM, NNET, CNN, LSTM
$mdl\ = linreg$ $m\ = 5$	model moving average size	CF
$alpha0\ = 0.9$ $beta\ = 0$ $l_c\ = 3$	maximum weighting weight attributed to the observation probability control bounds	EWMA
$n.train\ = 50$	training set size	EWMA, KNN, SVM, ELM, NNET, CNN, LSTM
$threshold\ = 1$ $k\ = 27$	anomaly bound number of candidate neighbors	KNN
$alpha\ = 3$	number of groups	K-MEANS
$mean.model\ = armaOrder$ $distribution.model\ = norm$ $variance.model\ = sGarch$	average model conditional density model variance model	GARCH

Table II: Results that support the window size choice

		NA		KNN		EWMA	
	Window size	1	2	1	2	1	2
Correct	50	2	<u>15</u>	<u>1</u>	1	2	6
	100	2	14	0	1	2	6
Priori Distance	50	9	<u>433</u>	24	<u>19</u>	35	89
	100	8	445	<u>9</u>	280	35	89

Most of the classification metrics are based on the confusion matrix, which holds four key-values of the results obtained in the model evaluation: *(1)* True Positive (*TP*); *(2)* False Negative (*FN*); *(3)* True Negative (*TN*); and *(4)* False Positive (*FP*). Although all the incorrect classifications are worrisome and can bring wrong decisions, the most alarming is the FP. FPs are classified as correct when they are not. It can cause major problems or catastrophes. The severity depends on the domain in which the model is implemented. Most classification metrics are derived from these four values (*TN*, *TP*, *FP*, *FN*). The balanced accuracy metric is used to balance the values obtained from the confusion matrix, whose average is obtained from the metrics of sensitivity and specificity, according to Equation 8. Sensitivity represents the amount of TP over $(TP + FN)$, while specificity describes the amount of TN over $(FP + TN)$.

$$balancedAccuracy = \frac{(sensitivity + specificity)}{2} \tag{8}$$

F1, as shown in Equation 11, consists of a harmonic average between two other metrics called precision and recall. The precision measure can be interpreted as the veracity of the truly detected events. In other words, it is a metric that evaluates among all the observations identified as positive, how many were correct (Equation 9). The recall represents how many of the actual events can be identified by a specific model. In short, such metric evaluates among all the positive occurrences marked as TP, how many were correct in fact (Equation 10).

$$precision = \frac{TP}{(TP + FP)} \tag{9}$$

$$recall = \frac{VP}{(VP + FN)} \tag{10}$$

The F1 value ranges between 0 and 1. The higher the value, the better the result is. F1 shows how accurate and robust the method is, i.e., how many observations it classified correctly and how many it failed to classify because they were difficult to label. However, when evaluating a method based only on F1, it must be paid attention to because it is inappropriate for an imbalance between positive and negative reference values.

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{11}$$

Metrics based on distance from the detected event to the real event called a prior and a posteriori have been proposed to measure temporal bias of detections. The before-mentioned metrics calculate the detections before (negative values) and after (positive values) of the reference event occurrence in time series. Let $E = \{e_1, e_2, \cdots, e_v\}$ be the events set detected by a method $m_h,\ h = 1, \cdots, |M|$, where M is the set with all methods, $D = \{D_1, D_2, \cdots, D_j\}$ is the set containing all selected datasets, and $R = \{re_1, re_2, \cdots, re_q\}$ is the set of real events present in the time series, given that $|R| = i$ represents the number of events present in the series, and $R_i,\ i = 1, \cdots, q$ each position of the event.

Hence, the set of detected events (E) was divided into two subsets: the first containing the event detections before each reference event in the series, and the second containing the detections located after the real event. Such subsets were defined as $\gamma = \{pr_1, pr_2, \ldots, pr_u\}$ and $\rho = \{ps_{u+1}, ps_{u+2}, \ldots, ps_v\}$, respectively. From these subsets, the distance metrics were calculated as follows:

(1) **A posteriori distance:** The metric named *posteriori* distance was determined in order to measure the absolute value (v_a) between the distance from a reference event (R_i), *i.e.*, real event present in the time series, to the later event detected by the method ($min(\rho)$), according to Equation 12.

$$post_i = v_a[min(\rho) - R_i] \tag{12}$$

(2) **A priori distance:** The metric named *priori* distance was determined to calculate the distance from a reference event (R_i) to the first event before detected by the method ($max(\gamma)$). It is shown in Equation 13.

$$prior_i = v_a[R_i - max(\gamma)] \tag{13}$$

The procedure for computing temporal bias of detections is described as follows. For each R event, the metrics *post* and *prior* are calculated. After that, the lowest value is computed. This lower value is assigned to the temporal bias metric. This process is repeated for each detected event until it reaches re_q, representing the last reference event in the series. Consequently, the result is a vector with the shortest detection distances with each event in the time series. The formalization of the temporal bias (bias, for short) is described by Equation 14.

$$bias_i = \begin{cases} post_i & \text{if} \quad post_i < prior_i \\ -prior_i & \text{otherwise} \end{cases} \quad \forall\ i \in R \tag{14}$$

5. EXPERIMENTAL EVALUATION

This section presents an experimental analysis conducted to compare the performance of different methods for detecting events. Such comparison aims to guide the most suitable detection methods to a time series and an application. The dataset created by *Yahoo Labs* was selected to evaluate the methods, from now on, referred to as *Yahoo*. This dataset consists of observations collected by hour containing anomalies identified manually by editors Webscope [2015]. Part of the data is synthetic, while the other part is based on service traffic. The dataset is divided into four *benchmarks*. The first one, named $A1$, is composed of real-world data, and the other three, called $A2$, $A3$, and $A4$, are comprised of synthetic data. $A1$ and $A2$ have three attributes: *(i)* the sequence in the series (timestamp), *(ii)* the observed value (value), and *(iii)* the indication of the presence/absence of an anomaly. In addition to these three attributes, $A3$ and $A4$ have six more. These attributes bring information about: *(iv)* noise, *(v)* trend, and *(vi – viii)* seasonality, and also labeling whether there is or not a change point. $A1$ is the benchmark with the highest number of detected events, with 25 out of 1416 ratings being labeled as such. $A2$ has five events in a total of 1421 observations. $A3$ and $A4$ have 1680 observations each, containing nine and eight labeled events, respectively.

Event detection processes were run for each time series under study. Therefore, the methods described in Section 2 were used, covering the event detection of different types. In the CF method, except when stated otherwise, the linear regression model is adopted to enable a fair comparison with the results produced by the SCP method. The experiments were performed in a shared computer, whose configuration consists of an Intel Core i7 processor with 16 cores, 128 GB of RAM, and the Ubuntu 20.04 operating system.

5.1 Detection Performance Comparison

Table III presents a comparison of quality metrics for event detections produced by the different detection methods under study in this work. Metrics F1 and balanced accuracy were selected as the basis for quantitative analysis of the detection accuracy of these methods. In the referred table, it is possible to observe, in a unified way, the performance of the methods regarding the time series contained in the *Yahoo* datasets. The best results for each dataset are underlined.

Table III: Comparison of the event detection quality with F1 and Balanced Accuracy metrics.

Method	**F1**				**Balanced accuracy**			
	A1	A2	A3	A4	A1	A2	A3	A4
AN	<u>0.58</u>	<u>1.00</u>	<u>0.67</u>	0.13	0.81	<u>1.00</u>	0.75	0.70
GARCH	0.05	0.06	0.01	0.02	0.66	0.71	0.48	0.48
EWMA	0.28	0.50	0.51	<u>0.50</u>	0.64	0.75	0.67	0.67
KNN-CAD	0.07	0.08	0.14	0.13	0.59	0.65	0.70	0.73
SCP	0.03	0.11	0.01	0.01	0.52	0.95	0.48	0.47
CF	0.30	0.72	0.20	0.08	0.89	<u>1.00</u>	0.79	0.67
K-MEANS	0.29	0.50	0.12	0.02	0.83	0.62	0.53	0.49
SVM	0.07	0.04	0.12	0.07	0.80	0.72	<u>0.96</u>	0.94
ELM	0.06	0.07	0.03	0.08	0.62	0.96	0.52	0.94
NNET	0.11	0.05	0.12	0.08	<u>0.94</u>	0.96	<u>0.96</u>	<u>0.95</u>
CNN	0.06	0.04	0.01	0.08	0.77	0.84	0.49	0.92
LSTM	0.09	0.08	0.11	0.10	0.84	0.96	0.83	0.83

By analyzing the results in Tab. III, it is possible to observe that based on the F1 metric, better detection performances were obtained by AN in almost all datasets. Nonetheless, the performances of EWMA, CF, and K-MEANS methods also stand out. It is noted that the methods above are specialized in detecting different types of events, including change points (EWMA, CF) and trend anomalies (AN, K-MEANS). This fact suggests that these methods can be complementary, and their combination can better understand the events. Besides, as the EWMA, AN, and CF methods are based on moving averages, this may indicate that the events in Yahoo datasets can be affected by the inertia of the data-generating phenomenon.

The balanced accuracy metric provides an analysis of the detection performance of the methods from another perspective. In this context, the methods based on machine learning NNET, ELM, SVM, CNN, and LSTM obtained better overall performance in all datasets. Among these, SVM and NNET are highlighted in the A3 dataset, whose data present seasonality. The performances of the AN and CF methods stood out in the A2 dataset while also performing well in the A1 dataset, composed of real-world data. The characteristic methodology of the methods that obtained the best performances indicates that the events in Yahoo can be mostly trend anomalies, except for the CF method. In this scenario, specialized methods for other types of events may not be the most suitable. Despite this, combining the methods above for performing event detection may offer more relevant information, including the semantics of change points detected by the CF method. Note that detection methods based on GARCH and SCP, which are specialized in anomalies of volatility and change points, respectively, found it challenging to detect events in Yahoo data.

The results suggest that the comparative analysis of the performance of detection methods based on well-defined metrics can guide more appropriate methods. Also, it can designate the need to combine methods with different methodologies. However, the metrics usually adopted, such as F1 and accuracy, focus mainly on a quantitative analysis of a method to "hit" accurately an event occurrence. Nonetheless, the inaccuracy can be interesting in specific contexts. It can often result from event anticipated or a delayed detection.

5.2 Discussion on Temporal Bias of Detections

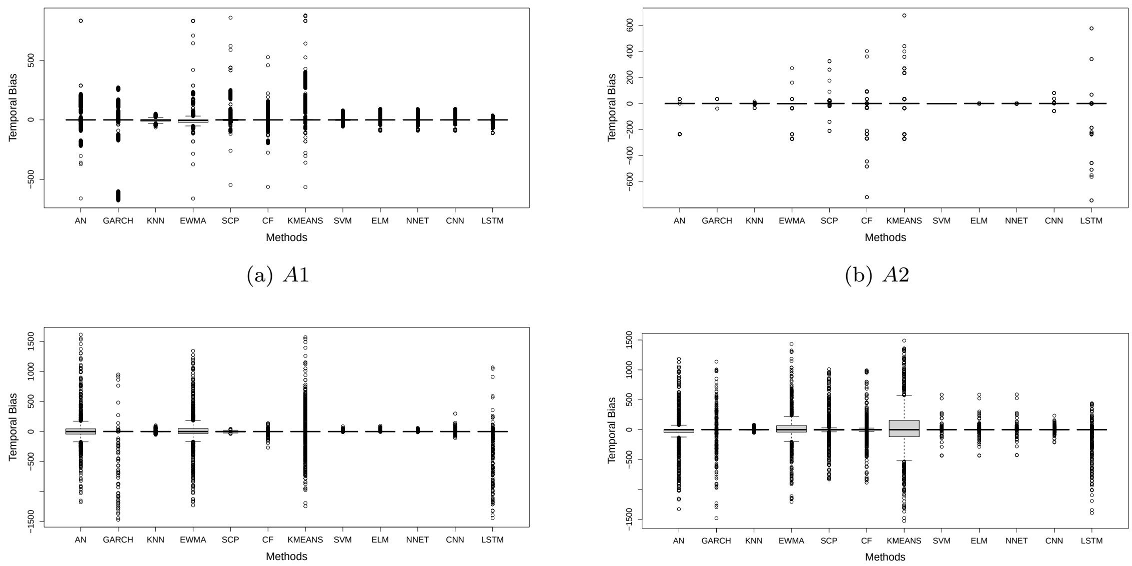
The comparative analysis of different event detection methods based only on accuracy is insufficient for a qualitative analysis of the tendency of a method to anticipate or delay the event detection. Such analysis can be interesting for several applications in which tolerance for comparing event detection methods is acceptable. Thereby, the method comparison process can benefit from a distance-based metric that indicates possible anticipations or delays in event detection.

In this context, the a *priori* and a *posteriori* distances introduced in Section 4 can be used for analysis. The before-mentioned distances were calculated for each event in the time series of the datasets under study. The minimum value between the two measurements refers to the nearest detection distance, from now on called the temporal bias. A minimum a *priori* distance results in an anticipation, while a minimum a *posteriori* distance results in a delay. Fig. 1 shows the distributions of temporal bias produced by the different methods applied to datasets $A1$, $A2$, $A3$, and $A4$.

Based on Fig. 1, it is possible to observe that the distributions of temporal bias are overall close to zero, which means the detections, if not accurate, are close to the actual events contained in the time series. However, we see many outliers, especially for datasets A1, A3, and A4. We also note that the machine learning-based methods, except K-MEANS and LSTM, could produce fewer outliers than other methods over all datasets. Still, we observe methods that display a wider distribution regarding their temporal bias, such as EWMA (A1, A3, and A4), AN (A3 and A4), KNN (A1), and K-MEANS (A4), where the last presents the more considerable variance of temporal bias.

Moreover, when analyzing Fig. 1, we observe that the better-evaluated methods based on the accuracy metrics presented in Tab. III may not be the ones whose detections are consistently closer to the actual time series events. That is the case for AN, EWMA, and K-MEANS, for example, which produced high F1 and balanced accuracy scores in all datasets but presented higher amounts of outliers and larger variance of temporal bias based on Fig. 1. In this context, the analysis of the distributions of temporal bias may complement the metrics presented in Section 5.1, giving an overview of how close the detections produced by each particular method are to the events of a time series. In that case, methods that can produce detections consistently closer to actual events, with narrow distributions, smaller variance, and lower outlier rates for temporal bias may be preferable and better evaluated.

On the other hand, one might also be interested to know the tendency of a particular detection



(a) $A1$ (b) $A2$

(c) $A3$ (d) $A4$

Fig. 1: Distribution of temporal bias for the datasets under study.

method to produce biased detections. Alternatively, in other words, the tendency of a method to anticipate or delay detections of a certain time series events. In this context, Tab. IV presents the mean and skewness of the distributions of the temporal bias of Fig. 1. A negative skewness coefficient indicates a tendency to delay event detections, while a positive skewness coefficient indicates a tendency to anticipate them.

Table IV: Mean and skewness of distributions of temporal bias.

Method	**A1**		**A2**		**A3**		**A4**	
	Mean	Skewness	Mean	Skewness	Mean	Skewness	Mean	Skewness
AN	16.33	2.45	-6.30	-5.66	23.54	1.31	-21.58	-0.17
GARCH	-17.85	-3.29	0.50	3.50	-34.82	-4.18	-6.69	-0.87
EWMA	2.33	5.63	-8.77	-3.96	12.58	0.30	13.96	0.39
KNN-CAD	-2.68	0.20	-0.45	-6.52	0.62	2.39	0.48	1.71
SCP	15.15	3.92	2.86	3.50	-0.11	0.07	11.44	0.82
CF	1.42	0.12	-17.25	-3.95	-0.21	-2.43	0.34	0.72
K-MEANS	51.56	2.37	-3.64	1.28	6.96	0.63	18.30	0.16
SVM	1.04	1.64	-0.70	-0.49	1.10	6.78	1.42	3.09
ELM	3.02	1.17	-0.05	-1.04	1.32	7.05	1.44	2.04
NNET	1.50	0.62	-0.26	-1.61	1.05	6.12	1.57	3.12
CNN	3.15	37.00	0.55	6.17	5.02	28.22	4.46	23.10
LSTM	-5.97	-40.10	-17.47	-22.42	-145.06	-30,06	-90.05	-27.96

Given that the distributions of temporal bias displayed in Fig. 1 are mainly centered around zero, the means presented in Tab. IV, indicate the effect of the outliers and the possible presence of a skewed distribution. For the A1 dataset, we can observe in Tab. IV that almost all methods resulted in positive skewness coefficients, which means that most of their detections anticipated the actual events contained in the dataset. The same can be said regarding the datasets A3 and A4. The exceptions correspond to the methods GARCH and LSTM (A1, A3, and A4), CF (A3), and AN (A4). In that case, if one is mainly interested in preceding events in those datasets, they might prefer to avoid these methods. For the A2 dataset, on the other hand, most methods resulted in negative skewness coefficients, meaning that most of their detections delayed the actual events. The exceptions were posed by GARCH, SCP, K-MEANS, and CNN. In that case, one might prefer to choose among these methods to anticipate events in A2. Furthermore, based on the skewness coefficients, LSTM detected events with temporal bias in all Yahoo datasets and, together with CNN, produced the highest values of skewness (either negative or positive).

The analysis of results presented in Tab. III, Fig. 1, and Tab. IV can be complementary, giving an overview of the performance and behavior of each detection method applied to the dataset under study. The combined analysis of detection accuracy measures, distributions of temporal bias, and skewness coefficients provides a more thorough benchmarking process. Such benchmarking analyzes detection accuracy and the temporal bias of different event detection methods.

6. FINAL REMARKS

This work presented a comparative analysis of different event detection methods. It includes quantitative analysis of both detection accuracy and temporal bias of different event detection methods. For that, in addition to metrics commonly found in the literature, metrics related to the temporal bias, *i.e.*, detection distance concerning the event were adopted. The latter are important for an analysis focused on anticipated or delayed of event detection. The metrics usually adopted detection accuracy. An evaluation was carried out on datasets containing synthetic and real-world data. It became possible to verify that temporal bias metrics help obtain a more thorough understanding of the performance and behavior of each detection method. It provides opportunities for choosing appropriate methods for a particular application. The benchmarking of different event detection methods using different metrics tends to avoid negligence or misidentification of events that can harm applications that depend on event monitoring. The extension of this study leads us to the event prediction problem based not only on detection techniques but also on solutions in the analysis and prediction of time series and machine learning.

Acknowledgments

The authors would like to thank CNPq, CAPES (finance code 001), and FAPERJ for the partial funding of the research.

REFERENCES

Agrawal, S. and Agrawal, J. Survey on anomaly detection using data mining techniques. In *Procedia Computer Science*. Vol. 60. pp. 708–713, 2015.

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

Assareh, H., Smith, I., and Mengersen, K. Change point detection in risk adjusted control charts. *Statistical Methods in Medical Research* 24 (6): 747–768, 2015.

Atashgar, K., Rafiee, N., and Karbasian, M. A new hybrid approach to panel data change point detection. *Communications in Statistics - Theory and Methods*, 2020.

Braei, M. and Wagner, S. Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art. *arXiv:2004.00433 [cs, stat]*, Apr., 2020.

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

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

Chauhan, V., Dahiya, K., and Sharma, A. Problem formulations and solvers in linear SVM: a review. *Artificial Intelligence Review* 52 (2): 803–855, 2019.

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

Dong, C., Jin, B., and Li, D. Predicting the heating value of MSW with a feed forward neural network. *Waste Management* 23 (2): 103–106, 2003.

Gammerman, A. and Vovk, V. Hedging predictions in machine learning. *The Computer Journal* 50 (2): 151–163, 2007.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems* 28 (10): 2222–2232, 2017.

Guo, T., Dong, J., Li, H., and Gao, Y. Simple convolutional neural network on image classification. In *2017 IEEE 2nd International Conference on Big Data Analysis, ICBDA 2017*. pp. 721–724, 2017.

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.

Gupta, V., Gupta, A., Kumar, D., and Sardana, A. Prediction of COVID-19 confirmed, death, and cured cases in India using random forest model. *Big Data Mining and Analytics* 4 (2): 116–123, 2021.

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. ACM, New York, NY, USA, pp. 33–42, 1999.

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

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

Huang, G.-B., Zhou, H., Ding, X., and Zhang, R. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 42 (2): 513–529, 2012.

Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing* 70 (1-3): 489–501, 2006.

Ismaeel, S., Miri, A., and Chourishi, D. Using the Extreme Learning Machine (ELM) technique for heart disease diagnosis. In *2015 IEEE Canada International Humanitarian Technology Conference, IHTC 2015*, 2015.

Lavin, A. and Ahmad, S. Evaluating real-time anomaly detection algorithms - The numenta anomaly benchmark. In *Proceedings - 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA 2015*. pp. 38–44, 2016.

Lim, B. and Zohren, S. Time-series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 379 (2194), 2021.

Muniyandi, A., Rajeswari, R., and Rajaram, R. Network anomaly detection by cascading k-Means clustering and C4.5 decision tree algorithm. In *Procedia Engineering*. Vol. 30. pp. 174–182, 2012.

Rahul and Choudhary, B. An Advanced Genetic Algorithm with Improved Support Vector Machine for Multi-Class Classification of Real Power Quality Events. *Electric Power Systems Research* vol. 191, 2021.

Raza, H., Prasad, G., and Li, Y. EWMA model based shift-detection methods for detecting covariate shifts in non-stationary environments. *Pattern Recognition* 48 (3): 659–669, 2015.

Riese, F. and Keller, S. Supervised, semi-supervised, and unsupervised learning for hyperspectral regression. *Advances in Computer Vision and Pattern Recognition*, 2020.

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

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)*. SBC, pp. 73–84, 2020.

Singh, N. and Olinsky, C. Demystifying Numenta anomaly benchmark. In *Proceedings of the International Joint Conference on Neural Networks*. Vol. 2017-May. pp. 1570–1577, 2017.

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.

Tang, J., Deng, C., and Huang, G.-B. Extreme Learning Machine for Multilayer Perceptron. *IEEE Transactions on Neural Networks and Learning Systems* 27 (4): 809–821, 2016.

Wang, Y., Vuran, M., and Goddard, S. Analysis of event detection delay in wireless sensor networks. In *Proceedings - IEEE INFOCOM*. pp. 1296–1304, 2011.

Webscope, Y. *Labeled anomaly detection dataset*, 2015.

Zhang, J., Jiang, R., Li, B., and Xu, N. An automatic recognition method of microseismic signals based on EEMD-SVD and ELM. *Computers and Geosciences* vol. 133, 2019.