Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full length article

Fusing anomaly detection with false positive mitigation methodology for predictive maintenance under multivariate time series

David López [a,c], Ignacio Aguilera-Martos [a,c], Marta García-Barzana [d], Francisco Herrera [a,c], Diego García-Gil [b,c], Julián Luengo [a,c,*]

[a] Department of Computer Science and Artificial Intelligence, University of Granada, Spain
[b] Department of Software Engineering, University of Granada, Spain
[c] Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI), University of Granada, 18071, Granada, Spain
[d] ArcelorMittal Global R&D, New Frontier, Digital Portfolio, Spain

ARTICLE INFO

Keywords:
Anomaly detection
Outlier detection
False positive mitigation
Deep learning
Multivariate time series
Predictive maintenance

ABSTRACT

Anomaly detection aims to identify observations that differ significantly from the majority of the data. Time series, which are data with a temporal component, is often used for anomaly detection. Identifying anomalies is not perfect and may produce many false positives, which labels standard data as anomalous. In this context, false positive mitigation is the task of reducing the number of false positives tagged by the anomaly detector, and thus both problems are closely linked. Moreover, current techniques for false positive mitigation are ad-hoc solutions for specific data sets. In this paper, we propose a novel two-stage methodology for Multivariate Anomaly Detection for Time Series and False Positive Mitigation, namely $FADFPM$ methodology, which creates the fusion of two learning models. The first stage is a multivariate anomaly detection stage. The second stage consists of training a new classifier on the false and true positives from the anomaly detector, which refines the observations labeled as anomalous by the anomaly detector to obtain more accurate and higher-quality results. Experiments using two benchmark data sets, as well as a real-world case study have shown the performance and validity of the proposal.

1. Introduction

To keep track of a system, a set of data is generated that reproduces the behavior of such a system. When the system begins to fail for some reason, anomalies begin to appear in the data. Therefore, detecting these anomalies in the data allows knowing if the system is facing a failure [1–3]. The task of finding observations that differ greatly from the rest of the data is known as anomaly detection [1]. Observations that share this unusual behavior are typically referred to as outliers or anomalies. There is a wide variety of domains in which anomaly detection is useful, such as intrusion detection [4], sensor networks [5], credit-card fraud detection [6], health care [7] or industrial anomalies [8]. For example, in a predictive maintenance scenario, detecting anomalous behavior in a motor may indicate that it is close to failure, therefore, detecting anomalies before it breaks down can greatly reduce repair costs. Anomaly detection is becoming increasingly important due to the relevance of the benefits it brings and the huge variety of domains in which it can be applied. Since anomaly detection tasks are typically tracking a system's behavior over time, data is rarely static. The most common scenario is to face a time component in the data.

Time series are data that have a temporal component, i.e., each observation is not independent, but related in time. Time series can be classified into univariate, which has only one feature, and multivariate, which has more than one feature. Most proposals for time series focus on univariate time series so there are not many alternatives for multivariate problems. Time series will show different values in different time periods without necessarily indicating an anomaly. For example, an engine may have a higher temperature than normal at one instant in time, but such overheating may simply be due to a higher workload and not to a failure. Learning the behavior of a time series can be used to analyze future data and thus anticipate a failure and prevent potential damage [9,10]. Within a time series, an anomaly is usually determined by several consecutive anomalous values in time [11], which is a major drawback for traditional anomaly detection problems since they deal with anomalies without taking into account the time component [12].

[*] Corresponding author at: Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI), University of Granada, 18071, Granada, Spain.

E-mail addresses: derwey@correo.ugr.es (D. López), nacheteam@ugr.es (I. Aguilera-Martos), marta.garcia-barzana@arcelormittal.com (M. García-Barzana), herrera@decsai.ugr.es (F. Herrera), djgarcia@ugr.es (D. García-Gil), julianlm@decsai.ugr.es (J. Luengo).

https://doi.org/10.1016/j.inffus.2023.101957
Received 28 March 2023; Received in revised form 12 July 2023; Accepted 28 July 2023
Available online 3 August 2023

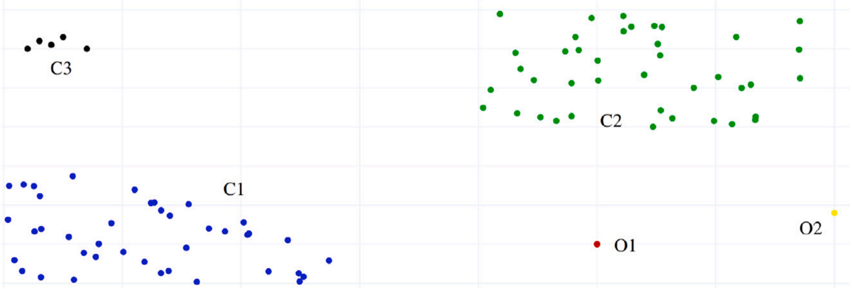


Fig. 1. Illustration of anomalies in a two-dimensional data set.

Considering the characteristics of the anomaly detection problems described above, it is appropriate to make use of an algorithm that takes into account the time component in the time series anomaly detection problems. When facing multivariate time series, current state-of-the-art is populated by recurrent neural networks such as Long Short-Term Memory networks (LSTMs) and Gated Recurrent Unit networks (GRUs) [13–15]. A recent and good performance algorithm is the *temporal convolutional networks* (TCN) [16,17]. However, although these approaches are models capable of obtaining quality results, they are not exempt from the presence of false positives. Therefore, in data sets that are difficult to analyze, a subsequent false positive mitigation stage may be applied to improve the quality of the results as stated in [18–20]. Nevertheless, current false positive mitigation techniques are ad-hoc solutions and are limited to the data sets they analyze.

In this paper, we tackle the false positive problem in anomaly detection scenarios by proposing a Fusing Anomaly Detection with Positive Mitigation methodology, namely $FADFPM$ methodology, for multivariate time series anomaly detection problems. It can effectively detect and reduce false positives in supervised data sets. Below, we provide an overview of how the $FADFPM$ methodology works in two stages:

1. The first stage is to train a multivariate anomaly detection model.
2. The second stage aims to reduce false positives. A classification model for tabular data is built from the predictions made.

We study the performance of $FADFPM$ methodology in two data sets commonly used in the literature (a constructed data set from SKAB [21] and a data set extracted from $kaggle$ [22]), as well as in a real-world case of study using sensor data provided by the company $ArcelorMittal$. The results achieved show that the methodology proposed in this paper is capable of achieving better results than the current state-of-the-art by reducing the number of false positives of the time series anomaly detection method tested.

The remainder of this paper is organized as follows: Section 2 presents the concept and the current state of anomaly detection problem and false positive mitigation problem. Section 3 explains in detail the proposed $FADFPM$ methodology followed by the study performed for this paper. Section 4 details the data sets analyzed and the framework setup. Section 5 shows the experiments carried out to assess the performance of the methodology in the benchmark data sets. Section 6 shows the performance of the methodology in a real-world case of study. Finally, Section 7 concludes the paper.

2. Predictive maintenance based on the anomaly detection problem

Predictive Maintenance accounts on machine learning models to determine when maintenance actions are necessary. It is based on continuous monitoring of a machine or process, feeding the anomaly detection model, and allowing maintenance to be performed only when it is needed. As such, the accuracy of the anomaly prediction model is key to ensuring precise maintenance scheduling and will be covered in the rest of this section.

In this Section, we describe the anomaly detection and false positive mitigation problems as well as the different evaluation metrics available for them. Section 2.1 details the problem of anomaly detection and its characteristics. Section 2.2 briefly summarizes the state-of-the-art techniques devised to deal with this problem. Examples of applications of interest are also included. Section 2.3 explains the evaluation metrics for anomaly detection problems and their characteristics. Finally, Section 2.4 describes the false positive mitigation problem in anomaly detection scenarios.

2.1. Anomaly detection fundamentals

An anomaly is an observation that does not follow the same pattern as the rest of the data. Fig. 1 depicts a graphical representation of anomalies in a two-dimensional data set. Clusters C1 and C2 are composed of normal observations since practically all of the points belong to these two regions. Cluster C3 contains very few observations since it is an anomalous cluster. Observations O1, and O2 are completely isolated and therefore are anomalous instances [1].

In the literature, we may find three different categorizations of anomalous instances [23]:

- Point anomaly: This is the most frequent scenario in anomaly detection. The anomalous instances are completely isolated from the rest. In Fig. 1, O1 and O2 are point anomalies.
- Collective anomaly: The anomaly is a mixture of several anomalous instances. For example, detecting a credit card theft may involve detecting multiple bank account extracts.
- Contextual anomaly: An instance that is not anomalous could be anomalous within a given context. For example, if we measure the temperature of an engine in a range from 50 to 120 degrees. A temperature of 80 degrees seems to be completely normal, but if that value is given when the engine has no working load, the estimated temperature should be lower.

Anomalies are related to noise, but the two concepts should not be confused. Noise has the same behavior described in Fig. 1, but, noise is of no interest to the data analyst while anomalies are. Noise is produced by an alteration in the data, therefore, it does not reflect the original distribution of the data. Moreover, noise damages the quality of the data and those observations should be either fixed or removed [24–26]. Anomalies are valuable information that has to be detected, extracted, and analyzed.

Anomaly detection is used in a wide variety of domains such as sensor networks [5], intrusion detection [4], industrial anomalies [8], credit-card fraud detection [6], health care [7], and much more [23]. The great number of domains that involve the anomaly detection problem, and the increasing number of sensors in all fields are making anomaly detection gain in popularity.

Regarding the output of an anomaly detection algorithm, it can be of two different types [1]:

- Labels: The return value is a binary label for each instance indicating which instances are normal and which are anomalies.
- Scores: The return value is an anomaly score for each instance. The score indicates the probability of the instance being anomalous, or it can also serve as a quantitative measure of the degree of anomaly in the instance. Which scores are anomalous should be studied as well.

As mentioned above, we focus on anomaly detection for time series. A time series is a collection of data that follows a chronological order. Some of its characteristics are huge size, high dimensionality, and continuous updating. Time series have become very important as they are used in many areas, such as the examples that follow. Fraud detection consists in finding unusual movements in commercial applications such as banks, phone companies, credit cards, etc [6]. Intrusion detection refers to the detection of anomalous activities in a computer network [4]. A high number of false alarms rate may arise due to a large amount of information flow. Those unusual movements are related to a thief's identity or theft attempts by a person. Moreover, anomaly detection can be applied in the world of the industry, detecting some damage in structures, engine sensor instrumentation errors, or unexpected behavior in the engines of an assembly line [8]. Other applications of interest are anomaly detection in text data, anomaly detection in sensor network [5], or image processing [27].

2.2. Deep learning for time series anomaly detection

Before the advent of Deep Learning, the types of algorithms used to solve anomaly detection problems typically include classical machine learning algorithms (KNN, k-Means, or HBOS), outlier detection algorithms (Isolation Forest or LOF), or Data Mining algorithms (STOMP or PST) among others [28].

Nowadays, state-of-the-art in time series anomaly detection focuses on the use of recurrent neural networks (LSTM and GRU) and temporal neural networks (TCN) [13–15]. Here, we briefly describe the functioning of some of the most recent and best-performing methods to be considered later on:

- TCN [17]: The TCN model focuses on the use of an encoding-decoding framework that uses a single set of computational mechanisms (1D convolutions, pooling, and channel normalization) to hierarchically capture low-, medium- and high-level temporal information. The 1D convolutions are applied to view the changes of the features at lower levels over time, pooling is used to compute long-range temporal patterns efficiently, and normalization achieves better robustness towards various environmental conditions.
- WeiXiaoyan [29]: To create a spatiotemporal deep learning model LSTM and a CNN are combined. CNN is used to extract relevant features. The features are decomposed into sequential components and provided to repetitive LSTM units for analysis. The output of the last step of the LSTM is provided to the fully connected layer for the prediction.
- YiboGao [30]: The use of a convolutional neural network works quite well on certain problems, however, it does not take into account the temporal characteristics of the problem. To solve this, a residual-based temporal attention block (RTA-block) is added to the architecture. The RTA-block uses residual learning to generate temporal attention weight, which allows the extraction of more information from the features.

Please note that TCN is devised for univariate time series. Therefore, for the sake of its use in more general scenarios, a modified multivariate model will be considered. This multivariate version of TCN works in a similar fashion to TCN by only adapting the network input to accept multiple input features. For the sake of simplicity, we will refer to this multivariate modification as TCN during the rest of this paper.

Table 1
Confusion matrix.

		Prediction	
		Negative	Positive
Actual	Negative	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Finally, since the current state-of-the-art is focused on neural networks, the above-described algorithms will be considered the main anomaly detectors in this work. Their implementation can be found in the "Time Series Feature Extraction using Deep Learning library" [31].

2.3. Evaluation of anomaly detection problems

To analyze the performance of an anomaly detection algorithm, four metrics are commonly used, which are represented in the confusion matrix shown in Table 1. Both true negatives and true positives are the most important and regarded values, they represent that the model is getting the prediction right.

High numbers of FPs or FNs are regarded as problematic. There are two main measures computed from the confusion matrix, sensitivity ($\frac{TP}{TP+FN}$) which indicates the ability to label as positive the TPs, and specificity ($\frac{TN}{TN+FP}$) which indicates the ability to label as negative the TNs. However, there is a trade-off between both measures, increasing one means decreasing the other, so finding the best possible adjustment is a challenging task and it will depend on the problem to be treated which measure will take precedence. There is another trade-off between TPs and the false positive rate ($FPR = \frac{FP}{FP+TP}$). Increasing TPs usually results in a higher FPR because more importance is given to positive observations and therefore although more positive observations are labeled correctly, negative observations will also be labeled as positive. A distinguishing feature of anomaly detection problems is that the number of normal observations is much larger than the number of anomalous observations so trying to adjust the model to focus on anomalous observations results in an increment of the number of FPs, thus generating unnecessary alarms. Moreover, it is possible for an observation to be labeled as positive when it is not depending on its context (leading to an FP), which is common when dealing with a problem involving time series [18,32]. Due to the complications described above, it will be difficult for the model to find a good fit in which the number of FPs is low, therefore, to correct models that generate a high number of FPs or to improve the quality of such a model, an FP reduction method should be applied.

2.4. False positive mitigation

As mentioned above, an FP occurs when the model predicts an instance as positive when it is actually negative, which is a scenario to be avoided. For example, in relation to the current coronavirus disease 2019 (COVID-19), it is common to find PCR tests that resulted in FPs. This means that these people must be quarantined, in many cases preventing them from going to work, which entails economic losses [33]. Another example focuses on the scope of cyber attacks in Industrial Control Systems (ICS), a high number of FPs results in a high rate of false alarms, consequently, the system has to be checked more than necessary resulting in lower performance. Therefore, the reduction of FPs is of great importance [32].

It is very frequent to encounter the problem of a high number of FPs when dealing with an anomaly detection problem. Typically, in anomaly detection problems the data sets have a large number of normal observations while the number of anomalous observations is small due to a lack of data since anomalies are unusual events and occur on exceptional occasions. This results in the detection model not being accurate enough, and therefore not being able to detect the anomalies

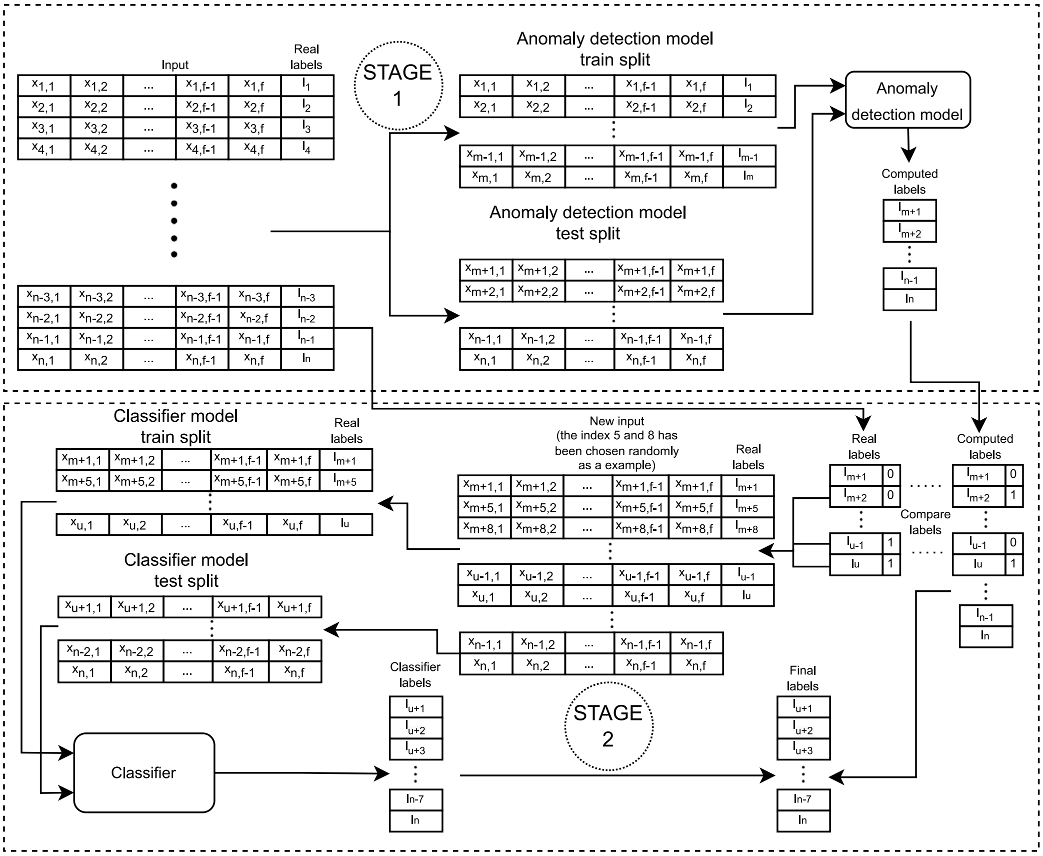

STAGE 1
Input
Real labels
Anomaly detection model train split
Anomaly detection model
Computed labels
Anomaly detection model test split
STAGE 2
Classifier model train split
Real labels
New input (the index 5 and 8 has been chosen randomly as a example)
Real labels
Computed labels
Compare labels
Classifier model test split
Classifier
Classifier labels
Final labels

Fig. 2. Flowchart of $FADFPM$ methodology.

as there are few observations. One way to fix this is to give more importance to anomalous observations or to generate more synthetic anomalous instances. However, this solution brings with it a higher number of FPs [34].

Regarding the aforementioned issues, FP mitigation has an important application in anomaly detection problems. However, most of the information in the literature that focuses on dealing with FP mitigation does not propose a general procedure but rather focuses on an ad-hoc solution for the problem that cannot be generalized to other areas.

In [35] the authors argue that a CNN has a limited learning capacity, so it may not learn all the fundamental features to distinguish the structure of a nodule from other non-nodules. However, working with several CNNs allows for dealing with more nodules. The proposal consists of an ensemble of CNNs (E-CNNs) to allow different types of nodules to be learned and thus reduce the number of FPs.

In [36] the authors reduce the FPs in the detection of pulmonary nodules in chest radiographs using morphological features, the edge of the rib, and nodule edge coverage feature. On the other hand, also in the area of pulmonary nodule detection, the author in [37] reduces the FP using the removal of the rib structure based on the left and right lung area symmetry.

As a final example, in [38] the authors describe a technique for anomaly detection that ensures that FPs are mitigated by pruning unrelated anomalies while maintaining an updated threshold to be used in the anomaly scoring system.

The examples provided above have good performance and quality, however, although their techniques achieve a low FP rate, they cannot be extrapolated to other problems. It is interesting and necessary to have mechanisms that enhance the results of current anomaly detection methods. That is, to have a proposal on FP mitigation that complements and is capable of improving the results of an already trained anomaly detection model, which would give great versatility to the study of anomalies.

One approach to address the problem is the one proposed in [18], which consists of pruning anomalies. In the problem addressed in the mentioned article, there are a number of error sequences calculated from an LSTM, whilst the idea is to prune these sequences to maintain a current data context and to reduce the cost of memory and computation time. The process consists of relabeling as normal those anomalous sequences whose values do not exceed a threshold consecutively.

Another approach described in [19] is the use of an initial component called “Process Action Monitoring Component”, which is available in anti-malware systems. A new data set is then generated with new attributes generated by the component. This new data set is trained using an artificial neural network model (ANN). The training of the artificial neural network corresponds to the FP mitigation stage.

In [20], the authors follow the last idea described, this publication proposes the use of a convolutional neural network (CNN) model for the detection of pulmonary nodules. From the CNN model, a series of attributes are extracted that will be the input of a new support vector machine (SVM) classifier. This SVM model achieves a lower FP rate.

Approaching the problem of mitigating FPs in the same way as in the three last-mentioned articles, applying a mitigation stage after the anomaly detection model, allows extrapolation of these mitigation tools to a wide variety of areas. In addition, there are a large number of tools to be employed at the mitigation stage that has not yet been studied.

The proposals in the three last-mentioned articles are limited to a specific domain and data sets, moreover, in [19,20] the proposed data sets do not have a time component. $FADFPM$ methodology is designed in such a way that it can be applied to time series. In addition, the proposed FP mitigation stage of the $FADFPM$ methodology can be applied to any data set. Given the novelty of this idea, there has not yet been an experiment in which an algorithm can achieve good mitigation of FPs. In [39], authors state that deep learning models are not always better in tabular data, which encourages the use of different methods

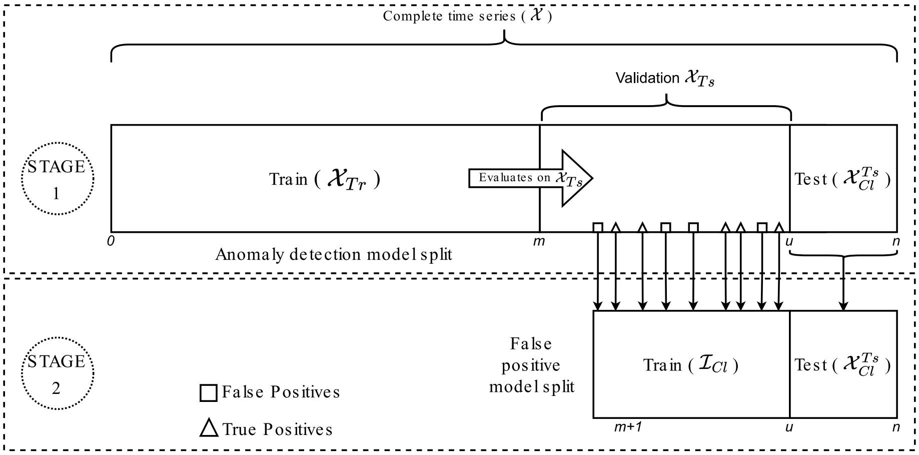

Complete time series ($\mathcal{X}$)
Validation $\mathcal{X}_{Ts}$
STAGE 1
Train ($\mathcal{X}_{Tr}$)
Evaluates on $\mathcal{X}_{Ts}$
Test ($\mathcal{X}_{Cl}^{Ts}$)
0
m
u
n
Anomaly detection model split
STAGE 2
□ False Positives
△ True Positives
False positive model split
Train ($\mathcal{I}_{Cl}$)
Test ($\mathcal{X}_{Cl}^{Ts}$)
m+1
u
n

Fig. 3. Diagram of how data is partitioned during the steps of $FADFPM$ methodology.

(SVMs, Random Forest, ANNs, and ensembles) to analyze their behavior for different types of problems.

3. FADFPM: Fusing anomaly detection with false positive mitigation methodology for predictive maintenance under multivariate time series

In this section, we describe in depth the $FADFPM$ methodology for solving anomaly detection problems in multivariate time series. Fig. 2 depicts the flowchart of the proposal. Our proposal consists of two stages: In the first stage, a portion of the training partition is fed to an anomaly detection algorithm. In the second stage, the FPs are mitigated using a trained classifier on the TPs and FPs generated by the anomaly detection model. Thus, for a given test series, the final output will comprise the decision from the anomaly detector and the FP mitigation classifier.

Since two models are created in $FADFPM$, namely the anomaly detector and the FP classifier, two splitting steps are also performed: one for the proper anomaly detection model and one for the classifier (FP mitigation stage). The full partitioning generated by the two $FADFPM$ stages is represented in Fig. 3, also naming each split with its own nomenclature that will be introduced in the subsequent sections.

The two stages involved in $FADFPM$ are detailed in their corresponding sections: Section 3.1 describes the application of the anomaly detection algorithm in the first stage, whereas Section 3.2 focuses on the FP mitigation stage. Finally, Section 3.3 describes how to create the validation set to apply a time series classifier if maintaining the temporality in the mitigation stage is advisable.

3.1. Anomaly detection stage

As mentioned earlier, the anomaly detection stage consists in training a base model to classify the data, which will also provide FPs that will be used to train the model for the FP mitigation stage. For this purpose, the data set is divided into three portions, train from 0 to m, validation from m to u and test from u to n, where n is the size of the complete data set. The anomaly detection model is trained with the train portion and the labels of the instances from m to n, i.e. validation and test, are computed. The computed labels from the validation portion are compared with the real labels to obtain the FPs along with the TPs, which will be used to build the new data set to be used by the FPs mitigation stage. The labels computed as positive for the test portion will be replaced based on the output of the false positive mitigation model.

It is interesting to mention that since the methodology is focused on dealing with time series, the concept drift problem can have a negative impact on the performance. Therefore, it would be convenient to retrain the anomaly detection model from time to time to try to avoid it. However, it will depend on the data set being dealt with.

The rest of this section describes the anomaly detection stage, which is represented in Fig. 2 by the term *Stage 1*.

Given a time series $\mathcal{X} \subseteq \mathbb{R}$ (representing the features of the time series to be analyzed) with its corresponding labels Y where $y \in Y$ is contained in $\{0, 1\}$ (representing the labels of the data set to be analyzed), two subsets called $\mathcal{X}_{Tr}$ (features of the training subset) and $\mathcal{X}_{Ts}$ (features of the test subset) are constructed with their corresponding subsets of labels Y_{Tr} (labels of the training subset) and Y_{Ts} (labels of the test subset). $\mathcal{X}_{Tr}$ corresponds to the first m instances of the data set, where m is defined by the analyst. $\mathcal{X}_{Ts}$ correspond to the instances from m to n where n is the size of the complete data set:

$$\mathcal{X}, Y \rightarrow \begin{cases} \mathcal{X}_{Tr}^{i}, Y_{Tr}^{i} \; : i = 0, \dots, m \\ \mathcal{X}_{Ts}^{i}, Y_{Ts}^{i} \; : i = m+1, \dots, n. \end{cases} \tag{1}$$

A function f is trained by $\mathcal{X}_{Tr}$ and applied to the subset $\mathcal{X}_{Ts}$, from which scores are obtained and subsequently transformed into labels. Let S_{T_s} be the result of applying the scoring function to the test subset. We define L as a labeling transformer from scores to labels, therefore a function from real numbers to the set conformed by 0 and 1. The process is as follows:

$$f(\mathcal{X}_{Ts}) = \underbrace{S_{Ts}}_{\subseteq \mathbb{R}} \rightarrow L(S_{Ts}) = L(f(\mathcal{X}_{Ts})) = A_{Ts} \; : \forall a \in A_{T_s} \; a \in \{0, 1\}. \tag{2}$$

After these steps, a first classification of the data set is obtained (in some cases this may be sufficient because the number of false positives obtained is very low or nonexistent.), however, the $FADFPM$ methodology involves applying a FP mitigation stage to improve the quality of these initial results. Such a stage requires a portion of the original training set in order to train the mitigation model, namely the validation split, in which the anomaly detection technique is evaluated and the TPs and FPs noted and extracted into a new training set. The details of the mitigation stage are described in the following Section 3.2.

3.2. False positive mitigation stage

As already discussed in Section 2.4, it is common in anomaly detection problems to reach a high number of FPs due to the large imbalance in the data. Therefore, applying techniques that reduce the number of FPs obtained by the anomaly detection model helps to achieve improved performance in the solution of the problem. A mechanism that gives good results is the one used in [19] and in [20], which consists of applying classification techniques with the results of the

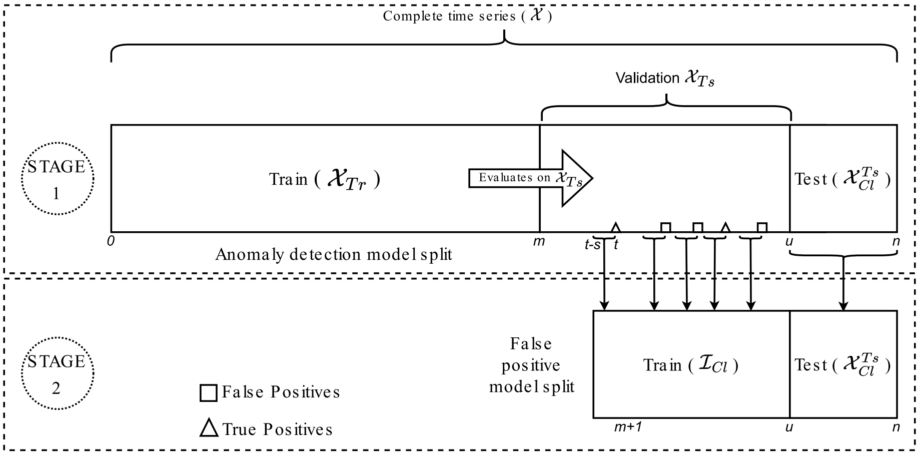

Complete time series ($\mathcal{X}$)
Validation $\mathcal{X}_{Ts}$
STAGE 1
Train ($\mathcal{X}_{Tr}$)
Evaluates on $\mathcal{X}_{Ts}$
Test ($\mathcal{X}^{Ts}_{Cl}$)
0
m
t-s
t
u
n
Anomaly detection model split
STAGE 2
False Positives
True Positives
False positive model split
Train ($\mathcal{I}_{Cl}$)
Test ($\mathcal{X}^{Ts}_{Cl}$)
m+1
u
n

Fig. 4. If temporality is to be maintained when training the FP mitigator, a window of instances preceding an FP or TP must be taken when building $\mathcal{I}_{Cl}$.

anomaly detection model. In this way, the anomaly model learns the most important features and the classification model (mitigation stage) refines the results.

However, the procedure used in [19,20] are limited to the characteristics of the analyzed data sets and to the algorithm used before the mitigation stage since this algorithm extracts specific characteristics of interest for the data set being treated. Therefore, it is interesting to consider an FP mitigation mechanism that is not dependent on the data set and the algorithm employed. The mitigation stage analyzes the FPs computed by the anomaly detection stage, using a classification model for tabular data. The model is trained with the original instances of the validation portion, but only those corresponding to FPs and TPs. The classification model computes the labels of the test portion and the labels computed by the anomaly detection model are compared. The final labels correspond to the labels of the anomaly detection model but those that were labeled as positive are replaced by the labels of the classification model (FPs mitigation model). The FP mitigation stage (see *Stage 2* in Fig. 2) is detailed below as a continuation of the anomaly detection stage.

The first u instances of Y_{Ts} (labels of the test subset) and A_{Ts} (computed labels of the test subset) are compared with each other to obtain which instances correspond to the FPs and real anomalies. Let $\mathcal{I}_{Cl}$ be the set of indices of FPs and real anomalies,

$$\mathcal{I}_{Cl} = \{ i : i \in [0,u) : \begin{cases} y^i_{Ts} = 0 \text{ and } a^i_{Ts} = 1 \\ y^i_{Ts} = 1 \end{cases} \}. \tag{3}$$

From the indices $\mathcal{I}_{Cl}$, a new set of data $\mathcal{X}^{Tr}_{Cl}$ (representing the features of the new data set used to train the classifier) with its corresponding labels Y^{Tr}_{Cl} (representing the labels of the new data set used to train the classifier) is constructed. The test subset for the classifier composed by $\mathcal{X}^{Ts}_{Cl}$ (representing the features of the test subset for the classifier) and Y^{Ts}_{Cl} (representing the labels of the test subset for the classifier) are the remaining instances from u to n. The definition of the sets will therefore be:

$$\mathcal{X}^{Tr}_{Cl} = \{x^i_{Ts} \quad i \in \mathcal{I}_{Cl}\}, \tag{4}$$

$$Y^{Tr}_{Cl} = \{y^i_{Ts} \quad i \in \mathcal{I}_{Cl}\}, \tag{5}$$

$$\mathcal{X}^{Ts}_{Cl} = \{x^i_{Ts} \quad i \in u, \dots, n\} \; and \tag{6}$$

$$Y^{Ts}_{Cl} = \{y^i_{Ts} \quad i \in u, \dots, n\}. \tag{7}$$

The last step will be to train a function g with the new set $\mathcal{X}^{Tr}_{Cl}$ and apply it to the new set $\mathcal{X}^{Ts}_{Cl}$. The result of the former will be a set of labels that are compared with the labels obtained by the previous function f. The labels labeled as positive by f are replaced by the labels obtained from the g function:

$$g(\mathcal{X}^{Ts}_{Cl}) = Y_{Rf} \; : \forall y \in Y_{Rf} \;\; y \in \{0,1\}. \tag{8}$$

The new set Y_F represents the output of the methodology, i.e. the labels that correspond to the last portion of the time series. These labels are the ones computed by the anomaly detection model but the ones that were labeled as positive are relabeled by the output of the FP mitigation model.

$$Y_F = \begin{cases} y^i_{Rf} & \text{if} \;\; a^i = 1 \\ a^i & \text{if} \;\; a^i = 0 \end{cases} \quad i = u, \dots, n \qquad y^i_{Rf} \in Y_{Rf}, \; a^i \in A_{Ts}. \tag{9}$$

In this way, the potentially most problematic instances are labeled by a model that has been specialized in that type of observation. Therefore, the number of FPs is reduced since those observations are reclassified by a model that has only trained with those observations in addition to the truly anomalous ones, which must be different from the observations labeled as FPs since they are actually normal.

3.3. Maintaining temporality in the false positive mitigation stage

Since we are focusing on a time series anomaly detection problem, applying the FP mitigation technique described in the previous section will result in the loss of the temporality of the data as the FPs obtained in the first model (anomaly detection stage) are most probably not consecutive.

This should not be a problem, however, two different approaches are proposed in view of this characteristic. The first only takes the FPs and TPs to build $\mathcal{I}_{Cl}$ as explained in Fig. 3. The second approach aims to maintain the temporality in the data, enabling a time series classifier to be used as an FP classifier, instead of a tabular classifier as described in the previous section. In order to do so, we consider the s instances prior to an FP or TP, thus building a data set of time sequences (Fig. 4) aiming to better exploit the original time dimension of the data. Both versions are analyzed in Section 5.

Please note that the $FADFPM$ methodology is focused on time series, the mitigation stage can be applied to both time series and non-time series problems due to the loss of temporality.

4. Experimental framework

This Section describes the data sets used for the analysis of the proposal, the algorithms used in the experimentation, and a description of the setup in which the experimentation has been performed. In Section 4.1, we describe the data sets that have been utilized in previous research studies and the real case of study data set. Section 4.2 details the algorithms and their parameters used in the experimentation. Finally, Section 4.3 describes the experimental setup.

4.1. Data sets description

Two different types of data sets have been selected for assessing the performance of the $FADFPM$ methodology.

Benchmark data sets. Two data sets previously analyzed in the literature have been used for comparison with other techniques. The first one is a data set constructed from the Skoltech Anomaly Benchmark (SKAB) data sets designed for evaluating the anomaly detection algorithms [21]. SKAB contains up to 35 data sets in csv format, the data sets from the valve1 section which contains a large number of measurements for a single day have been merged. The final data set is therefore obtained with **18,163 observations** and **10 features**.

The second data set has been extracted from $kaggle$ [22]. There is no information as to what the data set represents, however, it is a multivariate data set in which there are a large number of observations, which are quite unbalanced since only 0.09% of observations are anomalous. Therefore, it is an ideal data set with which to test the efficacy of $FADFPM$ methodology as well as having the possibility to compare with studies already carried out with this data set. The data set contains **509,633 observations** and **11 features**. Both data sets have been preprocessed, scaling it to the zero–one range. No features have been removed.

ArcelorMittal real case of study. Data from one of ArcelorMittal's machinery predictive maintenance[1] has been provided directly by the company. They suffer breakdowns frequently, because of the hostile environment where these assets are deployed. The main problem is that some failures involve machinery being down for several days. That period when the machinery is idle results in big losses for the company. The data provided is composed of the sensor information of one of their production machines, as well as related information such as contextual information, failures, etc. The features of the data set are built from the information of the individual sensors. These variables, model different properties of the machinery, most of them being of a real nature. The data set has 40 million instances corresponding to almost 2 years of measurements, and each instance has more than 100 variables. The objective is to detect these failures early enough to minimize the repair periods for these machines. We are going to work with subsets of one month as this is sufficient to represent the behavior of the complete data set. Furthermore, training the model every month avoids problems such as concept drift, since depending on the period of the year the range of values of the variables is different, for example, in summer values such as temperature or vibration will be higher than in winter. The data set has been preprocessed, scaling it to the zero–one range. Also, six features have been removed from the total of 112, because they have a constant value. Thus the data set used has **168,956** observations and **106 features**.

4.2. Algorithms used in the experimentation

This section contains the algorithms used in the experimentation as well as the parameters that have been optimized for each algorithm.

In addition to the TCN model, we have selected two novel recurrent neural network models for the anomaly detection stage: $YiboGao$ [30] and $WeiXiaoyan$ [29]. However, as mentioned earlier, any anomaly detection method can be applied to *FADFPM* methodology. The same is applicable to the FP mitigation stage, so several classical algorithms have been selected in addition to more recent algorithms to test different approaches.

In Table 2 we can observe the complete list of the parameters optimized for all algorithms employed. The parameters for $FADFPM$ methodology are the necessary parameters for one anomaly detection model and for one classification algorithm, these parameters will be different according to the mitigation algorithm to be used. Anomaly detection problems are data-dependent. Due to this fact, the parameters can hardly be generalized to another problem. Therefore, to find the optimal performance, the correct combination of parameters must be computed. For this task, we have chosen a hyper-parameter optimization framework called Optuna [40] to perform a hyper-parameter optimization. This framework aims to build the parameter search space for the hyper-parameters dynamically. We have employed the F1-score measure as an objective value to optimize.

Table 2
Complete list of all parameters optimized for all algorithms used in the experimentation.

Algorithm	Parameters
Anomaly Detection	
TCN	batch_size, dropout, epochs, kernel_size, levels, learning_rate, optimizer, number_of_hidden_units
YiboGao	batch_size, epochs
WeiXiaoyan	batch_size, epochs
Classification	
KNN	n_neighbors, weights, algorithm, leaf_size, p (power parameter for the Minkowski metric)
SVM	C (regularization parameter), kernel, tol (tolerance for stopping criterion), gamma
RANDOM FOREST	min_samples_split, criterion, min_samples_leaf, min_weight_fraction_leaf, max_depth, n_estimators
XGBOD	learning_rate, min_child_weights, max_delta_step, subsample, subsample_bytree, gamma, max_depth, n_estimators
XGBOOST	booster, eta (learning_rate), min_child_weight, max_delta_step, sampling_method, reg_lambda, alpha, num_round, threshold, gamma, depth
TABNET	n_d (dimension of the prediction layer), n_a (dimension of the attention layer), n_steps, learning_rate, gamma
NODE	num_layers, num_trees, dropout, threshold_init_beta, learning_rate, gamma, depth
1D-CNN	batch_size, epochs

4.3. Experimental setup

This section will detail the evaluation measure used, the validation scheme when splitting the data set for each $FADFPM$ methodology stage, and the hardware specifications used. Performance is evaluated using the F1-score metric. This metric has been widely employed in outlier research [1,2,23,41]. The F1-score is the harmonic mean of the precision (the number of TP results divided by the number of all positive results) and recall (the number of TP results divided by the number of all samples that should have been identified as positive).

$$F_1 = \frac{2}{precision^{-1} \cdot recall^{-1}} = 2\frac{precision \cdot recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} \tag{10}$$

In the experimentation, we divide the complete data set into two subsets, train, and test, which contain 80% and 20% of the data respectively. The training subset (X_{Tr}) is divided into two subsets again, the size of the subsets will be 60%–40% for the training set and the validation set respectively for the anomaly detection model partitioning. With the FP and TPs of the validation subset, the data set for the classification stage is built. Finally, we measure the quality of the methodology with the test subset. Percentages have been chosen to train the anomaly and FP detection models correctly. However, this is a problem-dependent parameter.

The experiments have been carried out in a server with the following hardware specs: 2 x Intel Xeon CPU E5-2698, 16 cores per processor (32 threads), 2.30 GHz (3.60 GHz in turbo mode), 512 GB RAM DDR4, and 8x Nvidia Tesla V100 32Gb GPUs. Regarding software, we have used the following configuration: Python 3.8, Pytorch 1.9.0, cuda toolkit 10.2 and scikit-learn 0.24.2.

[1] The complete, anonymized dataset with a brief explanation of its structure is available at https://github.com/ari-dasci/OD-TINA

Table 3
F1-score for the SKAB and Kaggle data sets. Cells in gray indicate a worse result with respect to the base without mitigation anomaly detection model while bold numbers indicate the best result for that data set.

F1-Score

	Data set	SKAB			Kaggle		
No mitigation	Original paper results	0.79			0.985		
	Anomaly detection algorithms	TCN 0.9982	Yibogao 0.9515	WeiXiaoyan 0.9298	TCN 0.9921	Yibogao 0.9914	WeiXiaoyan 0.9896
Mitigation	KNN	0.9982	0.9496	0.9333	0.9997	0.9988	0.9987
	SVM	**0.9989**	0.9518	0.9296	**0.9999**	**0.9999**	0.9997
	RANDOM FOREST	0.9982	0.9515	0.9390	0.9987	0.9985	0.9987
	XGBOD	0.9982	0.9503	**0.9409**	**0.9999**	**0.9999**	**0.9999**
	XGBOOST	0.9982	0.9416	0.9260	**0.9999**	**0.9999**	**0.9999**
	TABNET	**0.9989**	**0.9525**	0.9296	**0.9999**	0.9988	0.9994
	NODE	0.9982	0.9515	0.9387	0.9987	**0.9999**	0.9994
	1D-CNN	0.9982	0.9508	0.9278	0.9987	0.9988	0.9987

Table 4
Amount of FPs obtained for the SKAB and Kaggle data sets. Cells in gray indicate a worse result with respect to the base without mitigation anomaly detection model while numbers in bold indicate the best result for that data set.

Amount of false positives

	Data set	SKAB			Kaggle		
No mitigation	Anomaly detection algorithms	TCN 3	Yibogao 16	WeiXiaoyan 158	TCN 15,164	Yibogao 15,276	WeiXiaoyan 15,452
Mitigation	KNN	3	14	58	1	5	5
	SVM	**1**	14	38	**0**	**0**	3
	RANDOM FOREST	3	16	101	6	8	6
	XGBOD	3	13	71	**0**	**0**	**0**
	XGBOOST	3	**7**	**4**	**0**	**0**	**0**
	TABNET	**1**	11	15	**0**	6	2
	NODE	3	16	119	8	**0**	5
	1D-CNN	3	15	47	7	6	7

5. *FADFPM* : anomaly detection analysis in benchmark data sets

This section analyzes the results achieved by the $FADFPM$ methodology in terms of the detection of anomalies as well as the mitigation of FPs.

- Section 5.1 shows an analysis of the evaluation metric of the public benchmark data sets.
- Section 5.2 shows the results of the FP mitigation.
- Section 5.3 compares the performance of the evaluation measure against FP mitigation.
- Section 5.4 shows the additional study regarding the temporality of the time series when constructing the data set for the FP mitigation stage.

5.1. Analysis of the evaluation metric

In this section, we show $FADFPM$ methodology performance for the two benchmark data sets in terms of F1-score.

Table 3 shows the F1-score for the anomaly detection models, the previous literature research, and the different classification models that have been evaluated for the FP mitigation stage. The first two rows equate to the previous literature research and the anomaly detection models, both of which do not contain an FP mitigation stage. The other rows refer to the classification models for FP mitigation. The results obtained, which can be seen in Table 3, are explained in the following itemize:

- For the SKAB data set and the TCN results, the mitigation stage maintains the same results with the exception of the SVM and TABNET models, which improves the results of the TCN.
- This behavior is due to the fact that the TCN results contain only 3 FPs (see Table 4), therefore, the improvement that can be obtained is very low.
- WeiXiaoyan's results do contain a notable number of FPs, so more models manage to improve the F1-score obtained and in greater quantity.
- For the Kaggle data set where all models achieve near-perfect F1, but the number of FPs is high (15,164), the mitigation stage always manages to improve the results.

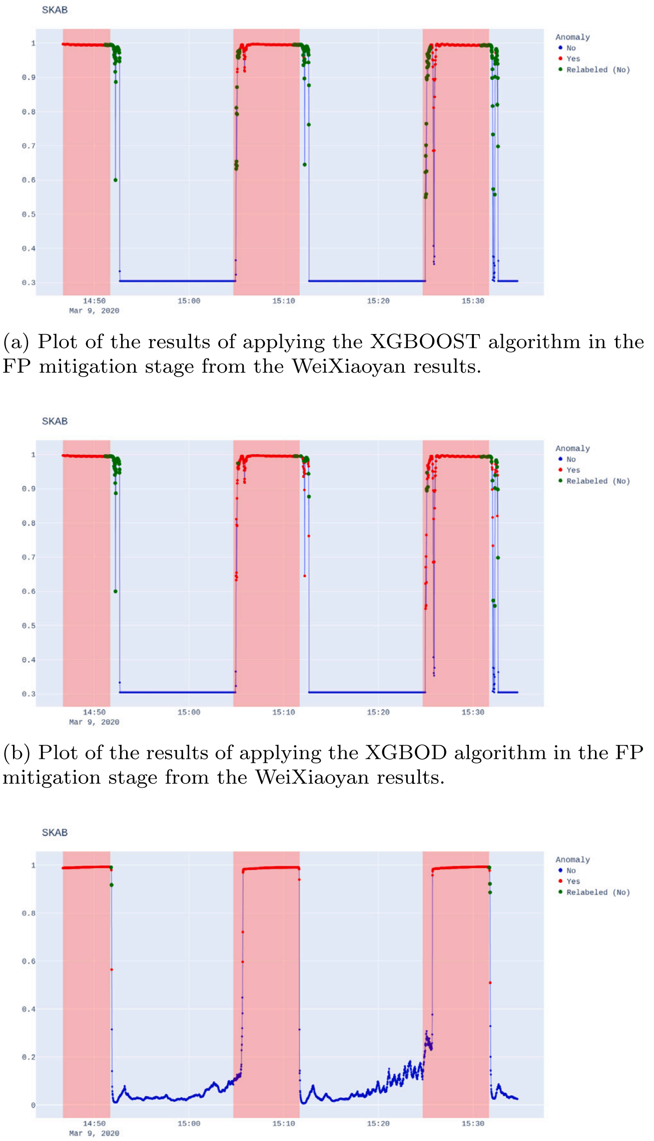
The anomaly detection algorithms alone are already able to improve the results of previous studies. Moreover, the mitigation stage manages to further improve the F1-score. It is important to emphasize the fact that if the number of FPs obtained by the anomaly detection model is very low, it will be more difficult for the mitigation stage to achieve better results. However, depending on the algorithm used, it is possible to improve the previous results. As the authors state in [39], deep learning models do not necessarily obtain the best results in classification problems.

5.2. False positive mitigation

Table 4 shows the number of FPs obtained for the anomaly detection models and for the different classification models. The first row equates to the anomaly detection models without a FP mitigation stage. The other rows refer to the classification models for FP mitigation.

As the results demonstrate, the FP mitigation stage reduces the number of FPs obtained by the anomaly detection model in most cases. As commented in the previous section, the number of FPs obtained by the anomaly detection model influences the performance of the classifier. The number of FPs mitigated in the Kaggle data set is bigger because of this reason.

In Fig. 5, we show a graphical representation of how the FP mitigation stage is reducing FPs in the SKAB data set. Fig. 5(a) shows how the XGBOOST algorithm is able to completely reduce the observations outside the anomalous period, i.e. to mitigate FPs. However, it is also reclassifying truly anomalous points as normal, therefore, the final F1-score obtained is lower than before the mitigation stage. On the other hand, in Fig. 5(b) the XGBOD algorithm, which does manage to improve the F1-score, is not mitigating all the FPs after the last two anomalous periods, but it is not failing to reclassify the TP observations. Fig. 5(c) shows how the TABNET algorithm, which starts from a more robust prior model, reduces the few FPs without encountering problems in reclassifying truly anomalous points.



(a) Plot of the results of applying the XGBOOST algorithm in the FP mitigation stage from the WeiXiaoyan results.

(b) Plot of the results of applying the XGBOD algorithm in the FP mitigation stage from the WeiXiaoyan results.

(c) Plot of the results of applying the TABNET algorithm in the FP mitigation stage from the YiboGao results.

Fig. 5. Plots to show the behavior of the mitigation stage algorithms. The red dots represent the observations detected as anomalous, the blue dots represent the observations represented as normal, and the green ones the observations that have been relabeled. The red bands represent the real anomalous area of the data set.

Therefore, the mitigation stage is capable of improving the performance of the anomaly detection algorithms. The quality of the improvement will be determined by the performance of the previous anomaly detection algorithm as it depends on the number of FPs it generates. In addition, it is interesting to apply one algorithm or another depending on the degree of mitigation to be applied.

5.3. Behavior of the evaluation metric when mitigating false positives

The results in Tables 3, 4, 6 and 7 depicts that depending on the algorithm used, the number of FPs and the improvement in F1-score is different. Moreover, in some cases, the number of FPs is reduced but the F1-score obtained is of poorer quality.

Table 5
Results obtained by the TCN mitigation model applying different time periods before each FP. The experimentation has been performed with the SKAB data set and for the results of WeiXiaoyan. The sampling rate of SKAB dataset is 1 s.

Interval Length (s)	10	20	30	40
F1 score	0.9314	0.9346	0.9352	**0.9387**
FP	**17**	29	41	68

The F1-score is related to precision and recall. Therefore, F1-score can improve even if the number of FPs is lower because the number of FNs is increasing. A good example of the trade-off between FPs and FNs can be seen in the behavior of the KNN model for the SKAB data set and for the results of YiboGao. By applying this model to these data set, FPs are reduced by almost a third, while the F1-score only varies by one-tenth. As mentioned above, if the F1-score is maintained, a decrease in FPs implies an increase in FNs. However, some algorithms such as TABNET for the Arcelor data set and for the TCN results manage to mitigate FPs completely without increasing FNs. The slight improvement in F1 is because the number of mitigated FPs is low.

In addition, depending on the type of problem to be treated, it may be interesting to reduce the number of FPs despite a small reduction in the accuracy measure. The reason for this is that when dealing with time series, the anomalies are usually clustered in a time window so that a small increase in the number of FNs may mean a slightly later detection (if it occurs at the beginning of the anomalous period) or even be unimportant if it occurs at the end of the anomalous period. However, mitigating FPs can result in eliminating false alarms, thus avoiding unnecessary system downtimes.

5.4. Considering temporality in the FP mitigation stage: SKAB case of study

As indicated in Section 3.2, the classifier in charge of learning how to mitigate FPs can be better contextualized by providing the prior examples to a FP in the time series. In this section, our goal is to analyze whether this added instances, which increase training time and complexity in the FP mitigation stage, result in a performance improvement.

In order to do so, we tested the behavior of a new data set constructed from the instances preceding the FPs. We have selected the results of the WeiXiaoyan model for the SKAB data set for the anomaly detection stage.

To better exploit the temporality of the window of instances preceding a FP, a TCN model has been trained for the FP mitigation stage instead of a tabular classifier, provided the latter would not take into account data temporality. Fig. 4 shows how this construction works. The SKAB data set has been chosen, varying the number of instances preceding a FP included in the window.

Results from Table 5 indicate that as the interval increases, i.e., there are more normal observations in the data set, the mitigation of FPs is reduced. However, the F1-score increases. With a smaller interval, the proportion of anomalous observations is higher, enabling the model to better classify the former. By increasing the interval and, therefore, the number of negative observations, the model focuses more on these, achieving a higher F1-score as it improves the normal instances accuracy at the cost of misclassifying more anomalous observations. This behavior can be useful depending on the data set and the requirements of the problem to be solved.

Using temporality in the mitigation stage still improves the results without mitigation, but it performs differently from the other tabular classifiers for FP mitigation considered in this research. The F1-score achieved using temporality is very similar to the F1-scores of the best classification models but at the cost of a significantly higher number of FPs. The explanation for this behavior may be due to two factors. The first factor is that the new data set constructed from intervals of observations does not fully maintain temporality as the intervals may be separated. The other factor is that the anomalies detected by both methods (with and without temporality) are different, in which case both methods could be complemented to obtain a possible better performance. The second factor will be analyzed as a future line of research.

Table 6
F1-score for the real-world data set (Arcelor). Cells in gray indicate a worse result with respect to the base without mitigation anomaly detection model while bold numbers indicate the best result for each anomaly detection algorithm.

F1-Score

	Data set	Arcelor		
No mitigation	Anomaly detection algorithms	TCN 0.5939	Yibogao 0.4348	WeiXiaoyan 0.3508
Mitigation	KNN	0.5899	0.6378	0.6103
	SVM	0.5969	0.6320	**0.7122**
	RANDOM FOREST	0.5975	0.5899	0.5830
	XGBOD	0.5971	0.6522	0.7074
	XGBOOST	0.5896	0.6410	0.5830
	TABNET	**0.5988**	**0.6771**	0.6989
	NODE	**0.5988**	0.5972	0.5922
	1D-CNN	0.5945	0.5908	0.5912

Table 7
Amount of FPs obtained for the real-world data set (Arcelor). Numbers in bold indicate the best result for each anomaly detection algorithm.

Amount of false positives

	Data set	Arcelor		
No mitigation	Anomaly detection algorithms	TCN 52	Yibogao 2,479	WeiXiaoyan 2,926
Mitigation	KNN	32	488	512
	SVM	20	235	395
	RANDOM FOREST	14	**0**	**0**
	XGBOD	18	356	488
	XGBOOST	16	142	**0**
	TABNET	**0**	208	301
	NODE	**0**	82	18
	1D-CNN	23	142	165

6. ArcelorMittal real case of study

Similarly to the previous section, this section shows the results achieved by $FADFPM$ methodology in terms of the detection of anomalies as well as the mitigation of FPs for the real-world case of study. Section 6.1 shows an analysis of the evaluation metric of the Arcelor dataset. Finally, the results of the FP mitigation are shown in Section 6.2.

6.1. Analysis of the evaluation metric

Table 6 shows the F1-score for the anomaly detection models and the different classification models that have been evaluated for the FP mitigation stage. The first row equates to the anomaly detection models which do not contain a FP mitigation stage. The other rows refer to the classification models for FP mitigation.

As can be seen in Table 6, similar to the behavior in the benchmark data sets, the TCN is the model that performs best before applying the FP mitigation step. However, for this data set, the F1-score value for all 3 algorithms shows that the anomaly detectors are not achieving good performance. As with the other data sets, the number of FPs obtained by TCN is low (52), so even if some algorithms manage to eliminate them completely, the F1-score improvement is very poor. On the other hand, when using YiboGao and WeiXiaoyan as anomaly detection models, which have a much lower F1-score and a higher number of FPs, the mitigation stage is able to greatly improve the results obtained. In fact, better results are obtained than those obtained using the TCN as an anomaly detection model despite obtaining significantly better initial results. Therefore, it is proved that for the mitigation stage, it is desirable that the anomaly detection stage model obtains a significant number of FPs so that the mitigation stage model is able to train correctly.

6.2. False positive mitigation

Table 7 shows the number of FPs obtained for the anomaly detection models and for the different classification models that have been evaluated for the FP mitigation stage. The first row equates to the anomaly detection models, without a FP mitigation stage. The other rows refer to the classification models for FP mitigation.

The results are very similar to those obtained for the benchmark data sets, in fact, for this data set, all algorithms reduce the number of FPs obtained by the anomaly detection algorithm. Regardless of the anomaly detection algorithm used, the number of FPs is reduced to 0 depending on the classification algorithm used. Similarly to the benchmark data sets, a lower number of FPs does not imply a better measure of accuracy. The best F1-score obtained belongs to the Yibogao anomaly detection algorithm after applying the TABNET algorithm in the mitigation stage. However, the number of FPs obtained is 208, which is far from the 0 value reached by other classification algorithms. Likewise, the reduction in the number of FPs obtained compared to the anomaly detection algorithm is quite remarkable (208 vs. 2,479).

7. Conclusions

This work presents a novel methodology for dealing with FPs in multivariate time series anomaly detection problems, which refines and improves the quality of the results, namely the $FADFPM$ methodology. It is divided into two stages. First, an anomaly detection model is trained on a fraction of the training partition of the time series and classifies the remaining validation part. The second stage builds a classification model (FP mitigator) from a portion of the FP and TPs obtained by the previous model in the validation portion. In the advent of new observations, the FP mitigator revises the positive predictions from the anomaly detector model, correcting the FP when necessary.

The results achieved show that the anomaly detection models chosen in $FADFPM$ methodology are able to obtain quality results for time series data sets, moreover, the mitigation stage achieves an FP rate reduction. Furthermore, the mitigation strategy for FPs by applying a classifier has been shown to work for a wide variety of algorithms. In fact, the algorithm of choice for the mitigation stage will be important depending on the final result to be obtained (reducing the highest number of FPs, improving the accuracy measure, or a balance between the two). The number of FPs obtained by the anomaly detection model has been found to be relevant since a low number of FPs limits the performance of the mitigation step. The reliability of the $FADFPM$ methodology is therefore shown, as well as allowing the possibility to continue the study of FP mitigation also on non-temporal data sets.

CRediT authorship contribution statement

David López: Conceptualization, Methodology/Study design, Software, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Ignacio Aguilera-Martos:** Validation, Formal analysis, Investigation, Writing – original draft, Writing –review & editing. **Marta García-Barzana:** Investigation, Resources, Data curation. **Francisco Herrera:** Conceptualization, Methodology/Study design, Validation, Investigation, Resources, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Diego García-Gil:** Conceptualization, Methodology/Study design, Software, Validation, Investigation, Writing – original draft, Writing – review & editing, Visualization, Supervision. **Julián Luengo:** Conceptualization, Methodology/Study design, Validation, Investigation, Resources, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

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

Data availability

The authors do not have permission to share data.

Acknowledgments

This work was supported by the Spanish Ministry of Science and Innovation under project TED2021-132702B-C21 funded by MCIN/AEI/10.13039/501100011033 “European Union PRTR”. This work has also been partially supported by project PID2020-119478GB-I00, the Contract UGR-AM OTRI-4260 and the EQC2018-005084-P project, granted by the Spain’s Ministry of Science and Innovation and European Regional Development Fund (ERDF), the SOMM17/6110/UGR project, granted by the Andalusian Consejería de Conocimiento, Investigación *y* Universidades and European Regional Development Funds (ERDF). Funding for open access charge: Universidad de Granada / CBUA.

References

[1] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. (CSUR) 41 (3) (2009) 1–58.

[2] C.C. Aggarwal, Outlier Analysis, second ed., Springer Publishing Company, Incorporated, 2016.

[3] L. Erhan, M. Ndubuaku, M. Di Mauro, W. Song, M. Chen, G. Fortino, O. Bagdasar, A. Liotta, Smart anomaly detection in sensor systems: A multi-perspective review, Inf. Fusion 67 (2021) 64–79.

[4] I.F. Kilincer, F. Ertam, A. Sengur, Machine learning methods for cyber security intrusion detection: Datasets and comparative study, Comput. Netw. 188 (2021) 107840.

[5] I. Kraljevski, F. Duckhorn, C. Tschöpe, M. Wolff, Machine learning for anomaly assessment in sensor networks for NDT in aerospace, IEEE Sens. J. 21 (9) (2021) 11000–11008.

[6] J. Forough, S. Momtazi, Ensemble of deep sequential models for credit card fraud detection, Appl. Soft Comput. 99 (2021) 106883.

[7] R.K. Dwivedi, R. Kumar, R. Buyya, A novel machine learning-based approach for outlier detection in smart healthcare sensor clouds, Int. J. Healthc. Inf. Syst. Inf. (IJHISI) 16 (4) (2021) 1–26.

[8] B. Bayram, T.B. Duman, G. Ince, Real time detection of acoustic anomalies in industrial processes using sequential autoencoders, Expert Syst. 38 (1) (2021) e12564.

[9] G.P. Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing 50 (2003) 159–175.

[10] F. Piccialli, F. Giampaolo, E. Prezioso, D. Camacho, G. Acampora, Artificial intelligence and healthcare: Forecasting of medical bookings through multi-source time-series fusion, Inf. Fusion 74 (2021) 1–16.

[11] J. Carrasco, D. López, I. Aguilera-Martos, D. García-Gil, I. Markova, M. García-Barzana, M. Arias-Rodil, J. Luengo, F. Herrera, Anomaly detection in predictive maintenance: A new evaluation framework for temporal unsupervised anomaly detection algorithms, Neurocomputing 462 (2021) 440–452.

[12] N. Tatbul, T.J. Lee, S.B. Zdonik, M. Alam, J.E. Gottschlich, Precision and recall for time series, in: Advances in Neural Information Processing Systems, Vol. 31, Curran Associates, Inc., 2018.

[13] S. Siami-Namini, N. Tavakoli, A.S. Namin, A comparison of ARIMA and LSTM in forecasting time series, in: 2018 17th IEEE International Conference on Machine Learning and Applications, ICMLA, IEEE, 2018, pp. 1394–1401.

[14] S. Siami-Namini, N. Tavakoli, A.S. Namin, The performance of LSTM and BiLSTM in forecasting time series, in: 2019 IEEE International Conference on Big Data, Big Data, IEEE, 2019, pp. 3285–3292.

[15] S. Gopali, F. Abri, S. Siami-Namini, A.S. Namin, A comparison of TCN and LSTM models in detecting anomalies in time series data, in: 2021 IEEE International Conference on Big Data, Big Data, IEEE, 2021, pp. 2415–2420.

[16] S. Bai, J.Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018, arXiv preprint arXiv:1803.01271.

[17] C. Lea, R. Vidal, A. Reiter, G.D. Hager, Temporal convolutional networks: A unified approach to action segmentation, in: European Conference on Computer Vision, Springer, 2016, pp. 47–54.

[18] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 387–395.

[19] A.M. Lungana-Niculescu, A. Colesa, C. Oprisa, False positive mitigation in behavioral malware detection using deep learning, in: 2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing, ICCP, IEEE, 2018, pp. 197–203.

[20] Z. Shi, H. Hao, M. Zhao, Y. Feng, L. He, Y. Wang, K. Suzuki, A deep CNN based transfer learning method for false positive reduction, Multimedia Tools Appl. 78 (1) (2019) 1017–1033.

[21] I.D. Katser, V.O. Kozitsin, Skoltech Anomaly Benchmark (SKAB), Kaggle, 2020, https://www.kaggle.com/dsv/1693952.

[22] Anomaly detection in multivariate time series, https://www.kaggle.com/drscarlat/anomaly-detection-in-multivariate-time-series.

[23] M. Goldstein, S. Uchida, A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, PLoS One 11 (4) (2016) e0152173.

[24] J. Luengo, D. García-Gil, S. Ramírez-Gallego, S. García, F. Herrera, Big Data Preprocessing - Enabling Smart Data, Springer, 2020.

[25] D. García-Gil, J. Luengo, S. García, F. Herrera, Enabling Smart Data: Noise filtering in Big Data classification, Inform. Sci. 479 (2019) 135–152.

[26] D. García-Gil, F. Luque-Sánchez, J. Luengo, S. García, F. Herrera, From big to smart data: Iterative ensemble filter for noise filtering in big data classification, Int. J. Intell. Syst. 34 (12) (2019) 3260–3274.

[27] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: A review, ACM Comput. Surv. 54 (2) (2021).

[28] S. Schmidl, P. Wenig, T. Papenbrock, Anomaly detection in time series: A comprehensive evaluation, Proc. VLDB Endow. 15 (9) (2022) 1779–1797.

[29] X. Wei, L. Zhou, Z. Zhang, Z. Chen, Y. Zhou, Early prediction of epileptic seizures using a long-term recurrent convolutional network, J. Neurosci. Methods 327 (2019) 108395.

[30] Y. Gao, H. Wang, Z. Liu, An end-to-end atrial fibrillation detection by a novel residual-based temporal attention convolutional neural network with exponential nonlinearity loss, Knowl.-Based Syst. 212 (2021) 106589.

[31] I. Aguilera-Martos, Á.M. García-Vico, J. Luengo, S. Damas, F.J. Melero, J.J. Valle-Alonso, F. Herrera, TSFEDL: A Python library for time series spatio-temporal feature extraction and prediction using deep learning, Neurocomputing 517 (2023) 223–228.

[32] S. Sridhar, M. Govindarasu, Model-based attack detection and mitigation for automatic generation control, IEEE Trans. Smart Grid 5 (2) (2014) 580–591.

[33] G.D. Braunstein, L. Schwartz, P. Hymel, J. Fielding, False positive results with SARS-CoV-2 RT-PCR tests and how to evaluate a RT-PCR-positive test for the possibility of a false positive result, J. Occup. Environ. Med. 63 (3) (2021) e159.

[34] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.

[35] C. Li, G. Zhu, X. Wu, Y. Wang, False-positive reduction on lung nodules detection in chest radiographs by ensemble of convolutional neural networks, IEEE Access 6 (2018) 16060–16067.

[36] B. Keserci, H. Yoshida, Computerized detection of pulmonary nodules in chest radiographs based on morphological features and wavelet snake model, Med. Image Anal. 6 (4) (2002) 431–447.

[37] H. Yoshida, Local contralateral subtraction based on bilateral symmetry of lung for reduction of false positives in computerized detection of pulmonary nodules, IEEE Trans. Biomed. Eng. 51 (5) (2004) 778–789.

[38] Z. Zohrevand, U. Glässer, Dynamic attack scoring using distributed local detectors, in: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2020, pp. 2892–2896.

[39] R. Shwartz-Ziv, A. Armon, Tabular data: Deep learning is not all you need, Inf. Fusion 81 (2022) 84–90.

[40] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019.

[41] G.O. Campos, A. Zimek, J. Sander, R.J.G.B. Campello, B. Micenková, E. Schubert, I. Assent, M.E. Houle, On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study, Data Min. Knowl. Discov. 30 (4) (2016) 891–927.