Expert Systems With Applications 297 (2026) 129285

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A data attribution approach for unsupervised anomaly detection on multivariate time series

Alessio Verdone, Simone Scardapane, Massimo Panella *

Department of Information Engineering, Electronics and Telecommunications (DIET), University of Rome "La Sapienza", Via Eudossiana 18, 00184, Rome, Italy

ARTICLE INFO

Keywords:
Unsupervised anomaly detection
Multivariate time series
Data influence
Explainable machine learning
Neural forecasting

ABSTRACT

Anomaly detection is a challenging task that manifests in several forms depending on its context (e.g., fraud detection, network security, fault monitoring): given a collection of data, the goal is discovering the anomalous patterns diverging from the majority. The time dimension adds complexity to defining an anomaly, especially with multivariate time series, where each channel represents possibly distinct quantities. Classical unsupervised anomaly detection approaches have been based on differences in data, sequences of data, distributions, or also by inspecting the prediction deviation errors. In this work, we propose an innovative unsupervised approach for discovering anomalies in multivariate time series by leveraging the concept of data attribution from the explainability literature. Our proposed method is flexible and data-driven, and it can be used across multiple scenarios without the necessity of domain experts. By training initially on a self-supervised task (e.g., forecasting), a model captures normal data behavior; then, by using a data attribution technique we identify potential anomalies at the time step level. We conduct experiments on several synthetic and real-world datasets, comparing the results with state-of-the-art unsupervised anomaly detection methods, achieving competitive or better performance on most datasets, with notable improvements on synthetic data and promising results on real-world data, despite certain challenges in highly anomalous scenarios. Finally, we show an in-depth investigation of this methodology, along different dimensions, in order to gain a greater understanding of the application of these functions and their characteristics within the anomaly detection landscape.

1. Introduction

Anomaly detection (AD) on multivariate time series (MTS) is the process of identifying unusual patterns or data points which differ significantly from expected or standard behavior. In today's data-driven world, the ability to detect and identify anomalies, deviations, or outliers (in the following as synonymous) is of paramount importance across various domains, including finance, cybersecurity, manufacturing, healthcare, and energy (Adiban et al., 2023; Ahmed et al., 2016; Copiaco et al., 2023; Namratha et al., 2023; Stracqualursi et al., 2023; Suh et al., 2023). These anomalies may represent critical insights, such as fraudulent activities, equipment malfunctions, or abnormal health conditions, making them crucial to preemptive decision-making and risk mitigation.

It is a highly challenging task for several reasons, which has been analyzed for a long time (Chandola et al., 2009; Gupta et al., 2014). First, the definition of anomaly is variable, as it depends on the particular distribution of data or the temporal process; furthermore, real-world time series are often subject to noise and characterized by complex temporal patterns that are difficult to identify. This condition typically requires the presence of domain experts to correctly identify them, making it difficult to produce generally applicable methods. Second, labels are rare: anomalies, by definition, are infrequent, requiring expertise in labeling each time series for a given application. In addition, the anomaly detection process generally happens in a continuous-way for real applications, highlighting the necessity to adjust the detection method according to the evolving data distribution over time (Gao et al., 2021).

The scarcity or total absence of labels, which makes many supervised methods ineffective, and the abundance of unlabeled data has led to the development of several works that attempt to detect anomalies in an unsupervised manner: these methods focus on identifying patterns, structures, or deviations that significantly differ from the norm for a dataset. Recent advances in signal processing have demonstrated the effectiveness of decomposition-based methods for anomaly detection in specific domains. For instance, Singular Spectrum Analysis (SSA) has shown promising results for detecting weak oscillations in rotary encoder signals from industrial systems (Algburi et al., 2021) and has been further enhanced through hierarchical approaches for improved fault detection in robotic systems (Algburi et al., 2025). While these

* Corresponding author.
E-mail addresses: alessio.verdone@uniroma1.it (A. Verdone), simone.scardapane@uniroma1.it (S. Scardapane), massimo.panella@uniroma1.it (M. Panella).

https://doi.org/10.1016/j.eswa.2025.129285
Received 23 September 2024; Received in revised form 29 July 2025; Accepted 2 August 2025
Available online 8 August 2025

domain-specific methods achieve good performance in their respective applications, they are primarily designed for specific signal types and may not generalize well to diverse multivariate time series scenarios without significant domain expertise and parameter tuning.

Unsupervised sequential outlier detection models can be divided into three categories based on their operational principles: prediction deviation, majority modeling, and discords analysis. Prediction Deviation methods identify the outliers by evaluating the differences between the predicted values and the observed data, while majority modeling and discords analysis put their focus on finding irregularities at the level of regular data or subsequences (Schölkopf & Smola, 2002; Senin, 2009; Steinwart et al., 2005).

Although these methods yield good results, majority modeling and discords analysis models do not fully leverage the temporal dynamics of the data, and prediction deviation methods are typically trained on anomaly-free datasets (Bontemps et al., 2016), in order to not learn also anomalies in the learning process, which is a non-ideal condition for real applications.

In this paper, we introduce a novel class of methods for multivariate time series unsupervised anomaly detection by leveraging the concept of data influence from the field of explainable machine learning (Hassija et al., 2024). The influence analysis of training patterns, that is data attribution (Dai & Gifford, 2023; Nguyen et al., 2023; Park et al., 2023), estimates the impact of individual training data points on the model outcomes, and it has already found several applications to deep learning (Koh & Liang, 2017; Yeh et al., 2018). Essentially, by analyzing the training process, data attribution methods quantify how much each training instance affects the final model on a given prediction. These functions have found application in detecting data poisoning (Koh et al., 2022), discerning helpful and harmful examples (Pruthi et al., 2020), detecting biases in word-embeddings (Brunet et al., 2019), and assessing the influence of different groups of instances (Koh et al., 2019). In (Thimonier et al., 2022), the TracIn technique is proposed to detect anomalies at the sample level in tabular data by employing a variational autoencoder. We propose instead to leverage influence values in temporal domains by introducing a novel class of anomaly detection methods for multivariate time series. By analyzing the influence of training points at the time-step level with respect to a model trained on an unsupervised task and a dataset that includes anomalies, we can identify the time steps that deviate most from normal behavior during the training process: these time steps are more likely to represent anomalies.

The novelty of this method lies in two key aspects. First, it focuses on analyzing the presence of anomalies, unlike most models, by taking into account the model's training process. This differs from traditional unsupervised anomaly detection models, such as prediction deviation models, which can learn anomalies during the training phase. In contrast, our approach relies on the difficulty of learning certain patterns with respect to the entirety of the data, a concept very close to the definition of an anomaly itself. As a second key aspect of novelty, this work introduces a unique application of data attribution techniques for the unsupervised detection of anomalies in multivariate time series. By analyzing and processing samples influences, the framework is capable of detecting deviations at a granular, time-step level, with the added flexibility of variable time windows. This allows for dynamic adjustment based on the length of the anomalies or dataset properties. Furthermore, the framework is versatile and generalizable, as it can be adapted to a wide range of existing and future forecasting models, making it suitable for diverse anomaly detection scenarios.

The contributions of this work can be summarized as in the following:

- We present a novel influence-based method for capturing anomalies in time series data. A neural model, trained on a forecasting task, embeds the normal dynamics of a time series. Then, we leverage checkpoints saved during the training to employ a powerful model-agnostic influence model, TracIn (Pruthi et al., 2020): we assign to each training data sample a score measuring the influence that a sample had on itself during the learning process. Finally, these scores are projected to each singular time step and, after a threshold procedure, we are able to detect anomalous time steps.
- We validate our model using multivariate synthetic and real-time series: we select the data carefully based on past benchmarks for the AD task for time series (Lai et al., 2021): synthetic data aids in comprehending the model's response to distinct dynamic anomaly patterns, while the real data validate its effectiveness in a real environment, characterized by increased noise and more unpredictable behaviors. We compare our method against various unsupervised AD approaches to evaluate its practical performance.
- Furthermore, we conduct a detailed analysis of the information related to the influence scores. We perform an ablation study on the proposed method's hyperparameters, such as input and output windows in the forecasting step, and asses their relationship with the task. Finally, we investigate the impact of threshold selection on the entire anomaly detection procedure.

The rest of the paper is organized as follows. In Section 2, we illustrate the basic concepts and the literature background related to anomaly detection and data attribution methods. The general methodology for time series anomaly detection is introduced in Section 3, while the proposed model is detailed in Section 4 in terms of implemented tasks up to the actual anomaly detection technique. The experimental setup for assessing the performances of the proposed approach is described in Section 5, while the obtained numerical results are reported and discussed in Section 6. Finally, our conclusions are drawn in Section 7.

2. Main background

In this section, we introduce the main elements associated with the proposed approach, starting from unsupervised anomaly detection up to data attribution analysis.

2.1. Unsupervised time series anomaly detection

Several unsupervised AD methods have been proposed to detect anomalies in sequential data during last years. Traditionally, AD was performed by applying a threshold, which identified data points that surpassed a boundary set by domain experts. However, this approach depends on expert knowledge and cannot be easily applied to detect unknown anomalies with varying patterns over time. Several statistical methods (Soule et al., 2005) were introduced to identify data points that do not pass statistical hypothesis tests, yet they depended on prior knowledge of the datasets. On the other hand, in more recent years, deep learning models have been increasingly employed to tackle this task, leading to significant improvements in performance and achieving promising results (Ceschini et al., 2021; Zhang et al., 2019; Zong et al., 2018).

Time series anomalies were studied and classified by Lai et al. (2021): this classification is crucial since it enables the categorization of the huge and heterogeneous scenario of time series anomalies into well-defined categories based on temporal characteristics (e.g., trend, seasonality, temporal context). It facilitates the development of new detection methods that effectively address the problem: for example, many conventional methods struggle to establish a threshold to distinguish between global point anomalies and contextual ones.

At present time, sequential algorithms for outlier detection can be classified based on three different types of approaches: prediction deviation, majority modeling, and discords analysis. The first one detects the outliers by evaluating the differences between predictions and real data. It is assumed that data can be reconstructed using regression and if a specific instance cannot be predicted correctly, then it is most likely an outlier. Autoregression (AR) assumes linearly correlated between time

steps (Rousseeuw & Leroy, 2005); Gradient boosting regression (GBRT) splits the time series into subsequences and tries to solve a regression problem on them (Elsayed et al., 2021). A recurrent neural network using Long Short-Term Memory layers (LSTM-RNN) is utilized in Bontemps et al. (2016) in order to increase the capability of the adopted model to learn temporal patterns and therefore increase also AD.

Majority modeling operates under the assumption that normal data instances distribution form compact clusters in hyperspace (Schölkopf & Smola, 2002; Steinwart et al., 2005): so, its goal is to detect the decision boundary among outliers and regular points by estimating the actual distribution of the latter ones. For example, One-class SVM (OCSVM) (Ma & Perkins, 2003; Schölkopf et al., 1999; Zhang et al., 2008) works by maximizing the margin between the origin and the normal data points, defining the decision boundary as the hyperplane that establishes this margin. Isolation Forest (IForest) (Liu et al., 2008, 2020), on the other hand, constructs several binary trees that aim at isolating the focused patterns based on how close an individual instance is to the root.

Autoencoders (AEs) (Sakurada & Yairi, 2014) project patterns onto a latent space with lower dimensionality, then reconstruct them from this compact representation. Outliers are identified considering that their reconstruction error will be greater than the one of normal points. Generative adversarial networks (GANs) (Adiban et al., 2023) use a min-max optimizer involving both a generator and a discriminator. The latter attempts to model normal data points, while the generator seeks to create outliers that the discriminator mistakenly identifies as normal. The discriminator's loss on single points is used to decide in this regard.

Discor analysis methods instead divide time series into subsequences, typically by using a simple sliding window: then, by measuring the similarity between them, they identify discords as outliers. It is useful to detect pattern-based outliers. Some discord analysis methods which perform subsequence clustering (Zolhavarieh et al., 2014) are OCSVM (Zhang et al., 2008), IForest (Liu et al., 2020) or Matrix Profile (MP) (Yeh et al., 2016; Zhu et al., 2018): in particular, MP generates distance profiles by calculating the minimum distances between each subsequence and finally identifies anomalous subsequences depending on this distance-based profile.

Due to the continuous increase in data volume and dimension, these methods are difficult to optimize for high-dimensional data (Lu et al., 2023; Xu et al., 2023). However, several challenges remain in real-world applications. For instance, AE and GAN models are particularly sensitive to noise within the data. Additionally, many generative approaches rely on being trained exclusively with normal data to learn well the true distribution of data, which is a requirement that is difficult to meet in real-world scenarios where datasets often contain unlabeled anomalies.

2.2. Data attribution

Data attribution methods (Dai & Gifford, 2023; Nguyen et al., 2023; Park et al., 2023) measure how the presence or absence of a particular training data point affects the trained model's performance. They have been studied for a long time (Cook, 1977; Hampel, 1974; Jaeckel, 1972) and recently they have been adapted to modern deep models and large datasets (Koh & Liang, 2017) in order to explain predictions and produce confidence intervals (Schulam & Saria, 2019), investigate model bias (Brunet et al., 2019; Wang et al., 2019), estimate Shapley values (Ghorbani & Zou, 2019; Jia et al., 2019), improve human trust (Zhou et al., 2019), and craft data poisoning attacks (Koh et al., 2019).

Retraining the model by removing each time a different sample is clearly unfeasible: several approximations were proposed during the years. In (Feldman & Zhang, 2020), a method is introduced that involves training multiple models on randomly selected data subsets while tracking which subsets each sample belongs to: in this way, this approach provides an unbiased estimate of the true influence, but, although this simple and easy to implement, it requires a significant number of model retraining. The technique described in Hara et al. (2019) determines the influence on the total loss of the training patterns: it leverages a specific Hessian-based formulation of model parameters to track the influence of a training point across the mini-batches where it is not present.

TracIn (Pruthi et al., 2020) measures the influence of a training sample on a model's prediction by tracking how the loss on a test point evolves during training whenever the specific training example is used. Compared to other influence functions, it offers several advantages: it is general, model-agnostic, easy to implement, and scalable. To make the whole procedure computationally efficient, the exact computation of the influence is approximated by a first-order gradient operation, saving checkpoints to replicate the training procedure. It requires a careful selection of the layers of the deep neural network on which to take the gradients, typically the last linear layer.

3. General methodology

In this section, we introduce the proposed influence-based unsupervised anomaly detection framework: it aims at discovering anomalies in MTS by employing self-influence scores generated by TracIn with respect to a preliminary forecasting task. Our method is composed by three fundamental steps: auxiliary task, data attribution, and anomaly detection, which we formulate and present in this section. The proposed solution integrates methodologies from diverse domains: time series forecasting, data attribution, and anomaly detection. In order to provide a cohesive understanding of our approach, we begin by presenting a unified formulation of our methodology.

3.1. Time series forecasting

A time series can be formulated as a matrix $x \in \mathbb{R}^{N \times C}$, with N and C equal to the length and the channels of the time series respectively, where each sample or time step is defined as $x_t \in \mathbb{R}^C$. When $C = 1$, the time series is *univariate*; otherwise, if $C > 1$, it is *multivariate*. At time index t, given a past sequence of L observations $x_{t-L:t}$, with L named input window or also lag, the forecasting problem consists into predicting the future P values of $y_{t:t+P}$, with P named output or prediction window: the values to predict can involve one or multiple channels of the time series x (Rosato et al., 2021). Given a model f parameterized by weights w_j, the training process is defined by the optimization of the model via the loss function $l(w_j, x)$, computed between the prediction of the model m at iteration j and the sample x. In the following, training and test samples will also be represented without the time index for clarity.

3.2. Data attribution

By defining a training process with the same notation presented before, TracIn represents the ideal influence $I_{\text{ideal}}(x, z)$ of training sample x over test sample z as the total reduction in loss of z induced by an ideal training process, which utilizes a singular sample at iteration j, whenever x is utilized or not at that iteration: $I_{\text{ideal}}(x, z) = \sum_{j:x_j=x} l(w_j, z) - l(w_{j+1}, z)$. By performing a first-order approximation to the idealized function of influence and employing k checkpoints to replicate the training process, an approximated and practical version of the influence is $I_{\text{approx}}(x, z) = \sum_{i=1}^{k} \eta_i \nabla l(w_{j_i}, z) \nabla l(w_{j_i}, x)$, where η_i is the learning rate. The gradients are taken from the last fully-connected layer of the model, which is typically present in almost all deep learning models.

Training examples x with a positive influence score with respect to test points z are named *proponents*, whereas training examples with a negative influence score are named *opponents*. When $z = x$, it is possible to consider the loss of a training pattern x in order to measure its influence also called *self-influence*; in our framework, this quantity will represent a measure for the presence of an anomaly. When the *self-influence* value significantly surpasses others, it indicates that the point is a robust proponent for itself because it is one of the few points that decrease its

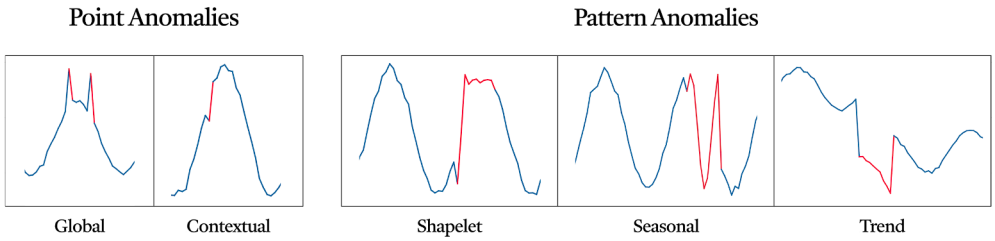

Fig. 1. Classification of anomalies in time series domain.

own loss, representing a behavior different from the majority, so more likely to represent an anomaly.

3.3. Time series anomaly detection

Having defined a multivariate time series as $x \in \mathbb{R}^{N \times C}$, the anomaly detection problem can be formulated as a binary classification task: we want to produce for each time step t of x a binary vector $\hat{a} \in \mathbb{R}^N$ depicting or not the presence of an anomaly: the presence of an anomaly in at least one channel defines a time step as anomalous.

Time series anomalies can symbolize single or multiple time steps, and in the last case, the length of the anomaly is not always the same. Traditional anomaly classification frameworks fail to accurately model the potential anomalous behaviors occurring within this domain. Lai et al. (2021) formalized this problem: *point-wise outliers*, which represent single-step anomalies, are divided into *global outliers*, points that significantly deviate from the rest of the points, and *contextual outliers*, points that deviate from its corresponding context; *pattern-wise outliers* instead are anomalous subsequences of variable length composed by *shapelet outliers*, subsequences with dissimilar basic shapelets compared with the normal ones, *seasonal outliers*, subsequences with unusual seasonalities compared with the overall seasonality, and *trend outliers*, in which the time series behavior is consistently modified by the subsequence. A visual representation of time series anomalies is given in Fig. 1.

4. Proposed model

The method proposed in this paper is composed of three different steps. The *Auxiliary Task* consists in learning the dynamics of a time series by performing a self-supervised task, such as the forecasting which best suits to temporal phenomena. The *Data Attribution* step assigns to each training sample a self-influence score, indicating how the presence of that sample in the training process influences itself. This measure is used as the anomaly score: in the *Detection* phase, the influence scores are first projected to each time step and then a threshold procedure produces a binary anomaly detection vector. An overview of our proposed system is given in Fig. 2. One important feature of our method is its versatility: each block of our method is model-agnostic and can be adaptable to different scenarios.

4.1. Auxiliary task

The concept itself of anomaly defines something which differs from normal behavior, so the process of detection anomalies starts by learning what is a normal behavior (Chalapathy & Chawla, 2019). Autoencoders or GANs were used extensively with non-temporal data to model data distribution and find anomalies: they perform well, but they do not fully exploit the temporal dimension. The initial step of our approach is to train a model to forecast the future behavior of a MTS: this is a self-supervised task, in which the supervision is generated by shifting the time series of the prediction window desired, so we do not need the labels to perform it (Bianchi et al., 2020). Given a sequence L of past input time values for each channel C, the model is optimized to predict P future time steps:

$$y_{t:t+P} = f(x_{t-L:t}, w)\,. \tag{1}$$

For our experiments, we predict the values at each channel C, but this setting can be changed if we know in advance the channels in which the anomalies are present.

During training, we save k checkpoints at predefined epochs to simulate the training process necessary for *Tracin*: these checkpoints, denoted as w_j for epoch j, store the model parameters at specific intervals. This way, the dynamics and causal behavior of the time series, encoded within the model's weights at various checkpoints during training, allow for a deep analysis of the concept of anomaly and its identification through Tracin methodology.

With respect to prediction deviation methods, which are negatively affected by the presence of anomalies inside the training dataset, our method is based on leveraging the impact they exert during training, thus turning a difficulty to our advantage; we show an example of this behavior in Fig. 3. By changing input and output windows, L and P, there is no constraint on anomaly lengths or type to predict: we can easily adapt them to any desired scenario. As a preliminary phase for a new application, it is possible to sweep between different forecasting parameters in order to find the combination that matches with the anomaly scenario. The model is not dependent on the prediction model: different forecasting models can be used to perform this step, allowing to stay updated with state-of-the-art advances in time series forecasting, enabling the discovery of anomalies even in more complex and different scenarios (Shaukat et al., 2021).

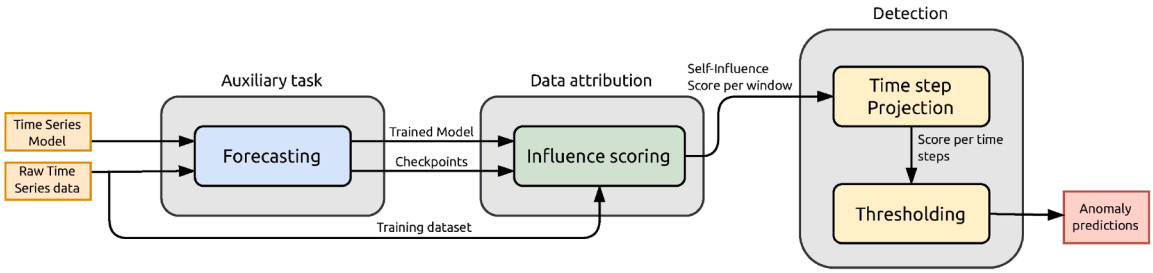

Fig. 2. Main architecture of the proposed method, which is divided into (I) auxiliary task, (II) data attribution, and (III) detection phase.

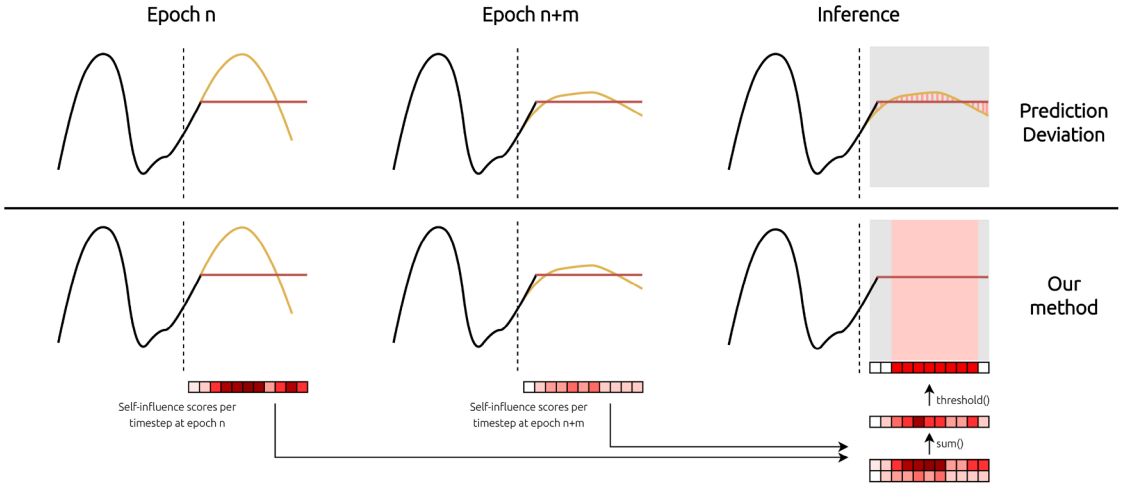

Fig. 3. Comparison between prediction-deviation methods, which may inadvertently learn anomalous patterns present in the training dataset during the forecasting step, thus reducing detection effectiveness, and our method, which instead exploits the influence scores computed at training checkpoints to identify and learn anomalies explicitly, turning their presence into a useful training signal.

4.2. Data attribution

The next step of our framework is the Data attribution: when training a model on a forecasting task, we collect k checkpoints of this process. Each checkpoint captures the state of the model at different epochs, reflecting how the model's parameters evolve and enabling an efficient computation of influence scores. We want to discover the influence of each training sample on itself, namely the *self-influence* (Pruthi et al., 2020). More in detail, it measures the impact of each training sample on its own prediction: training samples exhibiting higher self-influence are considered more influential and, by extension, more likely to represent anomalies in the dataset

The TracIn method allows to map the influence between the elements of the training dataset by only employing the loss function, gradients and checkpoints of the training process of the forecasting task. Following the formulation introduced herein, the self-influence score can be formulated as follows:

$$SI(x) = I_{\text{approx}}(x, x) = \sum_{i=1}^{k} \eta_i \nabla l(w_{j_i}, x) \nabla l(w_{j_i}, x)\,. \tag{2}$$

Evaluating the model's gradients can be heavy in terms of computational complexity; in order to speed-up the computation time and evaluating quickly the influence scores, we employ the gradients at the final fully-connected layer. Those gradients can be computed without the need for back-propagation. Moreover, many forecasting models have a fully-connected layer as a final step of the model, in order to adapt itself to different prediction windows. The final formula can be expressed as:

$$SI(x) = \sum_{i=1}^{k} \eta_i \nabla l(w_{j_i}, x)^2 x^2\,. \tag{3}$$

Consequently, samples that have higher self-influence are strong proponents on themselves, since they tend to reduce the loss with respect to the incorrect behavior which is represented by the anomaly.

4.3. Detection

The last part of our framework deals with the final detection of anomalies at the time step level: this part is composed of a projection and a threshold phase. The influence values, returned from TracIn for each training instance, are defined at the sample level: we need to project these values at the time step level. This technique is shown in Fig. 4, where iteratively we assign at each time step the sum of self-influence values of the samples they belong to:

$$S_t = \sum_{x_i \in \text{Near}_t} SI(x_i), \tag{4}$$

where S_t is the anomaly score for time step t and Near_t is the set of samples with at least one time step belonging to t. Obviously, for a fair comparison, the samples at the extremities of the dataset are discarded. The score is projected in the same way as time steps belonging to the input and output windows since the presence of an anomaly in both of them can alter the prediction of the model, and so influence the loss value. The anomaly scores are then scaled with respect to the maximum score value of the whole training dataset:

$$S_t^{\text{scaled}} = \frac{S_t}{\max(S)}\,. \tag{5}$$

Once we have scalar values for each time step as a measure for anomaly, we need to perform a threshold operation to discriminate between anomaly time steps and normal ones. The choice of the threshold

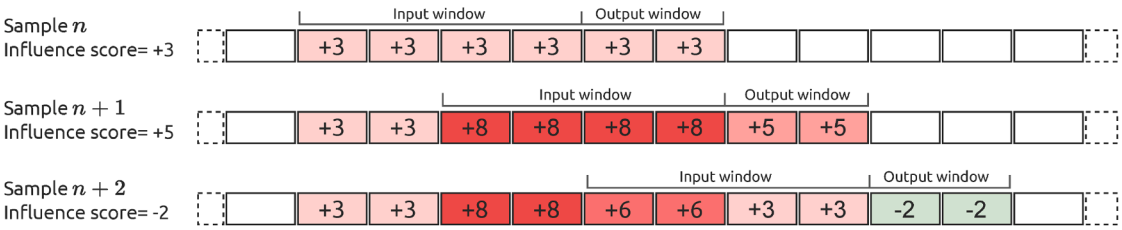

Fig. 4. Projection step: influence scores are projected from sample- to time-step level.

value becomes crucial: two types of thresholds are presented and were used to generate our predictions. The first creates a fixed threshold T_s across diverse datasets, window lengths and outlier ratios (anomalies percentage in the dataset): we call this the *static* threshold It is easy and simple to implement and can be quickly adapted to different scenarios, but it does not fully harness the potential of anomaly scores. The second method, *dynamic* threshold T_d, adapts the value of the threshold to the specific dataset.

The training set of the forecasting model is divided into two separate sets: a train subset, which is used to find the optimal value for the threshold, and a test set on which a threshold with this value is applied. For both of the methods, the threshold operation is made on the qth percentile of the influence values. Finally, the threshold operation is defined as follows:

$$\text{Anomaly}(t) = \begin{cases} 1, & \text{if } S_t^{\text{scaled}} \geq T \\ 0, & \text{if } S_t^{\text{scaled}} < T \end{cases} \tag{6}$$

5. Experimental setup

We extensively evaluate our method on 4 real and 15 synthetic datasets, using 4 forecasting models for the auxiliary step and 10 CE baselines for comparison. Experimental settings are presented in detail in the section below. We evaluate our framework on benchmark datasets proposed by Lai et al. (2021) so as to generate fair comparisons between models performance, in both real and synthetic environments.

5.1. Adopted datasets

The motivation behind this choice is twofold: we use synthetic datasets, that by definition comprehend a specific type of anomaly or combination of them, to analyze the method in domain-agnostic scenarios and observe how it behaves with different types of anomalies; we then utilize real datasets with significantly higher dimensionality to enhance the level of complexity and rigorously evaluate the results across diverse real-world scenarios. This approach enables a deep assessment of the methodology's effectiveness in practical applications.

The 15 synthetic datasets represent multivariate time series of 400 time steps, composed of 5 channels of sinusoidal-like signals generated with different types of outliers, from 1 to 5 types of anomaly defined by the classification previously introduced: point outliers as global (0) and contextual (1); pattern outliers as shapelet (2), seasonal (3) and trend (4). The names of the synthetic datasets indicate the type of anomaly or the combinations of them that define it, according to the previous categorization. The time series have different outlier ratios, from 0.05 to 0.25. Real datasets comprehend time series from four different domains: 'Credit Card', credit card transactions and the anomalies represent the frauds; 'CICIDS', web time series and the anomalies represent intrusion attacks; 'GECCO', IoT time series data related to water quality; 'SWAN-SF', Solar photospheric vector magnetograms with solar flares as outliers. The first two datasets are composed of point-wise outliers, while the last two of pattern-wise outliers. The properties of each dataset are shown in Table 1.

Table 1
Synthetic and real dataset properties (Lai et al., 2021).

Dataset	Type	Length	Channels	Outliers
0-1-2-3-4	Synthetic	400	5	0.050 %
01-12-23-34	Synthetic	400	5	0.100 %
012-123-234	Synthetic	400	5	0.150 %
0123–1234	Synthetic	400	5	0.200 %
01234	Synthetic	400	5	0.250 %
Credit Card	Real	284,807	29	0.173 %
CICIDS	Real	170,231	79	1.281 %
GECCO	Real	138,521	10	1.246 %
SWAN-SF	Real	120,000	39	23.80 %

5.2. Baselines and evaluation metrics

We accurately compare our model by using 10 baselines from (Lai et al., 2021). Three of these models belong to the *prediction deviation* group: Autoregression (AR) (Rousseeuw & Leroy, 2005); Gradient boosting regression (GBRT) (Elsayed et al., 2021); LSTM-RNN (Bontemps et al., 2016). Four models are from the *majority modeling* group: One-class SVM (OCSVM) (Ma & Perkins, 2003; Schölkopf et al., 1999; Zhang et al., 2008); Isolation forest (IForest) (Ding & Fei, 2013; Liu et al., 2008); Autoencoder (AE) (Sakurada & Yairi, 2014); Generative adversarial network (GAN) (Liu et al., 2020). The final three baseline models are from the *discords analysis* one: Matrix profile (MP) (Yeh et al., 2016; Zhu et al., 2018); OCSVM-DA (Ma & Perkins, 2003); IForest-DA (Ding & Fei, 2013).

For the real datasets setup, we have also compared our model with three well-know baselines in the time series anomaly detection landscape: AnomalyTransformer (Xu et al., 2022), THOC (Shen et al., 2020) and USAD (Audibert et al., 2020). These models represent state-of-the-art approaches in unsupervised anomaly detection for multivariate time series, each leveraging distinct architectures and techniques such as adversarial training, hierarchical clustering, and attention mechanisms to address the challenges of detecting anomalies in complex temporal data.

Different metrics are used in the experiments, Mean Squared Error (MSE) and Mean Absolute Error (MAE) are employed for the forecasting task with y_i and $\hat{y}_i$ target and prediction respectively:

$$\begin{aligned} \text{MSE} &= \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2, \\ \text{MAE} &= \sum_{i=1}^{N}|\hat{y}_i - y_i|\,. \end{aligned} \tag{7}$$

Anomaly detection tasks can be defined as binary classification tasks: by denoting true/false positive/negative with {TP, TN, FP, FN}, respectively, we can define further detection metrics:

$$\begin{aligned} &\text{P} = \frac{\text{TP}}{\text{TP}+\text{FP}}, \quad \text{R} = \frac{\text{TP}}{\text{TP}+\text{FN}}, \\ &\text{F1} = \frac{2\,\text{P}\,\text{R}}{\text{P}+\text{R}} = \frac{2\,\text{TP}}{2\,\text{TP}+\text{FP}+\text{FN}}\,. \end{aligned} \tag{8}$$

Precision (P) evaluates the correctness of the model's positive predictions. Recall (R) measures the ability of the model to capture all the actual anomalies in the dataset, while F1-score is the harmonic mean of Precision and Recall. The F1-score is meaningful when normal and anomalous instances are imbalanced. The perfect performance is achieved when F1 is 1.

5.3. Forecasting and details of experiments

We use different models for the forecasting phase: an LSTM (Hochreiter & Schmidhuber, 1997) model is tested on both synthetic and real datasets; state-of-the-art transformer models as Transformer (Vaswani et al., 2017), Informer (Zhou et al., 2021), and Autoformer (Wu et al., 2021) are employed in the more complex real datasets only. We train the models to forecast all the future channels of a time series on different scenarios, by varying the input and output windows that are also named for the sake of clarity 'lags' and 'prediction windows', respectively. Precisely, we test all configurations with lags equal to {4, 8, 16} and prediction windows {1, 2, 4}. The capability to use different windows allows the framework to be adapted to any type of anomaly.

For each experiment, we sweep between different outlier ratios, namely {0.1, 0.01} for the real datasets and {0.05, 0.1, 0.15, 0.2, 0.25} for the synthetic ones, and between different combinations of input and output windows as listed above. Finally, we show the best results in terms of F1-score achieved by the models. We performed 20 and 3 iterations per experiment for synthetic and real datasets, respectively, with

Table 2
Anomaly detection results on synthetic datasets.

Model/Dataset	0	1	2	3	4	01	12	23	34	012	123	234	0123	1234	01234
Autoregression	0.450	0.450	0.300	0.300	0.100	0.615	0.415	0.300	0.420	0.620	0.444	0.352	0.549	0.561	0.637
GBRT	0.400	0.350	0.250	0.350	0.250	0.560	0.260	0.280	0.360	0.460	0.440	0.320	0.490	0.390	0.510
LSTM	0.150	0.200	0.470	0.530	0.670	0.151	0.122	0.490	0.680	0.130	0.160	0.270	0.240	0.310	0.240
Iforest	0.850	0.600	0.350	**0.700**	0.300	0.759	0.538	0.450	0.753	0.615	0.593	0.383	0.645	0.675	0.521
Ocsvm	**0.900**	0.750	0.350	**0.700**	0.450	**0.810**	0.435	0.525	0.753	0.649	0.542	0.433	0.696	0.594	0.673
Autoencoder	0.200	0.050	0	0	0	0.202	0.051	0.075	0.103	0.136	0.101	0.133	0.167	0.243	0.217
GAN	0.170	0.150	0	0.250	0	0	0.100	0.080	0.130	0.090	0.090	0	0.170	0.189	0.150
IForest-DA	0.100	0.100	0.250	0.350	0.350	0.150	0.260	0.430	0.550	0.260	0.340	0.270	0.410	0.570	0.390
Ocsvm-DA	0.100	0.100	0.200	0.450	0.300	0.250	0.210	0.330	0.620	0.240	0.410	0.380	0.410	0.570	0.420
MatrixProfile	0.100	0.050	0.050	0.550	0.200	0.278	0.128	0.125	0.394	0.341	0.271	0.283	0.400	0.364	0.467
LSTM + S (ours)	0.660	0.644	0.619	0.648	0.615	0.693	**0.655**	0.653	0.667	0.690	0.671	**0.655**	0.681	0.706	0.678
std. dev. (±)	0.069	0.077	0.102	0.065	0.121	0.012	0.070	0.063	0.069	0.013	0.054	0.076	0.003	0.028	0.001
LSTM + D (ours)	0.807	**0.807**	**0.700**	0.695	**0.746**	0.778	0.642	**0.770**	**0.781**	**0.701**	**0.673**	0.636	**0.737**	**0.720**	**0.687**
std. dev. (±)	0.133	0.179	0.172	0.107	0.149	0.108	0.123	0.102	0.070	0.071	0.083	0.067	0.056	0.059	0.063

different random seeds so as to collect mean and variance. For the forecasting models, we employ the Adam optimizer with a learning rate initially set to 10^{-3}, decreased of a factor of 0.9 every 3 validation epochs without improvements, also using early stopping.

All experiments were performed on a system equipped with a 6-core Intel® Core™ i5-9400F CPU @ 2.90 GHz, 16 GB of RAM, and a Nvidia® RTX 2060 GPU featuring 6144 MB of RAM @ 1.830 GHz.

6. Numerical results

In this section, we present both the numerical and graphical obtained results. We test our method on both synthetic and real datasets; moreover, we analyze the impact of different elements that constitute our framework, such as the window lengths, the length of the dataset, the type and the number of anomalies and the threshold function.

6.1. Synthetic dataset results

Numerical results on synthetic datasets are presented in Table 2, where the best results for each column (or dataset) are highlighted in bold. The table is divided into two groups: the first one for the results of baseline AD models; the last two rows for the results of the proposed approach, which are in turn associated with a static and a dynamic threshold technique represented as 'LSTM + S' and 'LSTM + D', respectively. In the static setup, we found the best combination of parameters that generate the highest F1-score (input window, output window and threshold value) among all datasets, so the best parameters are {4, 1, 0.85} and the average F1-score is around to 0.654. This setup of the proposed LSTM + S model yields better results than the baseline AD models in the first group on 8 datasets out of 15, achieving anyway competitive results in the other datasets as well. The dynamic version of our method sets the threshold value depending on the training set; the performances of the proposed LSTM + D model increase, outperforming the baseline AD models on 12 datasets out of 15 and achieving an average F1-score of 0.725.

Overall, the proposed method is able to capture anomalous time steps that appear in different forms. However, it exhibits higher values of standard deviation in particular for the dynamic LSTM + D version. Although previous studies have shown that training data attribution methods can be sensitive to the randomness associated with training dynamics and model initialization (Deng et al., 2024; Karthikeyan & Søgaard, 2021), we demonstrate that the higher variance in results is primarily attributable to the scarcity of data points in the dataset. This is proved by replicating the procedure with the same dataset properties but increasing the total number of samples. The related results are reported in Table 3 as an average over all the 15 considered datasets: increasing the size of the datasets implies a general reduction of standard deviation like any classical deep learning model, thus yielding a less noisy method. The portion of time steps used for determining the optimal threshold value, by varying the relative length of the training time series, is also important: a high percentage of 0.8 leads to an increase in variance and a decrease in the F1-score with respect to a value of 0.5. Finally, an example of the detection results is shown in Fig. 5.

Table 3
Average results of proposed LSTM + D method with different time series length.

Dataset samples	Tr. set proportion	F1-score
400	0.5	$\mathbf{0.725 \pm 0.10}$
400	0.8	0.708 ± 0.15
4000	0.5	0.714 ± 0.04
4000	0.8	0.706 ± 0.07

6.2. Real dataset results

The numerical results of experiments performed on the real datasets are summarized in Table 4. This time the table is divided into three groups: the first one for the results of baseline AD models; the second group for the proposed method with static thresholds as before; the third group for the results with dynamic thresholds. In addition to LSTM + S and LSTM + D models as adopted for synthetic datasets, here we consider also Transformer (TF), Informer (IF), and Autoformer (AF) models, still combined with static and dynamic threshold detection. We evaluated P, R, and F1-score metrics; the higher the better, best results are represented in bold.

The results in terms of F1-score achieved by the methods proposed in this paper are better than the baseline ones in three real datasets out of four, achieving excellent results in particular for the GECCO dataset with an F1-score of 0.604 and 0.761 for static and dynamic setups, respectively. All of the models tested for the forecasting task are well-performing, in particular LSTM and Transformer. Standard deviation values are not displayed since for the static case they are always smaller than 0.006, while for the dynamic case, standard deviation has an average value around 0.017. The threshold value for the static version is set to 0.87 and the input/output window length pairs are 4 and 1. The poorest performance was observed with the SWAN-SF dataset; the likely explanation lies in the composition of the time series and the dataset outlier ratio: in fact, it comprises around 24 % anomalies, which almost fall outside the definition of the anomaly itself. The predictive model learns to forecast these anomalies alongside the regular forecasting task, contradicting the premises at the basis of the proposed method.

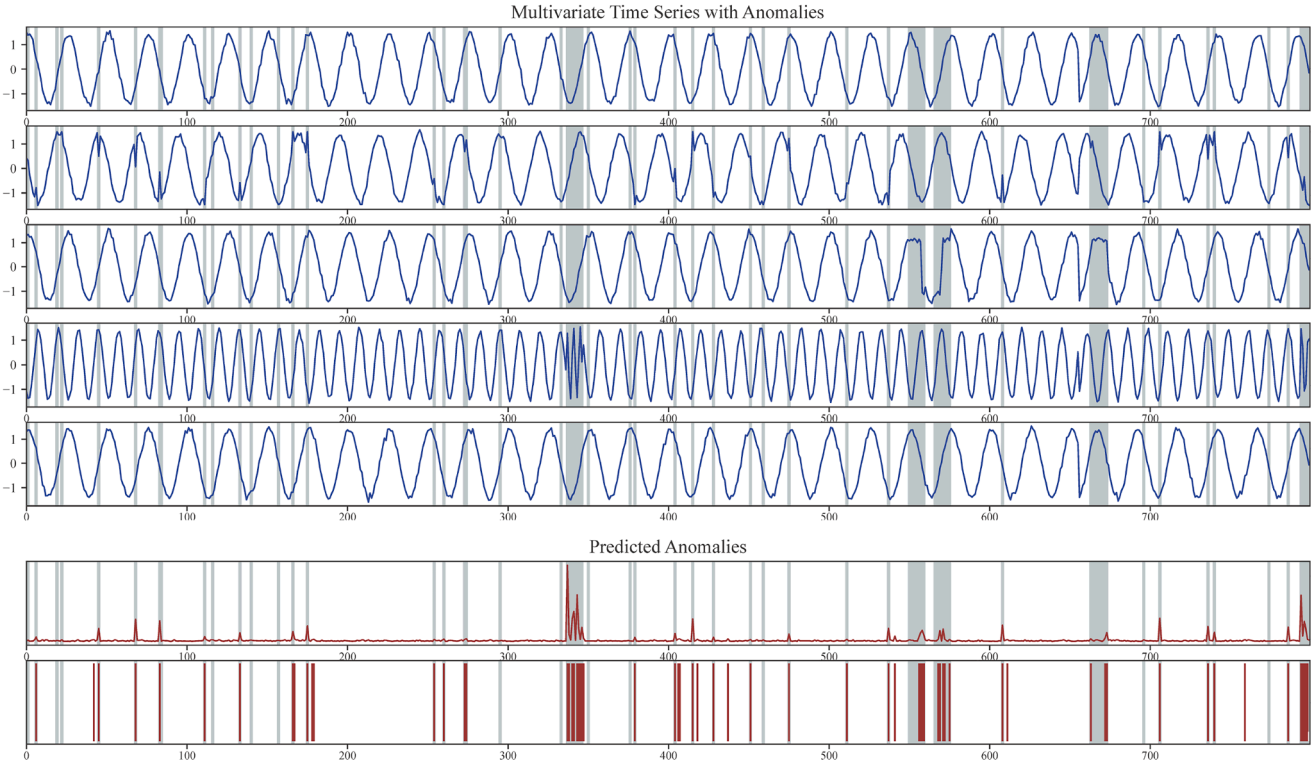

Fig. 5. Visual example of anomaly prediction on synthetic multivariate time series: original time series composed by 5 channels (blue); anomaly scores before and after the threshold operation (red); evidenced anomalous time steps (gray).

Table 4
Anomaly detection results on real datasets.

Dataset	Credit card			CICIDS			GECCO			SWAN-SF		
Metric	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Autoregression	0.113	0.652	0.192	0.016	0.310	0.03	0.392	0.314	0.349	0.421	0.354	0.385
GBRT	0.113	**0.657**	0.193	0.018	0.351	0.034	0.175	0.140	0.156	0.447	0.375	0.408
LSTM	0.004	0.110	0.007	0.024	0.383	0.046	0.343	0.275	0.305	0.527	0.221	0.312
Iforest	0.098	0.569	0.168	0.010	0.016	0.016	0.439	0.353	0.391	**0.569**	**0.598**	**0.583**
Ocsvm	0.107	0.620	0.183	0.004	0.007	0.007	0.185	0.743	0.296	0.474	0.498	0.485
AutoEncoder	0.103	0.598	0.176	0.011	0.017	0.017	0.424	0.340	0.377	0.497	0.522	0.509
Iforest-DA	0.039	0.226	0.066	0.011	0.020	0.02	0.392	0.315	0.390	0.406	0.425	0.416
Ocsvm-DA	0.002	0.305	0.004	0.000	0.000	0.00	0.021	0.341	0.040	0.193	0.001	0.001
MatrixProfile	0.006	0.514	0.012	0.080	0.013	0.013	0.046	0.185	0.074	0.167	0.175	0.171
AnomalyTransformer	0.056	0.054	0.054	0.002	0.033	0.004	0.326	1.000	0.492	0.540	0.029	0.054
THOC	0.005	0.117	0.005	0.001	0.339	0.002	0.326	1.000	0.492	0.018	0.066	0.026
USAD	0.355	0.170	0.230	0.002	0.848	0.004	0.517	0.609	0.558	0.799	0.270	0.403
LSTM + S	0.119	0.609	0.200	0.059	0.130	0.081	0.539	0.686	0.604	0.148	0.190	0.167
TF + S	0.117	0.647	0.198	**0.087**	0.186	0.119	0.506	0.676	0.579	0.191	0.253	0.218
IF + S	0.128	0.649	0.213	0.085	0.188	0.117	0.483	0.635	0.548	0.162	0.209	0.183
AF + S	0.109	0.586	0.183	0.084	0.183	0.115	0.386	0.498	0.435	0.172	0.222	0.194
LSTM + D	**0.280**	0.204	**0.236**	0.073	**0.961**	0.135	**0.750**	**0.784**	**0.761**	0.143	0.335	0.200
TF + D	0.153	0.533	0.233	0.080	0.931	0.147	0.618	0.663	0.639	0.171	0.393	0.238
IF + D	0.196	0.286	0.232	0.078	0.948	0.144	0.694	0.672	0.681	0.160	0.414	0.231
AF + D	0.195	0.282	0.230	0.082	0.829	**0.149**	0.618	0.650	0.632	0.162	0.295	0.209

6.3. Ablation study

We want to further investigate the properties of our method, by analyzing the relevance of the threshold selection procedure and the influence of model parameters such as the window lengths.

6.3.1. Threshold profile and datasets

The final results that we obtain are highly dependent on the threshold value. In Fig. 6, we show the metrics versus the threshold value: the input and output windows length is fixed and the mean value of P, R and F1-score reached at that threshold is picked. We can observe that the profile is dependent upon the dataset, but for three datasets out of four (Credit Card, CICIDS and SWAN-SF) it exhibits a smoother and more linear behavior, thus not impairing significantly the final performance of the F1-score. Only the GECCO dataset displays a more abrupt trend, changing more rapidly over time. This analysis allows for a more detailed examination of the datasets and the types of anomalies they comprise.

To further investigate the robustness of the proposed detection pipeline, we conducted an ablation study focused on the role of the threshold parameter in the final anomaly decision. In particular, we analyzed how the optimal threshold value and the corresponding F1-score

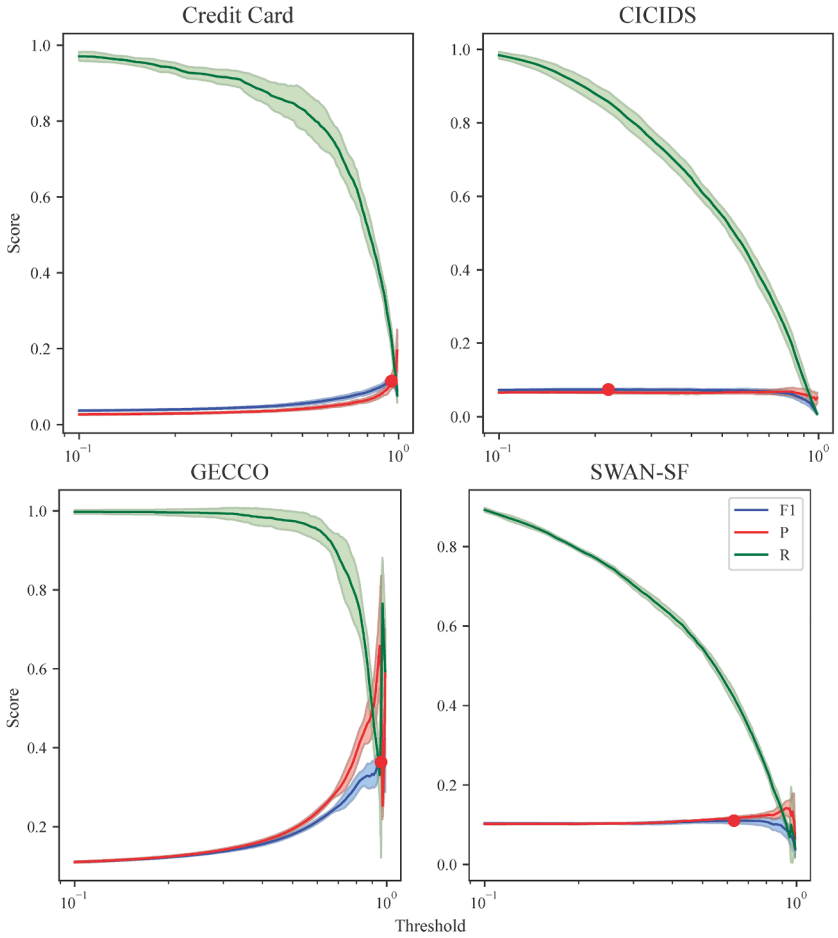

Fig. 6. Performance metrics versus the selected threshold: F1-score (blue); Precision (red); Recall (green). The shaded area represents the standard deviation.

vary across the 15 synthetic datasets, considering increasing levels of anomaly complexity. The synthetic datasets are defined with a gradual accumulation of anomaly types, from datasets containing only a single type of outlier to those combining up to five distinct temporal anomaly patterns.

The experiments were carried out using LSTM as base model, and each configuration was executed over 20 random seeds to estimate mean and standard deviation. The results summarized in Fig. 7 show the best-performing threshold, which is expressed as a percentile of the self-influence distribution across datasets.

A clear trend emerges as the dataset complexity increases. In the simpler datasets (e.g., containing only global or contextual outliers), the F1-score profiles exhibit a sharp peak, suggesting that the optimal threshold is well-defined and leads to a clear separation between normal and anomalous time steps. The optimal threshold values for these cases are typically high (e.g., 96.2 and 93.6 in the first two datasets) indicating that only the most extreme influence scores are indicative of anomalies.

As the number and heterogeneity of anomaly types increase, the F1-score profiles become progressively flatter. This effect reflects a growing ambiguity in threshold selection, as multiple types of deviations coexist and reduce the contrast between normal and abnormal behaviors. Correspondingly, the optimal thresholds tend to decrease (e.g., down to 83.8 and 86.2 in the last two plots) and the standard deviation of the F1-scores increases. These observations suggest that the dynamic thresholding strategy becomes increasingly relevant in complex scenarios, as fixed thresholds may no longer generalize well across datasets with diverse anomaly characteristics.

This analysis provides useful insights into the interaction between thresholding and dataset complexity, highlighting how the influence-based anomaly scores retain discriminative power even in the presence of multiple concurrent anomaly types, albeit requiring more careful tuning in such cases.

6.3.2. *Practical considerations on thresholding and dataset size*

The practical deployment of our method relies on two key components: the choice of the thresholding strategy and the size of the available dataset. These aspects directly impact both the stability and effectiveness of anomaly detection.

Thresholding strategy. As shown in Section 6, static and dynamic thresholds display complementary behaviors depending on dataset complexity. The static strategy based on a fixed percentile (e.g., 87th) performs reliably in simple scenarios, with a clear separation between normal and anomalous time steps. It is ideal for real-world cases in which domain knowledge is limited or cross-validation is not feasible. Conversely, the dynamic strategy adapts the threshold based on a subset of the training set, yielding higher F1-scores in the presence of complex or overlapping anomaly types, at the cost of an additional tuning step. Table 5 summarizes the conditions under which each method is more appropriate.

Dataset size and performance stability. As discussed in Section 6, results on synthetic datasets exhibit increased variance when the dataset length is small (e.g., 400 time steps). Our experiments show that extending the time series to 4000 points significantly reduces variance (see Table 3). This aligns with known behavior in deep learning models, where small datasets amplify instability due to training noise and model initialization.

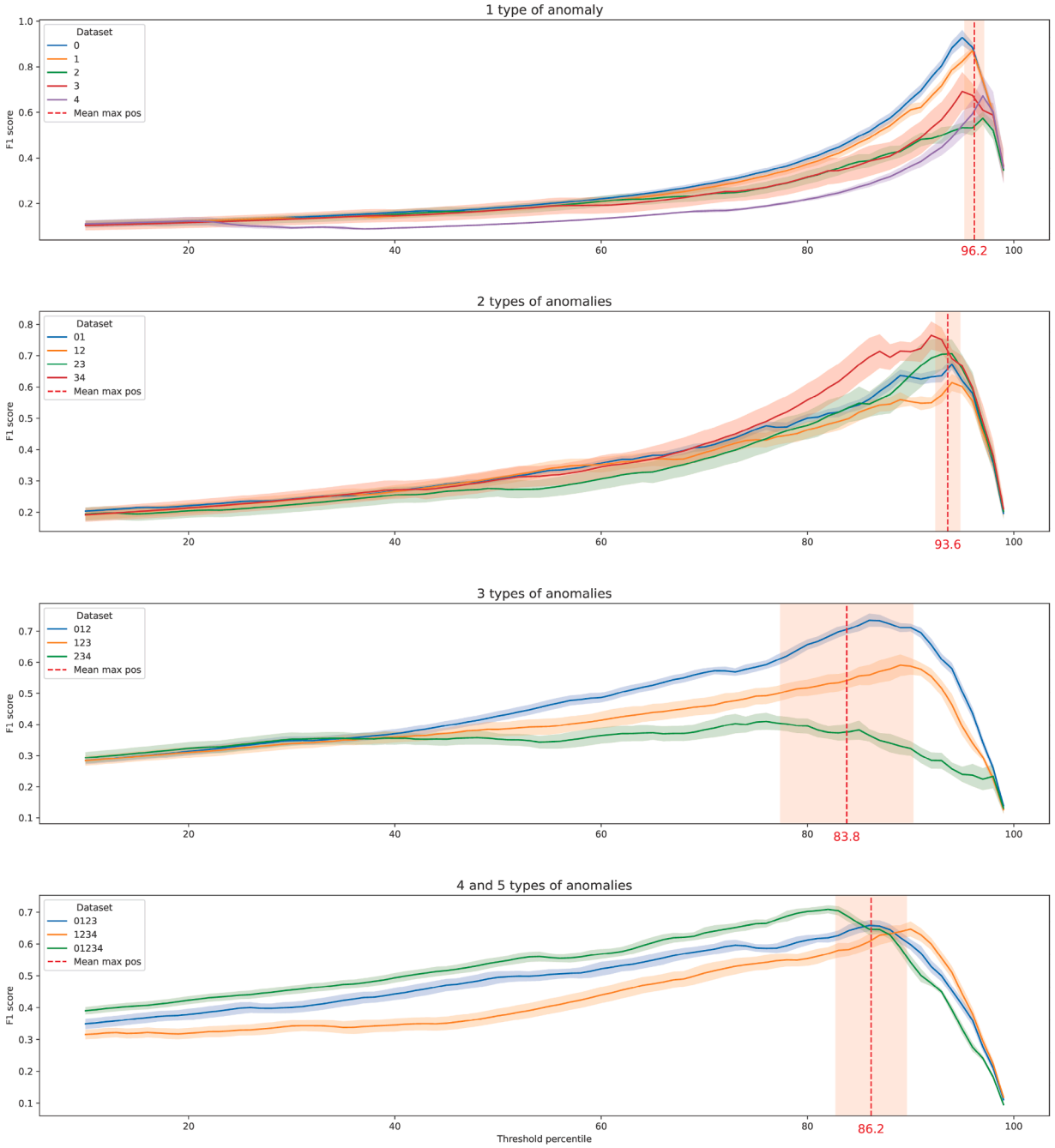

Fig. 7. F1-score vs. threshold percentile across synthetic datasets. As anomaly complexity increases, optimal thresholds decrease and score curves flatten, indicating reduced separability.

For smaller datasets, robustness can be enhanced by increasing the number of effective training samples. This can be achieved by increasing the overlap between forecasting windows, which augments the number of training sequences without altering the original data. Data augmentation techniques specific to time series, such as time warping, jittering, cropping, or magnitude scaling, can synthetically expand the dataset while preserving temporal structure needed for the forecasting task. Additionally, duplicating the dataset with small perturbations or segment recombinations can help expose the model to more variability (Iglesias et al., 2023), improving the reliability of both forecasting and influence estimation. While no strict minimum can be defined, our experiments suggest that simple multivariate sequences of at least 4000 time steps tend to yield more stable performance. Moreover, combining this with the aforementioned techniques can further enhance robustness and accuracy.

6.3.3. *Windows length and anomalies*

The length of the input and output window in the forecasting model has a significant impact on the model performances, it allows us to adapt the framework to different shapes of anomalies. We can observe in Fig. 8 the change in the F1-score by varying the windows length in the real datasets. Credit Card and CICIDS datasets reach the best F1-score with the input/output pair {4, 1}, they are populated by point outliers and so, wrong predictions of single points are the most useful to detect anomalies. GECCO and SWAN-SF, which are composed of pattern anomalies, achieve better results with bigger windows able to capture

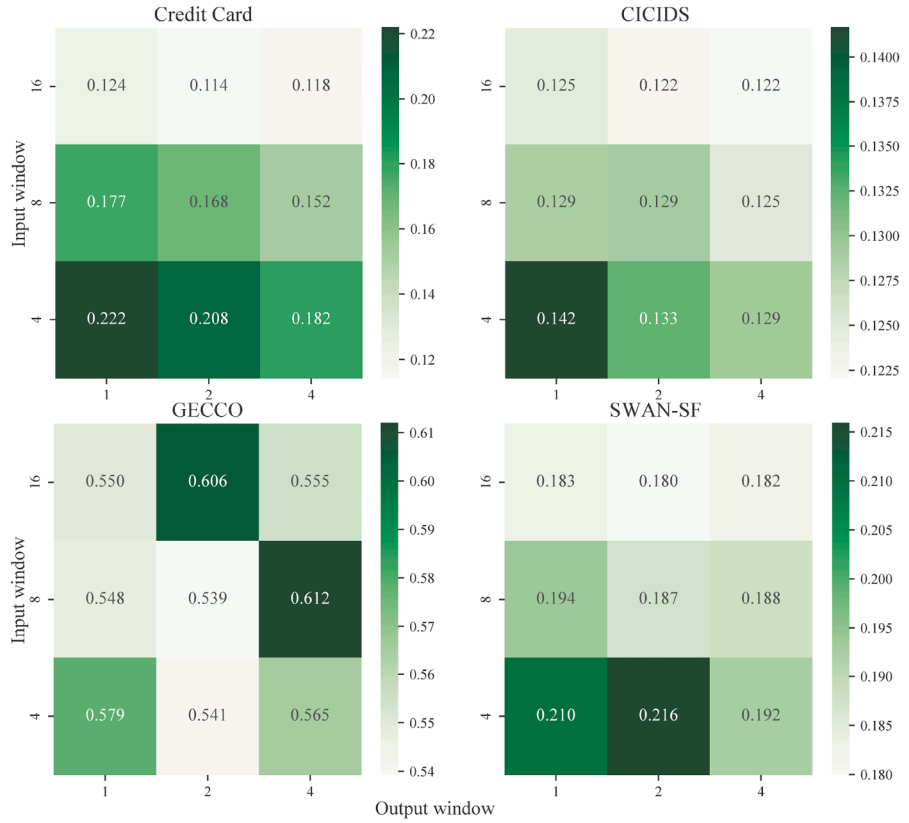

Fig. 8. F1-score of AD outcomes versus the input and output windows lengths on real datasets: in particular, we tested input windows of {4,8,16} time steps and output windows of {1,2,4} time steps.

Table 5
Guidelines for choosing thresholding strategy.

Scenario	Recommended strategy	Notes
Low-complexity data or single anomaly type	Static	Generalizable, no tuning needed
Multiple or overlapping anomaly types	Dynamic	Adapts threshold to dataset structure
No validation data or online detection	Static	Suitable for deployment with limited supervision
Offline setting with sufficient data	Dynamic	Improves F1-score at the cost of tuning

different anomaly lengths. It is interesting to note that, differently with respect to previous datasets, here the {4, 1} pair achieves still competitive results with respect to other combinations.

The same analysis conducted on synthetic datasets do confirm our assumptions, as proved by the results shown in Fig. 9. Our method succeeds in detecting point anomalies by employing shorter prediction windows while showing a slight preference for larger input windows. Pattern anomalies instead, as for the real case, prefer a broad combination of window lengths but still achieve better performance by utilizing smaller output window lengths.

6.3.4. Learning rate and scheduler

To assess the impact of the learning rate on anomaly detection performance, we conducted an ablation study using the LSTM model trained with five different learning rates: $1e-2$, $5e-3$, $1e-3$, $5e-4$, $1e-4$. For each learning rate, we also evaluated the use of a scheduler that reduces the learning rate by a factor of 0.9 when validation metrics do not improve for more than three consecutive epochs. The experiments were conducted on five synthetic datasets (0–4), each characterized by a distinct type of temporal anomaly. For each configuration, we measured the resulting F1 score in the anomaly classification task; the results are summarized in Fig. 10. The top row displays the individual F1 scores obtained for each dataset and learning rate, while the bottom row shows the mean and standard deviation across datasets. Results with and without scheduler are presented on the left and right, respectively.

As shown in the plots, the F1 score reaches its highest values when the initial learning rate is relatively large (e.g., $1e-2$), and progressively decreases as the learning rate becomes smaller. This suggests that larger learning rates may produce stronger and more expressive gradients, which in turn lead to better more informative influence scores for anomaly detection. Moreover, the results highlight a slight but consistent improvement when using a learning rate scheduler. This likely helps to balance fast convergence at the beginning with improved generalization later, which can enhance forecasting accuracy and, consequently, the reliability of the downstream anomaly classification method.

6.3.5. Checkpoint sampling strategy

As described in Section 3, our method computes element-wise influence scores by simulating the training dynamics through intermediate model checkpoints. However, the number and position of these checkpoints are not explicitly defined, and may affect the quality of the resulting anomaly classification. For this reason, we conducted a sensitivity analysis on six different checkpoint sampling strategies, presented in Table 6.

We evaluated each of these strategies using the LSTM model on four real-world datasets. To assess the robustness of each method across datasets of varying difficulty, we report both the raw mean F1 score and a normalized F1 score based on min–max normalization across datasets.

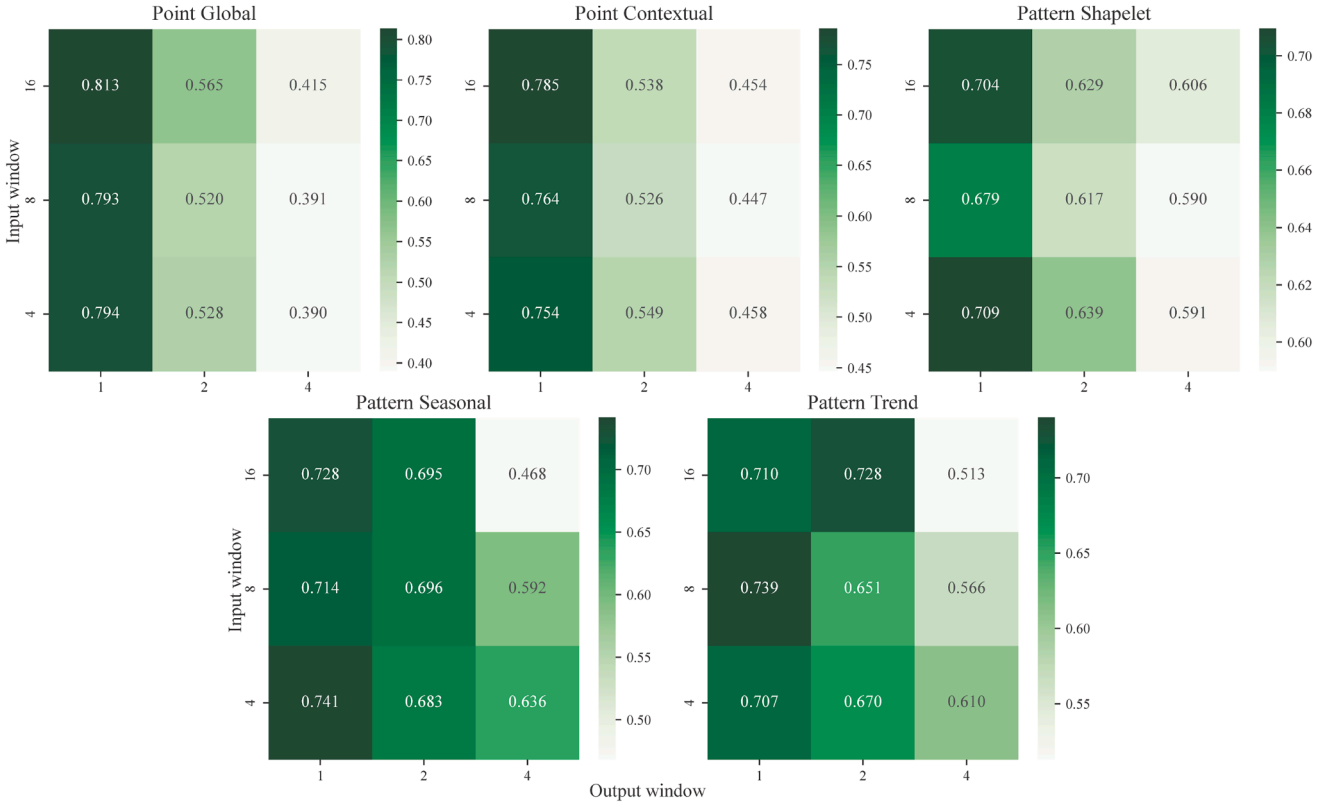

Fig. 9. F1-score of AD outcomes versus the input and output windows lengths on synthetic datasets: the same sizes of windows are considered as for real datasets.

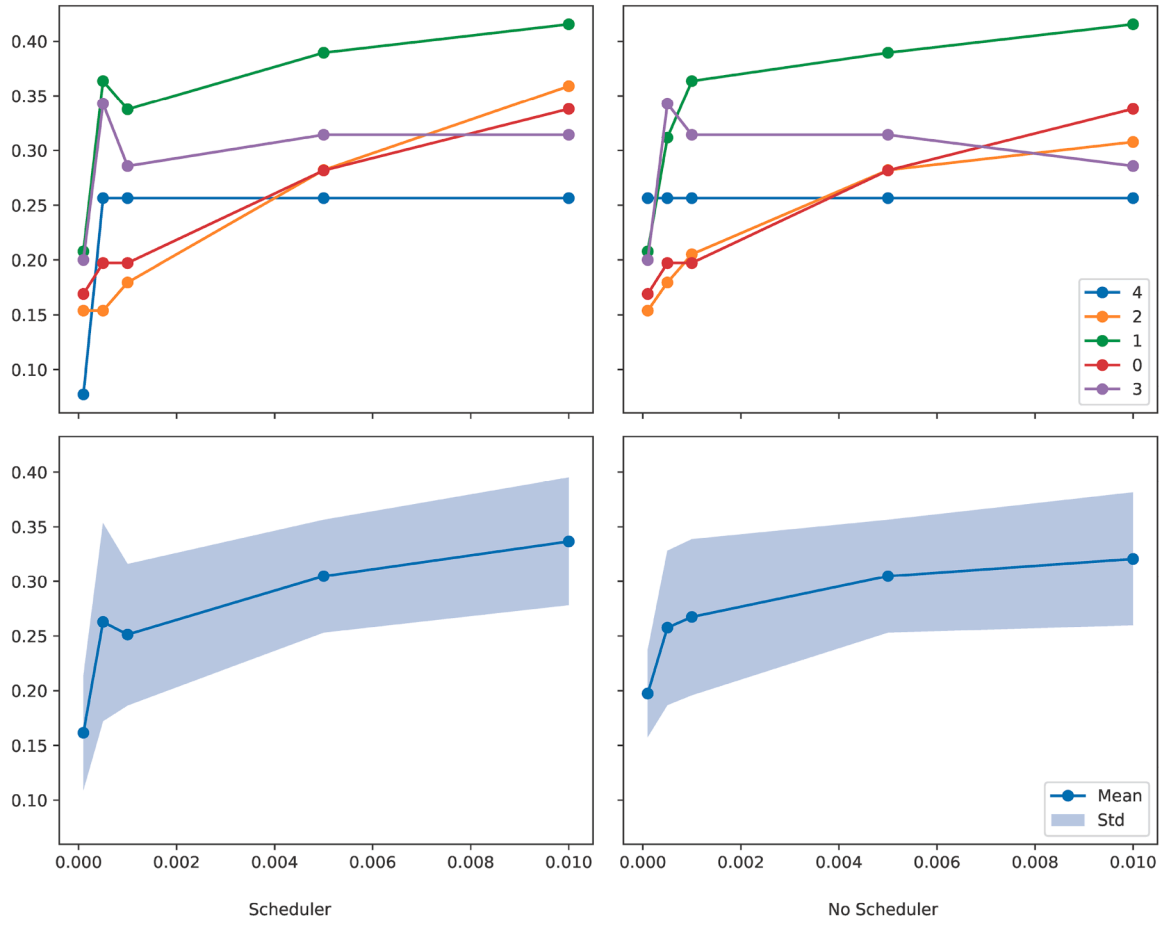

Fig. 10. Ablation study on the effect of different learning rates on final F1 scores and the use of a scheduler on anomaly detection performance using LSTM.

Table 6
Overview of checkpoint retention strategies.

Strategy	Checkpoint retention description
0	Retain all available checkpoints
1	Retain only the first and last checkpoint
2	Retain the first, last, and middle checkpoint
3	Retain the first and last, plus 25 % of central checkpoints
4	Sample exponentially from the beginning (more early checkpoints)
5	Sample exponentially from the end (more late checkpoints)

Table 7
Comparison of checkpoint sampling strategies based on Mean F1 and Min–Max Normalized F1 scores across four real-world datasets.

Strategy	Mean F1 Score	Normalized F1 Score
0 (All checkpoints)	0.2287	0.8875
1 (First & last)	0.2270	0.6441
2 (First, middle, last)	0.2015	0.5019
3 (First, last, +25 % mid)	0.2092	0.8219
4 (Exponential from start)	0.1300	0.0678
5 (Exponential from end)	0.2031	0.6028

The results are reported in Table 7. We observe that Strategy 0 (all checkpoints) achieves the best performance both in terms of absolute and normalized F1 score. Strategy 3, which retains a richer but still reduced set of checkpoints, also performs well, suggesting that some internal training dynamics are essential for effective attribution. By contrast, strategies that heavily compress the checkpoint sequence (such as Strategy 1, 2, and 4) show a noticeable drop in performance, likely due to the loss of temporal resolution in the attribution process.

7. Conclusion

Multivariate time series anomaly detection is an important task that is gathering increased attention over the years, due to many potential applications. In this paper, we faced this problem by introducing an innovative way to detect anomalies by employing influence functions: we developed a versatile framework, able to be employed in different scenarios and to discover efficiently different types of time series anomalies. We validated our method on both synthetic datasets, in order to understand more clearly the relationship between anomaly type and performance, and real datasets to assess its effective performance in a real context. Overall, we achieved competitive or higher results with respect to other classical unsupervised anomaly detection methods. This methodology has the potential for further improvement in the future, particularly in terms of computational cost, by exploring the trade-off between adaptability, to the various scenarios that may arise in real-world contexts, and training costs, which is essential to avoid that the model learns also anomalies.

Nonetheless, the impact that data attribution methods can have in the field of anomaly detection is very promising. These methods may also be integrated with a fully supervised method in order to achieve superior performance by exploiting the information encoded in the influence score.

CRediT authorship contribution statement

Alessio Verdone: Conceptualization, Methodology, Validation, Investigation, Writing – original draft, Software, Visualization; **Simone Scardapane:** Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Software, Visualization; **Massimo Panella:** Conceptualization, Methodology, Formal analysis, Data curation, Writing – original draft, Supervision.

Data availability

Public datasets have been used.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Massimo Panella reports financial support was provided by Government of Italy Ministry of University and Research. Alessio Verdone reports financial support was provided by Government of Italy Ministry of University and Research. Simone Scardapane reports financial support was provided by University of Rome La Sapienza. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The contribution of A. Verdone in this work was supported in part by the “ECS Rome Technopole” (FP6, Spoke 6) within the Italian “Piano Nazionale di Ripresa e Resilienza (PNRR)”, Mission 4 Component 2 Investment 1.5 funded by the European Union - NextGenerationEU - ECS 0000024 - CUP B83C22002820006.

The research of S. Scardapane in this work was funded in part by the University of Rome “La Sapienza”, grant no. RM1221816BD028D6 (DESMOS: DEsigning Self-Explainable AI MOdels for Scientific applications) and CHIST-ERA grant no. CHIST-ERA-19-XAI-009 (Multidisciplinary Use Cases for Convergent new Approaches to AI explainability).

The research of M. Panella in this work was funded in part by the Italian Ministry of University and Research (MUR), PRIN 2022 grant no. 2022HMYX2C (MESSI - Management Energy Systems for Smart Islands, CUP no. B53D23002650006).

References

Adiban, M., Siniscalchi, S. M., & Salvi, G. (2023). A step-by-step training method for multi generator GANs with application to anomaly detection and cybersecurity. *Neurocomputing*, *537*, 296–308. https://doi.org/10.1016/j.neucom.2023.03.056

Ahmed, M., Mahmood, A. N., & Islam, M. R. (2016). A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, *55*, 278–288. https://doi.org/10.1016/j.future.2015.01.001

Algburi, R. N. A., Aljibori, H. S. S., Al-Huda, Z., Gu, Y. H., & Al-antari, M. A. (2025). Advanced fault diagnosis in industrial robots through hierarchical hyper-laplacian priors and singular spectrum analysis. *Complex & Intelligent Systems*, *11*(6), 282.

Algburi, R. N. A., Gao, H., & Al-Huda, Z. (2021). Implementation of singular spectrum analysis in industrial robot to detect weak position fluctuations. *Fluctuation and Noise Letters*, *20*(03), 2150010.

Audibert, J., Michiardi, P., Guyard, F., Marti, S., & Zuluaga, M. A. (2020). Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 3395–3404).

Bianchi, F. M., Scardapane, S., Løkse, S., & Jenssen, R. (2020). Reservoir computing approaches for representation and classification of multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(5), 2169–2179.

Bontemps, L., Cao, V. L., McDermott, J., & Le-Khac, N.-A. (2016). Collective anomaly detection based on long short-term memory recurrent neural networks. In T. K. Dang, R. Wagner, J. Küng, N. Thoai, M. Takizawa, & E. Neuhold (Eds.), *Future data and security engineering*, Lecture notes in computer science (pp. 141–152). Cham: Springer International Publishing.

Brunet, M.-E., Alkalay-Houlihan, C., Anderson, A., & Zemel, R. (2019). Understanding the origins of bias in word embeddings. In *Proceedings of the 36th international conference on machine learning* (pp. 803–811). PMLR. ISSN: 2640–3498.

Ceschini, A., Rosato, A., Succetti, F., Luzio, F. D., Mitolo, M., Araneo, R., & Panella, M. (2021). Deep neural networks for electric energy theft and anomaly detection in the distribution grid. In *2021 IEEE international conference on environment and electrical engineering and 2021 IEEE industrial and commercial power systems Europe (EEEIC / i&CPS Europe)* (pp. 1–5). https://doi.org/10.1109/EEEIC/ICPSEurope51590.2021.9584796

Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, *41*(3), 15:1–15:58. https://doi.org/10.1145/1541880.1541882

Cook, R. D. (1977). Detection of influential observation in linear regression. *Technometrics*, *19*(1), 15–18. Publisher: Taylor & Francis _eprint:. https://doi.org/10.1080/00401706.1977.10489493

Copiaco, A., Himeur, Y., Amira, A., Mansoor, W., Fadli, F., Atalla, S., & Sohail, S. S. (2023). An innovative deep anomaly detection of building energy consumption using energy time-series images. *Engineering Applications of Artificial Intelligence*, *119*, 105775. https://doi.org/10.1016/j.engappai.2022.105775

Dai, Z., & Gifford, D. K. (2023). Training Data Attribution for Diffusion Models.

Deng, J., Li, T.-W., Zhang, S., & Ma, J. (2024). Efficient ensembles improve training data attribution. arXiv preprint arXiv:2405.17293

Ding, Z., & Fei, M. (2013). An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes*, *46*(20), 12–17. https://doi.org/10.3182/20130902-3-CN-3020.00044

Elsayed, S., Thyssens, D., Rashed, A., Jomaa, H. S., & Schmidt-Thieme, L. (2021). Do we really need deep learning models for time series forecasting?

Feldman, V., & Zhang, C. (2020). What neural networks memorize and why: discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, *33*, 2881–2891.

Gao, J., Song, X., Wen, Q., Wang, P., Sun, L., & Xu, H. (2021). RobustTAD: Robust Time Series Anomaly Detection via Decomposition and Convolutional Neural Networks. [cs, eess, stat] https://doi.org/10.48550/arXiv.2002.09545

Ghorbani, A., & Zou, J. (2019). Data shapley: Equitable valuation of data for machine learning. In *Proceedings of the 36th international conference on machine learning* (pp. 2242–2251). PMLR. ISSN: 2640–3498.

Gupta, M., Gao, J., Aggarwal, C. C., & Han, J. (2014). Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *26*(9), 2250–2267. https://doi.org/10.1109/TKDE.2013.184

Hampel, F. R. (1974). The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, *69*(346), 383–393. Publisher: Taylor & Francis _eprint:. https://doi.org/10.1080/01621459.1974.10482962

Hara, S., Nitanda, A., & Maehara, T. (2019). Data cleansing for models trained with SGD. In *Advances in neural information processing systems* (pp. 1–10). Curran Associates, Inc. (vol. *32*).

Hassija, V., Chamola, V., Mahapatra, A., Singal, A., Goel, D., Huang, K., Scardapane, S., Spinelli, I., Mahmud, M., & Hussain, A. (2024). Interpreting black-box models: A review on explainable artificial intelligence. *Cognitive Computation*, *16*, 45–74.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Iglesias, G., Talavera, E., González-Prieto, Á., Mozo, A., & Gómez-Canaval, S. (2023). Data augmentation techniques in time series domain: A survey and taxonomy. *Neural Computing and Applications*, *35*(14), 10123–10145.

Jaeckel, L. A. (1972). Estimating regression coefficients by minimizing the dispersion of the residuals. *The Annals of Mathematical Statistics*, *43*(5), 1449–1458. Publisher: Institute of Mathematical Statistics.

Jia, R., Dao, D., Wang, B., Hubis, F. A., Hynes, N., Gürel, N. M., Li, B., Zhang, C., Song, D., & Spanos, C. J. (2019). Towards efficient data valuation based on the shapley value. In *Proceedings of the twenty-second international conference on artificial intelligence and statistics* (pp. 1167–1176). PMLR. ISSN: 2640–3498 .

Karthikeyan, K., & Søgaard, A. (2021). Revisiting methods for finding influential examples. https://arxiv.org/abs/2111.04683.

Koh, P. W., & Liang, P. (2017). Understanding black-box predictions via influence functions. In *Proceedings of the 34th international conference on machine learning* (pp. 1885–1894). PMLR. ISSN: 2640–3498.

Koh, P. W., Steinhardt, J., & Liang, P. (2022). Stronger data poisoning attacks break data sanitization defenses. *Machine Learning*, *111*(1), 1–47. https://doi.org/10.1007/s10994-021-06119-y

Koh, P. W. W., Ang, K.-S., Teo, H., & Liang, P. S. (2019). On the accuracy of influence functions for measuring group effects. In *Advances in neural information processing systems* (pp. 1–11). Curran Associates, Inc. (vol. *32*).

Lai, K.-H., Zha, D., Xu, J., Zhao, Y., Wang, G., & Hu, X. (2021). Revisiting time series outlier detection: Definitions and benchmarks. In J. Vanschoren, & S. Yeung (Eds.), *Proceedings of the neural information processing systems track on datasets and benchmarks* (pp. 1–13). Curran (vol. *1*).

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE international conference on data mining* (pp. 413–422). ISSN: 2374–8486. https://doi.org/10.1109/ICDM.2008.17

Liu, Y., Li, Z., Zhou, C., Jiang, Y., Sun, J., Wang, M., & He, X. (2020). Generative adversarial active learning for unsupervised outlier detection. *IEEE Transactions on Knowledge and Data Engineering*, *32*(8), 1517–1528. Conference Name: IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2019.2905606

Lu, T., Wang, L., & Zhao, X. (2023). Review of anomaly detection algorithms for data streams. *Applied Sciences*, *13*(10), 6353. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute. https://doi.org/10.3390/app13106353

Ma, J., & Perkins, S. (2003). Time-series novelty detection using one-class support vector machines. In *Proceedings of the international joint conference on neural networks, 2003.* (pp. 1741–1745). (vol. *3*). ISSN: 1098–7576. https://doi.org/10.1109/IJCNN.2003.1223670

Namratha, M., Anusree, M. K., Niha, Pooja, S., & Arpana, M. R. (2023). Anomaly detection in medical IoT devices using federated learning. In T. Senjyu, C. So-In, & A. Joshi (Eds.), *Smart trends in computing and communications*, Lecture Notes in Networks and Systems (pp. 259–270). Singapore: Springer Nature. https://doi.org/10.1007/978-981-99-0769-4-25

Nguyen, E., Seo, M., & Oh, S. J. (2023). A Bayesian approach to analysing training data attribution in deep learning.

Park, S. M., Georgiev, K., Ilyas, A., Leclerc, G., & Madry, A. (2023). TRAK: Attributing model behavior at scale.

Pruthi, G., Liu, F., Kale, S., & Sundararajan, M. (2020). Estimating training data influence by tracing gradient descent. In *Advances in neural information processing systems* (pp. 19920–19930). Curran Associates, Inc. (vol. *33*).

Rosato, A., Araneo, R., Andreotti, A., Succetti, F., & Panella, M. (2021). 2-D convolutional deep neural network for the multivariate prediction of photovoltaic time series. *Energies*, *14*(9). https://doi.org/10.3390/en14092392

Rousseeuw, P. J., & Leroy, A. M. (2005). Robust regression and outlier detection. John Wiley & Sons.

Sakurada, M., & Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis, MLSDA'14* (pp. 4–11). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2689746.2689747

Schulam, P., & Saria, S. (2019). Can you trust this prediction? Auditing pointwise reliability after learning. In *Proceedings of the twenty-second international conference on artificial intelligence and statistics* (pp. 1022–1031). PMLR. ISSN: 2640–3498.

Schölkopf, B., & Smola, A. J. (2002). Learning with kernels: Support vector machines, regularization, optimization, and beyond. MIT Press. Google-Books-ID: y8ORL3DWt4sC.

Schölkopf, B., Williamson, R. C., Smola, A., Shawe-Taylor, J., & Platt, J. (1999). Support vector method for novelty detection. In *Advances in neural information processing systems* (pp. 1–7). MIT Press (vol. *12*).

Senin, P. (2009). Dynamic Time Warping Algorithm Review. https://seninp.github.io/assets/pubs/senin_dtw_litreview_2008.pdf.

Shaukat, K., Alam, T. M., Luo, S., Shabbir, S., Hameed, I. A., Li, J., Abbas, S. K., & Javed, U. (2021). A review of time-series anomaly detection techniques: A step to future perspectives. In K. Arai (Ed.), *Advances in information and communication* Advances in intelligent systems and computing (pp. 865–877). Cham: Springer International Publishing.

Shen, L., Li, Z., & Kwok, J. (2020). Timeseries anomaly detection using temporal hierarchical one-class network. *Advances in Neural Information Processing Systems*, *33*, 13016–13026.

Soule, A., Salamatian, K., & Taft, N. (2005). Combining filtering and statistical methods for anomaly detection. In *Proceedings of the 5th ACM SIGCOMM conference on Internet measurement - IMC '05* (p. 1). Berkeley, California: ACM Press. https://doi.org/10.1145/1330107.1330147

Steinwart, I., Hush, D., & Scovel, C. (2005). A classification framework for anomaly detection. *Journal of Machine Learning Research*, *6*(8), 211–232.

Stracqualursi, E., Rosato, A., Di Lorenzo, G., Panella, M., & Araneo, R. (2023). Systematic review of energy theft practices and autonomous detection through artificial intelligence methods. *Renewable and Sustainable Energy Reviews*, *184*, 113544. https://doi.org/10.1016/j.rser.2023.113544

Suh, W. H., Oh, S., & Ahn, C. W. (2023). Metaheuristic-based time series clustering for anomaly detection in manufacturing industry. *Applied Intelligence*, *53*(19), 21723–21742. https://doi.org/10.1007/s10489-023-04594-5

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2022). Tracinad: Measuring influence for anomaly detection. In *2022 international joint conference on neural networks (IJCNN)* (pp. 1–6). IEEE.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., & Gomez (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 1–11). Curran Associates, Inc. (vol. *30*).

Wang, H., Ustun, B., & Calmon, F. (2019). Repairing without retraining: Avoiding disparate impact with counterfactual distributions. In *Proceedings of the 36th international conference on machine learning* (pp. 6618–6627). PMLR. ISSN: 2640–3498.

Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in neural information processing systems* (pp. 22419–22430). Curran Associates, Inc. (vol. *34*).

Xu, H., Pang, G., Wang, Y., & Wang, Y. (2023). Deep isolation forest for anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, *35*(12), 12591–12604. Conference Name: IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2023.3270293

Xu, J., Wu, H., Wang, J., & Long, M. (2022). Anomaly transformer: Time series anomaly detection with association discrepancy. https://arxiv.org/abs/2110.02642.

Yeh, C.-C. M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H. A., Silva, D. F., Mueen, A., & Keogh, E. (2016). Matrix profile I: All pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)* (pp. 1317–1322). ISSN: 2374–8486. https://doi.org/10.1109/ICDM.2016.0179

Yeh, C.-K., Kim, J., Yen, I. E.-H., & Ravikumar, P. K. (2018). Representer point selection for explaining deep neural networks. In *Advances in neural information processing systems* (pp. 1–11). Curran Associates, Inc. (vol. *31*).

Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., & Chawla, N. V. (2019). A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1409–1416). AAAI Press (vol. *33*). https://doi.org/10.1609/aaai.v33i01.33011409

Zhang, R., Zhang, S., Lan, Y., & Jiang, J. (2008). Network anomaly detection using one class support vector machine. *Hong Kong*.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(12), 11106–11115. Number: 12. https://doi.org/10.1609/aaai.v35i12.17325

Zhou, J., Li, Z., Hu, H., Yu, K., Chen, F., Li, Z., & Wang, Y. (2019). Effects of influence on user trust in predictive decision making. In *Extended abstracts of the 2019 CHI conference on human factors in computing systems, CHI EA '19* (pp. 1–6). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3290607.3312962

Zhu, Y., Yeh, C.-C. M., Zimmerman, Z., Kamgar, K., & Keogh, E. (2018). Matrix profile XI: SCRIMP + +: Time series motif discovery at interactive speeds. In *2018 IEEE international conference on data mining (ICDM)* (pp. 837–846). ISSN: 2374–8486. https://doi.org/10.1109/ICDM.2018.00099

Zolhavarieh, S., Aghabozorgi, S., & Teh, Y. W. (2014). A review of subsequence time series clustering. *The Scientific World Journal*, *2014*, e312521. Publisher: Hindawi. https://doi.org/10.1155/2014/312521

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., & Chen, H. (2018). Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations* (pp. 1–19). https://openreview.net/forum?id = BJJLHbb0-.