A Stitch in Time Saves Nine: Enabling Early Anomaly Detection with Correlation Analysis

Conference Paper · April 2023

DOI: 10.1109/ICDE55515.2023.00143

4 authors, including:

Qiang Huang
National University of Singapore
30 PUBLICATIONS **471** CITATIONS
SEE PROFILE

Anthony Tung
National University of Singapore
211 PUBLICATIONS **8,769** CITATIONS
SEE PROFILE

Zhiyong Huang
National University of Singapore
82 PUBLICATIONS **1,268** CITATIONS
SEE PROFILE

A Stitch in Time Saves Nine: Enabling Early Anomaly Detection with Correlation Analysis

Yihao Ang,[†‡] Qiang Huang,[†*] Anthony K. H. Tung,[†] Zhiyong Huang[†‡]
[†]School of Computing, National University of Singapore, Singapore
[‡]NUS Research Institute in Chongqing, Chongqing, China
{yihao_ang, huangq, atung, huangzy}@comp.nus.edu.sg

***Abstract*—Early detection of anomalies from sensor-based Multivariate Time Series (MTS) is vital for timely response to the signs of operation failures and errors. While many interesting works have been done toward solving this problem, existing methods typically detect such anomalies as outliers by making certain assumptions that allow efficient and easily understandable solutions to be used but might not be applicable to time series. Meanwhile, unsupervised deep learning-based methods might be highly accurate but often lead to challenges for real-time industrial scenarios, e.g., requiring a large amount of training data and producing unstable output.**

In this paper, we propose a new approach, CAD, to detect anomalies from sensor-based MTS. We aim to leverage the latent correlations between sensors by first converting the MTS into a sequence of Time-Series Graphs (TSGs) that connect sensors to their highly correlated neighbors within a certain time period. Then, we track the unusual correlation variations between sensors on the sequence of TSGs. By analyzing the correlation variations with a theoretical guarantee, CAD can detect the time of occurrence for the anomalies simultaneously with the sensors that are affected as early as possible.

Extensive experiments over eight real-world datasets show that CAD is effective, scalable, yet stable compared to nine state-of-the-art methods while keeping comparable efficiency. Moreover, it maintains above 85% accuracy on large-scale datasets with over 1,000 sensors. Notably, CAD can determine relevant sensors in a very early stage of the anomaly so that timely predictive maintenance can be done. The code is available at https://github.com/YihaoAng/CAD.

***Index Terms*—Anomaly Detection, Multivariate Time Series, Outlier Detection, Predictive Maintenance, Correlation Analysis**

I. Introduction

Sensor networks have extensive applications in industries, including monitoring manufacturing processes for assembly lines [56], [68] and recording the operating states of critical infrastructures such as wind turbines [25]. Within these applications, one important function of sensor networks is to facilitate predictive maintenance by detecting anomalies from sensor-based Multivariate Time Series (MTS) as early as possible, which is vital for the timely discovery and restoration of failures before severe damages can occur. In this paper, we investigate the problem of unsupervised anomaly detection over sensor-based MTS. We aim to detect two things from MTS: (1) the time period of anomaly occurrence (abnormal time) and (2) the affected components (abnormal sensors). Together, these two things can provide valuable information for predictive maintenance.

Doing so, however, can be challenging. First, since the MTS is generated from a heterogeneous set of sensors measuring readings from physical entities subjected to wear and tear and usually contains noise [88], [90], trying to relate a fixed set of patterns or models in the MTS that correspond to both normal and abnormal events is difficult. Second, the high sampling rate of the sensors results in a high-velocity MTS, which must be processed efficiently to ensure real-time anomaly detection. For example, the sampling rate of sensors from a water treatment plant is around 1 Hz [29], [57], while the ones from a bearing system can be up to 25.6k Hz [82]. Third, only detecting the anomalies is not adequate for achieving timely predictive maintenance. A (small) failure in a machine might propagate and spread to other nearby components over time, leading to serious damages eventually [23], [34], [37]. Thus, once the anomaly occurs, it is imperative to alarm and take action to avoid further system deterioration.

Over the past few decades, the problem of unsupervised anomaly detection has attracted significant research attention. Many pioneer works have been proposed to improve both accuracy and efficiency [4], [15], [35], [44], [60], [69], [86], [87]. These works can roughly be divided into two categories: data mining-based methods and unsupervised deep learning-based methods. The idea of data mining-based methods [15], [44], [50], [69], [74] comes from outlier detection and is usually based on specific assumptions of input data, e.g., a data point is treated as an outlier if it is substantially different from other points [4]. Even though they are efficient and easy to interpret, their specific assumptions limit them from supporting other kinds of data. Also, most of them only detect outliers (without time points), so they might not explicitly consider the temporal dependency of the time series and fail to detect the anomalies in an early stage.

In recent years, due to the prevalence of deep neural networks, many unsupervised deep learning-based methods have been proposed for anomaly detection [3], [9], [26], [28], [35], [49], [60], [77], [79], [86], [87]. Since they extract latent features and utilize temporal dependencies, most methods have high detection accuracy on benchmark datasets. Nevertheless, deploying them in real-world industries for timely predictive maintenance is difficult. The reasons are four folds. First, these methods usually need many training data and a long time for

*Corresponding author

model training. Moreover, in many real-world applications, the data distributions often change constantly. To detect anomalies precisely, they require retraining to adapt to the new data distribution. The time for multiple retrains should also be considered. Thus, they are typically less efficient than data mining-based methods. Second, as they require training models with well-defined data, they might be unstable when dealing with noisy real-world data or the data without (enough) ground truth labels to support finetuning [49], [77]. Third, most of them aim to only detect the periods of anomalies without identifying the relevant sensors, which makes it challenging to localize the root cause of the anomalies. Finally, many methods only report an anomaly when significant abnormal patterns occur, while others simply output a probability for each time point to be abnormal. However, in the early stage of an anomaly, it might affect only a small number of sensors. Therefore, they might not be sensitive to these small changes and cannot discover the anomalies precisely in time.

In this work, we aim to identify the periods of anomalies as well as the relevant sensors. In many predictive maintenance applications, sensors are usually installed on a machine (e.g., an assembly line in a factory) with certain relationships [46]. We argue that correlations exist between sensors in most sensor networks based on the following two pieces of evidence: (1) In a machine, different components typically affect and interact with each other, resulting in correlations between the sensors installed on these components [49], [77]. (2) Many sensors have high spatial proximity as they are deployed on the same machine. Thus, the time series collected by the nearby sensors are often correlated [5], [6], [17].

Contributions. In this paper, we propose a new Correlation analysis-based Anomaly Detection method named CAD for sensor-based MTS. The central intuition is to monitor the correlation between sensors and identify anomalies by looking at the changes in correlation between them. To do so, we first transform the MTS into a series of Time Series Graphs (TSGs), where each vertex corresponds to a single sensor, and an edge between two vertices represents the correlation between the time series of the two sensors within a time period. TSGs only connect sensors to their highly correlated neighbors. To detect the anomalies, we first find subsets of affected vertices that have changed correlations to their neighbors; then, we track the unusual correlation variations of these affected vertices through the sequence of TSGs. Finally, based on the analysis of the correlation variations with a theoretical guarantee, CAD discovers the time period when the anomaly happens together with the affected sensors.

Moreover, CAD is practical yet reliable for early anomaly detection from noisy data. Once the time series contain unusual changes at a specific time point, the correlation of the affected sensors with other sensors will change accordingly. As a result, the edges related to these affected vertices on the TSG will change immediately. Thus, a small abnormal pattern in the sensor-based MTS can be reflected by the affected vertices on TSG once the anomaly occurs. Nonetheless, as the MTS often contains noise, it might not be accurate to report anomalies based on (a few) affected vertices. Therefore, we track the correlation variations through the sequence of TSGs, and only the time when many vertices have significant variations will be considered an actual anomaly. Thus, CAD can promptly reveal anomalies from noisy real-world data.

Point Adjustment (PA) is a commonly-used evaluation for anomaly detection [9], [77], [83]. Nevertheless, PA overlooks the chronological order of the time period of an anomaly, which cannot precisely measure how early one method detects the anomaly. Moreover, it can only assess a single method, but a relative comparison is also a vital need in industry [40]. To evaluate the ability of early anomaly detection and compare different anomaly detection methods, we propose a new Delay-aware Evaluation (DaE) scheme beyond PA. The DaE scheme consists of two parts: (1) the Delay-Point Adjustment (DPA) to reflect the ability to detect anomalies as early as possible and (2) the relative evaluation $Ahead$ and $Miss$ to compare the results of two anomaly detection methods.

Based on the DaE, we conduct a systematic comparison of CAD with three efficient data mining-based methods LOF [15], ECOD [48], and IForest [50], two state-of-the-art unsupervised deep learning-based methods USAD [9] and RCoders [2], [3], and four advanced univariate anomaly detection methods Series2Graph (S2G) [13], SAND [14], its online version SAND*, and NormA [12] over eight real-world datasets to verify the effectiveness, efficiency, robustness, and scalability of CAD. As a deterministic method, CAD achieves a better (or similar) detection performance than the nine benchmark methods while keeping comparable efficiency. Furthermore, CAD can support real-time anomaly detection yet maintain $F1_{DPA} > 85\%$ on large-scale datasets. Most importantly, CAD can detect the anomalies in a very early stage on noisy real-world data.

Organization. The rest of this paper is organized as follows. Section II surveys related work. Section III formulates the problem of anomaly detection. We present CAD in Section IV. The DaE scheme is introduced in Section V. Experimental results are reported and analyzed in Section VI. Finally, we conclude this work in Section VII.

II. Related Work

Anomaly detection is a fundamental problem with extensive studies. A comprehensive survey can be referred to [20], [59], [72]. As this work focuses on unsupervised anomaly detection, we briefly review three types of representative methods: data mining-based methods, univariate anomaly detection methods, and unsupervised deep learning-based methods.

Data Mining-based Methods. These methods usually leverage similarity measures and statistics for outlier detection [4], [7], [8], [30], [43], [69], [74]. For example, [69] finds k nearest neighbors given a data point and calculates the anomaly score based on the similarity towards its neighbors. LOF [15], [33], [61] has been proposed to detect local outliers from sparse data. It splits the whole space into several areas with different

densities. Data points that have significantly low densities are regarded as outliers. Another kind of method is based on linear models. PCA [4], [76] transforms original data into a low-dimensional space and adopts the degree of deviations of data along each dimension in the transformed space to detect anomalies. OC-SVM [74] extends the SVM classifier on unlabeled data to detect outliers. Other methods [30], [31], [44], [47] are based on probabilistic models and have certain assumptions on data distributions. For instance, ECOD [48] estimates tail probabilities and assumes outliers are located in the tails of the distributions. Recently, ensemble methods [10], [67], [91] have been proposed to improve the robustness of anomaly detection, e.g., IForest [50] separates an anomaly from other normal points based on the assumption that anomalies are minorities and have different values, which performs well on high-dimensional data. One of the common grounds of CAD with most existing data mining-based methods is that they detect anomalies by simple rules on data. Moreover, CAD captures the changes in correlations and utilizes the changes in community structures to detect anomalies in an early stage.

Univariate Anomaly Detection Methods. Besides the data mining-based methods for anomaly detection on MTS, some other methods focus on detecting abnormal time periods on Univariate Time Series (UTS) [12]–[14], [38], [42], [75], [78], [80], [84], [85]. For example, S2G [13] uses a graph representation of the data to identify unusual patterns and detect anomalies of various lengths. SAND [14] leverages k-Shape clustering [63] and statistical analysis to maintain a set of weighted subsequence and detects anomalies at every time point. It can be extended into the online setting to detect anomalies from data streams. NormA [12] first builds a normal model to represent the normal behavior from the input. Then, it computes the distance of each time point to the normal pattern and detects the abnormal ones. Although they focus on detecting anomalies from UTS input, they can be extended into the MTS setting for anomaly detection. Similar to CAD, univariate methods are mainly unsupervised and usually do not require additional training data.

Unsupervised Deep Learning-based Methods. These methods have been proposed to deeply exploit data information for anomaly detection, which can be divided into two major categories. (1) Feature Extraction [28], [35], [79], [87]: Deep learning models are designed to extract latent features from input data. Then, vanilla data mining-based methods are applied for anomaly detection. (2) End-to-end Detection: These methods follow a two-step approach: they first learn the (feature) representations from input data; then, they utilize the reconstruction error [16], [19], [36], [41], [53], [59], [71], [81], prediction error [32], [39], [51], [52], [54], [88], or adversarial learning [73], [86], [93] to identify the potential anomalies.

Most deep learning-based methods discussed above are developed for general purposes. Recently, there exist some methods tailored for time series anomaly detection [3], [9], [49], [77], [88]. The primary contributions of these methods lie in novel network architectures, loss functions, or training strategy design. Their intuitions mostly come from the inherent characteristics of MTS. For example, USAD [9] is an autoencoder-based method that adopts a two-stage adversarial training process to minimize both reconstruction error and the difference between two autoencoders; RCoders [3] is based on the observations that most real-world time series exhibit asynchronous and consistent repetitive variations. These methods usually treat each time point individually or extract features implicitly by neural networks. In contrast, CAD explicitly considers the features from time series by analyzing and tracking how outliers change across time.

TABLE I
THE SUMMARY OF COMMONLY USED NOTATIONS IN THIS PAPER.

Notations	Descriptions
$\boldsymbol{T}$, n	an MTS $\boldsymbol{T}$ with n sensors, i.e., $\boldsymbol{T} = (\boldsymbol{s}_1, \cdots, \boldsymbol{s}_n)^\mathsf{T}$
w, s, R	sliding window, step, number of rounds
$G_r = (V, E_r)$	a TSG G_r in round r with vertices V and edges E_r
k, τ	number of nearest neighbors, correlation threshold
V_Z, R_Z	abnormal sensors, abnormal time
ξ	abnormal time threshold
$C_{r,c}$	the c-th community in round r
$S_r(\boldsymbol{v})$	co-appearance number of vertex $\boldsymbol{v}$ in round r
$RC_{\boldsymbol{v},r}$	ratio of co-appearance number of vertex $\boldsymbol{v}$ in round r
O_r, θ	outlier set in round r, outlier threshold
n_r	number of outlier variations

III. Problem Formulation

In this section, we formulate the problem of anomaly detection. The commonly used notations throughout this paper are summarized in Table I.

A. Anomaly Detection in MTS

A Multivariate Time Series (MTS) $\boldsymbol{T}$ with n sensors is often represented as a matrix, i.e., $\boldsymbol{T} = (\boldsymbol{s}_1, \cdots, \boldsymbol{s}_n)^\mathsf{T}$. Let $|\boldsymbol{T}|$ be the length of the time series of $\boldsymbol{T}$. Then, each sensor $\boldsymbol{s}_i$ $(1 \leq i \leq n)$ can be denoted as a $|\boldsymbol{T}|$-dimensional vector, i.e., $\boldsymbol{s}_i = (x_{i,1}, \cdots, x_{i,|\boldsymbol{T}|})$, where $x_{i,j}$ $(1 \leq j \leq |\boldsymbol{T}|)$ is a sensor reading of $\boldsymbol{s}_i$ from a single time point t_j. For simplicity, we assume that the time interval between any two consecutive time points t_j and t_{j+1} is the same.

Operation states in MTS can be of two types: normal state and abnormal state (anomaly). An anomaly comprises two parts: *abnormal sensors* and *abnormal time*. Abnormal sensors are the sensors that result in the anomaly being detected and which are likely to link to the root cause of the anomalous events. The abnormal time corresponds to the time period in which sensor readings are considered abnormal.

Given an MTS $\boldsymbol{T}$ with n sensors, the goal of anomaly detection is to find l sub-matrices out of $\boldsymbol{T}$ as anomalies, where l is the number of anomalies. Each sub-matrix represents an *anomaly* Z, where each column denotes a single abnormal time point; each row corresponds to the time series of an abnormal sensor resulting in the anomaly. To better represent the anomalies, the sub-matrix can be formed from non-adjacent rows of $\boldsymbol{T}$, but it should be formed from the adjacent columns to keep the abnormal time consecutive.

Example 1: Figure 1 shows an MTS $\boldsymbol{T}$ with $n = 4$ sensors, where $|\boldsymbol{T}| = 8$. The sensor readings of $\boldsymbol{s}_4$ in t_7 and t_8 drop significantly compared to those from t_1 to t_6. Thus, the sub-matrix $(20\ 20)$ is an anomaly Z, where the abnormal time is from t_7 to t_8, and the abnormal sensor is $\boldsymbol{s}_4$. △

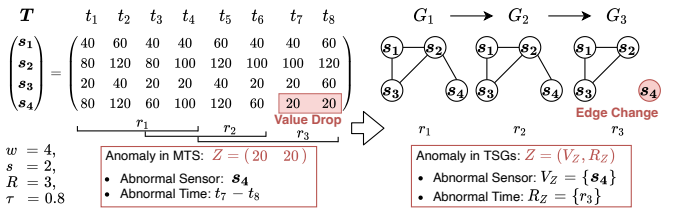

Fig. 1. From MTS to TSGs: an example.

B. From MTS to TSGs

MTS Partitioning. The MTS is usually very long, but the anomalies often occur within a short period. It is hard to detect anomalies directly from the long MTS. Thus, given a sliding window w and a step s ($s < w$), we first partition the long MTS $\boldsymbol{T}$ into a series of short sub-matrices $\{\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3, \cdots\}$, where $\boldsymbol{T}_1 = \boldsymbol{T}[1 : w]$, $\boldsymbol{T}_2 = \boldsymbol{T}[1+s : w+s]$, $\boldsymbol{T}_3 = \boldsymbol{T}[1+2s : w+2s]$, etc. Each sub-matrix $\boldsymbol{T}_i$ has the same length w.

It should be noted that sometimes $(|\boldsymbol{T}| - w) \not\equiv 0 \bmod s$. For this case, since $s < w \ll |\boldsymbol{T}|$, we remove the last few columns of $\boldsymbol{T}$ to make it divisible. For ease of illustration, we assume that $R = (|\boldsymbol{T}| - w)/s + 1$. Then, we convert the MTS $\boldsymbol{T}$ into R overlapping sub-matrices $\{\boldsymbol{T}_1, \cdots, \boldsymbol{T}_R\}$.

Time Series Graph (TSG). The sub-matrix $\boldsymbol{T}_r$ ($1 \le r \le R$), however, only shows the time series of each individual sensor. To monitor the correlations between sensors and detect the anomalies by the unusual correlation variations, we first convert each $\boldsymbol{T}_r$ into a k-Nearest Neighbor Graph (k-NNG) $G_r = (V, E_r)$, where V is a set of vertices with each of which is corresponding to a single sensor, i.e., $n = |V|$; E_r is a set of edges that connect each vertex to its k highest correlated neighbors based on Pearson Correlation [66]. Moreover, we represent G_r as a weighted graph, where the weight $\omega(e)$ of each edge $e = (\boldsymbol{u}, \boldsymbol{v}) \in E_r$ is the Pearson Correlation of the time series of the two sensors $\boldsymbol{u}, \boldsymbol{v}$ in $\boldsymbol{T}_r$.

Nonetheless, some vertices in G_r still have weak correlations within their k neighbors. Such edges might cause frequent changes in consecutive rounds, leading to extra noise to the actual correlations between sensors. To detect anomalies precisely, we further prune the edge e whose absolute weight is less than a correlation threshold τ, i.e., $|\omega(e)| < \tau$. Note that this step is necessary, especially when we encounter a large number of sensors, which allows CAD only to keep track of the strong correlations between sensors and perform real-time anomaly detection. For each $\boldsymbol{T}_r$, we call the k-NNG G_r after pruning as a *Time Series Graph* (TSG). As such, we convert the long MTS $\boldsymbol{T}$ into R rounds of TSGs $\{G_1, \cdots, G_R\}$.

Example 2: The top of Figure 1 displays an example of transforming an MTS $\boldsymbol{T}$ into $R = 3$ rounds of TSGs $\{G_1, G_2, G_3\}$. Given a sub-matrix $\boldsymbol{T}_r$, once there exist unusual changes in some rows (i.e., the time series of sensor $\boldsymbol{s}_4$), as their correlations with other sensors will change accordingly, the edges related to those vertices in G_r (i.e., edge $e = (\boldsymbol{s}_2, \boldsymbol{s}_4)$) will change immediately. Thus, in the form of TSGs, we can achieve the same detection goal as using the form of MTS yet detect the anomalies as early as possible.

C. Anomaly Detection in TSGs

In the form of TSGs, we use the affected vertices to represent abnormal sensors and leverage the consecutive rounds with abnormal sensors to denote the abnormal time. We first provide an overview of the anomaly as below.

Definition 1 (Anomaly)**:** *Given a set of TSGs $\{G_1, \cdots, G_R\}$, an anomaly Z is defined by a pair (V_Z, R_Z), i.e.,*

$$Z = (V_Z, R_Z),$$

where $V_Z \subseteq V$ are all vertices affected by the anomaly; R_Z is a set of consecutive rounds when the anomaly occurs.

Definition 1 only illustrates an overall representation of the anomaly in TSGs. For a certain round r, the TSG G_r can be split into c_r disjoint subgraphs $\{C_{r,c}\}_{c=1}^{c_r}$. We say these c_r subgraphs are c_r communities if the vertices from the same subgraphs are densely connected, while those from different subgraphs only have a loose connection. A formal definition of the community structure can be referred to [27], [58].

Various studies have shown that sensor networks commonly present community structures [1], [18], [21], [22], [89]. Thus, the TSGs constructed by sensor networks often exhibit community structures. With the community structures, one can determine whether the connectivity of different vertices is stable by detecting their changes in communities in two consecutive rounds. Based on this insight, we add specific constraints to (V_Z, R_Z), leading to Definitions 2 and 3, respectively.

Definition 2 (Affected Vertices)**:** *Given a TSG $G_r = (V, E_r)$ in round $r \in (1, R]$ with c_r communities, the subset of vertices $V_r \subseteq V$ are affected vertices if every vertex $\boldsymbol{v} \in V_r$ moves in or moves out the community $C_{r,c}$ in two consecutive rounds, i.e., $V_r = \{\boldsymbol{v} \in V \mid (\boldsymbol{v} \notin C_{r-1,c}$ and $\boldsymbol{v} \in C_{r,c})$ or $(\boldsymbol{v} \in C_{r-1,c}$ and $\boldsymbol{v} \notin C_{r,c})\}$.*

According to Definition 2, we denote each vertex $\boldsymbol{v} \in V_r$ as one *variation* if it changes across two consecutive rounds. Thus, one can find some consecutive rounds that contain a larger number of variations than others and regard this period as abnormal time. Formally,

Definition 3 (Abnormal Time and Abnormal Sensors)**:** *Given an abnormal time threshold ξ ($1 \le \xi \le n$) and R rounds of affected vertices $\{V_1, \cdots, V_R\}$, the abnormal time R_Z is a set of consecutive rounds such that for $\forall r \in R_Z$, the number of variations is at least ξ, i.e., $|V_r| \ge \xi$; the abnormal sensors V_Z are the union of the affected vertices V_r for $\forall r \in R_Z$.*

In Definition 3, a round r is regarded as abnormal if there exist many affected vertices. In the worst case, all vertices could be affected. Thus, the range of ξ is $1 \le \xi \le n$. Note that

Algorithm 1: OutlierDetection

Input: An MTS $\boldsymbol{T}_r$ of round r, an outlier set O_{r-1} of round $(r-1)$, outlier threshold θ;

1 Convert $\boldsymbol{T}_r$ into a TSG G_r;
2 Apply Louvain [11] on G_r to get c_r communities;
3 $O_r \leftarrow \emptyset$;
4 **foreach** $\boldsymbol{v} \in V$ **do**
5 Compute $S_r(\boldsymbol{v})$ and $RC_{\boldsymbol{v},r}$ by Equations 2 and 3;
6 **if** $RC_{\boldsymbol{v},r} < \theta$ **then** $O_r \leftarrow O_r \cup \{\boldsymbol{v}\}$;
7 Compute n_r according to Definition 8;
8 **return** $\{O_r, n_r\}$;

ξ is a key parameter for anomaly detection. We will explain how to set ξ for CAD in Subsection IV-E.

Example 3: The bottom of Figure 1 shows that each abnormal sensor $\boldsymbol{v} \in V_Z$ corresponds to a single row of the MTS $\boldsymbol{T}$, and R_Z is the time period when the anomaly occurs. Thus, the anomaly in TSGs is consistent with the abnormal sub-matrix in MTS. For example, in the form of MTS, the abnormal sensor is $\boldsymbol{s}_4$, and the abnormal time is from t_7 to t_8, whereas in the form of TSGs, $V_Z = \{\boldsymbol{s}_4\}$ and $R_Z = \{r_3\}$.

IV. Method

A. Overview

We now present CAD for anomaly detection. As discussed in Subsection III-B, the MTS has been converted into a series of TSGs. CAD first detects potential outliers one round at a time. For each round, the outlier detection has three phases:

- In Phase 1, we detect different communities from the TSG based on the correlations of vertices (Subsection IV-B).
- In Phase 2, we mine the co-appearance relationship of each vertex and other vertices in every two consecutive rounds and discover outliers (Subsection IV-C).
- In Phase 3, we define the number of outlier variations and analyze its properties. (Subsection IV-D).

The pseudo-code of the outlier detection process is shown in Algorithm 1. It returns the outlier set and the number of outlier variations. Based on the analysis of the number of outlier variations with a theoretical guarantee, CAD detects the abnormal time together with the abnormal sensors. An overview of CAD is depicted in Figure 2.

B. Phase 1: Community Detection

For each round $r \in [1, R]$, we first adopt Louvain [11] to partition the TSG G_r into c_r communities such that the vertices that are more correlated than others are split into the same communities (Line 2). With the communities, we can track the unusual correlation variations of vertices. Among numerous methods for community detection, we choose Louvain as it is a prevalent yet very efficient method, which takes $O(n \log n)$ time only [11]. Figure 2 displays the first 10 rounds of TSGs with 12 sensors. After Phase 1, each TSG has been partitioned into three communities, but the communities in different rounds can be different. When an anomaly occurs, although the sensors might not exhibit significant abnormal behaviors initially, their correlations are likely to break down, leading to changes in communities. Next, we will mine the co-appearance relationship of vertices from their communities.

C. Phase 2: Co-appearance Mining

We first introduce the concept of *co-appearance*. We say two vertices co-appeared if they are in the same communities in both rounds $(r-1)$ and r. Formally,

Definition 4 (Co-appearance)**:** *Given a round $r \in (1, R]$ and a vertex $\boldsymbol{v}$ with its previous community $C_{r-1,c}$ and current community $C_{r,c'}$, the co-appearance $S_r(\boldsymbol{v}, \boldsymbol{u})$ of $\boldsymbol{v}$ and each $\boldsymbol{u} \in V$ is computed as below:*

$$S_r(\boldsymbol{v}, \boldsymbol{u}) = \begin{cases} 1, & \text{if } \boldsymbol{u} \in C_{r-1,c} \text{ and } \boldsymbol{u} \in C_{r,c'} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Definition 4 depicts one co-appearance of two vertices in two consecutive rounds. One can monitor the community modifications of all vertices in each round by computing the co-appearance of each vertex and other vertices. Next, we define the *co-appearance number* $S_r(\boldsymbol{v})$ of a vertex $\boldsymbol{v}$ to count its total number of co-appeared vertices.

Definition 5 (Co-appearance Number)**:** *The $S_r(\boldsymbol{v})$ of each vertex $\boldsymbol{v} \in V$ in a round $r \in (1, R]$ is computed as follows:*

$$S_r(\boldsymbol{v}) = \textstyle\sum_{\boldsymbol{u} \in V \wedge \boldsymbol{u} \neq \boldsymbol{v}} S_r(\boldsymbol{v}, \boldsymbol{u}). \tag{2}$$

The co-appearance number $S_r(\boldsymbol{v})$ of each vertex $\boldsymbol{v} \in V$ in different rounds could be different, especially when there exists an anomaly. Thus, we define the *Ratio of Co-appearance Number* (RC) to analyze how the co-appearance number of a vertex changes by considering all r rounds so far.

Definition 6 (Ratio of Co-appearance Number)**:** *For each $\boldsymbol{v} \in V$, the ratio of co-appearance number of $\boldsymbol{v}$ with other vertices for round $r \in (1, R]$ is computed as follows:*

$$RC_{\boldsymbol{v},r} = \tfrac{1}{r(n-1)} \textstyle\sum_{i=1}^{r} S_i(\boldsymbol{v}) \tag{3}$$

Intuitively, if a vertex $\boldsymbol{v}$ consistently co-appears with a certain group of vertices in the same community across r rounds, it has a large $RC_{\boldsymbol{v},r}$ and can be considered normal. However, for a specific round r, if $\boldsymbol{v}$ suddenly becomes a member in a new community, it should be regarded as an outlier because this new community might only contain a few (or does not contain) vertices that co-appear with $\boldsymbol{v}$ in the previous round. As such, $S_r(\boldsymbol{v})$ will be very small, and meanwhile $RC_{\boldsymbol{v},r}$ will drop drastically. Thus, one can compare $RC_{\boldsymbol{v},r}$ with a pre-specified outlier threshold to determine whether $\boldsymbol{v}$ is an outlier in a round r.

Definition 7 (Outlier Set)**:** *Given an outlier threshold θ, the outlier set O_r of a round $r \in (1, R]$ is defined as below:*

$$O_r = \{\boldsymbol{v} \in V \mid RC_{\boldsymbol{v},r} < \theta\}. \tag{4}$$

The phase of co-appearance mining is described as follows. We first initialize the outlier set O_r as an empty set (Line

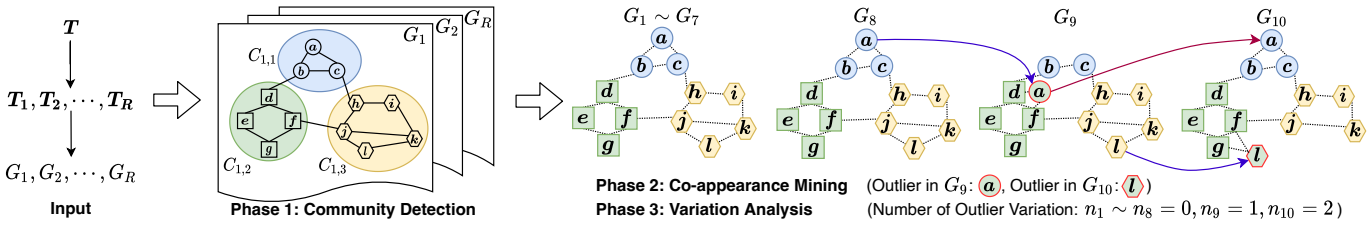

Fig. 2. An overview of CAD.

3). For each vertex $\boldsymbol{v} \in V$, we compute its co-appearance number $S_r(\boldsymbol{v})$ and the ratio of co-appearance number $RC_{\boldsymbol{v},r}$ (Line 5). Then, we add $\boldsymbol{v}$ to O_r if its $RC_{\boldsymbol{v},r}$ is below the outlier threshold θ (Line 6). Note that as the MTS usually contains noise, the existence of (a few) outliers might not be reliable to determine a round to be abnormal. Next, we will analyze the outlier variation and alleviate the impact of noise.

D. Phase 3: Variation Analysis

There are three states for a vertex in two consecutive rounds: (1) a normal one in both rounds, (2) an outlier in both rounds, and (3) a transition between a normal one and an outlier. For example, vertices $\boldsymbol{b}$ and $\boldsymbol{c}$ in Figure 2 are normal in both rounds 9 and 10, while vertex $\boldsymbol{a}$ becomes an outlier in round 9 and turns normal in round 10. To determine the abnormal time precisely, we focus on the transition state and further consider the *number of outlier variations* n_r.

Definition 8 (Number of Outlier Variations)**:** *For a round $r \in (1, R]$, the number of outlier variations n_r is determined by the number of vertices that are in a transition state, i.e., $n_r = \sum_{\boldsymbol{v} \in V}((\boldsymbol{v} \notin O_{r-1}$ and $\boldsymbol{v} \in O_r)$ or $(\boldsymbol{v} \in O_{r-1}$ and $\boldsymbol{v} \notin O_r))$.*

According to Definition 8, we compute the number of outlier variations n_r based on the outlier sets O_{r-1} and O_r from rounds $(r-1)$ and r, respectively (Line 7).

Moreover, based on Definition 3, a round r is abnormal if $n_r > \xi$. Suppose that N_r is a random variable of the number of outlier variations of round r. n_r can be regarded as an actual value drawn from N_r. When an anomaly occurs, as N_r is usually affected by its previous rounds, it is *not* an independent random variable, which might correlate with $N_{r-\Delta r}$ from its previous round $(r-\Delta r)$, where $\Delta r \in [1, r)$. We use the covariance [70] to represent this correlation, i.e., $Cov(N_r, N_{r-\Delta r})$.

Before deriving a reliable ξ for anomaly detection, we first show that the *Weak Law of Large Numbers* applies to n_r.

Theorem 1 (Weak Law of Large Numbers)**:** *Given a round $r \in (1, R]$, suppose there exists a certain correlation between N_r and $N_{r-\Delta r}$ for any $\Delta r \in [1, r)$. The sample average of N_r tends to its expectation within the following two constraints:*

(1) N_r is a bounded non-negative variable;
(2) $|Cov(N_r, N_{r-\Delta r})| \leq h(\Delta r)$, where $h(\Delta r)$ is a function of Δr that decays rapidly to 0 as Δr gets to r.

Proof. According to Definition 8, $n_r \in [0, n]$. Thus, constraint (1) holds. As the random variable N_r is bounded, we can infer that both $Var(N_r)$ and $Cov(N_r, N_{r-\Delta r})$ are also bounded. Note that the anomaly is often affected by the outliers from its previous rounds. Thus, we use $h(\Delta r)$ to reflect the effect of outliers from its previous round $(r-\Delta r)$, and the effect usually decays rapidly as Δr increases, i.e., $\lim_{\Delta r \to \infty} h(\Delta r) = 0$. Thus, we have $|Cov(N_r, N_{r-\Delta r})| \leq h(\Delta r)$ for all $\Delta r \in [1, r) > 0$. Therefore, constraint (2) also holds.

We then show that the sample average of N_r tends to its expectation. For simplicity, we assume the expectation of N_i for all $i \in [1, r]$ is $E[N_i] = \mu$. The sample average of r random variables is $A_r = \frac{1}{r}\sum_{i=1}^{r} N_i$. Then, $E[A_r] = \mu$, and the variance is $Var(A_r) = E[(A_r - \mu)^2] = E[(\frac{1}{r}\sum_{i=1}^{r}(N_i - \mu))^2]$, which can be rewritten as $Var(A_r) = \frac{1}{r^2}\sum_{i=1}^{r}\sum_{j=1}^{r} E[(N_i - \mu)(N_j - \mu)]$. With constraint (2), we have $E[(N_i - \mu)(N_j - \mu)] = Cov(N_i, N_j) \leq h(|i-j|)$. Thus, the variance of A_r satisfies the following inequality:

$$Var(A_r) \leq \frac{1}{r^2}\sum_{i=1}^{r}\sum_{j=1}^{r} h(|i-j|) \to 0.$$

Based on Chebyshev's inequality, we have $P(|A_r - \mu| < \epsilon) = 1 - P(|A_r - \mu| \geq \epsilon) \geq 1 - \frac{Var(A_r)}{\epsilon^2} \to 1$. Thus, the sample average of N_r tends to its expectation. □

E. Overall Algorithm

Anomaly Detection. Given a round r, after computing the number of outlier variations n_r, we need to determine whether the current round r is abnormal or not. According to Theorem 1, the Weak Law of Large Numbers is valid for n_r. Thus, based on Chebyshev's inequality, we have

$$P(|n_r - \mu| \geq \eta \cdot \sigma) \leq \frac{1}{\eta^2}, \tag{5}$$

where η is a constant; μ and σ are the mean and standard deviation of n_r, respectively. To detect anomalies precisely, we set $\eta = 3$ by default. Thus, the abnormal time threshold is $\xi = 3\sigma$. A round r is abnormal if $|n_r - \mu| \geq 3\sigma$ as the communities vary apparently in this round compared to previous rounds, which is unusual according to Inequality 5.

The pseudo-code of CAD for anomaly detection is depicted in Algorithm 2. We use $\mathcal{Z}$ to store the anomalies and adopt V_Z and R_Z to store the affected sensors and abnormal rounds for each anomaly, respectively (Line 2). For each round $r \in [1, R]$, we first call Algorithm 1 for outlier detection and get the outlier set O_r and the number of outlier variations n_r (Line 6). We use $\mathcal{N}$ to store a series of n_r's. Note that μ and σ are two critical parameters to determine the abnormal time. If $|n_r - \mu| \geq 3\sigma$, this round is considered abnormal, and hence we add r to R_Z and update V_Z by O_r (Line 8); otherwise, if V_Z is not empty, which means that the current anomaly is over, we add (V_Z, R_Z) to $\mathcal{Z}$ and reinitialize V_Z and R_Z as

Algorithm 2: CAD

Input: MTS $\boldsymbol{T}$, historical MTS $\boldsymbol{T}_{his}$, sliding window w, step s, outlier threshold θ;

1 $\mathcal{N}$, μ, and $\sigma \leftarrow$ WarmUp$(\boldsymbol{T}_{his}, w, s, \theta)$;
2 $\mathcal{Z} \leftarrow \emptyset$; $V_Z \leftarrow \emptyset$; $R_Z \leftarrow \emptyset$; $O_0 \leftarrow \emptyset$;
3 $R = (|\boldsymbol{T}| - w)/s + 1$;
4 **for** $r = 1$ **to** R **do**
5 $\boldsymbol{T}_r \leftarrow \boldsymbol{T}[1 + (r-1) \cdot s, w + (r-1) \cdot s]$;
6 $\{O_r, n_r\} \leftarrow$ OutlierDetection$(\boldsymbol{T}_r, O_{r-1}, \theta)$;
7 **if** $r > 1$ *and* $|n_r - \mu| \geq 3\sigma$ **then**
8 $V_Z \leftarrow V_Z \cup O_r$; $R_Z \leftarrow R_Z \cup \{r\}$;
9 **else**
10 **if** $V_Z \neq \emptyset$ **then**
11 $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{(V_Z, R_Z)\}$; $V_Z \leftarrow \emptyset$; $R_Z \leftarrow \emptyset$;
12 $\mathcal{N} \leftarrow \mathcal{N} \cup \{n_r\}$;
13 Update μ and σ from $\mathcal{N}$;
14 **return** $\mathcal{Z}$;
15 **Function** WarmUp $(\boldsymbol{T}_{his}, w, s, \theta)$ **:**
16 $\mathcal{N} \leftarrow \emptyset$; $O_0 \leftarrow \emptyset$;
17 $R_{his} = (|\boldsymbol{T}_{his}| - w)/s + 1$;
18 **for** $r = 1$ **to** R_{his} **do**
19 $\boldsymbol{T}_r \leftarrow \boldsymbol{T}_{his}[1 + (r-1) \cdot s, w + (r-1) \cdot s]$;
20 $\{O_r, n_r\} \leftarrow$OutlierDetection$(\boldsymbol{T}_r, O_{r-1}, \theta)$;
21 $\mathcal{N} \leftarrow \mathcal{N} \cup \{n_r\}$;
22 Compute μ and σ from $\mathcal{N}$;
23 **return** $\mathcal{N}$, μ, and σ;

empty sets (Line 11). Then, we add n_r to $\mathcal{N}$ (Line 12) and update μ and σ accordingly (Line 13). In the end, we return all detected anomalies $\mathcal{Z}$ as the answer (Line 14).

Compared to directly using the number of outliers defined in Definition 7 to determine abnormal rounds, using the inequality $|n_r - \mu| \geq 3\sigma$ is more reliable and better helpful in handling noise data. For a certain machine, the noise is usually at the same level. Thus, we use $|n_r - \mu|$ to alleviate the impact of noise and leverage 3σ to distinguish the actual anomalies from noisy data. Nonetheless, it might lead to a significant variance to determine μ and σ by only a few n_r's, especially when r is small. Thus, we introduce a warm-up process (Line 1) with the historical MTS to obtain sufficient n_r's.

Warm-up Process. The pseudo-code of the warm-up process is shown in Lines 16–23. The input historical MTS $\boldsymbol{T}_{his}$ is from the same data source of $\boldsymbol{T}$. Similar to the process of anomaly detection, it calls Algorithm 1 in each round r to get n_r, but it does not detect anomalies. In this way, $\mathcal{N}$ could store sufficient n_r's so that we can get the precise μ and σ for anomaly detection even for the first few rounds.

F. Discussions

Time Complexity Analysis. We first analyze the time complexity of outlier detection, which contains four parts: (1) We can construct the TSG G_r in $O(n \log n)$ time when w is small [55]. (2) The community detection phase uses $O(n \log n)$ time. (3) The co-appearance mining phase spends $O(n)$ time. (4) The variation analysis phase needs $O(n)$ time. Thus, Algorithm 1 takes $O(n \log n + n \log n + n + n) = O(n \log n)$ time. CAD performs R_{his} rounds of outlier detection for a warm-up and R rounds of outlier detection for anomaly detection. As $R = (|\boldsymbol{T}| - w)/s$, CAD uses $O(n \log n \cdot (R + R_{his})) = O(n \log n \cdot (|\boldsymbol{T}| + |\boldsymbol{T}_{his}|))$ time in total.

Generalization. Until now, we have presented CAD to detect anomalies from sensor-based MTS. However, it should be noted that CAD can be extended to process streaming data. For example, when a new round of data $\boldsymbol{T}_{R+1}$ arrives, one can repeat Lines 6–11 in Algorithm 2 to detect anomalies.

This property further allows CAD to apply to various industrial scenarios for the following reasons. (1) As CAD is performed for each round, the detection process can be run concurrently with new data collection. As long as the running time of CAD for one round is smaller than the time period for one step s, CAD can detect anomalies in real-time, which will be validated in Subsection VI-D. (2) During the anomaly detection process (Algorithm 2), we determine the μ and σ by maintaining a series of n_r's. In a streaming setting, the number of n_r's increases as more and more data streams come in. Thus, the values of μ and σ can be estimated more precisely, and hence, CAD can detect anomalies more accurately.

Limitations. CAD can detect anomalies in an early stage. This is because when an anomaly occurs, although the sensors might not exhibit significant abnormal behaviors, the correlations between the affected sensors are likely to break down, and CAD can capture such changes. However, CAD might fail to detect anomalies if there is no correlation in the sensor network or the set of affected sensors remain the same correlation to each other. Fortunately, these two cases are rare, and CAD can be used in parallel with other anomaly detection methods to provide an additional check for the anomalies.

V. Delay-aware Evaluation Scheme

Point Adjustment. The evaluation for time series anomaly detection is usually based on *Point Adjustment* (PA) [9], [77], [83]. Given a ground truth of time series, once a single time point from the ground truth is detected, this anomaly is regarded as detected. After that, a direct comparison against the ground truth is performed to compute the $F1$ score.

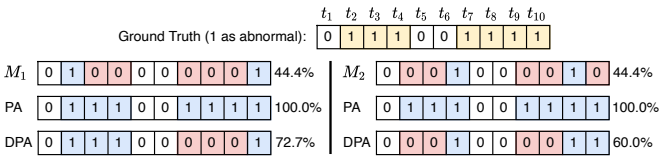

Fig. 3. The difference between PA and DPA.

Figure 3 shows an example of two anomalies (i.e., two parts of consecutive 1's (yellow color) in Ground Truth) and the detection results of two methods M_1 and M_2. As M_1 only detects two True Positives (TPs) (blue color), $F1_{M_1}$ is only 44.4%. By applying PA, as the five False Negatives (FNs) (red color) of M_1 are adjusted as TPs, $F1_{PA}$ is increased to 100%.

Challenges. In many industrial applications, detecting the anomaly as early as possible is critical before it leads to severe damage to the whole system. Thus, a feasible evaluation should not only measure how many anomalies are detected but also consider how early the anomalies are detected. The PA evaluation, however, comes with two issues.

(1) As PA does not consider which time point of the anomaly is first detected, it overlooks the chronological order when the anomaly is detected and does not assess how early it is detected. For example, in Figure 3, the $F1_{PA}$ of both M_1 and M_2 is 100%. It cannot reflect the fact that M_1 detects the first anomaly earlier than M_2, while M_2 detects the second one earlier than M_1.

(2) PA is only applied for a single method. Nonetheless, when several methods exist, we are also interested in which method can detect more anomalies earlier yet miss fewer ones than others.

Delay-aware Evaluation (DaE). To remedy the above issues in PA, we propose a new Delay-aware Evaluation (DaE) scheme, which consists of two parts:

(1) Motivated by Abdulaal et al. [3], we use *Delay-Point Adjustment* (DPA) to reflect the chronological order when anomalies are detected. Compared to PA, only the FNs after the first TP are adjusted for each anomaly. For example, considering the same M_1 in Figure 3, by using DPA, only two FNs in t_3 and t_4 are adjusted as TP, while the three FNs from t_7 to t_9 remain the same. Thus, $F1_{DPA} = 72.7\%$. Compared to PA, DPA is a more rigorous evaluation as $F1_{DPA} \leq F1_{PA}$.

(2) We propose two new measures, $Ahead$ and $Miss$, for relative comparison between the detection results of two methods. Let I_{ahead} be the number of anomalies M_1 detected *ahead* of M_2 and I_{miss} be the number of anomalies M_1 *miss* but M_2 detected. Formally,

$$Ahead = I_{ahead}/I_d, \;\; Miss = I_{miss}/(I - I_d),$$

where I is the total number of anomalies; $I_d \in [0, I]$ is the number of anomalies M_1 detected. We set $Miss = 0$ if $I_d = I$. In Figure 3, M_1 detects the first anomaly earlier than M_2, and it does not miss any anomaly. Thus, $Ahead = 50\%$ and $Miss = 0$. In the ideal case, $Ahead = 100\%$ and $Miss = 0$.

VI. Experiments

In this section, we systematically evaluate the performance of CAD for anomaly detection against state-of-the-art methods with the aim of answering the following questions:

- **Abnormal Time Detection:** Does CAD detect abnormal time accurately and early against state-of-the-art methods? (Subsection VI-B)
- **Abnormal Sensor Detection:** Can CAD localize abnormal sensors in an effective manner? (Subsection VI-C)
- **Efficiency:** How efficient is CAD and can CAD detect anomalies in real-time? (Subsection VI-D)
- **Robustness:** Can CAD produce stable results? (Subsection VI-E)

TABLE II
The statistics of the eight datasets.

Datasets	# Sensors	$\lvert\boldsymbol{T}_{his}\rvert$	$\lvert\boldsymbol{T}\rvert$	k	Data Source
PSM	26	132,480	87,840	10	Server Node
SMD	38	304,168	304,174	10	Server Machine
SWaT	51	495,000	449,919	20	Network Attack
IS-1	143	5,664	11,712	20	Electric Meters
IS-2	264	5,664	34,560	20	Electric Meters
IS-3	406	5,664	34,560	30	Assembly Line
IS-4	702	5,664	34,560	50	Assembly Line
IS-5	1,266	5,664	34,560	50	Assembly Line

- **Scalability:** Can CAD keep the performance for datasets with an increasing number of sensors? (Subsection VI-F)

In addition, we include a case study (Subsection VI-G) to demonstrate how CAD detects an anomaly in time. Finally, we study the impacts of parameters in Subsection VI-H.

A. Experimental Setup

Datasets. We study the performance of CAD over three public real-world MTS datasets PSM [2], [3], SWaT [29], [57], and SMD [77] and two private datasets IS-1 and IS-2 from industrial sensors with labeled information. For SMD, since it is made up of 28 different subsets, we directly conduct experiments on each of them without the warm-up process.

Note that the anomaly information (e.g., machinery fault) is typically very sensitive, and many manufacturers are unwilling to disclose their data. It is hard to find large-scale public datasets with labeled anomalies [24]. Thus, we include three larger private datasets IS-3–IS-5 from different assembly lines with real-world anomalies to validate the scalability of CAD. Table II summarizes the statistics of the eight datasets.

Benchmark Methods. As a comprehensive baseline to evaluate CAD, we select three popular data mining-based methods LOF [15], ECOD [48], and IForest [50] and two state-of-the-art deep learning-based methods USAD [9] and RCoders [2], [3] for comparison. Moreover, we consider four advanced univariate methods, i.e., S2G [13], SAND [14], its online version SAND*, and NormA [12]. To detect anomalies on MTS, we perform these methods on each time series and treat the mean of the abnormal scores as the output.

Evaluation Metrics. For the abnormal time detection, we report (1) $F1_{PA}$ and $F1_{DPA}$ for direct comparison against ground truth, (2) $Ahead$ and $Miss$ for relative comparison by setting M_1 as CAD, (3) Volume Under the Surface (VUS) for both ROC and PR curves (i.e., VUS-ROC and VUS-PR) [62], [64] after PA and DPA. To compute $F1_{PA}$ and $F1_{DPA}$, we grid search the optimal abnormal threshold from 0 to 1 with an interval of 0.001. For the abnormal sensor detection, we merge all detected abnormal sensors into one ground truth period for each abnormal time and use $F1_{sensor}$ for evaluation.

To assess the detection efficiency, we report the wall-clock time of training and testing for benchmark methods and the warm-up and detection time for CAD. For univariate methods, as they do not require training data, we only report their testing time, i.e., the average wall-clock time for each time series.

TABLE III
ABNORMAL TIME DETECTION BY PA AND DPA.

Methods	PSM		SWaT		IS-1		IS-2		Average Rank
	$F1_{PA}$	$F1_{DPA}$	$F1_{PA}$	$F1_{DPA}$	$F1_{PA}$	$F1_{DPA}$	$F1_{PA}$	$F1_{DPA}$	
CAD	**95.0**	89.7	83.8	**78.2**	**100.0**	**97.4**	98.2	**91.1**	**1.6**
LOF	76.2	71.4	80.1	74.1	**100.0**	83.8	**99.9**	70.3	6.1
ECOD	87.1	80.5	84.6	77.6	99.9	97.3	71.1	64.0	4.3
IForest	91.4 ± 0.9	85.7 ± 1.7	**84.9** ± 0.5	76.9 ± 0.4	99.1 ± 1.3	94.4 ± 2.4	71.3 ± 2.2	67.3 ± 1.9	3.9
USAD	90.6 ± 0.7	76.3 ± 0.6	80.3 ± 0.3	73.5 ± 2.4	99.8 ± 0.0	84.0 ± 0.0	82.6 ± 0.0	60.2 ± 0.0	6.3
RCoders	94.2 ± 1.1	**90.6** ± 1.2	81.9 ± 1.0	76.3 ± 1.2	99.4 ± 0.8	89.8 ± 4.9	64.6 ± 2.3	62.4 ± 1.9	4.6
S2G	93.6	85.0	**84.9**	78.0	95.3	85.9	72.7	67.9	3.8
SAND	85.7 ± 0.6	73.5 ± 2.4	80.5 ± 4.2	60.6 ± 3.2	78.6 ± 4.3	67.2 ± 2.3	68.7 ± 2.7	50.5 ± 0.7	8.6
SAND*	86.0 ± 1.6	72.8 ± 1.3	80.0 ± 0.9	77.8 ± 0.8	93.2 ± 2.0	87.5 ± 2.6	61.5 ± 1.2	51.1 ± 1.0	7.6
NormA	85.7 ± 0.2	76.7 ± 0.6	80.4 ± 0.6	75.2 ± 1.3	95.3 ± 1.5	82.2 ± 1.1	55.5 ± 0.3	52.3 ± 0.1	7.8

Parameter Settings. For USAD and RCoders, we refer to the examples provided by authors [2], [3], [9] and tune their models to reach the best performance. For LOF, ECOD, and IForest, we use the optimal settings suggested in their papers [15], [48], [50] and TODS [45], [92]. For S2G, we set the query length and the period length as 100 for all datasets to obtain optimal results, as it is longer than the length of most anomalies. For SAND, SAND*, and NormA, we first estimate their pattern length l based on the autocorrelation function [65]; then, we set the centroids length as $4l$ for SAND and SAND* and the length of the normal model as $4l$ for NormA; besides, for SAND*, we set the update rate $\alpha = 0.5$ and the initial length as $0.5|\boldsymbol{T}|$ with a batch size of $0.1|\boldsymbol{T}|$. For other parameters, we use the settings suggested in their papers [12]–[14]. To alleviate the randomness issue, we repeat 10 times for all baselines and report the average of their results. For CAD, we vary (1) w from $0.005|\boldsymbol{T}|$ to $0.2|\boldsymbol{T}|$, (2) s from $0.005w$ to $0.2w$, and (3) τ and θ from 0.1 to 0.9 with a step of 0.05; the settings of k for different datasets are shown in Table II.

Environment. All experiments are conducted on a machine with Intel® Xeon® Gold 6342 CPU @ 2.80GHz, 64 GB memory, and one NVIDIA GeForce RTX 3090.

B. Effectiveness of Abnormal Time Detection

The Evaluations of PA and DPA. Table III shows the results of abnormal time detection on PSM, SWaT, IS-1, and IS-2, where the standard deviations (std) are calculated based on $F1_{PA}$ and $F1_{DPA}$ for 10 repeats. As the results of CAD, LOF, ECOD, and S2G do not change with different repeats, their std's are always 0. We observe that CAD outperforms all baselines on IS-1 and achieves the highest average rank of $F1_{PA}$ and $F1_{DPA}$. The results on SMD are depicted in Table IV. We also find that CAD outperforms USAD, RCoders, S2G, SAND, SAND*, and NormA for at least 17 out of 28 subsets and is comparable to LOF, ECOD, and IForest.

When # sensors increases, e.g., from IS-1 to IS-2, although some methods (e.g., LOF and USAD) are comparable to CAD under PA, they decrease largely under DPA. It is partly because the MTS with more sensors usually includes more complex relationships and contains more noise [88], [90]. The four univariate methods also decrease significantly under PA and DPA, which indicates that a trivial extension of these methods to the MTS setting might not be effective. As most competitors find outliers based on certain assumptions, they might become less stable and more sensitive to noise, whereas CAD detects anomalies based on unusual correlation variations between sensors through a series of TSGs. Also, not all outliers are considered abnormal sensors. We analyze the number of outlier variations with a theoretical guarantee (Theorem 1). Thus, CAD can reduce noise and produce more robust results, which will be further confirmed in Subsection VI-E.

TABLE IV
ABNORMAL TIME AND ABNORMAL SENSOR DETECTION ON SMD (OP DENOTES #SUBSETS CAD OUTPERFORMS IN 28 SUBSETS).

Methods	$F1_{PA}$		$F1_{DPA}$		$F1_{sensor}$
	OP	mean ± std	OP	mean ± std	OP
CAD	–	80.7 ± 14.4	–	71.7 ± 15.9	–
LOF	16	75.0 ± 24.4	13	65.5 ± 26.0	/
ECOD	14	83.0 ± 14.5	12	73.2 ± 16.8	28
IForest	7	**86.5** ± 13.7	8	**76.5** ± 18.4	/
USAD	20	75.3 ± 18.4	17	66.5 ± 20.7	/
RCoders	21	75.3 ± 17.4	19	63.6 ± 20.2	28
S2G	25	64.0 ± 21.4	23	52.5 ± 21.4	/
SAND	26	45.3 ± 30.3	26	33.5 ± 25.7	/
SAND*	26	51.1 ± 28.8	24	38.7 ± 25.8	/
NormA	25	55.3 ± 29.3	26	42.6 ± 26.5	/

TABLE V
THE EVALUATIONS OF $Ahead$ (Ah) AND $Miss$ (Ms).

CAD vs. Methods	PSM		SWaT		IS-1		IS-2	
	Ah	Ms	Ah	Ms	Ah	Ms	Ah	Ms
LOF	60.0	40.4	100.0	0.0	100.0	0.0	77.8	0.0
ECOD	73.3	19.3	100.0	35.3	100.0	0.0	66.7	0.0
IForest	50.0	40.4	100.0	37.9	100.0	0.0	36.7	0.0
USAD	66.4	17.5	100.0	9.7	100.0	0.0	77.8	0.0
RCoders	64.7	31.8	100.0	13.5	100.0	0.0	43.3	0.0
S2G	73.3	12.3	100.0	31.2	100.0	0.0	100.0	0.0
SAND	96.0	8.2	100.0	59.1	100.0	0.0	100.0	0.0
SAND*	86.7	12.8	100.0	4.7	100.0	0.0	100.0	0.0
NormA	82.0	19.5	100.0	15.7	100.0	0.0	100.0	0.0

The Evaluations of $Ahead$ and $Miss$. We then look at $Ahead$ and $Miss$ in Table V. CAD achieves at least 50% $Ahead$ against others on PSM and SWaT, and meanwhile, its $Miss$ remains smaller than 50% (except for SAND). These results indicate that (1) for the anomalies we detected, CAD manages to discover at least half of them ahead of others; (2) for those we missed, at most half of them can be detected by

others. CAD achieves 100% for $Ahead$ and 0 for $Miss$ on IS-1, which is the ideal case. For the cases that $Ahead < 50\%$ or $Miss > 50\%$, the reason might be that $Ahead$ and $Miss$ treat each ground truth anomaly equally and disregard its length.

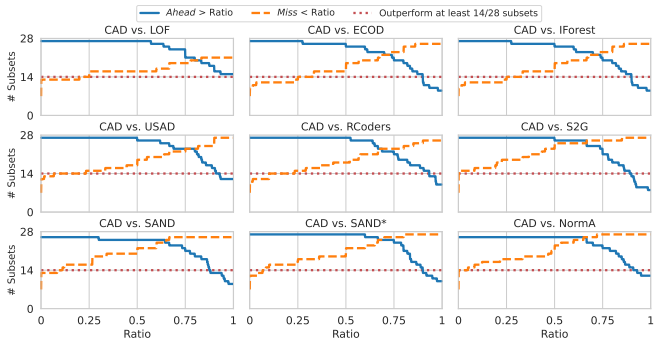

Fig. 4. The evaluations of $Ahead$ and $Miss$ on SMD.

For SMD, we vary the ratio of $Ahead$ and $Miss$ from 0 to 1 with an interval of 0.01 and count the number of subsets that CAD outperforms from all 28 subsets. The results are shown in Figure 4. Most subsets in SMD can achieve $Ahead > 50\%$ compared to other methods, and more than half of subsets obtain $Miss < 50\%$. These results further validate that CAD can reveal the abnormal time early with few misses.

The Evaluations of VUS-ROC and VUS-PR. At last, we report the evaluations by VUS-ROC and VUS-PR after PA and DPA in Figure 5. We observe that CAD achieves the highest VUS-ROC and VUS-PR after both PA and DPA with only a few exceptions, which is consistent with the evaluations by $F1_{PA}$ and $F1_{DPA}$. Also, CAD enjoys a little drop from PA to DPA for both VUS-ROC and VUS-PR. These results confirm that CAD can detect anomalies earlier than the baselines, which is more favorable for timely maintenance.

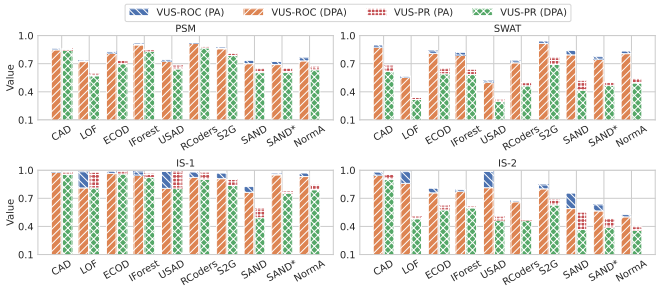

Fig. 5. The evaluations of VUS-ROC and VUS-PR.

Moreover, when dealing with large datasets, e.g., IS-2, CAD maintains VUS-ROC and VUS-PR over 90%, whereas other methods drop significantly. When datasets become larger and more complex, it becomes harder to detect anomalies in an early stage, as the readings from many normal sensors could balance off the abnormal ones from a few abnormal sensors. However, CAD detects anomalies one round at a time, and a round is considered abnormal if it contains much more outliers than previous rounds. Thus, CAD can maintain its performance and detect anomalies in an early stage for large datasets. We will further validate its scalability in Subsection VI-F.

TABLE VI
TRAINING TIME OF ALL MTS METHODS (IN SECONDS).

Methods	PSM	SWaT	IS-1	IS-2	SMD
CAD	9.9	9.3	0.6	15.7	/
LOF	309.3	3,377.4	0.5	0.5	7.2
ECOD	1.4	5.4	0.2	0.3	0.2
IForest	5.8	36.6	0.7	1.5	1.2
USAD	228.8	270.9	2.2	2.5	44.4
RCoders	508.5	2,327.6	79.0	163.8	118.7

TABLE VII
TESTING TIME OF ALL METHODS (IN SECONDS).

Methods	PSM	SWaT	IS-1	IS-2	SMD
CAD	6.4	10.3	1.0	104.0	3.4
TPR	1.1 (ms)	2.3 (ms)	1.7 (ms)	24.2 (ms)	1.3 (ms)
LOF	120.4	2,167.8	0.9	2.9	6.9
ECOD	2.9	9.8	0.6	3.7	0.5
IForest	1.5	11.3	0.5	4.3	0.5
USAD	0.6	3.1	0.2	0.3	0.2
RCoders	1.5	11.0	1.6	47.0	0.6
S2G	6.4	34.3	0.9	2.8	1.9
SAND	12.7	20.3	3.8	5.0	4.6
SAND*	7.9	21.7	1.7	6.2	3.2
NormA	8.6	105.5	1.3	4.6	3.2

C. Effectiveness of Abnormal Sensor Detection

Recall that CAD can not only detect the abnormal time but also reveal the abnormal sensors. The last column of Table IV shows the results of abnormal sensor detection on SMD. We observe that CAD outperforms **28/28** against ECOD and RCoders, which are the *only* two competitors that can provide some interpretations for abnormal sensors. We further conducted experiments on IS-1 and IS-2. We found that CAD also outperforms ECOD and RCoders, and its $F1_{sensor}$ is over 60%. Thus, at least half of normal sensors can be safely ruled out by CAD, which greatly reduces the pressure on human operators for predictive maintenance.

D. Detection Efficiency

Tables VI and VII depict the training and testing time of all methods, respectively. CAD completes the entire detection process in a few seconds (except for IS-2), while some baselines (e.g., USAD, RCoders, and LOF on PSM and SWaT) require spending hundreds (or even thousands) of seconds in training their models with extra training data. As the length of the time series grows, e.g., from PSM to SWaT, the testing time of all MTS methods increases significantly by around 3.3∼18×, while the detection time of CAD increases by about 1.6× only. Moreover, from Table VII, we find that CAD takes less testing time than the four univariate methods on PSM and SWaT. When the number of sensors is considered, their total testing time is much larger than that of CAD.

The third row of Table VII shows the detection time of CAD for each round, i.e., Time Per Round (TPR). To detect anomalies in real-time, TPR should be less than the time period of a step s, i.e., TPR $< s/freq$, where $freq$ is the sampling frequency of the sensors. Thus, $freq < s/\text{TPR}$ Hz. According to Table VII, CAD can support real-time anomaly detection. For example, the maximum frequency of CAD on

TABLE VIII
THE MINIMUM $F1_{PA}$ AND $F1_{DPA}$ FOR ABNORMAL TIME DETECTION.

Methods	PSM		SWaT		IS-1		IS-2	
	$F1_{PA}$	$F1_{DPA}$	$F1_{PA}$	$F1_{DPA}$	$F1_{PA}$	$F1_{DPA}$	$F1_{PA}$	$F1_{DPA}$
CAD	**95.0**	**89.7**	83.8	**78.2**	**100.0**	**97.4**	98.2	**91.1**
LOF	76.2	71.4	80.1	74.1	**100.0**	83.8	**99.9**	70.3
ECOD	87.1	80.5	84.6	77.6	99.9	97.3	71.1	64.0
IForest	89.9	83.2	84.3	76.3	95.8	89.1	66.7	64.1
USAD	89.1	75.3	78.0	68.3	99.8	84.0	82.6	60.2
RCoders	92.7	89.1	80.6	74.2	97.4	83.1	62.5	60.1
S2G	93.6	85.0	**84.9**	78.0	95.3	85.9	72.7	67.9
SAND	85.0	71.2	69.6	55.8	70.2	64.2	65.2	49.7
SAND*	83.7	70.9	78.9	75.7	89.1	82.9	60.3	49.5
NormA	85.3	75.5	79.9	74.2	93.2	79.8	55.1	52.1

SWaT and IS-2 is around 43k Hz and 331 Hz, which is much larger than their $freq$ in 1 Hz and 1/900 Hz, respectively.

E. Robustness

Besides presenting the mean and std values in Table III, we report the minimum $F1_{PA}$ and $F1_{DPA}$ of all methods in Table VIII to validate the robustness of CAD. Except for the four deterministic methods CAD, LOF, ECOD, and S2G, there exists a (large) gap between the mean and minimum values of $F1_{PA}$ and $F1_{DPA}$ for the rest methods, which indicates their results are *unstable*. Thus, they need multiple runs to get reliable results. Nonetheless, as the sensors often generate a huge amount of real-time data, it is challenging to deploy them in a real-world setting. In contrast, CAD only depends on the changes in correlations between sensors and does not introduce any randomness in anomaly detection.

F. Scalability

Next, we evaluate the scalability of CAD. In particular, we run CAD on five large-scale real-world labeled datasets (i.e., IS-1–IS-5) by increasing the number of sensors and report $F1_{PA}$, $F1_{DPA}$, and TPR for anomaly detection.

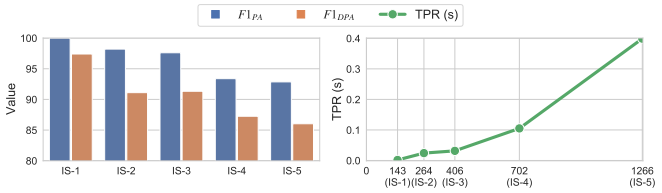

Fig. 6. Scalability test on IS-1–IS-5 datasets.

The left side of Figure 6 shows the results of abnormal time detection. When the number of sensors increases, both $F1_{PA}$ and $F1_{DPA}$ drop. However, even if IS-5 contains over 1,000 sensors, CAD is still capable of detecting anomalies with $F1_{DPA} > 85\%$. These results further confirm that CAD can detect anomalies precisely and as early as possible.

The right side of Figure 6 displays the TPR for anomaly detection on IS-1–IS-5. We observe that the TPR of CAD is subquadratic to the number of sensors, which is consistent with the time complexity analysis as discussed in Subsection IV-F. Thus, CAD is scalable to a large number of sensors. As discussed in Subsection VI-D, although CAD requires larger detection time for more sensors, as long as the TPR is less than the time period of one step (which are the cases for IS-1–IS-5), CAD can achieve real-time anomaly detection.

G. Case Study

We select a subset SMD 1_6 from SMD to conduct a case study. The time series of partial sensors for a labeled anomaly is shown in Figure 7. We first illustrate whether the anomalies detected by CAD differ from normal patterns. The orange part covers the abnormal time period of this anomaly and indicates which sensors are affected. When this anomaly occurs, the time series of abnormal sensors (e.g., Sensors 2–4, 9, 12, and 13) become (very) different from their previous ones, whereas the normal sensors (e.g., Sensors 19–21) still keep their original trends, which are not affected by this anomaly.

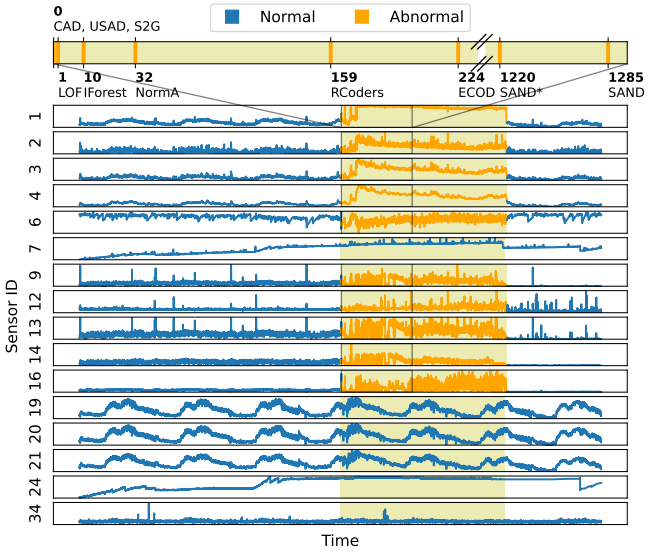

Fig. 7. Case study of a detected anomaly on SMD 1_6.

We then explain why detecting anomalies based on the changes in correlations between sensors would work better than other methods. Sensors 2–4 initially act similarly to Sensors 9, 12, and 13, and hence, they have a strong correlation. When this anomaly occurs, as their time series change, Sensors 2–4 no longer correlate with Sensors 9, 12, and 13. Since CAD uses edges in TSGs to represent the strong correlations between sensors, unusual changes in some sensors can be immediately reflected by the changes in the edges of TSGs. Thus, CAD can precisely report them as abnormal sensors in an early stage. However, when this anomaly occurs, the change in time series might initially be insignificant, so other methods only based on specific rules might not detect it early.

Finally, we illustrate how early CAD can reveal the anomalies. The first row of Figure 7 depicts the time point at which this anomaly is detected. CAD, USAD, and S2G identify this anomaly once it occurs, while other methods take at most 1,285 time points to alarm, which might not be suitable in the real-world industrial setting as a failure from one component will propagate to other adjacent components over time if not maintained in time [23], [34], [37].

H. Parameter Study

Finally, we study the effects of five key parameters in CAD, i.e., the sliding window w, step s, correlation threshold τ, outlier threshold θ, and the number of nearest neighbors k.

Effects of w and s. We first analyze the effects of w and s and study how they affect $F1_{PA}$ and $F1_{DPA}$ of CAD through experiments on PSM, SMD, and SWaT. To concisely visualize SMD, we only report the results of SMD 1_7. Similar trends can be observed from other subsets in SMD. In the first row of Figure 8, we show the optimal $F1_{PA}$ and $F1_{DPA}$ of different ratio of w and $|\boldsymbol{T}|$ (i.e., $w/|\boldsymbol{T}|$) from all s's. A small to moderate $w/|\boldsymbol{T}|$ (e.g., 0.01∼0.03) tends to get large $F1_{PA}$ and $F1_{DPA}$, but a large $w/|\boldsymbol{T}|$ will lead to low accuracy because given a fixed $|\boldsymbol{T}|$, a large w incurs a very long time series, making CAD less effective in detecting the anomalies within a short period. As w and s are closely related to each other, we typically set s based on w. Thus, we further study the effect of s/w. The results are shown in the second row of Figure 8. We find that a small s/w (e.g., 0.01∼0.02) results in large $F1_{PA}$ and $F1_{DPA}$ because a large w with a small s helps to track the variations of correlations at fine granularity. However, a very small s/w will be highly expensive as many windows need to be processed. Thus, we suggest setting a moderate $w \in [0.01|\boldsymbol{T}|, 0.03|\boldsymbol{T}|]$ and $s \in [0.01w, 0.02w]$ to balance the trade-off.

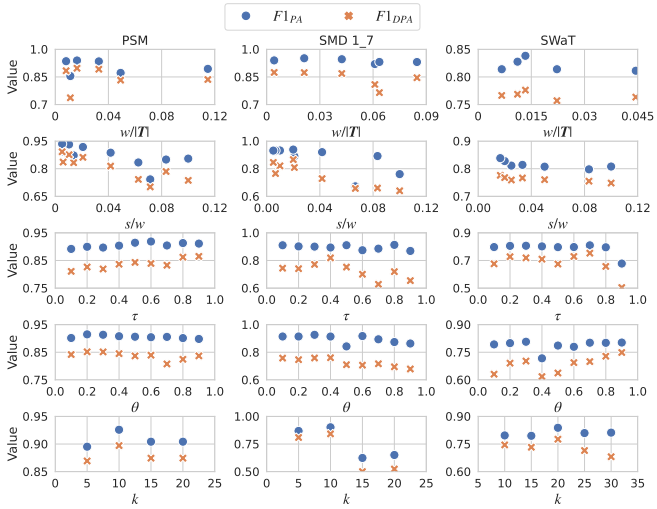

Fig. 8. Parameter study on PSM, SMD 1_7, and SWaT.

Effects of τ. To study the effect of τ, we use the same settings of w/s and vary τ from 0.1 to 0.9 with a step of 0.1. The results are displayed in the third row of Figure 8. The τ around 0.4∼0.6 often leads to large $F1_{PA}$ and $F1_{DPA}$. The reason is that τ controls the sparsity of the TSG: (1) if we set a very large τ, some vertices might not have enough neighbors, and accordingly, CAD lacks sufficient correlation information to detect anomalies; (2) if we set a very small τ, the TSG might contain many edges with weak correlations, leading to extra noise. Thus, a moderate τ around 0.4∼0.6 is preferred.

Effects of θ. We then vary θ from 0.1 to 0.9 with a step of 0.1 to investigate the effect of θ. The results are depicted in the fourth row of Figure 8. A small θ around 0.3 usually leads to large $F1_{PA}$ and $F1_{DPA}$. When $\theta = 0.5$ (0.4), CAD has the smallest $F1_{PA}$ on SMD 1_7 (SWaT). As θ continues to increase from 0.6, $F1_{PA}$ drops (especially on SMD 1_7) because a large θ will rule out many vertices that should be considered outliers, leading to low detection accuracy. Thus, for most datasets, we suggest setting a small θ around 0.3.

Effects of k. Finally, we vary k from 5 to 20 (from 10 to 30 for SWaT) with a step of 5. The results are shown in the fifth row of Figure 8. We find that k cannot be too large as some sensors in TSGs might contain weak correlations to their k neighbors, leading to extra noise and worse results for anomaly detection. For example, in SMD 1_7, $F1_{PA}$ and $F1_{DPA}$ drop significantly when k is larger than 10. On the other hand, if k is very small, TSGs might not contain enough correlation information between sensors. Thus, we suggest setting a moderate k as depicted in Table II.

I. Summary

Based on the experimental results, we have six observations. (1) CAD has the best performance for abnormal time detection. Also, compared to the nine baseline methods, CAD can detect anomalous events in an earlier stage while having fewer misses. (2) For the detection of abnormal sensors, CAD outperforms all competitors. (3) CAD can support real-time anomaly detection. Specifically, CAD is faster than the two deep learning-based methods and the four univariant methods in terms of the total running time by at least an order of magnitude and is comparable to the fastest data mining-based methods. (4) For robustness, as a deterministic method, the results of CAD are stable. (5) Scalability test shows that CAD can detect anomalies efficiently from a large number of sensors with little drop in $F1_{PA}$ and $P1_{DPA}$. (6) A case study on a real-world anomaly demonstrates that CAD can capture the common expectation of what anomalies are like in industrial applications and detect anomalies in an early stage.

VII. Conclusions

In this paper, we investigate the fundamental problem of anomaly detection for sensor-based MTS. We introduce a novel method CAD to detect the time period of anomaly occurrence together with the relevant sensors based on the analysis of correlation variations with a theoretical guarantee. CAD is a deterministic method, which has stable output without model training. Moreover, CAD does not require rigorous assumptions on data distribution, so it can be easily extended to process streaming data. We also propose a new DaE scheme to systematically evaluate different anomaly detection methods. Extensive experiments validate the improvements of CAD against nine state-of-the-art methods, especially the early alarm ability for timely predictive maintenance of failures.

Acknowledgment

We sincerely thank the anonymous reviewers for their insightful comments. This research is supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

REFERENCES

[1] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*, vol. 30, no. 14-15, pp. 2826–2841, 2007.

[2] A. Abdulaal and T. Lancewicki, "Real-time synchronization in neural networks for multivariate time series anomaly detection," in *ICASSP*, 2021, pp. 3570–3574.

[3] A. Abdulaal, Z. Liu, and T. Lancewicki, "Practical approach to asynchronous multivariate time series anomaly detection and localization," in *KDD*, 2021, pp. 2485–2494.

[4] C. Aggarwal, *Outlier Analysis*. Springer International Publishing, 2017.

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Comput. Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[6] I. F. Akyildiz, M. C. Vuran, and Ö. B. Akan, "On exploiting spatial and temporal correlation in wireless sensor networks," in *Proceedings of WiOpt: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2004, pp. 71–80.

[7] Y. Almardeny, N. Boujnah, and F. Cleary, "A novel outlier detection method for multivariate data," *TKDE*, vol. 34, no. 9, pp. 4052–4062, 2022.

[8] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *PKDD*, 2002, pp. 15–26.

[9] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "Usad: Unsupervised anomaly detection on multivariate time series," in *KDD*, 2020, pp. 3395–3404.

[10] T. R. Bandaragoda, K. M. Ting, D. W. Albrecht, F. T. Liu, Y. Zhu, and J. R. Wells, "Isolation-based anomaly detection using nearest-neighbor ensembles," *Comput. Intell.*, vol. 34, no. 4, pp. 968–998, 2018.

[11] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[12] P. Boniol, M. Linardi, F. Roncallo, T. Palpanas, M. Meftah, and E. Remy, "Unsupervised and scalable subsequence anomaly detection in large data series," *VLDBJ*, vol. 30, no. 6, pp. 909–931, 2021.

[13] P. Boniol and T. Palpanas, "Series2Graph: Graph-based subsequence anomaly detection for time series," *PVLDB*, vol. 13, no. 11, pp. 1821–1834, 2020.

[14] P. Boniol, J. Paparrizos, T. Palpanas, and M. J. Franklin, "SAND: streaming subsequence anomaly detection," *PVLDB*, vol. 14, no. 10, pp. 1717–1729, 2021.

[15] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," in *SIGMOD*, 2000, pp. 93–104.

[16] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in β-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[17] W. Cai and M. Zhang, "Spatiotemporal correlation-based adaptive sampling algorithm for clustered wireless sensor networks," *Int. J. Distributed Sens. Networks*, vol. 14, no. 8, 2018.

[18] S. Chai, Z. Wang, B. Zhang, L. Cui, and R. Chai, "Community detection in complex networks," in *Wireless Sensor Networks*, 2020, pp. 189–240.

[19] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[20] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *CSUR*, vol. 41, no. 3, pp. 1–58, 2009.

[21] Z. Chen, D. Chen, X. Zhang, Z. Yuan, and X. Cheng, "Learning graph structures with transformer for multivariate time series anomaly detection in iot," *IEEE Internet of Things Journal*, 2021.

[22] Z. Chen, M. Jia, Y. Wang, and X. Yan, "A security topology protocol of wireless sensor networks based on community detection and energy aware," in *IEEE Trustcom/BigDataSE/ISPA*, 2015, pp. 1284–1291.

[23] W. Cheng, K. Zhang, H. Chen, G. Jiang, Z. Chen, and W. Wang, "Ranking causal anomalies via temporal and dynamical analysis on vanishing correlations," in *KDD*, 2016, pp. 805–814.

[24] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *AAAI*, 2021, pp. 4027–4035.

[25] H. S. Dhiman, D. Deb, S. M. Muyeen, and I. Kamwa, "Wind turbine gearbox anomaly detection based on adaptive threshold and twin support vector machines," *IEEE Transactions on Energy Conversion*, vol. 36, no. 4, pp. 3462–3469, 2021.

[26] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[27] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[28] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, "Unsupervised scalable representation learning for multivariate time series," in *NeurIPS*, 2019, pp. 4652–4663.

[29] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *International conference on critical information infrastructures security (CRITIS)*, 2016, pp. 88–99.

[30] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI-2012: poster and demo track*, vol. 9, 2012.

[31] J. S. Hardin and D. M. Rocke, "Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator," *Comput. Stat. Data Anal.*, vol. 44, no. 4, pp. 625–638, 2004.

[32] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *CVPR*, 2016, pp. 733–742.

[33] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognit. Lett.*, vol. 24, no. 9-10, pp. 1641–1650, 2003.

[34] H. Huang, S. Yoo, and Y. Xu, "System deterioration detection and root cause learning on time series graphs," in *CIKM*, 2019, pp. 2537–2545.

[35] A. Hyvärinen and H. Morioka, "Unsupervised feature extraction by time-contrastive learning and nonlinear ica," in *NIPS*, 2016, pp. 3765–3773.

[36] N. Japkowicz, C. Myers, and M. A. Gluck, "A novelty detection approach to classification," in *IJCAI*, 1995, pp. 518–523.

[37] G. Jiang, H. Chen, and K. Yoshihira, "Discovering likely invariants of distributed transaction systems for autonomic system management," in *Proceedings of the 3rd International Conference on Autonomic Computing (ICAC)*, 2006, pp. 199–208.

[38] E. Keogh, J. Lin, and A. Fu, "Hot sax: Efficiently finding the most unusual time series subsequence," in *ICDM*, 2005, pp. 226–233.

[39] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier detection for time series with recurrent autoencoder ensembles," in *IJCAI*, 2019, pp. 2725–2732.

[40] S. Kim, K. Choi, H. Choi, B. Lee, and S. Yoon, "Towards a rigorous evaluation of time-series anomaly detection," in *AAAI*, 2022, pp. 7194–7201.

[41] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *ICLR*, 2014.

[42] M. Kontaki, A. Gounaris, A. N. Papadopoulos, K. Tsichlas, and Y. Manolopoulos, "Continuous monitoring of distance-based outliers over data streams," in *ICDE*, 2011, pp. 135–146.

[43] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Outlier detection in axis-parallel subspaces of high dimensional data," in *PAKDD*, 2009, pp. 831–838.

[44] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *KDD*, 2008, pp. 444–452.

[45] K.-H. Lai, D. Zha, G. Wang, J. Xu, Y. Zhao, D. Kumar, Y. Chen, P. Zumkhawaka, M. Wan, D. Martinez *et al.*, "Tods: An automated time series outlier detection system," in *AAAI*, 2021, pp. 16 060–16 062.

[46] S. Li, D. Hong, and H. Wang, "Relation inference among sensor time series in smart buildings with metric learning," in *AAAI*, 2020, pp. 4683–4690.

[47] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu, "Copod: Copula-based outlier detection," in *ICDM*, 2020, pp. 1118–1123.

[48] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. Chen, "Ecod: Unsupervised outlier detection using empirical cumulative distribution functions," *TKDE*, 2022.

[49] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen, and D. Pei, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *KDD*, 2021, pp. 3220–3230.

[50] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *ICDM*, 2008, pp. 413–422.

[51] W. Lu, Y. Cheng, C. Xiao, S. Chang, S. Huang, B. Liang, and T. Huang, "Unsupervised sequential outlier detection with deep architectures," *TIP*, vol. 26, no. 9, pp. 4321–4330, 2017.

[52] W. Luo, W. Liu, and S. Gao, "Remembering history with convolutional lstm for anomaly detection," in *ICME*, 2017, pp. 439–444.

[53] A. Makhzani and B. J. Frey, "K-sparse autoencoders," in *ICLR*, 2014.

[54] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[55] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *TPAMI*, vol. 42, no. 4, pp. 824–836, 2018.

[56] L. Martí, N. S. Pi, J. M. Molina, and A. C. B. Garcia, "Anomaly detection based on sensor data in petroleum industry applications," *Sensors*, vol. 15, no. 2, pp. 2774–2797, 2015.

[57] A. P. Mathur and N. O. Tippenhauer, "Swat: A water treatment testbed for research and training on ics security," in *2016 International Workshop on Cyber-physical Systems for Smart Water Networks*, 2016, pp. 31–36.

[58] M. E. Newman, "Modularity and community structure in networks," *PNAS*, vol. 103, no. 23, pp. 8577–8582, 2006.

[59] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *CSUR*, vol. 54, no. 2, pp. 1–38, 2021.

[60] G. Pang, C. Shen, and A. van den Hengel, "Deep anomaly detection with deviation networks," in *KDD*, 2019, pp. 353–362.

[61] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, "Loci: Fast outlier detection using the local correlation integral," in *ICDE*, 2003, pp. 315–326.

[62] J. Paparrizos, P. Boniol, T. Palpanas, R. S. Tsay, A. Elmore, and M. J. Franklin, "Volume under the surface: a new accuracy evaluation measure for time-series anomaly detection," *PVLDB*, vol. 15, no. 11, pp. 2774–2787, 2022.

[63] J. Paparrizos and L. Gravano, "k-shape: Efficient and accurate clustering of time series," in *SIGMOD*, 2015, pp. 1855–1870.

[64] J. Paparrizos, Y. Kang, P. Boniol, R. S. Tsay, T. Palpanas, and M. J. Franklin, "TSB-UAD: an end-to-end benchmark suite for univariate time-series anomaly detection," *PVLDB*, vol. 15, no. 8, pp. 1697–1711, 2022.

[65] E. Parzen, "On spectral analysis with missing observations and amplitude modulation," *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 383–392, 1963.

[66] K. Pearson and F. Galton, "Notes on regression and inheritance in the case of two parents," *Proceedings of the Royal Society of London*, vol. 58, pp. 240–242, 1895.

[67] T. Pevný, "Loda: Lightweight on-line detector of anomalies," *Machine Learning*, vol. 102, no. 2, pp. 275–304, 2016.

[68] S. Prasanna and S. Rao, "An overview of wireless sensor networks applications and security," *International Journal of Soft Computing and Engineering*, vol. 2, no. 2, pp. 2231–2307, 2012.

[69] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *SIGMOD*, 2000, pp. 427–438.

[70] J. A. Rice, *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[71] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *ICML*, 2011, pp. 833–840.

[72] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller, "A unifying review of deep and shallow anomaly detection," *Proc. IEEE*, vol. 109, no. 5, pp. 756–795, 2021.

[73] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "f-anogan: Fast unsupervised anomaly detection with generative adversarial networks," *Medical Image Anal.*, vol. 54, pp. 30–44, 2019.

[74] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, 2001.

[75] P. Senin, J. Lin, X. Wang, T. Oates, S. Gandhi, A. P. Boedihardjo, C. Chen, and S. Frankenstein, "Time series anomaly discovery with grammar-based compression." in *EDBT*, 2015, pp. 481–492.

[76] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," Miami Univ Coral Gables Fl Dept of Electrical and Computer Engineering, Tech. Rep., 2003.

[77] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *KDD*, 2019, pp. 2828–2837.

[78] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *VLDB*, 2006, pp. 187–198.

[79] S. Tonekaboni, D. Eytan, and A. Goldenberg, "Unsupervised representation learning for time series with temporal neighborhood coding," in *ICLR*, 2021.

[80] L. Tran, L. Fan, and C. Shahabi, "Distance-based outlier detection in data streams," *PVLDB*, vol. 9, no. 12, pp. 1089–1100, 2016.

[81] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *JMLR*, vol. 11, pp. 3371–3408, 2010.

[82] B. Wang, Y. Lei, N. Li, and N. Li, "A hybrid prognostics approach for estimating remaining useful life of rolling element bearings," *IEEE Transactions on Reliability*, vol. 69, no. 1, pp. 401–412, 2020.

[83] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, J. Chen, Z. Wang, and H. Qiao, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *WWW*, 2018, pp. 187–196.

[84] D. Yankov, E. Keogh, and U. Rebbapragada, "Disk aware discord discovery: finding unusual time series in terabyte sized datasets," *KAIS*, vol. 17, no. 2, pp. 241–262, 2008.

[85] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. Keogh, "Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets," in *ICDM*, 2016, pp. 1317–1322.

[86] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar, "Adversarially learned anomaly detection," in *ICDM*, 2018, pp. 727–736.

[87] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *KDD*, 2021, pp. 2114–2124.

[88] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *AAAI*, 2019, pp. 1409–1416.

[89] D. Zhang, G. Li, K. Zheng, X. Ming, and Z.-H. Pan, "An energy-balanced routing method based on forward-aware factor for wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 766–773, 2013.

[90] Y. Zhang, Y. Chen, J. Wang, and Z. Pan, "Unsupervised deep anomaly detection for multi-sensor time-series signals," *TKDE*, 2021.

[91] Y. Zhao, Z. Nasrullah, M. K. Hryniewicki, and Z. Li, "Lscp: Locally selective combination in parallel outlier ensembles," in *SDM*, 2019, pp. 585–593.

[92] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *JMLR*, vol. 20, no. 96, pp. 1–7, 2019.

[93] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017, pp. 2242–2251.