Compatible Unsupervised Anomaly Detection with Multi-Perspective Spatio-Temporal Learning

Tingyang Chen[1], Bolong Zheng[1,*], Shuncheng Liu[2], Zhujiong Fan[1], Zhi Xu[2], Lingsen Yan[1], Kai Zeng[2], Tao Ye[2], Xiaofang Zhou[3]
[1]*Huazhong University of Science and Technology, Wuhan, China*
[2]*Huawei, Shenzhen, China*
[3]*Hong Kong University of Science and Technology, Hong Kong, China*

2025 IEEE 41st International Conference on Data Engineering (ICDE) | 979-8-3315-3603-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICDE65448.2025.00303

***Abstract*—Anomaly detection is one of the most significant tasks in industrial automatic maintenance, such as in distributed cloud systems. However, the implementation of existing anomaly detection methods is still challenging in (i) capturing the complex spatial and temporal correlations of multivariate time series, (ii) effectively adapting to the unsupervised condition, and (iii) generalizing across nodes in distributed systems. To address these challenges, we design a multi-perspective spatio-temporal attention model, called STAMP, which consists of a prediction module ST-ATTN, a reconstruction module AutoEncoder, and an adversarial optimizing module. Specifically, ST-ATTN leverages multiple attention mechanisms to perform spatio-temporal learning from both local and global perspectives, AutoEncoder is utilized to fit implicit representations, and the adversarial optimization module employs a *min-max* training strategy to enhance the learning capability. By introducing pre-training strategies, STAMP can be effectively adapted to distributed systems with a strong generalization ability. Furthermore, to cope with the practical unlabeled data conditions, we propose an unsupervised framework compatible with not only STAMP but also other advanced detection models. In this framework, a screening process is first conducted by traditional methods to generate a training set of pseudo-normal samples. Second, the models are trained and then used for detection. The framework can be further optimized by performing feature selection based on model-derived information for a better detectability. Extensive experiments in real-world datasets demonstrate that the proposed model and framework achieve superior performance when compared with baselines under both semi-supervised and unsupervised conditions. In particular, the detection framework has already been applied in Huawei's GaussDB (DWS) system.**

***Index Terms*—multivariate time series, anomaly detection, spatio-temporal correlation, unsupervised learning**

I. Introduction

In large-scale industrial systems, the occurrence of sub-health and malfunction is inevitable, which is closely related to the system reliability. A prompt identification of such abnormal events is critical to preventing loss escalation. Hence, the development of a fast, accurate, and automated anomaly detection method becomes an urgent need [1]–[3]. Additionally, with the expansion of business and the diversity of services, the operational data collected by industrial (distributed) systems continues to increase with various types of key performance indicators (KPIs). In this context, the data further forms complex multivariate time series, where timestamps and KPI features are regarded as logical points in the temporal and spatial domains [4], respectively.

The existing multivariate time series anomaly detection methods mainly include traditional statistical and machine learning methods, as well as deep learning models. Traditional methods [5]–[7] generally process the input in a linear manner to detect outliers based on factors such as distance and density. More recently, a series of deep learning models have been proposed to further learn the non-linear relationship of timestamps and features. These models are typically categorized into two types: prediction-based and reconstruction-based. In particular, the prediction-based models [8], [9] mainly learn the trends of historical data and predict future behaviors. The reconstruction-based models [10], [11] focus on learning latent representations of normal patterns to reconstruct the original input. With these approaches, the anomalies are distinguished using prediction or reconstruction errors, respectively. However, existing methods are not suitable for modern large-scale distributed systems for the following reasons:

Ignore spatio-temporal correlations in a multi-KPI environment. Complex spatio-temporal dependencies exist among timestamps and KPI features in the multivariate time series, which are highly related to the abnormal patterns [12]. The occurrence of an abnormal KPI could be root in either its temporal trend over a period of time or the impact of changes in other KPIs. In this context, traditional methods face inherent limitations in extracting joint correlations due to their linear and discrete processing manners, while deep learning models have sought to address the limitation through various techniques. For instance, Graph Neural Networks (GNNs) are employed to capture spatial dependencies by constructing topological graphs among features in [9], [13]. Meanwhile, models based on Recurrent Neural Networks (RNNs) [10], and Transformer [14], [15] focus on identifying long and short-term temporal trends. However, these approaches often struggle to comprehensively capture both spatial and temporal dependencies, thus limiting their performance in multivariate time series detection scenarios.

Not applicable to unlabeled data conditions. According to the study [16], most of existing deep learning models are actually **semi-supervised**. They are generally trained on normal samples to minimize prediction or reconstruction errors, with an overreliance on fitting normal patterns in the training

Bolong Zheng is the corresponding author.

2375-026X/25/$31.00 ©2025 IEEE
DOI 10.1109/ICDE65448.2025.00303

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

phase. However, in most practical detection scenarios, the availability of labeled data is limited due to high labor costs and ambiguous judgment criteria. As a result, the training sets often contain a mix of unlabeled normal and abnormal samples, rendering semi-supervised models not applicable to such unsupervised conditions.

Ineffective in real-world anomaly detection tasks. The benchmark study [4] reveals that traditional unsupervised methods generally exhibit weak performance. The few unsupervised deep detection models [17]–[19] suffer from sensitivity to the proportion of anomalies and parameter settings, leading to constrained detection capabilities. Moreover, when the detection scenario involves a distributed system with multiple nodes, existing methods typically struggle to extract the common patterns across nodes. Instead, they are deployed independently on each node, resulting in increased resource consumption and significant delays during cold starts. Therefore, it is challenging to implement an effective detection method in real-world unsupervised and distributed scenarios.

Considering the above limitations, an excellent anomaly detection method should meet following requirements: (**i**) Thoroughly explore the complex spatio-temporal correlations and patterns; (**ii**) Demonstrate effective detection performance under unsupervised conditions with unlabeled samples; (**iii**) Exhibit strong generalization across target nodes to minimize deployment costs and cold-start delays.

In response to aforementioned requirements, we first design a multi-perspective spatio-temporal attention model (**STAMP**) for semi-supervised anomaly detection. STAMP consists of a prediction module ST-ATTN, a reconstruction module AutoEncoder, and an adversarial optimization module. To capture the complex spatio-temporal correlations among timestamps and features, ST-ATTN incorporates multiple attention components thereby taking account of both local and global perspectives (*Requirement i*). AutoEncoder maps the input into a latent space to further learn the normal representation. In the adversarial optimization module, we build positive and negative data pairs and then conduct adversarial training via *min-max* strategy between ST-ATTN and AutoEncoder.

Second, we introduce two pre-training strategies on the data sampled from all target nodes in distributed systems. These strategies aim to extract common correlations and patterns so that enhance the generalization ability of STAMP. Given that the correlations of KPI features are typically consistent across target nodes, the trained weights of **(a)** the graph structure or **(b)** the spatial learning components are loaded into a new STAMP model and are frozen. This approach enables the deploying of a single model that can effectively adapt to the detection task of distributed systems (*Requirement iii*).

Third, based on STAMP, we propose an unsupervised anomaly detection framework. The framework initially employs traditional unsupervised methods to screen raw samples and generate a training set for STAMP. Subsequently, the trained STAMP is utilized to perform the detection task. To further enhance detection performance, the framework is optimized through a feature selection process based on three types of model-derived information, ensuring higher-quality training samples. The optimized framework can effectively adapt to the non-label detection scenarios with high performance (*Requirement ii, iii*).

Overall, the contributions are summarized as follows:

(i) We propose a multi-perspective spatio-temporal attention model STAMP for multivariate time series anomaly detection. The model adopts attention and convolutional networks to focus on capturing the spatio-temporal correlations from multiple perspectives.

(ii) We further design a novel unsupervised detection framework that integrates traditional unsupervised methods with STAMP, thereby enabling adaption to the practical scenarios with unlabeled samples. In addition, the framework is compatible for other existing semi-supervised detection models.

(iii) We introduce two pre-training strategies to improve the generalization ability of STAMP so that achieve satisfying detection performance and efficiency across different target nodes.

(iv) We conducted extensive experiments under both semi-supervised and unsupervised conditions in multiple real-world datasets. The experimental results show that our model and framework outperform baseline methods.

II. Related Work

A. Semi-Supervised Anomaly Detection

The existing semi-supervised anomaly detection methods are generally based on deep learning models. Compared to traditional statistical or machine learning methods, deep models can better handle non-linear and high-dimensional data patterns, leading to higher accuracy and stronger robustness.

Some reconstruction based models [20]–[22] map the samples to the latent space and learn the representation of normal patterns. These methods attempt to recover the original input and detect anomalies with the reconstruction error. OmniAnomaly [10] utilizes VAE as the reconstruction method, which ensures the latent space is continuous and regularized to enhance the robustness. USAD [23] introduces an Encoder-Decoder based adversarial training procedure, so that effectively addresses the problem of distinguishing the reconstruction errors between normal and abnormal samples. Some other models are based on the prediction task. They try to mine the trends of multivariate time series to predict future behaviors, and then detect anomalies with the error between predicted values and true values. LSTM-NDT [8] extracts a time series for each feature and then utilizes LSTM for single-step prediction. GDN [9] and GTA [24] attempt to build self-learning graphs and then capture feature correlations with GNNs.

B. Unsupervised Anomaly Detection

The unsupervised anomaly detection methods are mostly based on traditional statistical and machine learning techniques, including distance-based methods [7], [25], distribution-based methods [5], [26], cluster-based methods

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

[27], [28], and isolation tree based methods [6]. However, these methods generally compute dependencies in a linear manner, which restricts the detection performance while dealing with high-dimensional samples.

In unsupervised detection tasks, the training set comprises a mix of unlabeled normal and abnormal samples, significantly increasing the learning difficulty. Consequently, deep learning based unsupervised detection methods remain relatively scarce. InterFusion [18] adds a pre-trained filter layer to the model while training. In the filter layer, an embedding procedure is performed to obtain a representation of abnormal input that approximates normal data. Bi-VQRAE [19] modifies the logarithmic likelihood term of the optimizing function ELBO. It introduces new divergence functions to suppress the impact of abnormal input on the training process. However, these methods require reconfiguration of the learning procedures when the data distribution changes. Moreover, they are typically ineffective in situations where the proportion of anomalies in the training set is relatively high.

C. Spatio-Temporal Relationship Learning

Recent studies generally address this task by designing deep models with spatial and temporal learning modules. In the temporal modules, RNNs, TCN, and Attention are usually utilized, and the spatial modules are popularly based on CNN, GNNs, and Attention. ConvLSTM [29] replaces the fully connected layer in LSTM with 2D-convolution to further learn spatial relationships. Graph WaveNet [30] and MTGNN [31] introduce adaptive adjacent matrix to build topology graph between feature variables. They both adopt GCN [32] for spatial learning and TCN [33] for temporal learning. Some other studies are based on attention mechanisms, ASTGNN [34] ulilizes Multi-Head Attention [35] to study the correlation among timestamps globally. MTAD-GAT [36] adopts parallel GATs [37] to capture feature dependencies and temporal trends via graph learning.

III. Preliminaries

A. Problem Settings

Definition 1 (Time-Windowed Sequence): The multivariate time series is denoted as the time sequence $\mathcal{X} = \{X_1, \ldots, X_t, \ldots, X_T\}$, where $X_t \in \mathbb{R}^N$ is the data collected at timestamp t with N features, and T denotes the length of each sequence. To capture the correlations among timestamps, we split the original sequence into overlapping time windows. A time window of length w corresponding to timestamp t is represented as $W_t = \{X_{t-w+1}, \ldots, X_t\}$, which includes data from $t-w+1$ to t. Consequently, a time-windowed sequence $\mathcal{W} = \{W_w, \ldots, W_t, \ldots, W_T\}$ is derived from $\mathcal{X}$, with each window corresponding to a specific timestamp.

Definition 2 (Time Window Prediction): Given the time window $W_t = \{X_{t-w+1}, \ldots, X_t\}$ at timestamp t and the target step p, we split W_t into $X_{t-w+1:t-p}$ and $X_{t-p+1:t}$, which represents the input and target respectively. Hence, the task of time window prediction can be defined as using a specific method f to generate the predicted result $\tilde{X}_{t-p+1:t}$ based on the historical data $X_{t-w+1:t-p}$:

$$[X_{t-w+1:t-p}] \xrightarrow{f} \tilde{X}_{t-p+1:t}, \tag{1}$$

The loss function is defined as:

$$Loss_{pred} = ||X_{t-p+1:t} - \tilde{X}_{t-p+1:t}||_2^2, \tag{2}$$

where $||\cdot||_2$ represents L2-normalization.

Definition 3 (Time Window Reconstruction): Given the time window $W_t = \{X_{t-w+1}, \ldots, X_t\}$ at timestamp t, we consider W_t as both the input and target. Hence, the task of time window reconstruction can be defined as using a specific method g to reduce the original W_t to the output $R(W_t)$:

$$W_t \xrightarrow{g} R(W_t), \tag{3}$$

The loss function is defined as:

$$Loss_{rec} = ||W_t - R(W_t)||_2^2, \tag{4}$$

B. Problem Statement

We aim to address the multivariate time series anomaly detection problem, where the samples of operational data are integrated into time sequences $\mathcal{X}_{train}$ for training and $\mathcal{X}_{test}$ for testing. There are a certain proportion of labeled anomalies in $\mathcal{X}_{test}$. As for $\mathcal{X}_{train}$, the samples are all normal under the semi-supervised detection condition, while there are a certain proportion of **unlabeled** anomalies under the unsupervised condition. We first derive $\mathcal{X}_{train}$ and $\mathcal{X}_{test}$ to time windowed sequences $\mathcal{W}_{train}$ and $\mathcal{W}_{test}$. Then, STAMP is trained on $\mathcal{W}_{train}$ with the time window prediction and reconstruction tasks. For a specific time window W_t in $\mathcal{W}_{test}$, the trained STAMP computes an anomaly score S_t based on the error between the ground truth and the model output. Given an anomaly threshold τ, if $S_t > \tau$, we label the observation at timestamp t as an anomaly ($y_t = 1$); Otherwise, it is labeled as a normal one ($y_t = 0$).

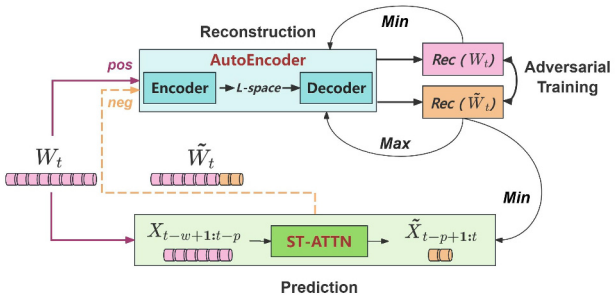


Fig. 1. The Overview of STAMP

IV. Methodology

To address the multivariate time series anomaly detection problem, we design a semi-supervised model and further propose an unsupervised framework.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

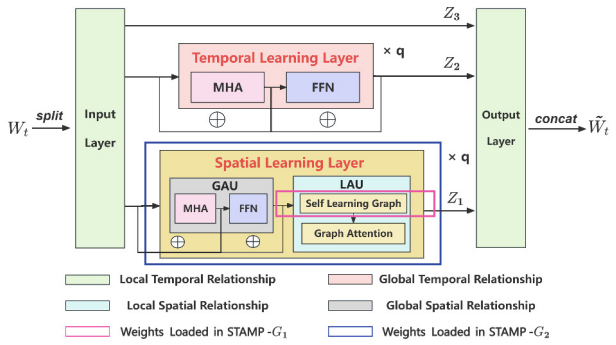


Fig. 2. The Architecture of ST-ATTN

A. Multi-Perspective Spatio-Temporal Attention Model

1) Network Architecture: First, we introduce a multi-perspective spatio-temporal attention model **STAMP**, which is suitable for semi-supervised conditions. STAMP is designed to learn the spatial and temporal relationship and mine the hidden patterns, thereby amplifying prediction and reconstruction errors of abnormal inputs. The architecture overview of STAMP is shown in Figure 1. It consists of three main modules:

- **ST-ATTN:** Utilize Multi-Head Attention, Graph Attention, and convolutional networks to capture spatio-temporal correlations from both local and global perspectives. ST-ATTN primarily addresses the time window prediction task.
- **AutoEncoder:** Map the input to a latent space using an Encoder to learn and capture implicit representations. Subsequently, a Decoder reconstructs the original input, accomplishing the time window reconstruction task.
- **Adversarial Optimization:** Conduct an adversarial training process between ST-ATTN and the AutoEncoder with *min-max* optimization strategy. As a result, the prediction performance of ST-ATTN is enhanced and the overfitting in AutoEncoder is mitigated.

2) ST-ATTN: To deeply explore the complex spatial and temporal correlations of multivariate time series, we design a spatio-temporal attention learning module (ST-ATTN) for the time window prediction task. The architecture of it is shown in Figure 2. ST-ATTN consists of three components, which are the Convolutional Input-Output Layer, the Temporal Learning Layer, and the Spatial Learning Layer.

Convolutional Input-Output Layer. In order to extract long-short-term patterns and reduce the impact of noise, we first perform a moving average operation by sub-windows of length $\in$ [1,3,5,10] to extend the original data to four individual channels. After that, we split the generated window W_t into the historical input $X_{t-w+1:t-p} \in R^{(w-p)\times N\times 4}$ and the target $X_{t-p+1:t} \in R^{p\times N\times 4}$. p represents the prediction step for determining how many future timestamps to be generated. Two Conv2D layers with kernel size $\in 5\times 1$ are further adopted to generate the predicted output. Hence, the overall computation process of Convolutional Input-Output Layers can be denoted as:

$$\begin{aligned} Z_t &= Conv2D_{input}(X_{t-w+1:t-p}), \\ \tilde{X}_{t-p+1:t} &= Conv2D_{output}([Z_t; TLL(Z_t); SLL(Z_t)]), \\ \tilde{W}_t &= [X_{t-w+1:t-p}; \tilde{X}_{t-p+1:t}] \end{aligned} \quad (5)$$

where $\tilde{W}_t$ is the predicted result and ";" denotes the splitting and concatenating operation. In addition to matching the input and output dimensions, the convolutional layers also preliminarily learn the temporal relationships from a local perspective with the receptive field size of 5.

Temporal Learning Layer. Since Conv2Ds only learn the correlations within adjacent timestamps, we ulteriorly design the Temporal Learning Layer (TLL). It utilizes Multi-Head Self-Attention mechanism (MHA) [35] to capture temporal relationships from a global perspective among all timestamps. In TLL, We first split $Z_t \in R^{(w-p)\times N}$ into h attention heads, thereby enriching the representation space. For head i, the self-attention learning process is denoted as:

$$head_i = Softmax(\frac{Z_t^i M_q \cdot Z_t^i M_k{}^T}{\sqrt{dim_{Z_t^i M_k}}}) Z_t^i M_v \quad (6)$$

where M_q, M_k, M_v are learnable linear transformation matrices. Subsequently, the results of all heads will be concatenated and then sent to the Feed-Forward Network (FFN). Finally, we can obtain the output of TLL as:

$$TLL(Z_t) = FFN([head_1; head_2; ...; head_h]) \quad (7)$$

Additionally, residual connections are incorporated in both the MHA and FFN to prevent gradient vanishing during the training process.

Spatial Learning Layer. Apart from temporal relationships, complex spatial dependencies also exist among feature variables. To address this, we design the Spatial Learning Layer (SLL). SLL introduces MHA and Graph Attention mechanism [37] to capture spatial correlations from the global and local view, respectively. The learning process of Global Attention Unit (GAU) is similar to TLL. But it is worth noting that since the learning targets are features instead of timestamps, we first transpose Z_t to $Z_t^T \in R^{N\times(w-p)}$. Following the learning process of MHA and FFN, the output is denoted as $O_t \in \mathbb{R}^{D\times(w-p)}$, where D represents the embedding size shared by both TLL and SLL.

In addition to global correlations, certain features exhibit stronger local dependencies, such as sensors within the same procedure in industrial systems or attribute indicators of the same server device. To capture these relationships, the Local Attention Unit (LAU) is incorporated in SLL. Inspired by [9], LAU leverages a self-learning graph to construct directed topological connections between feature nodes. Specifically, we generate two node embedding matrices E_1 and $E_2 \in R^{N\times v}$

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

using learnable linear layers. Then, the graph structure A will be obtained through the following process:

$$\begin{aligned} &\textit{for } i = 1, 2, \ldots, N: \\ &A[:, i] = \varphi(E_2 E_1^T[i, :]), \\ &A[i, :] = \varphi(E_1[i, :] E_2^T) \end{aligned} \tag{8}$$

where $A[:, i]$ denotes the in-degree edge weights of feature node i, $A[i, :]$ denotes the out-degree edge weights, and φ is the activation function *Softmax* for normalization. For each node, we only retain k neighbor nodes with the top-k largest edge weights to outstand local dependencies. Building on this foundation, a graph attention learning process is performed to facilitate spatial learning, which is expressed as follows:

$$\begin{aligned} e_{ij} &= \frac{< O_t^i M_1, O_t^j M_1 >}{\sqrt{dim_{O_t^i M_1}}}, \\ \alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})}, \\ SLL(Z_t^i) &= \sigma(O_t^i M_2 + \sum_{j \in \mathcal{N}(i)} \alpha_{ij} O_t^j M_2) \end{aligned} \tag{9}$$

where $<, >$ is the inner product operation, α_{ij} is the attention weight between node i and j, $\mathcal{N}(i)$ contains the neighbors of node i, M_1 and M_2 are linear transformation matrices.

3) AutoEncoder: To further capture the implicit data patterns, we utilize an AutoEncoder [38] as the time window reconstruction method g. It consists of a pair of Encoder and Decoder. Encoder maps the input window to *L-space* to obtain the latent representation L, while Decoder reduces L back to the original space to obtain the reconstructed output. In our STAMP, Encoder and Decoder are symmetric networks each consisting of three linear layers.

4) Adversarial Optimization: The Adversarial Optimization Module is designed to address the overfitting issue commonly observed when training the AutoEncoder on normal samples by minimizing $Loss_{rec}$. The trained AutoEncoder tends to perform completely lossless reconstruction on any input window. This behavior results in insignificant reconstruction errors for abnormal samples during inference, making it challenging to effectively distinguish anomalies. The problem can be solved if AutoEncoder is able to determine whether the input sample is normal and adjust the optimization target during the training phase [13]. Inspired by former studies [13], [23], we perform the adversarial training process between ST-ATTN and AutoEncoder based on a *min-max* optimizing strategy to overcome the performance bottlenecks of both modules.

Specifically, in the Adversarial Optimization Module, the AutoEncoder receives pairs of positive and negative samples. The positive sample is the original time window W_t, while the negative sample $\tilde{W}_t$, is generated by ST-ATTN. $\tilde{W}_t$ is formed by concatenating the historical data $X_{t-w+1:t-p}$ and the predicted values $\tilde{X}_{t-p+1:t}$. The adversarial optimization process is divided into two stages. In the first stage, a minimization strategy is employed to train both ST-ATTN and AutoEncoder, with the corresponding optimization objective:

$$\begin{aligned} Loss_1 &= \underset{Min}{Loss}\ (\textit{ST-ATTN}) + \underset{Min}{Loss}\ (\textit{AutoEncoder}) \\ &= \eta_1 ||X_{t-p+1:t} - \tilde{X}_{t-p+1:t}||_2^2 + \eta_2 ||W_t - R(\tilde{W}_t)||_2^2 \end{aligned} \tag{10}$$

where η_1 and η_2 are weight factors that change with training epochs. This stage aims to enhance the prediction performance of ST-ATTN in order to generate negative samples that closely approximate the true values thereby confusing AutoEncoder. The more accurate the prediction results are, the higher training quality is achieved in the subsequent stage.

In the second stage, we train AutoEncoder via minimum (for W_t) and maximum (for $\tilde{W}_t$) strategies, respectively. The corresponding optimization objective is:

$$\begin{aligned} Loss_2 &= \underset{Min}{Loss}\ (\textit{AutoEncoder}) - \underset{Max}{Loss}\ (\textit{AutoEncoder}) \\ &= \eta_1 ||W_t - R(W_t)||_2^2 - \eta_2 ||W_t - R(\tilde{W}_t)||_2^2 \end{aligned} \tag{11}$$

It can be observed that AutoEncoder is trained to distinguish between positive and negative inputs. Since the optimization objective is no longer minimizing the reconstruction loss for any input, the overfitting of AutoEncoder can be avoided so that the detection performance is improved.

5) Anomaly Detection: During the training phase, the prediction, reconstruction, and adversarial modules of STAMP are set to be conducted sequentially. For the inference phase, given a trained STAMP model and a specific time window W_t as the detection target, we compute the prediction error Err_{pred}, reconstruction error Err_{rec}, and adversarial error Err_{adv} using the following formulas:

$$\begin{aligned} \textit{Err}_{\textit{pred}} &= \sum_{i \in top\tilde{k}(N)} AGG_{t-p+1:t} \, |X_{t-p+1:t}^i - \tilde{X}_{t-p+1:t}^i|, \\ \textit{Err}_{\textit{rec}} &= \sum_{i \in top\tilde{k}(N)} AGG_t \, |W_t^i - R(W_t^i)|, \\ \textit{Err}_{\textit{adv}} &= \sum_{i \in top\tilde{k}(N)} AGG_t \, |W_t^i - R(\tilde{W}_t^i)| \end{aligned} \tag{12}$$

To solve the anomaly detection task, we first aggregate the errors of each feature across different timestamps. The aggregation operator AGG can be configured to compute either the maximum or the average value. Subsequently, we retain the features with top-$\tilde{k}$ largest errors and sum the errors up. The three kinds of errors are then combined to obtain the anomaly score S_t as:

$$S_t = \alpha \textit{Err}_{\textit{pred}} + \beta \textit{Err}_{\textit{rec}} + \gamma \textit{Err}_{\textit{adv}} \tag{13}$$

where α, β, and γ are weight factors. Finally, the observation at timestamp t is labeled as an anomaly if S_t exceeds the threshold τ.

6) Model Generalization: The generalization ability is an important indicator in real-world anomaly detection scenarios. Training the model separately requires a large memory space and a long deployment time delay when there are multiple nodes in a distributed system. The most efficient solution is

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

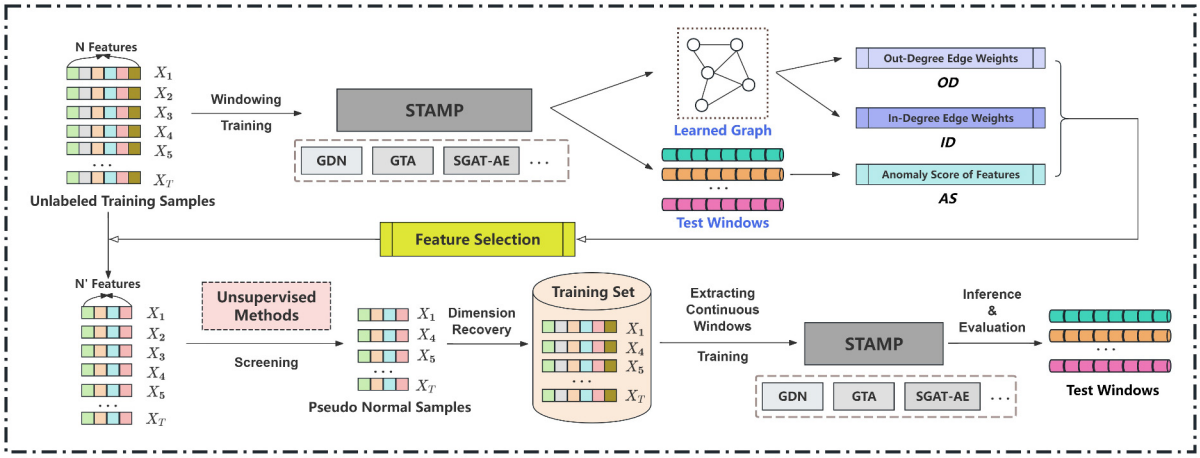


Fig. 3. The unsupervised framework integrates the basic unsupervised screening procedure with the feature selection process based on model-derived information. Notably, STAMP must be trained in advance to ensure the extraction of effective information. Additionally, the screened samples need to be recovered to the original feature dimension to maintain data integrity.

Algorithm 1: Generalization in a Distributed System

Input:
Independent nodes $\{N_1, N_2, \ldots, N_n\}$, training target N_0;
Obtain the pre-training set $\mathcal{X}_p$ by sparse sampling from $\{N_1, N_2, \ldots, N_n\}$;
$\mathcal{X}_p \xrightarrow{input}$ **STAMP** for pre-training;
Reinitialize STAMP;
if *Strategy I* **then**
    Load the learned weights of graph A (**STAMP-**G_1);
else if *Strategy II* **then**
    Load the learned weights of SLLs (**STAMP-**G_2);
Freeze the loaded weights;
$N_0 \rightarrow$ training set $\mathcal{X}_0 \xrightarrow{input}$ STAMP for training;
Apply STAMP to $\{N_0, N_1, N_2, \ldots, N_n\}$ for detection;

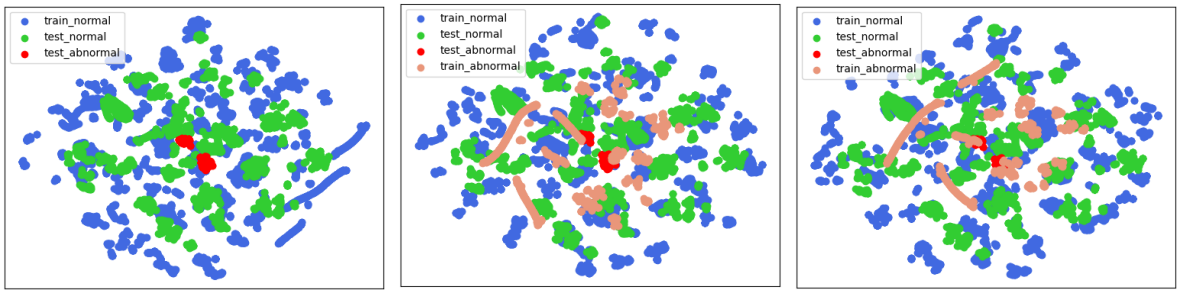
to train on a specific target node and then apply it to the others, which requires a strong generalization effectiveness to meet the performance requirements. We posit that the spatial learning capability of the model has a more significant impact on generalization. The reason is that the correlations of features are generally consistent across nodes, whereas that of timestamps often vary. Consider the detection scenario of a cluster comprising independent servers, the interactions between KPIs are consistent across all servers. For example, response time is influenced by throughput, and high utilization often results in longer queue lengths. In contrast, the temporal trends of each KPI can differ significantly among servers due to variations in parameter settings and application conditions.

Based on the above analysis, we introduce two pre-training strategies to enhance the generalization ability of STAMP across distributed nodes. In the strategies, STAMP is first pre-trained on the data collected from all nodes to enable its spatial learning components to extract unified correlations and patterns. To minimize pre-training costs, the number of epochs is kept small and the data is obtained through sparse sampling. After pre-training, certain learned parameter weights are loaded into a new STAMP model. As shown in Figure 2, the pre-training strategy I **only involves loading the learned graph structure** A (the pink square frame), while the rest components of STAMP are reinitialized. In the pre-training strategy II, **we consider loading all the component weights of SLLs, including GAU and LAU** (the blue square frame). In the subsequent process of training on a single target node, the loaded weights are frozen. We name the model versions corresponding to the two strategies as STAMP-G_1 and STAMP-G_2. Comparatively, STAMP-G_1 can be applied more flexibly, while STAMP-G_2 is able to capture richer feature characteristics. The detailed generalization procedures are shown in Algorithm 1.

B. Unsupervised Anomaly Detection Framework

1) Basic Detection Framework: As mentioned in *Section II*, most existing deep learning models rely on capturing normal patterns from the training samples to achieve optimal detection performance. However, it is challenging to assign prior labels in most real-world scenarios due to high labor costs and ambiguous judgment criteria. As a result, the training sets often comprise a mixture of unlabeled normal and abnormal samples, which is referred to as the unsupervised detection condition. Some traditional statistical and machine learning methods such as IForest [6], COPOD [5], and KMeans [27] can be applied under this condition, but their detection performance is generally too weak to meet practical needs. A small number of deep learning models achieve unsupervised detection by modifying the optimization objective or incorporating pre-trained data transformation networks [18], [19]. However, these models are often sensitive to the anomaly distribution and parameter configurations, resulting in weaker detection capabilities compared to advanced semi-supervised models.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.



(a) Without Screening (b) Basic Screening by IForest (c) Screening by IForest after Feature Selection

Fig. 4. We utilize the manifold learning technique t-SNE to visualize normal and abnormal samples in both training and test sets. In this case, the test set with true labels in three subfigures are the same. The training set in (a) is derived without screening so that there are no training samples labeled as abnormal; In contrast, we perform a basic screening process on training samples without relying on any model-derived information in (b); Furthermore, feature selection is performed before screening based on the information type *ID* in (c).

Algorithm 2: Unsupervised Detection Framework

Input:
Training samples $\mathcal{X}_{train} = \{X_1, X_2, \ldots, X_T\} \in R^{T \times N}$;
Test windows $\mathcal{W}_{test} = \{W_{T+w}, W_{T+w+1}, \ldots, W_{T+T'}\}$;
Output:
Labels $Y'_{test} = \{Y'_{T+w}, Y'_{T+w+1}, \ldots, Y'_{T+T'}\}$
if *Optimizing with model information* **then**
  $\mathcal{X}_{train} \rightarrow \mathcal{W}_{train} = \{W_w, W_{w+1}, \ldots, W_T\}$;
  $\mathcal{W}_{train} \xrightarrow{input}$ **STAMP** for training;
  Obtain ***OD*** an ***ID*** from trained STAMP;
  $\mathcal{W}_{test} \xrightarrow{input}$ STAMP for inference and obtain ***AS***;
  Conduct feature selection and derive $\mathcal{X}_{train} \in R^{T \times N'}$
Perform a **screening method** on $\mathcal{X}_{train}$ for pseudo labels;
if *Have conducted feature selection* **then**
  Recover $\mathcal{X}_{train}$ to the original feature dimension $R^{T \times N}$
$\mathcal{X}'_{train} \in R^{T' \times N} \leftarrow \mathcal{X}_{train}$ by selecting samples labeled as normal;
$\mathcal{W}'_{train} \leftarrow \mathcal{X}'_{train}$ by extracting continuous timestamps;
$\mathcal{W}'_{train} \xrightarrow{input}$ STAMP for training;
$\mathcal{W}_{test} \xrightarrow{input}$ trained STAMP $\xrightarrow{output} Y'_{test}$;
Return Y'_{test};

To address the unsupervised anomaly detection task, we propose an integrated framework that combines traditional unsupervised methods with our semi-supervised STAMP. The core design intuition is to leverage the flexibility and efficiency of traditional methods while harnessing the superior detection performance of STAMP, thereby overcoming the limitations of both approaches. Initially, traditional unsupervised methods are employed to assign pseudo-labels to the original training data. Samples identified as normal are then selected to form a new training set, a process referred to as "screening". Given that the selected samples are often discretely distributed across timestamps, we further segment them into continuous time windows to serve as the inputs of STAMP.

2) Framework Optimization Based on Model Information: The above basic detection framework has two notable shortcomings. Firstly, the screening methods are unable to completely filter out all abnormal samples. This limitation becomes especially pronounced when the feature dimension is high, resulting in the inclusion of more anomalies in the training set and subsequently impairing the detection performance of the model. Secondly, the screening process and the detection model operate independently without any exchange of information. To address these limitations, we optimize the basic framework by incorporating model-derived information. Specifically, given the model information, we perform feature selection on the original training samples to reduce the dimension of samples prior to the screening process. This enhancement aims to improve the screening accuracy and the overall detection capability.

To achieve feature selection, we define three types of model-derived information to quantify the importance of each feature:

- **Out-Degree (*OD*):** Derived from the learned graph A in LAU. For feature i, the *OD* importance is calculated as the sum of its out-degree edge weights:

$$OD(i) = \sum_{j \in \mathcal{N}(i)} A[i, j] \tag{14}$$

- **In-Degree (*ID*):** Similarly, for feature i, the *ID* importance is calculated as the sum of its in-degree weights:

$$ID(i) = \sum_{i \in \mathcal{N}(j)} A[j, i] \tag{15}$$

- **Anomaly Score (*AS*):** Given the time-windowed sequences $\{W_1, \ldots, W_t, \ldots, W_T\}$ in the test set, the *AS* importance for feature i is computed as the sum of its anomaly scores across all time windows:

$$AS(i) = \sum_{t=1}^{T} S_t^i \tag{16}$$

Once the feature importance is calculated, a subset of less significant features will be discarded prior to screening .The feature selection process not only reduce the dimension of samples so that improve the screening accuracy, but also emphasize the features deemed critical by the detection model. As a result of this optimization, the abnormal proportion in the formed training set will decrease and the model will receive

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

TABLE I
COMPARISON WITH BASELINE MODELS FOR SEMI-SUPERVISED DETECTION

Method	SWaT			WADI			SMD			MSL			Avg. F1
	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	
IForest	0.731	0.847	0.785	0.380	0.798	0.514	0.565	0.976	0.716	0.539	0.965	0.692	0.677
COPOD	0.618	0.926	0.783	0.345	0.796	0.482	0.537	0.966	0.690	0.546	0.969	0.699	0.664
HBOS	0.688	0.870	0.768	0.294	0.650	0.405	0.549	0.956	0.697	0.732	0.845	0.784	0.664
GTA	0.924	0.899	0.911	0.805	0.811	0.808	0.885	0.974	0.927	0.608	0.623	0.616	0.816
GDN	0.912	0.799	0.851	0.747	0.588	0.612	0.972	0.915	0.942	0.876	0.809	0.841	0.812
TranAD	0.975	0.778	0.863	0.706	0.601	0.649	0.919	0.997	0.956	0.904	1.000	**0.950**	0.855
RANSynCoders	0.964	0.715	0.821	0.432	0.450	0.441	0.732	0.945	0.825	0.859	0.889	0.874	0.740
SGAT-AE	0.893	0.975	0.932	0.916	0.789	0.848	0.934	0.978	0.955	0.860	0.919	0.889	0.906
AnomalyTransformer	0.922	0.894	0.907	0.762	0.687	0.722	0.907	0.979	0.942	0.918	0.951	0.934	0.876
LUNAR	0.722	0.991	0.836	0.721	0.682	0.701	0.957	0.918	0.937	0.911	0.888	0.900	0.843
STAMP	0.991	0.975	**0.983**	0.978	0.804	**0.883**	0.955	0.984	**0.969**	0.776	0.998	0.873	**0.928**

samples with higher quality. The procedures of the framework are outlined in Algorithm 2. The detailed architecture of the framework is illustrated in Figure 3.

To demonstrate the effectiveness of the screening process and its optimization, we visualize samples from an unsupervised dataset using t-SNE [39]. The manifold distributions of normal and abnormal samples in both training and test sets are shown in Figure 4. In Figure 4(a), the manifold regions of training and test samples are mixed together. Specifically, setting the judgment radius to 5, 74% of the test anomalies (the red region) overlap with the training samples (the blue region), thus make it difficult for the model to distinguish them. However, it can be observed in Figure 4(b) that some of the training samples are assigned to be abnormal (the brown region) and then filtered out via basic screening, reducing the overlap proportion of test anomalies to 48%. Further optimization of feature selection based on information type *ID* (Figure 4(c)) leads to a minor overlap proportion of 17%, while the retained samples align more closely with test normals (the green region). This refined separation of training samples can significantly improve the detection performance.

In practical terms, the key advantage of the proposed framework is that it is compatible for a considerable number of other semi-supervised detection models such as GDN [9], GTA [24], SGAT-AE [13], and FuSQGNET [40]. These models can be integrated into the framework in replace of STAMP, thereby breaking through their limitation of unsupervised detection.

V. EXPERIMENT

To validate the effectiveness of the proposed detection model and framework, we conduct multiple experiments under both semi-supervised and unsupervised conditions. The experimental results are presented and analyzed in this section. All experiments are run on a Linux server equipped with a NVIDIA RTX A6000 GPU.

A. Experiments on Models

We begin by validating the detection performance of STAMP through comparisons with baseline methods under semi-supervised conditions. In these experiments, STAMP and other deep learning models are trained on normal inputs and evaluated on a labeled test set containing anomalies.

1) Experimental Setup:

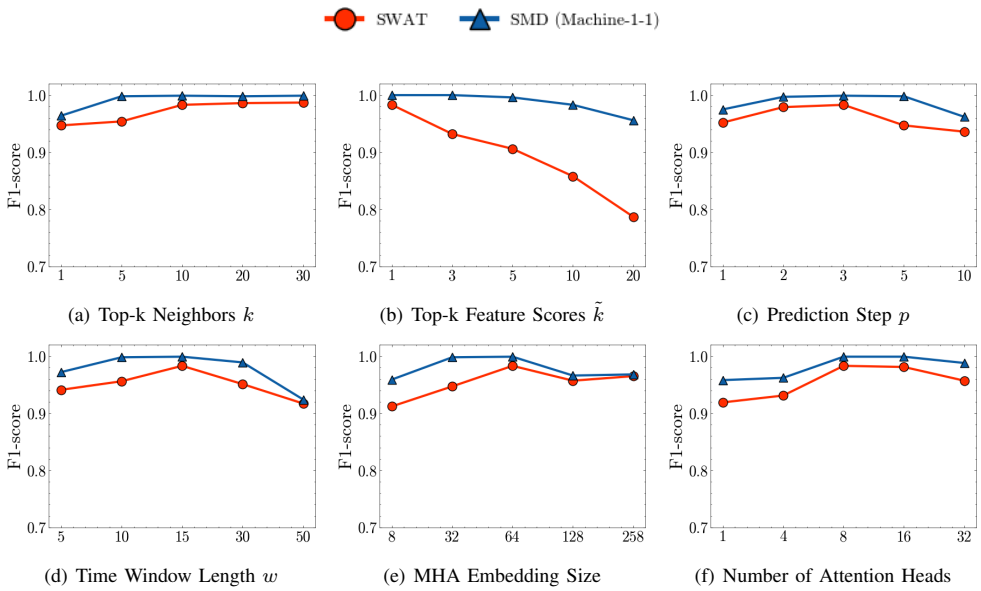
Datasets. We use the following four publicly available datasets with labeled anomalies to evaluate our model and other baseline methods. Note that the original sequence of samples should be split into time windows.

- **SWaT:** This dataset comes from the operational data of a real water treatment plant over a period of 11 days. The first 7 days collected under normal operations are regarded as the training set and the last 4 days collected with attack scenarios are regarded as the test set.
- **WADI:** This dataset is an extension of SWaT, which includes 127 KPIs. It is collected from 16 days of operational data, including 14 days of normal operation for training and 2 days with attack for testing.
- **SMD:** This dataset originates from the operational data of a large internet company spanning over a period of 5 weeks. It consists of 28 subsets and each of them corresponds to an individual server machine. We only consider four nontrivial subsets (Machine-1-1, 2-1, 3-2 and 3-7) pointed out by [15].
- **MSL:** This dataset is provided by NASA and collected from sensors and operators of the Mars rover. The size of the test set is approximately 1.2 times of training set.

Baselines. We compare our STAMP with nine baseline methods. There are three traditional detection methods: the outlier-based method IForest [6], the probability-based method COPOD [5], and the proximity-based method HBOS [41]. These methods are training-free and can be directly applied to the inference and evaluation phase.

The other seven methods are advanced deep learning models. GDN [9] and GTA [24] are both prediction-based models that combine self-learning graphs with GCN; TranAD [15] is based on the Encoder-Decoder architecture of Transformer and utilizes adversarial training to enhance stability; RANSynCoders [42] bulids asynchronous time series and then performs the Fourier transform; SGAT-AE [13] adopts convolutional layers and GAT to learn inherent relationships for accurate pre-

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.



(a) Top-k Neighbors k (b) Top-k Feature Scores $\tilde{k}$ (c) Prediction Step p

(d) Time Window Length w (e) MHA Embedding Size (f) Number of Attention Heads

Fig. 5. The Effects of Model Parameters

diction; AnomalyTransformer [14] initiatively introduces the association discrepancy for detection; LUNAR [17] integrates GNN with local outlier algorithms and can also be applied to unsupervised conditions by setting the abnormal proportion.

Evaluation Metrics. We choose three classification metrics, Precision (Pre), Recall (Rec), and F1-score (F1) to evaluate the detection performance of each method:

$$Pre = \frac{TP}{TP+FP}, \; Rec = \frac{TP}{TP+FN}, \; F1 = \frac{2*Pre*Rec}{Pre+Rec}$$

where TP, TN, FP, and FN are the sample numbers of true positives, true negatives, false positives, and false negatives, respectively. While evaluating, we adopt the widely-used point-adjustment strategy proposed by [43] to each method: if a timestamp in a certain successive abnormal segment is detected, all anomalies in this abnormal segment are viewed to be correctly detected.

Parameter Setting. The parameter settings of STAMP model and related detection process are as follows:

- **Data Preprocessing:** For dataset SWaT and WADI, the training and test sets are down-sampled at a ratio of 1:50 to reduce training and inference time and mitigate overfitting. To ensure fair comparison, all baseline methods use the same down-sampled data. The time window length w is set to 15, with a prediction step p of 3. When labeling for the test set, a window is considered as normal only if all timestamps within it are normal.
- **STAMP:** In ST-ATTN, the convolutional layers have 32 (Input) and 64 (Output) channels, and the number of TLL and SLL layers is set to 2. For attention components, the embedding size is 64, along with 8 heads. In LAU, the node embedding dimension is set to 32, and the number of neighbors is 10. In AutoEncoder, we set the dimension of *L-space* to 1.
- **Training and Inference:** In the training phase, MSE Loss is used as the loss function, with a batch size of 64. STAMP is able to achieve optimal detection performance with relatively few epochs, thus ensuring efficient deployment (Specifically, 10 epochs in SWaT, 20 in WADI, 5 in SMD, and 40 in MSL). In the inference phase, we set $\tilde{k}$ to 1. The allocation of error weight factors varies in different datasets, which are α = 0.8, β = 0.1, and γ = 0.1 in SWaT; 0.1, 0.1, 0.8 in WADI; 0.5, 0.1, 0.4 in SMD; and 0.1, 0.8, 0.1 in MSL. Consistent with [10], [44], we adopt grid searching to get possible anomaly thresholds τ for each test set and report the best F1-score along with the corresponding Precision and Recall.

2) Comparison with Baseline Methods: Table I shows the performance of different methods in the four datasets. We have the following observations:

(i) The performance of traditional methods in most datasets is not significantly weaker than deep models. Additionally, we find that the metrics remain stable without point-adjustment, indicating that they are able to capture segmented anomalies. These findings demonstrate the anomaly screening capability of traditional methods, thus ensure the reliability of our unsupervised framework.

(ii) The performance of different deep learning models varies across datasets. GTA focuses on graph learning to capture the spatial relationship. It achieves better results in SWaT and WADI, of which the hidden dependencies among features are complex; RANSynCoders focuses on learning in the frequency domain but exhibits significantly poor performance in WADI when the variation of data dissatisfies periodicity; TranAD and AnomalyTransformer utilize

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

TABLE II
COMPARISON WITH VARIANTS OF STAMP

Variant	SWaT			SMD (Machine-1-1)		
	Pre	Rec	F1	Pre	Rec	F1
w/o Adversarial	0.985	0.858	0.917	0.999	0.832	0.908
w/o GAU	0.950	0.828	0.884	0.953	0.989	0.970
w/o LAU	0.985	0.779	0.870	0.959	0.927	0.943
w/o TLL	0.980	0.926	0.952	0.989	0.968	0.978
w/o SLL	0.816	0.926	0.868	0.967	0.882	0.923
w/o Attention	0.795	0.905	0.846	0.927	0.882	0.904
STAMP	0.991	0.975	**0.983**	1.000	1.000	**1.000**

Transformer to conduct reconstruction thorough temporal learning. They achieve optimal detection performance in the low-dimensional datasets (i.e. SMD and MSL) with less feature inter-correlation. SGAT-AE, with a network architecture similar to STAMP, performs slightly worse than STAMP due to its weaker capability of learning spatio-temporal relationships.

(iii) Benefited from the multi-perspective spatio-temporal learning components, the integration of prediction and reconstruction tasks, and the adversarial optimization based on *min-max* strategy, our STAMP achieves the best F1 performance on average. More specifically, it improves the F1-score by **5.5**% in SWaT, **5.1**% in WADI, and **1.4**% in SMD. However, STAMP fails to achieve the best performance in MSL. A probable reason is that the spatial learning components get over-parameterized since the feature correlations are relatively weak and sparse.

3) Effect of Parameters: We analyze the effects of different parameters on the performance of STAMP. The experiments are carried out in dataset SWaT and the subset Machine-1-1 of SMD. Note that, when we vary the value of a parameter, the other parameters are set to the default values. According to the results in Figure 5, we provide the following analysis:

Effect of k and $\tilde{k}$. Our model incorporates two top-k aggregation operations, which are the number of neighbor nodes k and the number of features $\tilde{k}$ used for error calculation. The learned graph tends to be sparse when k is too small, which limits the effect of capturing spatial correlations. As k increases, the performance continues to improve until it reaches an optimal point. Beyond this, additional neighbors contribute little to performance while increasing computational complexity. Based on this analysis, k is set to 10 to balance spatial learning capability and computational efficiency. As for $\tilde{k}$, it can be observed that the performance significantly decreases as $\tilde{k}$ increases. This is because only a small number of features exhibit anomalous states and the discrimination of anomalies will be weakened if we take the normal features into account. Therefore, we restrict $\tilde{k}$ to 1, considering only the feature with the largest error at each timestamp.

Effect of prediction step p and window length w. The two parameters play crucial roles in temporal learning performance. A smaller p limits STAMP's ability to capture sufficient temporal correlations during the short-term prediction process. In contrast, a larger p raises the difficulty of generating long sequences, particularly when combined with a smaller w. Additionally, as w increases, the attention weights between associated timestamp pairs become diluted, thus reducing the effectiveness of temporal correlation learning. To balance these factors, we adopt the settings of $p = 3$ and $w = 15$, which can achieve the best detection performance.

TABLE III
COMPARISON WHILE GENERALIZING AMONG MACHINES

Method	Avg. Pre	Avg. Rec	Avg. F1	Num of Machines with F1 > 0.9
GDN	0.885	0.778	0.828	7
SGAT-AE	0.858	0.835	0.846	9
STAMP	0.872	0.920	0.895	14
STAMP-G_1	0.907	0.925	0.916	20
STAMP-G_2	0.923	0.929	**0.926**	**21**

Effect of the embedding size and heads in attention components. We can infer from the resluts that the representation space of hidden spatio-temporal correlations is limited when we set the two parameters too small. However, the emphasis of representation are diluted and the corresponding model components could be stuck in over-parameterizing once they exceed a certain limit. Hence, we set them to moderate values.

4) Ablation Study: To assess the impact of different components, we compare several variants of STAMP, including:

- **w/o Adversarial.** Remove the Adversarial Optimization Module. STAMP will degenerate into a model that solely focus on the prediction and reconstruction task.
- **w/o GAU.** Replace the Global Attention Unit in SLLs with a linear layer.
- **w/o LAU.** Replace the Local Attention Unit in SLLs with a linear layer.
- **w/o TLL.** Remove the Temporal Learning Layers.
- **w/o SLL.** Remove the Spatial Learning Layers.
- **w/o Attention.** Replace all Multi-Head Attention and Graph Attention components with Conv2D layers.

The results are shown in Table II. We have the following three observations:

(i) The detection performance drops significantly in the absence of the adversarial optimization process. It indicates that adversarial training can effectively improve the learning ability and detection effectiveness. Moreover, when calculating the anomaly score, the adversarial part should also be taken into account for the optimal performance.

(ii) Learning spatial correlations with SLLs is the focal point of STAMP. Compared to GAU, LAU provides a more substantial improvement to the model. This is likely because spatial correlations are typically sparse, with each feature being associated with only a limited number of others. The graph learning process in LAU is better suited for capturing such sparse correlations compared to the global attention mechanism.

(iii) Even with the model architecture integrating prediction, reconstruction, and adversarial optimization modules, the

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

performance of STAMP would be far from optimal without the inclusion of multi-perspective attention components. This highlights the critical role of spatio-temporal relationship learning in enhancing the detectability.

Overall, STAMP consistently outperforms its variants across both datasets, demonstrating the importance and necessity of each component in the proposed model.

5) Generalization Study: We conduct experiments in the distributed dataset SMD that includes 28 independent server machines to evaluate the generalization performance of STAMP. Machine 1-1 is selected as the training target, and the trained model's detection performance is evaluated on the other 27 machines using average Precision, Recall, and F1-score as metrics. Additionally, we compare STAMP with the baseline models GDN and SGAT-AE. It can be observed from Table III that our STAMP has better performance across target nodes than baselines. Furthermore, the performance is further enhanced through optimization with pre-training strategies.

Ulteriorly, we define *F1-score>0.9* as the criterion for determining whether the performance meets the applicable requirement. We then count the number of machines on which the model achieves this criterion. Results show that the original STAMP meets the requirement on over 50% of the machines, while the optimized versions, STAMP-G_1 and STAMP-G_2, exceed the criterion on over 75% of the machines. This highlights the effectiveness and efficiency of STAMP in practical tasks benefited from the pre-training strategies.

B. Experiments on Frameworks

We conduct experiments on our unsupervised detection framework through three aspects. First, we analyze the detection performance of both the basic and optimized frameworks under unsupervised conditions. Some other semi-supervised and unsupervised detection methods are also introduced for comparison to prove the superiority of our framework. Second, we conduct a runtime evaluation of each segment in the framework and then provide options for acceleration. Third, we study a real anomaly case in Huawei DWS, which is successfully detected by our framework.

1) Experimental Setup:

Datasets. Under unsupervised conditions, the initial training data comprises unlabeled samples with a certain proportion of anomalies. Hence, we split the original test sets of SWaT and SMD to derive new datasets for unsupervised detection. For SWaT, we extract the first 300,000 out of 449,919 test samples for unsupervised training, where approximately 16% of the samples are anomalies. The remaining samples are used for testing with true labels. For SMD, we focus on the test set of Machine-3-5 with an uniform anomaly distribution. The first 15,000 out of 23,691 samples are used for training with an anomaly proportion of roughly 1%, while the others for testing. We refer to the two datasets as **SWaT-U** and **SMD-U**.

Experimental Background. We independently fit STAMP on the training sets generated by different screening methods and evaluate their detection performance on the test set. In this phase, we follow the settings in Section V-A1 and adjust the down-sampling ratio according to the amount of samples.

TABLE IV
EXPERIMENTAL RESULTS OF UNSUPERVISED DETECTION

Method	SWaT-U				SMD-U			
	Screening Info.	Pre	Rec	F1	Screening Info.	Pre	Rec	F1
KNN	-	0.488	0.912	0.636	-	0.613	1.000	0.760
LODA	-	0.992	0.466	0.635	-	0.594	0.962	0.735
LOF	-	0.668	0.905	0.767	-	0.681	1.000	0.810
ABOD	-	0.643	0.911	0.754	-	0.783	0.956	0.861
LUNAR	-	0.852	0.816	0.833	-	0.791	0.977	0.874
DIF	-	0.706	0.837	0.765	-	0.877	0.929	0.902
ALAD	-	0.828	0.917	0.870	-	0.734	0.961	0.832
InterFusion	-	0.916	0.858	0.885	-	0.921	0.932	0.926
GDN*	w/o Screening	0.821	0.410	0.547	w/o Screening	0.996	0.725	0.839
	COPOD-*Basic*	0.709	0.634	0.670	COPOD-*Basic*	0.839	0.960	0.895
	COPOD-*OD*	0.687	0.732	0.709	COPOD-*OD*	0.987	0.878	0.927
	HBOS-*Basic*	0.677	0.683	0.680	IForest-*Basic*	0.744	0.878	0.805
	HBOS-*AS*	0.682	0.732	0.706	IForest-*AS*	0.974	0.878	0.924
SGAT-AE*	w/o Screening	1.000	0.665	0.799	w/o Screening	0.983	0.679	0.803
	IForest-*Basic*	0.990	0.900	0.943	DBSCAN-*Basic*	0.857	0.842	0.849
	IForest-*AS*	1.000	0.912	0.954	DBSCAN-*OD*	0.931	0.888	0.909
	CBLOF-*Basic*	1.000	0.797	0.887	COPOD-*Basic*	0.914	0.842	0.876
	CBLOF-*OD*	0.960	0.900	0.929	COPOD-*ID*	0.917	0.897	0.905
STAMP*	w/o Screening	0.957	0.647	0.772	w/o Screening	0.945	0.799	0.868
	IForest-*Basic*	1.000	0.876	0.934	IForest-*Basic*	0.975	0.842	0.903
	IForest-*ID*	0.991	0.994	**0.992**	IForest-*OD*	0.993	0.897	**0.942**
	COPOD-*Basic*	1.000	0.888	0.941	ECOD-*Basic*	0.958	0.842	0.896
	COPOD-*ID*	0.964	0.992	0.977	ECOD-*ID*	0.993	0.842	0.911

$*$ denotes the method is integrated into our unsupervised detection framework.

It is worth noting that grid searching for the anomaly threshold is omitted to reflect real-world unsupervised conditions, as referring to information from the test set is inappropriate. It will lead to an inflated performance even if the model is affected by anomalies while training. Instead, we employ the Peaks-Over-Threshold (POT) theorem [45] to predefine the threshold before testing. The parameters of POT are configured based on the anomaly proportion and the error distribution within the training set. For parameter q, it is set to $1e^{-4}$ in both datasets, while the parameter *low quantile* is set to 0.9978 and 0.9999 in SWaT-U and SMD-U, respectively. Thus, we can obtain τ by performing POT on the **training scores**.

Screening Methods. The unsupervised screening methods adopted in the experiments include IForest [6], COPOD [5], CBLOF [7], ECOD [46], HBOS [41], KMeans [27], and DBSCAN [47]. The first five methods are wrapped in the PyOD library [48], where we can directly conduct the screening process after setting the anomaly proportion. As for the two cluster-based methods, we retain the samples from three primary clusters based on the distance and size.

Methods for Comparison. To evaluate the performance of the proposed unsupervised detection framework, we compare it against three categories of methods:

- **Unsupervised anomaly detection methods.** We adopt eight frequently-used unsupervised anomaly detector: *a)* the statistical methods: KNN [49], LODA [50], LOF [25], and ABOD [51]; *b)* the deep learning models: LUNAR [17], DIF [52], ALAD [53], and InterFusion [18].
- **W/O screening.** We train the model directly without applying the screening process. As a result, there is a

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

TABLE V
THE RUNTIME EVALUATION OF UNSUPERVISED DETECTION

Dataset	Preprocessing	Getting Model Information	Screening	Training	Inference
SWaT-U	8s	547s	10s	336s	79s
SMD-U	2s	263s	2s	120s	82s

higher proportion of anomalies in the training samples, which leads to elevated anomaly thresholds since the parameters of POT are invariant.

- **Integrating other semi-supervised detection models.** We combine GDN [9] and SGAT-AE [13] into our framework in replace of STAMP. In general, any semi-supervised detection model that adopts graph learning is suitable for the framework.

2) Detection Performance: Table IV presents the best evaluation results of the unsupervised detection methods. The results lead to the following observations:

(i) For each semi-supervised model, the detection performance improves significantly when the basic screening process is applied. It demonstrates the effectiveness and robustness of the unsupervised framework. Specifically, the average F1-score increases by **18.0**%, **16.3**%, and **17.9**% of GDN, SGAT-AE, and STAMP in the two datasets compared to the non-screening situation.

(ii) Introducing feature selection based on model information significantly enhances detection results. Specifically, the Recall of the basic frameworks tends to be lower, indicating that the models are influenced by abnormal patterns during training. In contrast, the optimized frameworks are trained on samples that better represent normal patterns and align with the models' characteristics, resulting in improved Recall and F1-score performance.

(iii) IForest and COPOD produce the largest performance gain among the screening methods. AS for the three types of model-derived information, there is no significant performance gap across datasets. We can flexibly adopt any one in practice.

(iv) Overall, our framework that adopts IForest as the screening method outperforms other unspervised anomaly detection methods in both datasets. It significantly improves the F1-score by **12.1**% in SWaT-U compared to InterFusion, where the anomaly proportion of training samples is relatively high.

3) Runtime Evaluation: Table V shows the runtime of each segment in the unsupervised detection process with IForest as the screening method. It can be observed that the time consumption of preprocessing and screening is relatively low. Apart from the computational complexity of STAMP, another important reason is that the throughput of them involves all the input samples, while the other procedures are conducted in batches. To shorten the overall runtime, the practicable options include: *a)* choosing a larger batch size; *b)* reducing the training epochs; *c)* increasing the down-sampling ratio.

4) Case Study: To demonstrate the effectiveness of our framework in the real-world scenario, we study an anomaly case of Huawei's GaussDB (DWS) system. DWS is a distributed online database platform. Hundreds of KPIs are collected in DWS, including both hardware status and database indicators. Before deploying our framework on independent database clusters, we adopt IForest as the screening method and then train STAMP-G_1 in accordance with Section IV-A6.

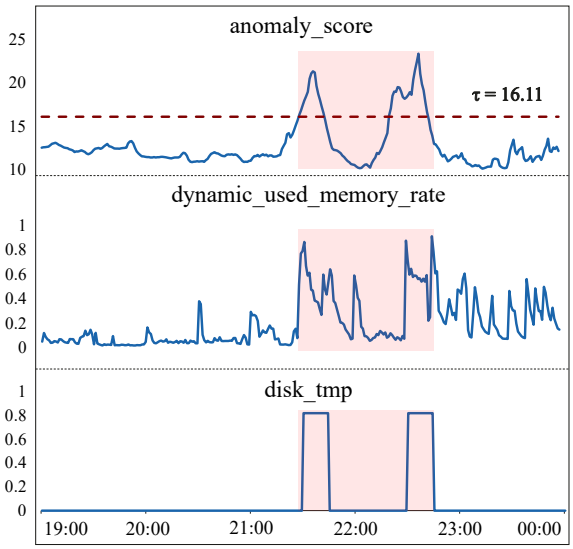


Fig. 6. A Real Anomaly Detection Case in Huawei DWS

The anomaly case is located in a certain cluster that contains three physical hosts. As shown in Figure 6, at around 22:00, the anomaly score inferred by the model exceeded the threshold τ for twice and then triggered the alarm. After examining the KPIs, the engineers discovered sharp rises of the indicator *dynamic_used_memory_rate* and *disk_tmp*. *dynamic_used_memory_rate* represents the memory usage for processing and executing SQLs, while *disk_tmp* represents the temporary storage size in the disk during the execution of SQLs. This phenomenon leads to the root case that users mistakenly submitted a large number of complex jobs. The cluster restored normal operation at around 23:00 after cleaning off the related jobs, thus avoided loss escalation.

VI. CONCLUSION

We design a multi-perspective spatio-temporal attention model STAMP for anomaly detection in multivariate time series, which focuses on spatio-temporal learning via attention networks to capture the complex correlations of KPIs and timestamps from both local and global perspectives. Building on STAMP, we further propose a novel unsupervised anomaly detection framework, in which traditional methods are adopted to generate a training set from unlabeled samples. Then the trained STAMP is employed to solve the detection task. Experimental results demonstrate that the proposed framework outperforms baselines in two unsupervised datasets.

ACKNOWLEDGEMENTS

This work was supported by NSFC Grant No. 62372194.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] R. Lu, E. Xu, Y. Zhang, F. Zhu, Z. Zhu, M. Wang, Z. Zhu, G. Xue, J. Shu, M. Li, and J. Wu, "Perseus: A fail-slow detection framework for cloud storage systems," in *FAST*, 2023, pp. 49–64.

[2] G. Zhang, C. Li, K. Zhou, L. Liu, C. Zhang, W. Chen, H. Fang, B. Cheng, J. Yang, and J. Xing, "Dbcatcher: A cloud database online anomaly detection system based on indicator correlation," in *ICDE*, 2023, pp. 1126–1139.

[3] M. Ma, S. Zhang, J. Chen, J. Xu, H. Li, Y. Lin, X. Nie, B. Zhou, Y. Wang, and D. Pei, "Jump-starting multivariate time series anomaly detection for online service systems," in *ATC*, 2021, pp. 413–426.

[4] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: A comprehensive evaluation," *PVLDB*, vol. 15, no. 9, pp. 1779–1797, 2022.

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

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

[7] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognit. Lett.*, pp. 1641–1650, 2003.

[8] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Söderström, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *KDD*, 2018, pp. 387–395.

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

[10] 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.

[11] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," in *IEEE. MDM*, 2018, pp. 125–134.

[12] Y. Zhang, Y. Chen, J. Wang, and Z. Pan, "Unsupervised deep anomaly detection for multi-sensor time-series signals," *TKDE*, vol. 35, no. 2, pp. 2118–2132, 2023.

[13] B. Zheng, L. Ming, K. Zeng, M. Zhou, X. Zhang, T. Ye, B. Yang, X. Zhou, and C. S. Jensen, "Adversarial graph neural network for multivariate time series anomaly detection," *TKDE*, pp. 1–14, 2024.

[14] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *ICLR*, 2022.

[15] S. Tuli, G. Casale, and N. R. Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *PVLDB*, vol. 15, no. 6, pp. 1201–1214, 2022.

[16] V. J. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artif. Intell. Rev.*, vol. 22, no. 2, pp. 85–126, 2004.

[17] A. Goodge, B. Hooi, S. Ng, and W. S. Ng, "LUNAR: unifying local outlier detection methods via graph neural networks," in *AAAI*, 2022, pp. 6737–6745.

[18] 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.

[19] T. Kieu, B. Yang, C. Guo, R. Cirstea, Y. Zhao, Y. Song, and C. S. Jensen, "Anomaly detection in time series with robust variational quasi-recurrent autoencoders," in *ICDE*, 2022, pp. 1342–1354.

[20] X. Chen, L. Deng, F. Huang, C. Zhang, Z. Zhang, Y. Zhao, and K. Zheng, "DAEMON: unsupervised anomaly detection and interpretation for multivariate time series," in *ICDE*, 2021, pp. 2225–2230.

[21] Y. Fang, J. Xie, Y. Zhao, L. Chen, Y. Gao, and K. Zheng, "Temporal-frequency masked autoencoders for time series anomaly detection," in *ICDE*, 2024, pp. 1228–1241.

[22] C. Lai, F. Sun, Z. Gao, J. H. Lang, and D. S. Boning, "Nominality score conditioned time series anomaly detection by point/sequential reconstruction," in *NIPS*, 2023.

[23] 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.

[24] 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 Things J.*, vol. 9, no. 12, pp. 9179–9189, 2022.

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

[26] A. Siffer, P. Fouque, A. Termier, and C. Largouët, "Anomaly detection in streams with extreme value theory," in *KDD*, 2017, pp. 1067–1075.

[27] T. Yairi, Y. Kato, and K. Hori, "Fault detection by mining association rules from house-keeping data," 2001. [Online]. Available: https://api.semanticscholar.org/CorpusID:17400289

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

[29] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *NIPS*, 2015, pp. 802–810.

[30] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *IJCAI*, 2019, pp. 1907–1913.

[31] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *KDD*, 2020, pp. 753–763.

[32] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[33] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *CoRR*, vol. abs/1803.01271, 2018.

[34] S. Guo, Y. Lin, H. Wan, X. Li, and G. Cong, "Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting," *TKDE*, vol. 34, no. 11, pp. 5415–5428, 2022.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[36] H. Zhao, Y. Wang, J. Duan, C. Huang, D. Cao, Y. Tong, B. Xu, J. Bai, J. Tong, and Q. Zhang, "Multivariate time-series anomaly detection via graph attention network," in *ICDM*, 2020, pp. 841–850.

[37] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.

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

[39] L. van der Maaten and G. E. Hinton, "Visualizing data using t-sne," *JMLR*, vol. 9, pp. 2579–2605, 2008.

[40] S. Han and S. S. Woo, "Learning sparse latent graph representations for anomaly detection in multivariate time series," in *KDD*, 2022, pp. 2977–2986.

[41] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," in *KI-2012: Poster and Demo Track*, 2012.

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

[43] 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.

[44] W. Zhang, C. Zhang, and F. Tsung, "GRELEN: multivariate time series anomaly detection from the perspective of graph relational learning," in *IJCAI*, 2022, pp. 2390–2397.

[45] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouët, "Anomaly detection in streams with extreme value theory," *KDD*, 2017.

[46] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. H. Chen, "ECOD: unsupervised outlier detection using empirical cumulative distribution functions," *TKDE*, vol. 35, no. 12, pp. 12 181–12 193, 2023.

[47] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996, pp. 226–231.

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

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

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

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

[52] H. Xu, G. Pang, Y. Wang, and Y. Wang, "Deep isolation forest for anomaly detection," *TKDE*, vol. 35, no. 12, pp. 12 591–12 604, 2023.

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

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on October 13,2026 at 12:15:58 UTC from IEEE Xplore. Restrictions apply.