SimAD: A Simple Dissimilarity-Based Approach for Time-Series Anomaly Detection

Zhijie Zhong, Zhiwen Yu, *Senior Member, IEEE*, Xing Xi, Yue Xu, Wenming Cao, *Member, IEEE*, Yiyuan Yang, Kaixiang Yang, *Member, IEEE*, and Jane You

***Abstract*—Despite the prevalence of reconstruction-based deep learning methods, time-series anomaly detection (TSAD) remains a tremendous challenge. Existing approaches often struggle with limited temporal contexts, insufficient representation of normal patterns, and flawed evaluation metrics, all of which hinder their effectiveness in detecting anomalous behavior. To address these issues, we introduce a simple dissimilarity-based approach for time-series anomaly detection (SimAD). Specifically, SimAD first incorporates a patching-based feature extractor capable of processing extended temporal windows and employs the EmbedPatch encoder to fully integrate normal behavioral patterns. Second, we design an innovative ContrastFusion module in SimAD, which strengthens the robustness of anomaly detection by highlighting the distributional differences between normal and abnormal data. Third, we introduce two robust enhanced evaluation metrics, unbiased affiliation (UAff) and normalized affiliation (NAff), designed to overcome the limitations of existing metrics by providing better distinctiveness and semantic clarity. The reliability of these two metrics has been demonstrated by both theoretical and experimental analyses. Experiments conducted on seven diverse time-series datasets clearly demonstrate SimAD's superior performance compared with state-of-the-art (SOTA) methods, achieving relative improvements of 19.85% on *F*1, 4.44% on Aff-F1, 77.79% on NAff-F1, and 9.69% on AUC on six multivariate datasets. Code and pretrained models are available at https://github.com/EmorZz1G/SimAD**

***Index Terms*—Anomaly detection, data mining, deep learning, evaluation metrics, outlier detection, time series.**

I. Introduction

TIME-SERIES anomaly detection (TSAD) is a critical component of time-series analysis, focused on accurately detecting abnormal patterns in time-series data and identifying their specific locations [1], [2]. TSAD methods utilize time-series data to identify anomalies in web traffic, which play a vital role in ensuring the stability, security, and efficient functioning of web services [3]. Unsupervised methods have garnered considerable attention in academic research, particularly for addressing this challenge through reconstruction-based approaches [4], [5], [6], [7], [8]. These methods assume that models are perfectly trained on normal data and assign higher anomaly scores to anomalous data during the testing phase. However, these methods have shown insufficient performance in practical applications. A thorough review of existing research [9], [10], [11], [12], coupled with comprehensive experiments (detailed in our analyses), has enabled us to identify several critical challenges in the field of TSAD.

Challenge 1: Many methods rely on the reconstruction assumption, which is inadequate for enhancing the detection performance and may not always hold true [9], [13]. Our experiments, including both sensitivity and ablation analyses, validate this limitation.

Challenge 2: Failure to adequately utilize extended time windows: The complexity of the attention mechanism has constrained previous methods, capping the window length at 200 or fewer [13], [14]. This limitation, in turn, prevents the capture of more informative data.

Challenge 3: Limited expressive power hinders the representation of normal features. On one hand, certain methods fail to effectively model either normal or abnormal data, or both. On the other hand, most models are constrained by a limited number of parameters, which restricts their expressive capacity.

To address the above challenges, we propose a simple dissimilarity-based approach for time-series anomaly detection (SimAD) method, in both univariate and multivariate settings. Specifically, to tackle Challenge 2, we design a feature extractor that can process longer time windows by splitting the sequence into multiple patches. This strategy enables SimAD to learn extended temporal receptive fields while using fewer parameters. To address Challenges 1 and 3, we design the EmbedPatch encoder and incorporate an enhanced attention mechanism for layerwise modeling of dissimilarities between normal and abnormal data features. Thus, the EmbedPatch encoder can learn discriminative representations of normal data more effectively. Essentially, the proposed EmbedPatch

Received 5 February 2025; revised 6 June 2025; accepted 14 July 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62476101, Grant 92467109, Grant U21A20478, and Grant 62306052; in part by the National Key Research and Development Program of China under Grant 2023YFA1011601; in part by the Major Key Project of the Peng Cheng Laboratory (PCL), China, under Grant PCL2023AS7-1; and in part by the Natural Science Foundation of Chongqing under Grant CSTB2023NSCQ-LZX0092. *(Corresponding author: Zhiwen Yu.)*

Zhijie Zhong and Zhiwen Yu are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, Guangdong 510650, China, and also with the Peng Cheng Laboratory, Shenzhen, Guangdong 518066, China (e-mail: zhwyu@scut.edu.cn).

Xing Xi, Yue Xu, and Kaixiang Yang are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, Guangdong 510650, China.

Wenming Cao is with the Department of Information and Computing Science, Chongqing Jiaotong University, Chongqing 400074, China.

Yiyuan Yang is with the Department of Computer Science, University of Oxford, OX1 2JD Oxford, U.K.

Jane You is with the Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hong Kong, China (e-mail: jane.you@polyu.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNNLS.2025.3590220, provided by the authors.

Digital Object Identifier 10.1109/TNNLS.2025.3590220

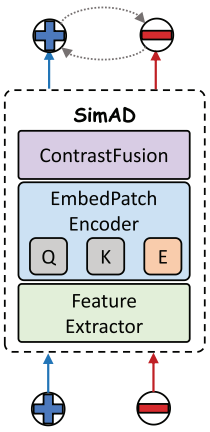


Fig. 1. Overview of SimAD.

encoder is a prototype, using V vectors to store prototypical features of normal data. The key differences between the EmbedPatch encoder and traditional prototypes can be highlighted in two aspects: 1) prototypes typically use the last layer to preserve features, while the EmbedPatch encoder, acting as the value matrices for the attention mechanism, not only preserves features at different layers but also serves as a querying repository for each level and 2) in traditional prototypes, query results are obtained by calculating the similarity between prototypes and features. In contrast, the EmbedPatch encoder generates query results by interacting the attention scores with the value matrices. With the EmbedPatch encoder, each layer of attention mechanism incorporates a series of embedded patches to learn features of that specific layer, giving it a stronger capability to learn richer and more distinctive representations, which are beneficial to performance improvement of anomaly detection. Finally, we introduce the ContrastFusion module to amplify the divergence of normal and abnormal data distributions for further accentuating their dissimilarity.

Furthermore, we have analyzed the shortcomings of existing evaluation metrics, including inflated metrics, low discriminative power, and insufficient semantic relevance. These issues have contributed to false perceptions of progress in TSAD [15], [16], [17]. To enable a fair comparison of performance, we propose two improved evaluation metrics: unbiased affiliation (UAff) and normalized affiliation (NAff). These metrics are designed to address the limitations of existing metrics by providing better distinctiveness and semantics. We have conducted analyses from both theoretical and experimental perspectives to establish the reliability of the newly proposed metrics. Extensive experiments indicate that the proposed approach outperforms the state-of-the-art (SOTA) methods in TSAD.

In summary, our article makes the following contributions.

1) We propose SimAD, a simple yet effective algorithm designed for TSAD that can handle extended time windows. The overall simplicity of SimAD is demonstrated in Fig. 1. SimAD is a straightforward framework based on dissimilarity measures, and its effectiveness has been validated through comprehensive evaluations from multiple perspectives, as illustrated in Fig. 2.
2) We introduce two improved TSAD evaluation metrics, UAff and NAff, which address challenges such as inaccurate assessments and lack of semantic clarity. The reliability of the proposed metrics is demonstrated through theoretical and experimental analyses. Furthermore, a comprehensive evaluation of the issues, limitations, and strengths of existing metrics is conducted.
3) Our algorithm outperforms significantly SOTA methods on real-world datasets, excelling in point-level evaluations, such as $F1$ and AUC, as well as sequence-level evaluations like Aff, UAff, and NAff (see Fig. 3). Moreover, we have released both our code and pretrained models.

II. Related Works

Unsupervised learning methods [4], [18] have garnered considerable attention due to the scarcity of labeled data for anomaly detection in time-series sequences [1], [9], [13], [19], [20], [21]. In past studies, anomaly detection algorithms have become one of the key data analysis tools and are widely used in fields like remote sensing and imaging [22], [23], [24], [25]. These methods can be categorized as follows (detailed discussion in Appendix A of the Supplementary Material): 1) algorithms based on classical machine learning [26], [27], [28] transform traditional machine learning approaches into deep networks, enhancing their ability to handle complex data; 2) reconstruction-based approaches [14], [19], [29] involve training models using normal data and leveraging reconstruction error as an anomaly score, attributing higher scores to anomalous data during testing; 3) prediction-based techniques, as demonstrated in [30], learn from historical data to predict future observations, considering prediction errors as the foundation for anomaly detection; and 4) generative adversarial learning-based methods [31], [32], [33] utilize generative models to learn the distribution of normal data and a discriminator network to detect anomalies.

Recently, some innovative algorithms have emerged in the field of anomaly detection, including: 1) Transformer-based approaches [13], [21], [34] leverage the power of Transformer, which has shown exceptional success in natural language processing tasks, and are increasingly being applied to anomaly detection; 2) contrastive learning-based methods [20], [35], [36], [37], [38] utilize contrastive learning to obtain robust representations, specifically tailored for anomaly detection; 3) diffusion-based methods [39], [40] model the propagation of anomalies in complex networks and time-series sequences by using the diffusion process; and 4) large language models (LLMs) [41] exploit cutting-edge models, like GPT-2, adapted specifically for anomaly detection tasks, capitalizing on the sophisticated architectures and knowledge representation capabilities.

These algorithms can be considered extensions or variations of the above categories, which incorporate advanced network structures and knowledge representation methods to enhance anomaly detection performance. However, existing methods have not effectively addressed the above three challenges. Most methods are constrained by a limited number of learnable parameters, which hinders them from capturing long-term dependencies in data. Moreover, some reconstruction-based methods solely focus on the task of reconstruction. The

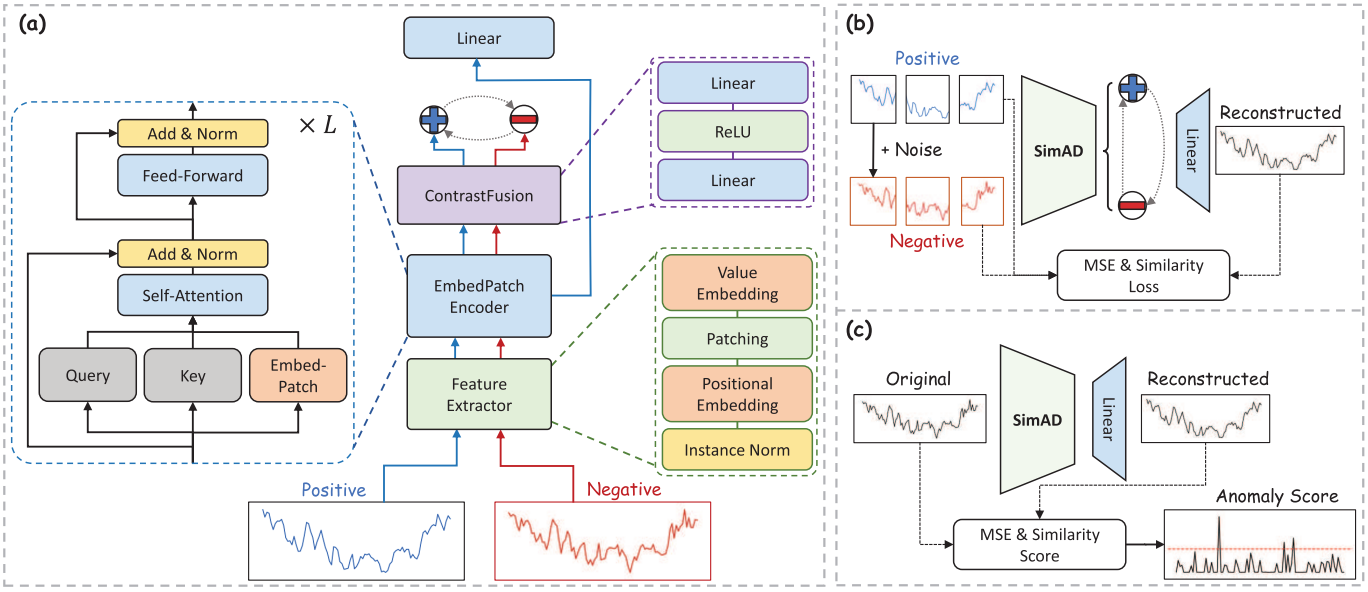


Fig. 2. Framework and workflow of SimAD. (a) SimAD framework. (b) Training phase. (c) Testing and detect anomalies.

differences between our dissimilarity-based approach and contrastive learning-based methods are in the framework (EmbedPatch and ContrastFusion), motivation (dissimilarity of perspectives [42]), and implementation, as discussed in Appendix I of the Supplementary Material.

III. Proposed Approach

A. Framework of SimAD

1) Overview: In Fig. 2(a), SimAD comprises a Feature Extractor, EmbedPatch encoder, and ContrastFusion module. In the context of temporal anomaly detection, the original time-series data is denoted as $\mathbf{X} \in \mathbb{R}^{T\times C}$, where T denotes the length of the time series and C denotes the number of channels. The goal of this task is to predict the label $\dagger \in \mathbb{R}^{T}$ for $\mathbf{X}$, where $y_i = 1$ indicates an anomaly at the ith time point and $y_i = 0$ otherwise. The input data $\mathbf{X}$ is first processed by the Feature Extractor to generate patch tokens $\mathbf{N}$. Next, at each layer of the EmbedPatch encoder, the attention mechanism combines the patch tokens from the previous layer with the current tokens to capture their interdependencies. We then utilize a linear layer to reconstruct the original features and compute anomaly scores. Finally, we design the ContrastFusion module to strengthen the distinction between normal and abnormal data distributions, thereby enhancing overall detection performance.

2) Feature Extractor: To extract unified sequence-level temporal features, we devised a patch-based feature extractor, which allows SimAD to process longer time windows and capture richer semantic information. Specifically, $\mathbf{X}$ is first fed to the feature extractor. To address distribution shifts [43], [44], we introduce instance normalization (IN) [43]. The improved positional embedding (PE) is then used to incorporate temporal positional encoding, enabling SimAD to learn the relationships between C channels actively. The improved PE applies sinusoidal positional encoding solely to the temporal positions, in contrast to the original PE [45], without the necessity of encoding the channel indices. This distinction arises from the differing implications of channels in time series and the word embeddings in the language model.

Then, for C channels, the temporal sequence is processed through two operations via Patching$(\cdot)$. First, the original input $\mathbf{X} \in \mathbb{R}^{T\times C}$ is segmented into M patches of length P, resulting in an intermediate representation $\mathbf{N}' \in \mathbb{R}^{M\times P\times C}$, where $T = M \times P$. Next, $\mathbf{N}'$ is reshaped to obtain $\mathbf{N}' \in \mathbb{R}^{M\times(P\cdot C)}$. In this article, we use the superscript "$\prime$" to denote intermediate process variables, such as $\mathbf{N}'$. Finally, value embedding (VE) is a simple linear transformation that maps all patches into a unified D-dimensional space, specifically $\mathbf{N} \in \mathbb{R}^{M\times D}$. It is defined as $\mathbf{N} = \mathrm{VE}(\mathbf{N}') = \mathrm{LayerNorm}(\mathrm{Linear}(\mathrm{LayerNorm}(\mathbf{N}')))$. Here, D is a predefined and represents the final dimension of the model. The term LayerNorm refers to layer normalization [45], which has been shown to be more suitable for sequential data. The term Linear refers to linear layer. The processing performed by the Feature Extractor can be described as follows:

$$\begin{aligned} \mathbf{X}' &= \mathrm{PE}\left(\mathrm{IN}\left(\mathbf{X}\right)\right), \\ \mathbf{N} &= \mathrm{VE}\left(\mathrm{Patching}\left(\mathbf{X}'\right)\right) \end{aligned} \tag{1}$$

where $\mathbf{N}$ refers to the *naive representation* and the original input of EmbedPatch encoder. With the designed feature extractor that maintains the framework's simplicity, SimAD can handle data with longer time windows, whose necessity will be demonstrated in Section IV-C.

3) EmbedPatch Encoder: To enhance SimAD's capability of modeling the dissimilarity between normal and abnormal samples, we made an improvement to the original attention mechanism. Inspired by Zhang et al. [9], You et al. [11], and Zhong et al. [12], this improvement involves incorporating the EmbedPatch $\mathbf{E}^{(i)} \in \mathbb{R}^{V\times D}$ with embedded queries into the EmbedPatch encoder backbone of SimAD, where V denotes the number of patch embeddings (EmbedPatch) and (i) for ith layer of encoder. Determining a proper V often requires

prior and a larger number of embeddings to accommodate different scenarios [46]. Therefore, a linear projection is stacked on EmbedPatch to obtain $\mathbf{E}^{(i),\prime} \in \mathbb{R}^{M^* \times D}$ for reducing the dimensionality, where $M^* \ll V$. Here, M^* represents the number of embeddings. It equals M from the Feature Extractor ($M^* = M$), but they serve different purposes in the context. Embedding (EmbedPatch) is learnable, whereas *naive representation* refers to the raw, high-dimensional features extracted directly from the original time series. Intuitively, each of the M patches finds a corresponding embedding from the V patch embeddings, resulting in M^* new embeddings for the original patches. This simple component allows SimAD to efficiently learn key information from normal samples without being overly dependent on prior knowledge.

To achieve this, we employ an improved multihead attention. For each head $k \in [1, 2, \ldots, U]$, we define the query matrix as $\mathbf{Q}_k^{(i)} = \mathbf{N}^{(i)}\mathbf{W}_k^Q$, the key matrix as $\mathbf{K}_k^{(i)} = \mathbf{N}^{(i)}\mathbf{W}_k^K$, and the value matrix as $\mathbf{V}_k^{(i)} = \mathbf{E}_k^{(i),\prime} = \mathbf{W}_k^V\mathbf{E}_k^{(i)}$, where $\mathbf{W}_k^Q, \mathbf{W}_k^K \in \mathbb{R}^{D\times d}$, $\mathbf{W}_k^V \in \mathbb{R}^{M^*\times V}$,$\mathbf{E}^{(i)} = [\mathbf{E}_1^{(i)}, \mathbf{E}_2^{(i)}, \ldots, \mathbf{E}_U^{(i)}], \mathbf{E}_k^{(i)} \in \mathbb{R}^{V\times d}$, and d is defined as $\lfloor \frac{D}{U} \rfloor$, which represents the dimension of the head. For the first layer of the EmbedPatch encoder, its input $\mathbf{N}^{(1)}$ is the output $\mathbf{N}$ from the Feature Extractor. Next, we define the attention calculation between different patches in each layer as follows:

$$\begin{aligned}\mathbf{Z}_k^{(i)} &= \text{Attention}\left(\mathbf{Q}_k^{(i)}, \mathbf{K}_k^{(i)}, \mathbf{V}_k^{(i)}\right) \\ &= \text{Softmax}\left(\frac{\left(\mathbf{Q}_k^{(i)}\mathbf{K}_k^{(i)\top}\right.}{\sqrt{d}}\right)\mathbf{V}_k^{(i)}. \end{aligned} \tag{2}$$

Note that $\mathbf{Q}_k^{(i)}$ and $\mathbf{K}_k^{(i)}$ here are generated by different parameters, so the attention scores are asymmetric. On the top of each layer, we utilize a linear layer to aggregate features $\mathbf{Z}_k^{(i)}$ from different heads, resulting in $\mathbf{Z}^{(i)} \in \mathbb{R}^{M\times D}$. Specifically, we first concatenate the features from U heads along the last dimension. For each $\mathbf{Z}_k^{(i)} \in \mathbb{R}^{M\times d}$, the concatenation yields $\mathbf{Z}^{(i),\prime}(\in \mathbb{R}^{M\times D}) = [\mathbf{Z}_1^{(i)}, \ldots, \mathbf{Z}_U^{(i)}]$. Subsequently, this concatenated feature is aggregated via a linear layer Linear, resulting in $\mathbf{Z}^{(i)} = \text{Linear}(\mathbf{Z}^{(i),\prime})$.

Subsequently, we proceed with the original "Add and Norm" operation to generate the input, $\mathbf{N}^{(i+1)}$, for the next layer

$$\begin{aligned}\mathbf{N}^{(i)\prime} &= \text{LayerNorm}\left(\mathbf{N}^{(i)} + \mathbf{Z}^{(i)}\right), \\ \mathbf{N}^{(i+1)} &= \text{LayerNorm}\left(\mathbf{N}^{(i)\prime} + \text{FFN}\left(\mathbf{N}^{(i)\prime}\right)\right)\end{aligned} \tag{3}$$

where FFN$(\cdot)$ denotes a two-layer feed-forward network (FFN), as in [45], $\mathbf{N}'$ denotes the intermediate variables, and (i) indicates the features at different layers.

Finally, a linear layer is used to restore $\hat{\mathbf{X}}$ from the last layer. As shown in Fig. 2(a), the final output of the EmbedPatch encoder is processed by a linear layer, which can be expressed as $\hat{\mathbf{X}} = \text{Linear}(\mathbf{N}^{(-1)})$. Here, $\mathbf{N}^{(-1)}$ represents the features from the last layer of the encoder.

To enable the model to detect anomalies in long-context time series, we need to optimize the parameters of SimAD's Feature Extractor and EmbedPatch encoder. During the training phase, the mean square error (mse) [13], [19] is used to measure the difference between $\mathbf{X}$ and $\hat{\mathbf{X}}$, and a patch-based similarity loss is employed to ensure continuity between patches. The mse term ensures that the model reconstructs the time series accurately locally. However, this point-to-point reconstruction alone cannot guide the model to detect anomalies over long time windows. The similarity term addresses this limitation. It first converts $\mathbf{X}$ into $\mathbf{N} \in \mathbb{R}^{M\times(P\cdot C)}$ using the Patching operation, and then calculates patch-based similarity along the last dimension. The final training loss is given by (4).

During the testing phase, SimAD utilizes $\mathcal{L}_{\text{rec}}$ to calculate anomaly scores and detect anomalies

$$\mathcal{L}_{\text{rec}} = \text{mse}\left(\hat{\mathbf{X}}, \mathbf{X}\right) + \left(1 - \text{Similarity}\left(\hat{\mathbf{X}}, \mathbf{X}\right)\right) \tag{4}$$

where cosine similarity is adopted as a similarity measure.

4) ContrastFusion Module: Although the EmbedPatch encoder enables SimAD to memorize normal samples, it does not substantially enhance the discrimination between normal and abnormal data. To mitigate this limitation, the ContrastFusion module is designed to employ contrastive learning to enlarge the feature distance between normal and abnormal data features.

To generate negative samples for contrastive learning without introducing prior, we adopt the simplest method by using Gaussian noise [1], [9]. Specifically, we use the equation $\mathbf{X}^- = \mathbf{X} + \alpha\cdot\mathbf{J}$, where $\mathbf{J}$ is sampled from a Gaussian distribution, and α controls the level of noise. This is a common time-series data augmentation method [1], [9]. From then on, we use variables with "–" to represent negative samples or features. Inspired by denoising autoencoders [47], we incorporate a denoising loss into the final objective function. Similarly, to guide the model to focus on contextual temporal associations during denoising, we added an extra similarity term

$$\mathcal{L}_{\text{denoise}} = \text{mse}\left(\hat{\mathbf{X}}^-, \mathbf{X}^+\right) + \left(1 - \text{Similarity}\left(\hat{\mathbf{X}}^-, \mathbf{X}^+\right)\right). \tag{5}$$

$\mathbf{X}^+$ indicates the branch opposite to the negative sample $\mathbf{X}^-$, emphasizing their opposition and being differentiated by colors/symbols in Fig. 2(a) for convenience and consistency. Keep in mind that $\mathbf{X}^+ = \mathbf{X}$ in this case. The two branches of ContrastFusion, which are optimized without regard to reconstruction, share structural similarities with previous image self-supervised techniques such as SimCLR [48] and SimSiam [49]. Therefore, our methodology can also be considered as a self-supervised (and/or unsupervised [4]) learning strategy.

To learn invariant features, we feed positive samples $\mathbf{X}^+$ and negative samples $\mathbf{X}^-$ into the Feature Extractor and EmbedPatch encoder, resulting in the acquisition of representations $\mathbf{N}^+$ and $\mathbf{N}^-$. Previous self-supervised works [1], [10], [48], [49] have shown that constructing different views through data augmentation can guide models to learn invariant features. For this purpose, we utilize a projection head $\mathscr{P}(\cdot)$, akin to prior contrastive learning methods, to facilitate the acquisition of meaningful and discriminative representations

$$\mathbf{H}^+ = \mathscr{P}\left(\mathbf{N}^+\right) = \text{Linear}\left(\text{ReLU}\left(\text{Linear}\left(\mathbf{N}^+\right)\right)\right) \tag{6}$$

where ReLU is the activation function. We derive low-dimensional representations $\mathbf{H}^+$ and $\mathbf{H}^-$ for $\mathbf{N}^+$ and $\mathbf{N}^-$,

respectively. To design an asymmetric feature contrastive loss, we utilize gradient stopping

$$\begin{aligned}\mathcal{L}_{\text{cont}} = &\operatorname{mse}\left(\mathbf{H}^{+}, \operatorname{StopGrad}\left(\mathbf{H}^{-}\right)\right) \\ &+ \left(1 - \operatorname{Similarity}\left(\mathbf{H}^{+}, \operatorname{StopGrad}\left(\mathbf{H}^{-}\right)\right)\right) \\ &+ \operatorname{mse}\left(\mathbf{H}^{-}, \operatorname{StopGrad}\left(\mathbf{H}^{+}\right)\right) \\ &+ \left(1 - \operatorname{Similarity}\left(\mathbf{H}^{-}, \operatorname{StopGrad}\left(\mathbf{H}^{+}\right)\right)\right). \end{aligned} \tag{7}$$

5) Joint Optimization for Training: As described above, SimAD aims to optimize the reconstruction of positive samples, denoise noisy samples, and amplify the feature differences between positive and negative samples [see Fig. 2(b)]. During the initial stages of training, the model focuses primarily on reconstruction and denoising. Since the contrastive loss may introduce additional training complexity, the weight of contrastive loss incrementally rises in the initial $N_{\text{warm-up}}$ iterations until it reaches the maximum value of $\beta_i = \min\left\{\frac{i+1}{N_{\text{warm-up}}}, \beta_{\max}\right\}$.

Algorithm 1 Pseudocode for Training SimAD

```
# J: i.i.d Gaussian Noise
# FE: Feature Extractor
# Enc: EmbedPatch Encoder
# CF: ContrastFusion
# SimAD: Combination of FE, Enc, and CF
# R: Rearrange Data
    for x in Data_Loader:
    x1, x2 = x, x + J
    x_out1, sim1 = SimAD(x1) # SimAD
    Processing
    x_out2, sim2 = SimAD(x2)
    # Reconstructs and Projects Features
    x_patch = R(x)
    rec_loss = loss_func(x_out1,x_patch)
    denoise_loss = loss_func(x_out2,x_patch)
    sim_loss = loss_func(sim1,sim2.detach())
    + loss_func(sim2,sim1.detach())
    loss = rec_loss + denoise_loss -
    sim_loss* warmup
    loss.backward()
    update(SimAD)

    # loss function
    def loss_func(s1,s2):
    return l2_loss(s1,s2) +
    (1-cos_loss(s1,s2)).mean()
```

The overall objective function combines the reconstruction loss, denoising loss, and contrastive loss that are given as follows:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{denoise}} - \beta\mathcal{L}_{\text{cont}}. \tag{8}$$

Algorithm 1 shows the training workflow of SimAD, highlighting the simplicity of our method in design and implementation.

B. Baselines

1) Inference: Anomaly Score: To calculate the anomaly score for query samples, we use the following equation:

$$\mathcal{L}_{\text{score}} = \operatorname{mse}\left(\hat{\mathbf{X}}, \mathbf{X}\right) + \left(1 - \operatorname{Similarity}\left(\hat{\mathbf{X}}, \mathbf{X}\right)\right). \tag{9}$$

TABLE I
DETAILS OF BENCHMARK DATASETS

Dataset	#Training	#Test (Labeled)	Dimension	Anomaly ratio (%)	Bias	Ideal Bias
MSL	58317	73729	55	10.5	51.34	50.55
SMAP	135183	427617	25	12.8	51.48	50.82
SWaT	99000	89984	26	12.2	52.94	50.74
WADI	241921	34561	123	5.74	54.81	50.16
PSM	132481	87841	25	27.8	53.17	53.86
Swan	60000	60000	38	32.6	54.83	55.31

Note that the first term (mse) has a length of T time points, while the second term (similarity) has a length of M patches. To match the length, each patch in the similarity term is replicated P times to reach T time points in the implementation.

As shown in Fig. 2(c), this process involves inputting time-series data into SimAD and performing feature extraction and representation to obtain the final feature $\mathbf{Z}^{(L)}$. This feature is then fed into a linear layer to reconstruct the data as closely as possible to the original: $\hat{X} = \text{Linear}(\mathbf{Z}^{(L)})$. We then compute the mse and cosine similarity between the reconstructed and original data. In inference stage, the ContrastFusion module is not used, eliminating the need to generate negative samples or compute similarities between positive and negative samples.

C. Improving Affiliation Metrics

Recent studies [15], [16], [17] have highlighted the limitations of conventional metrics used in TSAD. The affiliation metric [50] is a parameter-free method that has shown promising performance. However, based on experiments and theoretical analysis (see Appendix C of the Supplementary Material), the affiliation precision (Aff-Pre) tends to approach 0.5, while the affiliation recall (Aff-Rec) tends to approach 0.99 when confronting with random anomaly scores. This leads to an affiliation $F1$ score (Aff-F1) of approximately 0.7, suggesting that the metric's discriminatory capability is insufficient. To address this limitation, we introduce UAff and NAff metrics. Both of them exhibit superior discriminatory ability and provide a more accurate reflection of an algorithm's performance.

Experiments in Table I indicate that the Aff-Pre varies across different datasets but consistently tends to approach 0.5, which can be interpreted as a dataset-specific bias, denoted by $\textbf{Aff-Pre}_{bias}$. Meanwhile, there has been insufficient encouragement for high-precision models, while the penalties for low-precision models are inadequate. In this context, we design unbiased Aff-Pre (UAff-Pre) and unbiased Aff-F1 (UAff-F1), both of which can alleviate the dataset-specific bias and offer a more equitable assessment of the algorithm's precision performance

$$\textbf{UAff-Pre} = \frac{\textbf{Aff-Pre} - \textbf{Aff-Pre}_{bias}}{1 - \textbf{Aff-Pre}_{bias}} \tag{10}$$

$$\textbf{UAff-F1} = \frac{2 \cdot |\textbf{UAff-Pre}| \cdot \textbf{Aff-Rec}}{|\textbf{UAff-Pre}| + \textbf{Aff-Rec}} \cdot (-1)^{|\textbf{UAff-Pre}<0|} \tag{11}$$

where $\textbf{UAff-F1} < 0$ when $\textbf{UAff-Pre} < 0$. Although **UAff-Pre** denotes an improvement over its original version and surpasses **UAff-F1**, it introduces additional parameters, deviating from its parameter-free nature. Since the majority of real-world

datasets exhibit **Aff-Pre**$_{bias} \leq 0.55$, we further propose a modified version of NAff metrics. The computation follows a similar process, with the exception that we set **Aff-Pre**$_{bias} = \beta$ for constant while maintaining other aspects unchanged. In other words, **NAff-Pre** can be calculated as $\frac{\textbf{Aff-Pre}-\beta}{1-\beta}$, and **NAff-F1** is calculated similar to **UAff-F1**. The mathematical and experimental analyses of the metrics are in Appendix C.

IV. Experiments

We evaluate the effectiveness of our proposed SimAD by conducting extensive comparison experiments against SOTA competing methods on six real-world multivariable datasets: MSL, SMAP, PSM, SWaT, WADI, and Swan. In addition, we utilize a univariate dataset UCR [51]. The details of the above six multivariable datasets are in Table I.

We employ the $F1$ without point adjustment, Aff-F1, and the improved UAff-F1 and NAff-F1 for performance evaluation. We exclude the point-adjusted $F1$ score due to its acknowledged potential for false improvements. It is noted that although the classical $F1$ score is not optimal for time-series data, it can still be employed for evaluation purposes [15], as detailed in Appendix D-A of the Supplementary Material. Furthermore, we present an analysis of VUS [52] scores (see Table S4 in the Supplementary Material) for all the datasets. In TSAD, the $F1$ score evaluates model performance by balancing precision (Prec) and recall (Rec). It is defined as

$$\text{Prec} = \frac{\text{TP}}{\text{TP}+\text{FP}}, \quad \text{Rec} = \frac{\text{TP}}{\text{TP}+\text{FN}},$$
$$F1 = 2 \times \frac{\text{Prec} \times \text{Rec}}{\text{Prec}+\text{Rec}}. \tag{12}$$

Here, TP, FP, and FN represent true positives, false positives, and false negatives. The $F1$ score ranges from 0 to 1, with higher values indicating better performance. Due to space limitations and the need for conciseness, the calculation of Aff-F1 and VUS refers to the original papers [50], [52]. We discuss UAff and NAff in Section C, comparing them with other metrics, and summarize the limitations and advantages of existing metrics in detail.

A. Baselines

Our model is compared against 20 baselines, which include machine learning-based models like LOF, isolation forest (IForest), and PCA; deep learning-enhanced models like Deep SVDD [26] and deep IForest (Deep IF) [27]; reconstruction-based models like USAD [19], TCN-ED [14], AdaMemBLS [1], and NPSR [53]; prediction-based methods like TimesNet [30] and M2N2 [54]; Transformer-based models such as Anomaly Transformer (AnomTrans) [13] and TranAD [21]; contrastive learning-based models like COUTA [35], NCAD [36], and DCdetector [20]; diffusion-based model D3R [39]; and LLM-based GPT2-Adapter [41].

B. Quantified Comparisons

Table II shows comparisons between SimAD and baselines on six real-world datasets. It is obvious that SimAD performs superior to other models. Compared with the best baseline on datasets SWaT, WADI, and Swan, SimAD exhibits an improvements of **5.15%** (76.88%→82.03%), **13.31%** (50.67%→63.98%), and **15.24%** (55.99%→71.23%) in F1 score. In contrast, the NAff-F1 exhibits absolute improvements of **11.09%** (55.73%→66.82%), **30.87%** (45.39%→76.26%), and **19.05%** (33.99%→53.04%), respectively. In addition, SimAD has shown slight superiority over other models on datasets MSL and SMAP. On these datasets, SimAD achieved an absolute improvement of 1.26% and 1.00% in Aff-F1, respectively, along with higher scores in other metrics as well.

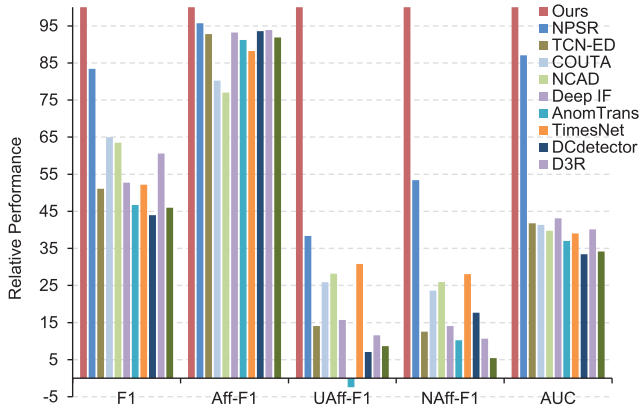


Fig. 3. Comparison analysis of relative performance.

Table III summarizes the evaluation metric scores of models across all datasets, clearly indicating that SimAD achieved improvements of **9.07%** on F1, **3.18%** on Aff-F1, **16.63%** on UAff-F1, and **20.55%** on NAff-F1, and **6.83%** AUC, compared with the SOTA baseline. In contrast, although models such as TimesNet and D3R exhibit good performance on specific evaluation metrics (UAff-F1 and Aff-F1), their performances on other metrics (Aff-F1 and NAff-F1) are inferior to those of SimAD, and even lower than those of random algorithms. Fig. 3 shows the advantages of our model, compared with other baseline models across multiple evaluation metrics. This further confirms the superior generalization capability of SimAD, whereas other algorithms may display larger performance fluctuations due to an excessive focus on specific dataset characteristics. (see a more detailed comparison in Appendix E-D of the Supplementary Material.)

In addition, we evaluate the effectiveness of our model in handling univariate dataset UCR in Table IV, which includes Avg.+ (calculated using only positive values) and Avg. scores. From this table, SimAD demonstrates superior performance. Specifically, SimAD achieves an improvement of **13.33%** (1.66%→14.99%) in **Avg.+ F1** and **13.38%** (1.61%→14.99%) in *Avg. F1*, compared with the best baseline. Furthermore, SimAD achieves an improvement of **25.75%** (9.41%→35.16%) in *Avg.+ UAff-F1* and **17.52%** (2.01%→19.53%) in *Avg. UAff-F1*, compared with the best baseline.

C. Sensitivity Analysis

We performed sensitivity analysis on hyperparameters of SimAD, which include the window size and the number of patch embeddings. Fig. 4 illustrates the influence of different

TABLE II

COMPARISON RESULTS. ALL RESULTS ARE IN %, THE BEST IN BOLD, AND THE SECOND IN UNDERLINED

Datasets	MSL							SMAP							SWaT						
Methods	F1	AUC	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1	F1	AUC	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1	F1	AUC	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1
Random	18.60	50.01	51.34	99.99	67.84	0.00	5.20	22.06	50.18	51.48	**100.0**	67.97	0.00	5.74	21.06	50.10	52.94	99.99	69.23	0.00	11.09
LOF	19.36	55.75	53.57	88.90	66.86	8.74	13.22	23.64	62.39	46.00	81.26	58.74	-19.84	-14.58	58.22	84.59	99.87	2.70	5.25	5.25	5.25
IForest	10.90	59.09	55.09	97.63	70.44	14.31	18.45	23.78	61.15	39.45	70.23	50.52	-36.64	-32.45	38.52	83.80	**100.0**	2.71	5.27	5.27	5.27
PCA	10.33	53.25	55.50	97.44	70.72	15.73	19.77	22.96	58.70	39.67	70.31	50.72	-36.15	-31.93	26.05	81.85	99.96	2.70	5.25	5.25	5.25
Deep SVDD	24.30	61.80	52.67	99.93	68.98	10.53	10.13	21.28	61.14	44.98	88.45	59.64	-17.60	-18.02	22.57	86.81	53.58	99.99	69.77	13.75	13.37
USAD	22.50	57.12	51.90	97.29	67.69	7.73	7.31	16.57	62.79	39.56	69.48	50.41	-31.78	-32.12	21.70	88.66	53.00	100.0	69.28	11.71	11.31
TCN-ED	19.66	51.14	51.44	99.99	67.93	6.04	5.61	22.90	58.91	51.39	99.98	67.89	5.86	5.42	21.65	89.23	52.98	100.0	69.26	11.64	11.24
COUTA	20.88	55.59	51.24	99.65	67.68	5.29	4.85	22.69	58.74	51.48	**100.0**	67.97	6.18	5.74	47.65	75.54	79.98	33.20	46.92	42.76	42.73
TranAD	22.81	50.03	52.82	99.34	68.97	11.07	10.66	22.74	59.74	39.41	70.30	50.51	-32.21	-32.56	25.50	88.90	55.65	99.66	71.42	20.64	20.30
NCAD	22.05	60.20	56.38	83.30	67.25	**22.46**	22.14	23.09	53.45	51.88	99.99	68.32	7.67	7.25	68.72	82.92	65.01	85.58	73.89	44.64	44.46
Deep IF	19.11	55.94	51.45	**100.0**	67.94	6.08	5.64	29.14	60.09	53.75	98.67	69.59	14.31	13.93	21.65	89.52	52.98	**100.0**	69.26	11.64	11.24
AnomTrans	18.39	52.61	50.21	99.83	66.81	-4.53	0.83	16.06	52.18	55.84	99.11	71.44	16.49	20.90	23.44	80.80	50.24	99.49	66.77	-10.83	0.96
TimesNet	21.24	57.18	51.12	99.95	67.64	4.81	4.36	24.37	53.73	49.50	99.86	66.19	-1.52	-1.99	21.66	73.24	57.84	93.77	71.55	27.16	26.86
DCdetector	11.62	50.31	51.96	97.77	67.85	2.52	7.52	26.56	58.50	55.55	99.78	71.37	15.48	19.97	23.24	52.78	52.63	98.30	68.56	-1.28	10.00
D3R	23.98	**63.00**	53.61	99.97	69.79	8.92	7.65	22.71	54.56	51.43	**100.0**	67.92	-0.21	0.07	45.89	79.95	61.47	78.52	68.96	29.47	36.02
GPT2-Adapter	13.72	52.03	53.31	97.01	68.81	7.80	6.75	24.12	55.48	53.28	99.84	69.48	7.18	3.67	22.30	52.30	52.51	98.13	68.41	-1.79	0.07
NPSR	23.72	61.16	52.05	99.81	68.42	2.89	7.88	22.68	61.26	51.46	**100.0**	67.95	-0.06	5.68	76.88	90.18	71.21	81.16	75.86	52.54	55.73
M2N2	21.76	59.15	51.38	**100.00**	67.88	0.17	5.36	22.68	54.05	51.46	**100.00**	67.95	0.52	5.69	38.50	67.79	59.15	96.83	73.44	27.54	30.77
AdaMemBLS	19.11	51.78	**57.07**	95.37	71.41	20.99	**24.64**	26.93	53.66	53.30	99.80	69.49	7.24	12.38	74.10	81.61	53.57	**100.00**	69.76	2.65	13.32
Ours	**30.02**	62.70	56.30	99.76	**71.98**	18.50	22.37	**29.39**	**65.46**	**56.84**	99.82	**72.44**	**19.91**	**24.07**	**82.03**	**90.31**	78.46	80.88	**79.65**	**64.93**	**66.82**
Datasets	**WADI**							**PSM**							**NIPS-TS-Swan**						
Methods	**F1**	**AUC**	**Aff-Pre**	**Aff-Rec**	**Aff-F1**	**UAff-F1**	**NAff-F1**	**F1**	**AUC**	**Aff-Pre**	**Aff-Rec**	**Aff-F1**	**UAff-F1**	**NAff-F1**	**F1**	**AUC**	**Aff-Pre**	**Aff-Rec**	**Aff-F1**	**UAff-F1**	**NAff-F1**
Random	10.79	50.03	54.81	99.95	70.79	0.00	17.54	40.93	50.16	53.17	99.95	69.41	0.00	11.91	46.25	50.13	54.83	96.82	70.01	0.00	17.56
LOF	10.58	65.44	51.76	25.72	34.36	-10.67	6.20	23.91	**81.28**	77.47	52.13	62.32	52.01	53.50	21.37	77.89	82.01	11.14	19.62	18.81	18.98
IForest	30.88	74.84	57.68	82.53	67.90	11.82	25.91	38.52	75.12	72.97	46.46	56.77	44.27	46.20	46.18	85.12	**94.11**	16.31	27.81	27.47	27.54
PCA	34.30	50.53	48.10	52.07	50.01	-23.09	-7.08	43.20	72.06	77.86	79.65	**78.74**	**63.44**	**65.57**	49.18	72.88	92.96	16.03	27.35	26.95	27.02
Deep SVDD	7.34	59.47	50.06	55.79	52.77	0.72	0.24	44.46	78.09	53.92	99.98	70.06	14.93	14.56	55.99	75.32	56.84	89.93	69.66	24.07	23.76
USAD	10.86	53.61	54.92	**100.0**	70.90	18.27	17.91	47.90	64.53	55.30	98.71	70.88	19.48	19.14	52.63	66.76	55.26	68.62	61.22	18.57	18.25
TCN-ED	10.86	53.01	54.92	**100.0**	70.90	18.27	17.91	43.46	63.97	53.16	**100.0**	69.42	12.29	11.89	49.22	71.37	55.10	99.92	71.03	18.87	18.52
COUTA	26.92	50.52	**99.18**	23.08	37.44	37.38	37.38	48.21	69.36	57.98	99.30	73.21	27.80	27.50	47.19	71.38	45.07	86.50	66.54	15.24	14.87
TranAD	11.78	52.60	55.91	91.96	69.54	21.27	20.94	43.20	65.22	60.88	92.09	73.30	35.44	35.19	51.93	70.00	57.86	90.18	70.49	27.07	26.78
NCAD	10.87	62.84	54.99	**100.0**	70.96	18.49	18.14	46.61	61.76	77.79	53.08	63.10	54.35	54.30	37.57	52.24	54.32	94.06	68.87	16.20	15.83
Deep IF	10.86	51.43	54.92	**100.0**	70.90	18.27	17.91	43.47	69.06	53.15	**100.0**	69.41	12.24	11.84	49.19	73.30	55.10	99.98	71.05	18.86	18.51
AnomTrans	11.24	52.98	51.43	95.15	66.77	-13.84	5.56	39.81	52.18	52.78	96.07	68.13	-1.65	10.51	44.86	54.62	55.21	92.36	69.11	1.68	18.73
TimesNet	17.65	65.08	65.25	39.30	49.05	34.45	34.34	43.60	58.74	**77.84**	67.23	72.15	60.97	60.91	43.16	53.62	60.75	81.29	69.53	34.23	33.99
DCdetector	11.33	50.12	59.72	94.64	73.23	19.51	32.26	22.72	50.38	53.05	94.98	68.08	-0.49	11.47	48.83	50.35	55.08	99.82	70.99	1.11	18.45
D3R	12.86	51.39	56.47	99.98	72.17	7.09	2.46	44.24	60.64	53.84	**100.0**	69.99	2.82	3.93	49.45	62.54	57.70	96.06	72.10	11.93	9.98
GPT2-Adapter	10.68	51.21	55.62	98.29	71.04	3.55	-0.27	35.24	51.23	54.33	95.16	69.17	4.84	2.18	45.23	58.77	61.17	69.55	65.09	23.36	18.28
NPSR	50.67	80.19	65.15	90.45	75.74	36.53	45.39	51.02	70.67	54.71	99.37	70.56	6.29	17.14	49.30	62.61	55.13	99.99	71.07	1.32	18.61
M2N2	13.55	55.68	54.92	**100.00**	70.90	13.73	17.93	47.71	54.91	55.43	94.36	69.84	15.45	19.48	38.40	70.12	60.94	93.68	**73.84**	32.60	35.47
AdaMemBLS	29.09	53.85	54.95	**100.00**	70.93	0.63	18.01	46.69	63.26	58.22	92.65	71.51	19.32	27.92	49.35	51.27	55.14	**100.00**	71.09	1.38	18.66
Ours	**63.98**	**87.97**	82.36	92.81	**87.27**	**73.59**	**76.26**	**52.07**	71.99	62.46	91.76	74.33	32.63	39.20	**71.23**	**85.17**	76.17	53.75	63.03	**50.29**	**53.04**

TABLE III

AVERAGE PERFORMANCE AND RANKING OF DIFFERENT ALGORITHMS. RK. DENOTES THE RANKING

Avg.	F1	RK.	Aff-F1	RK.	UAff-F1	RK.	NAff-F1	RK.	AUC	RK.	V_PR	RK.	Avg.	RK.
Random	26.62	16	69.21	9	0.00	19	11.51	15	50.10	20	27.71	18		16.17
LOF	26.18	17	41.19	20	9.05	13	13.76	10	71.23	3	39.75	9		12.00
IForest	31.46	7	46.45	19	11.09	11	15.15	9	73.19	2	43.39	7		9.17
PCA	31.00	8	47.13	18	8.69	14	13.10	13	64.88	8	37.84	13		12.33
Deep SVDD	29.32	11	65.15	15	7.73	15	7.34	18	70.44	5	50.13	3		11.17
USAD	28.69	13	65.06	16	7.33	17	6.97	19	65.58	7	51.95	2		12.33
TCN-ED	27.96	15	69.41	8	12.16	10	11.76	14	64.61	9	46.07	6		10.33
COUTA	35.59	4	59.96	17	22.44	4	22.18	5	63.52	11	36.75	14		9.17
TranAD	29.66	10	67.37	13	13.88	8	13.55	11	64.41	10	47.66	5		9.50
NCAD	34.82	5	68.73	10	27.30	2	27.02	2	62.24	12	38.21	12		7.17
Deep IF	28.90	12	69.69	7	13.57	9	13.18	12	66.56	6	39.69	10		9.33
AnomTrans	25.63	18	68.17	12	-2.11	20	9.58	17	57.56	17	26.49	19		17.17
TimesNet	28.61	14	66.02	14	26.68	3	26.41	3	60.27	15	36.47	15		10.67
DCdetector	24.05	20	70.01	6	6.14	18	16.61	8	52.07	19	28.33	17		14.67
D3R	33.19	6	70.16	5	10.00	12	10.02	16	62.01	13	39.59	11		10.50
GPT2-Adapter	25.22	19	68.67	11	7.49	16	5.11	20	53.51	18	28.33	16		16.67
NPSR	45.71	2	71.60	2	16.58	7	25.07	4	71.01	4	49.29	4		3.83
M2N2	30.43	9	70.64	4	17.58	6	19.12	7	60.28	14	21.55	20		10.00
AdaMemBLS	40.88	3	70.70	3	18.58	5	19.16	6	59.24	16	42.17	8		6.83
Ours	54.79	1	74.78	1	43.31	1	46.96	1	77.27	1	52.96	1		1.00

TABLE IV

COMPARISON OF DIFFERENT MODELS ON DATASET UCR

Datasets		UCR					
Metrics	Methods	F1	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1
Avg.+	AnomTrans	1.17	50.66	98.94	66.88	9.41	10.88
	DCdetector	1.66	50.90	99.96	67.41	6.71	5.52
	Ours	**14.99**	57.83	99.91	**72.52**	**35.16**	**34.38**
Avg.	AnomTrans	1.15	50.39	97.73	66.55	1.30	1.23
	DCdetector	1.61	50.63	83.30	67.06	2.01	2.64
	Ours	**14.99**	57.83	97.47	**72.52**	**19.53**	**19.46**

window sizes on the model's performance on the WADI dataset. As the window size increases, the model's loss exhibits a decreasing trend, and its accuracy consistently improves. A larger window size incorporates more temporal and contextual information, enabling the model to capture patterns and trends and improve the detection performance. It is worth noting that when the window size increases from 1024 to 2048, the model's loss further decreases while the improvement in the $F1$ score is limited. This suggests that larger window sizes also increase the optimization difficulty.

Fig. 5(a) depicts impacts of the number of patch embeddings on performance on the WADI dataset. When the number of patch embeddings is set to 0 (i.e., the original attention mechanism), the model's performance is significantly lower. This demonstrates the effectiveness of EmbedPatch in enhancing the memory of normal samples. As the number of patch embeddings increases, both the model's Rec Loss and $F1$ score exhibit fluctuating changes. It can be inferred that the model achieves the optimal performance when the

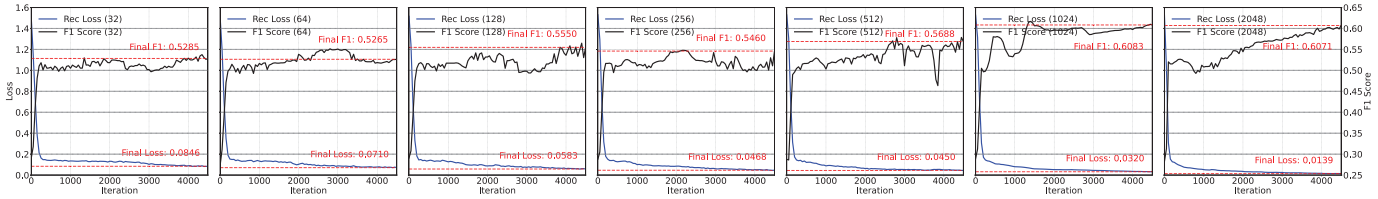


Fig. 4. Impact of window size on $F1$-score and the reconstruction loss when the window size varies from 32 to 2048.

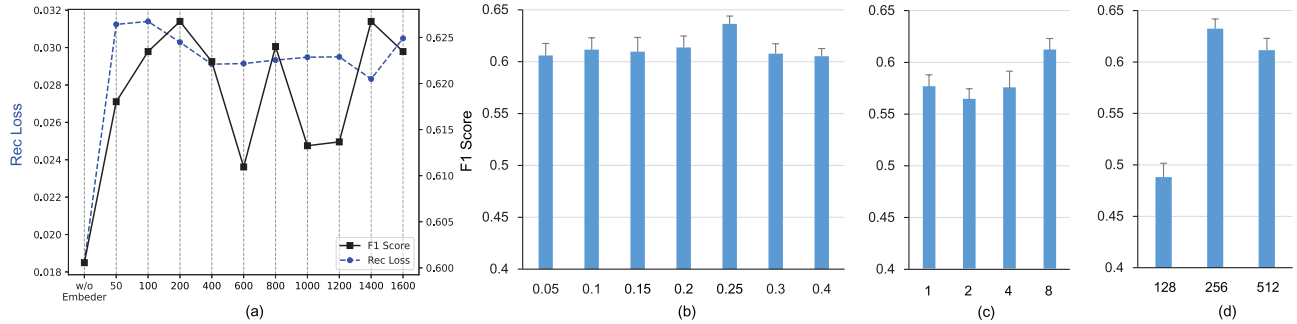


Fig. 5. Different numbers of embeddings and other hyperparameters. The error bar represents the standard deviation. (a) Number of patch embedding. (b) Noise level. (c) Number of layers. (d) Dimension of model.

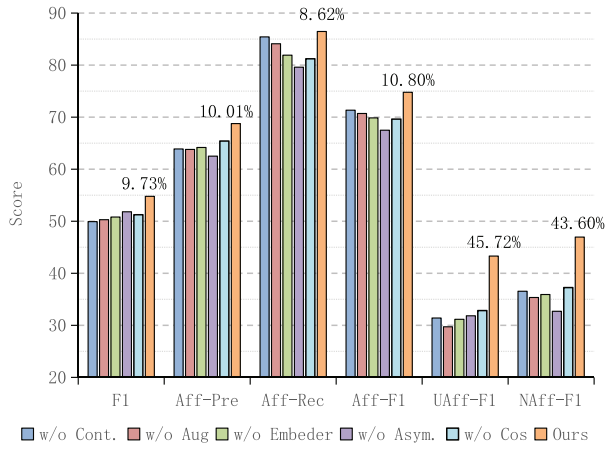


Fig. 6. Ablation study. The data labeling denotes an upper bound on the performance improvement of our model.

number of patch embeddings is between 100 and 200. From Fig. 5(b)–(d), it is evident that increasing the number of layers and setting an appropriate level of noise and dimension of features can result in improved model performance on the WADI dataset. However, a notable decline in performance is observed when the dimension is set to 128. This indicates that models with smaller dimensions face challenges in learning complex features.

D. Ablation Analysis

To validate the effectiveness of our model, we conducted ablation studies on ContrastFusion, data augmentation, EmbedPatch, and asymmetric optimization in Fig. 6. Here we provide descriptions about the model's variants.

1) *w/o Contrastive (Cont.):* It indicates the removal of contrastive learning in the ContrastFusion. By removing this, $\mathcal{L}_{\text{cont}}$ [see (7)] no longer works, and the similarity relationships between positive and negative samples will not be learned.
2) *w/o Aug:* It signifies negative sample generation without denoising learning, where $\mathcal{L}_{\text{denoise}}$ [see (5)] does not work. It is noteworthy that despite generating negative samples, it is still possible to conduct contrastive learning between positive and negative samples, i.e., $\mathcal{L}_{\text{cont}}$ [see (7)] remains active.
3) *w/o Embeder:* When the Embeder, i.e., $\mathbf{E}$ is removed, the architecture of SimAD regresses to a Transformer. Originally, the value matrix in SimAD was $\mathbf{V} = \mathbf{W}^V\mathbf{E}$. After regressing to a Transformer, the generation of $\mathbf{V}$ aligns with matrices $\mathbf{Q}$ and $\mathbf{K}$, i.e., $\mathbf{V} = \mathbf{N}\mathbf{W}^V$. For simplicity, $\mathbf{W}^V$ represents learnable parameters, and $\mathbf{N}$ is the preinput. When EmbedPatch is removed, SimAD's backbone reverts to a Transformer, where self-attention is uniform across Q, K, and V.
4) *w/o Asymmetric (Asym.):* Our asymmetric optimization relies on gradient stop. Removing the asymmetric optimization eliminates the StopGrad from the equation. As both mse and Similarity losses are symmetric, after removing this operation, (7) regresses to a symmetric optimization method.
5) *w/o Cos:* It indicates the removal the cosine terms from (4), (5), and (7) while keeping the mse loss.

The detailed ablation studies are in Table V. The observations from ablation studies are summarized as follows.

1) After removing the *w/o Contrastive*, i.e., the removal of positive and negative sample contrastive learning, the $F1$ score of the model decreased by 7.88%, indicating that ContrastFusion is important in learning distinct features to differentiate between normal and abnormal samples.

TABLE V
FULL ABLATION RESULTS

Datasets	MSL						SMAP						SWaT					
Variations	F1	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1	F1	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1	F1	Aff-Pre	Aff-Rec	Aff-F1	UAff-F1	NAff-F1
w/o Cont.	23.90	54.13	99.08	70.01	10.84	15.23	22.56	51.94	99.86	68.34	1.89	7.47	80.31	77.66	76.52	77.09	62.29	64.21
w/o Aug	29.07	54.64	98.88	70.39	12.72	16.98	23.78	52.66	98.11	68.53	4.75	10.09	79.14	**80.60**	64.14	71.43	61.34	62.63
w/o Embeder	27.20	54.12	98.85	69.94	10.80	15.20	17.89	50.04	99.64	66.62	-5.76	0.15	80.31	77.66	76.52	77.09	62.29	64.21
w/o Asym.	28.23	54.13	98.94	69.98	10.86	15.25	22.39	51.46	99.77	67.90	-0.05	5.69	82.01	77.35	83.37	**80.25**	63.95	66.06
w/o Cos	28.50	53.95	99.10	69.87	10.20	14.64	23.32	52.24	**99.91**	68.61	3.11	8.59	77.77	75.02	**83.53**	79.05	60.09	62.59
Ours	**30.02**	**56.30**	**99.76**	**71.98**	**18.50**	**22.37**	**29.39**	**56.84**	99.82	**72.44**	**19.91**	**24.07**	**82.03**	78.46	80.88	79.65	**64.93**	**66.82**
Datasets	**WADI**						**PSM**						**NIPS-TS-Swan**					
Variations	**F1**	**Aff-Pre**	**Aff-Rec**	**Aff-F1**	**UAff-F1**	**NAff-F1**	**F1**	**Aff-Pre**	**Aff-Rec**	**Aff-F1**	**UAff-F1**	**NAff-F1**	**F1**	**Aff-Pre**	**Aff-Rec**	**Aff-F1**	**UAff-F1**	**NAff-F1**
w/o Cont.	62.68	70.55	84.04	76.71	49.26	55.20	45.78	58.83	96.05	72.96	21.47	29.83	64.34	70.24	57.00	62.93	42.69	47.34
w/o Aug	63.19	75.92	81.53	78.63	59.40	63.38	41.79	56.43	**96.15**	71.12	12.98	22.67	64.83	62.55	**65.81**	64.14	27.13	36.34
w/o Embeder	61.36	70.73	59.61	64.70	44.29	48.91	49.78	61.47	94.80	**74.58**	29.86	36.93	68.32	71.04	62.03	**66.23**	45.47	50.14
w/o Asym.	56.89	63.33	63.98	63.65	56.26	37.64	**52.26**	59.93	91.27	72.35	24.93	32.62	69.14	68.85	40.28	50.82	35.06	38.95
w/o Cos	63.13	72.66	74.79	73.71	51.69	56.43	44.00	59.83	86.97	70.89	24.46	32.07	70.79	**78.76**	42.99	55.62	47.47	49.21
Ours	**63.98**	**82.36**	**92.81**	**87.27**	**73.59**	**76.26**	52.07	**62.46**	91.76	74.33	**32.63**	**39.20**	**71.23**	76.17	53.75	63.03	**50.29**	**53.04**

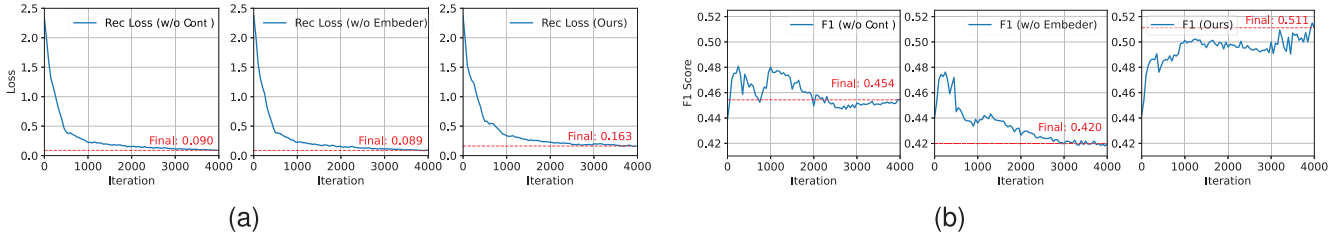


Fig. 7. (a) Loss and (b) $F1$-score variations on dataset PSM.

TABLE VI
KL DIVERGENCE BETWEEN NORMAL AND ANOMALOUS FEATURES BOTH BEFORE AND AFTER PROJECTION

Datasets	SWaT		WADI	
Variations	Before Proj.	After Proj.	Before Proj.	After Proj.
w/o Cont.	7.34E-01	1.66E-02	4.95E-05	2.85E-03
w/o Embeder	4.34E-01	2.04E-02	3.25E-03	1.59E-01
Ours	4.25E-01	5.04E-01	3.00E-02	9.89E-01

2) By disabling *w/o Aug*, i.e., the removal of data augmentation, the detection performance of SimAD shows a decrease of 43.69% in UAff-F1, demonstrating its ability to improve the model's generalization.
3) For *w/o Embeder*, the EmbedPatch in the attention mechanism degrades to a simple value matrix. The ablation of this module has the most significant impact on the model's $F1$ score, which indicates that incorporating EmbedPatch into the attention mechanism is crucial to learning appropriate representations of temporal data.
4) *w/o Asymmetric* indicates the model without asymmetric optimization. The ablation of this module results in a decrease in all metrics, particularly a significant drop of 43.69% in NAff-F1. This suggests that asymmetric optimization helps the model focus on abnormal samples.
5) Experimental results of *w/o Cos* demonstrate that upon removing the cosine similarity penalty, both the predictive continuity and detection performance of SimAD decline.

The comprehensive results of ablation studies, along with the setting of the experiment, are in Table V.

To verify the effectiveness of the EmbedPatch encoder and ContrastFusion modules in our proposed SimAD framework for addressing the first Challenge outlined in the introduction, i.e., enhancing the dissimilarity between features of both normal and abnormal data, we extracted feature representations before and after the ContrastFusion projection head. Next, we conducted a multisampling to calculate the KL divergence between normal and abnormal features at both stages. The results are in Table VI. From this table, it can be observed that removing the ContrastFusion module (w/o Cont.) leads to an increase in KL divergence between the feature distributions of normal and abnormal data on the SWaT and WADI datasets, indicating that the representations become more distinct. Moreover, when both components are integrated into our model, the separation between normal and abnormal feature distributions is maximized. This demonstrates that SimAD effectively overcomes the limitations of previous approaches.

Moreover, Fig. 7 provides an illustration of the training on dataset PSM. It is evident that the *w/o Embedder* module achieves the lowest loss but with the lowest $F1$ score. This can be attributed to that EmbedPatch contributes SimAD to memorizing normal samples and enhancing generalization. By capturing the characteristics of normal data, EmbedPatch improves the effectiveness of anomaly detection by leveraging distinct differences between normal and abnormal samples, as demonstrated in Fig. 8(b) and Fig. S4 in Appendix E of the Supplementary Material. The *w/o Contrastive* module also exhibits a low loss but suffers from negative optimization, resulting in a relatively low $F1$ score. In contrast, SimAD, despite having a higher loss, achieves the highest $F1$ score. This demonstrates that optimizing solely for reconstruction is

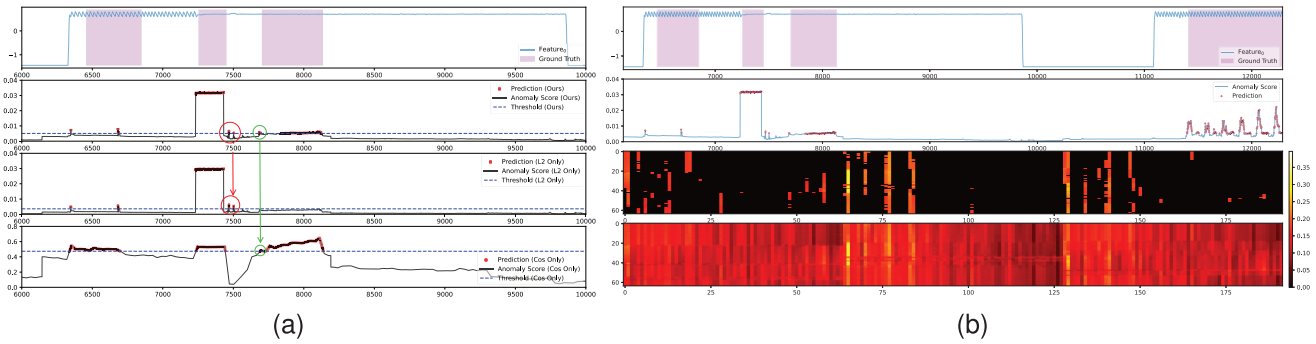


Fig. 8. Real-world case study on dataset SWaT. (a) Detection performances in the real world. (b) Similarity scores of query and embedding.

inadequate, and SimAD effectively addresses the challenges outlined in Section I.

E. Visualization Analysis

1) Real-World Cases Analysis: The performance of SimAD on dataset SWaT is in Fig. 8(a) and Fig. S2 (in Appendix E-B of the Supplementary Material). The first row of images showcases the original data and the ground truth labels. Its y-axis represents the magnitude of the first channel in the time series. The second row of images denotes the predicted results, which effectively detects most anomalies by assigning significantly higher scores to abnormal data. The y-axis represents the magnitude of the anomaly score. The third and fourth rows of images illustrate the results obtained using $L2$ loss and patch-based cosine similarity loss as anomaly scores, respectively.

It is noticed that the former focuses more on short-duration anomalies, while the latter can capture long-duration anomalies. In Appendix E-B of the Supplementary Material, Fig. S3 shows the detection results of AnomTrans and NPSR. Our method exhibits superior performance compared with AnomTrans, as the latter only detects a few anomalies and encounters difficulties in accurately detecting segment anomalies. The case study highlights the effectiveness and rationale behind our anomaly score design.

2) Explanation of Patch Embedding: Fig. 8(b) shows the detection performance of SimAD, where the fourth row denotes the similarity between the query and the embedding of the last layer, while the vertical axes of third and fourth regions illustrate correlations between the query embedding and the learned embedding of value matrices. In the third and fourth regions of the right subfigure, the horizontal axis denotes the ordinal number of the embedding. The colors in the heatmap indicate the strength of these correlations, with precise values corresponding to the color bar for reference. Specifically, given a time window of 2048, we divide it into 64 nonoverlapping patches, each with a length of 32. There are 64 learned embeddings in the last layer of our EmbedPatch encoder. In the following, we consider the 64 nonoverlapping patches from the input window as the query embedding. Correspondingly, the 64 learned embedding produced by the final layer of the EmbedPatch encoder serves as the value matrices. Next, we compute the pairwise correlations between 64 nonoverlapping patches and 64 learned embedding representations, resulting into an attention feature map. This feature map is depicted in the fourth region of the right subfigure. In contrast, the third region of the right subfigure highlights the top five most relevant query embeddings, selected from the 64 patches (i.e., segments of the 2048-length time window), for each learned embedding. In the third region, patches that exhibit weak relevance to all embeddings are shown in black. Conversely, patches with stronger relevance to certain embeddings are displayed in vibrant colors, highlighting their higher correlations relative to other patches. It is important to note that the attention feature map denotes the correlations between the query ($\mathbf{Q}$) vectors and the value ($\mathbf{V}$) vectors in the last layer of the EmbedPatch encoder, i.e., the relationships between the input patches from the time series and the learned embedding, rather than the correlations between the query ($\mathbf{Q}$) vectors and the key ($\mathbf{K}$) vectors. Combined with Fig. S4 in Appendix E-B of the Supplementary Material, it can be observed that in the lower layers, the embedding establishes relationships with a majority of queries. However, only the normal data can establish relationships with EmbedPatch in the higher layers. This means that in the shallow layers, SimAD learns general representations. While in the higher layers, SimAD learns features associated with normal data, which results in less similarity between abnormal data and EmbedPatch. Consequently, the patch embeddings enhance the detection performance of SimAD.

V. Conclusion

This article introduces a SimAD. The proposed framework is distinguished by its simplicity and versatility, allowing for the use of longer time windows. Experimental results demonstrate the superiority of our method compared with existing approaches. In addition, we present two improved evaluation metrics, UAff and NAff, which effectively assess the algorithm's performance and overcome many shortcomings of previous metrics. In future work, we plan to extend this detection scheme to various other scenarios and aim to develop a unified model for anomaly detection.

References

[1] Z. Zhong, Z. Yu, Z. Fan, C. L. Philip Chen, and K. Yang, "Adaptive memory broad learning system for unsupervised time series anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 5, pp. 8331–8345, May 2025.

[2] Z. He et al., "A spatiotemporal deep learning approach for unsupervised anomaly detection in cloud systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 1705–1719, Apr. 2023.

[3] Y. Zheng, "Correlation-aware spatial–temporal graph learning for multivariate time-series anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 11802–11816, Sep. 2024.

[4] Z. Yu, Z. Zhong, K. Yang, W. Cao, and C. L. P. Chen, "Broad learning autoencoder with graph structure for data clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 1, pp. 49–61, Jan. 2024.

[5] B. Kim, M. A. Alawami, E. Kim, S. Oh, J. Park, and H. Kim, "A comparative study of time series anomaly detection models for industrial control systems," *Sensors*, vol. 23, no. 3, p. 1310, Jan. 2023.

[6] K. Zhu, P. Song, and C. Zhao, "Fuzzy state-driven cross-time spatial dependence learning for multivariate time-series anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 3, pp. 4532–4544, Mar. 2025.

[7] Z. Yang et al., "A multi-scale mask convolution-based blind-spot network for hyperspectral anomaly detection," *Remote Sens.*, vol. 16, no. 16, p. 3036, Aug. 2024.

[8] R. Zhao and L. Zhang, "GSEAD: Graphical scoring estimation for hyperspectral anomaly detection," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 2, pp. 725–739, Feb. 2017.

[9] Y. Zhang, J. Wang, Y. Chen, H. Yu, and T. Qin, "Adaptive memory networks with self-supervised learning for unsupervised anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 1–13, Dec. 2023.

[10] Z. Liu, Y. Zhou, Y. Xu, and Z. Wang, "SimpleNet: A simple network for image anomaly detection and localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 20402–20411.

[11] Z. You et al., "A unified model for multi-class anomaly detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 4571–4584.

[12] Z. Zhong, K. Yang, Z. Yu, Y. Shi, and C. L. Philip Chen, "Towards efficient anomaly detection using memory broad learning system," in *Proc. 9th Int. Conf. Control Sci. Syst. Eng. (ICCSSE)*, Jun. 2023, pp. 252–257.

[13] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–20.

[14] A. Garg, W. Zhang, J. Samaran, R. Savitha, and C.-S. Foo, "An evaluation of anomaly detection and diagnosis in multivariate time series," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2508–2517, Jun. 2022.

[15] S. Kim, K. Choi, H.-S. Choi, B. Lee, and S. Yoon, "Towards a rigorous evaluation of time-series anomaly detection," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 7194–7201.

[16] K. Doshi, S. Abudalou, and Y. Yilmaz, "Reward once, penalize once: Rectifying time series anomaly detection," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2022, pp. 1–8.

[17] R. Wu and E. J. Keogh, "Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress (extended abstract)," in *Proc. IEEE 38th Int. Conf. Data Eng. (ICDE)*, May 2022, pp. 1479–1480.

[18] R. Zhao, B. Du, L. Zhang, and L. Zhang, "A robust background regression based score estimation algorithm for hyperspectral anomaly detection," *ISPRS J. Photogramm. Remote Sens.*, vol. 122, pp. 126–144, Dec. 2016.

[19] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: Unsupervised anomaly detection on multivariate time series," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2020, pp. 3395–3404.

[20] Y. Yang, C. Zhang, T. Zhou, Q. Wen, and L. Sun, "DCdetector: Dual attention contrastive representation learning for time series anomaly detection," in *Proc. 29th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2023, pp. 3033–3045.

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

[22] R. Zhao, B. Du, and L. Zhang, "Hyperspectral anomaly detection via a sparsity score estimation framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 6, pp. 3208–3222, Jun. 2017.

[23] R. Zhao, B. Du, L. Zhang, and L. Zhang, "Beyond background feature extraction: An anomaly detection algorithm inspired by slowly varying signal analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1757–1774, Mar. 2016.

[24] R. Zhao, B. Du, and L. Zhang, "A robust nonlinear hyperspectral anomaly detection approach," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 4, pp. 1227–1234, Apr. 2014.

[25] R. Zhao, Z. Yang, X. Meng, and F. Shao, "A novel fully convolutional auto-encoder based on dual clustering and latent feature adversarial consistency for hyperspectral anomaly detection," *Remote Sens.*, vol. 16, no. 4, p. 717, Feb. 2024.

[26] L. Ruff et al., "Deep one-class classification," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4393–4402.

[27] H. Xu, G. Pang, Y. Wang, and Y. Wang, "Deep isolation forest for anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12591–12604, Dec. 2023.

[28] B. Du, R. Zhao, L. Zhang, and L. Zhang, "A spectral–spatial based local summation anomaly detection method for hyperspectral images," *Signal Process.*, vol. 124, pp. 115–131, Jul. 2016.

[29] Z. Tian, M. Zhuo, L. Liu, J. Chen, and S. Zhou, "Anomaly detection using spatial and temporal information in multivariate time series," *Sci. Rep.*, vol. 13, no. 1, p. 4400, Mar. 2023.

[30] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-variation modeling for general time series analysis," in *Proc. 11th Int. Conf. Learn. Represent.*, 2022, pp. 1–23.

[31] F. Zeng, M. Chen, C. Qian, Y. Wang, Y. Zhou, and W. Tang, "Multivariate time series anomaly detection with adversarial transformer architecture in the Internet of Things," *Future Gener. Comput. Syst.*, vol. 144, pp. 244–255, Jul. 2023.

[32] A.-H. Shin, S. T. Kim, and G.-M. Park, "Time series anomaly detection using transformer-based GAN with two-step masking," *IEEE Access*, vol. 11, pp. 74035–74047, 2023.

[33] B. Du, X. Sun, J. Ye, K. Cheng, J. Wang, and L. Sun, "GAN-based anomaly detection for multivariate time series using polluted training set," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12208–12219, Dec. 2023.

[34] J. Kim, H. Kang, and P. Kang, "Time-series anomaly detection with stacked transformer representations and 1D convolutional network," *Eng. Appl. Artif. Intell.*, vol. 120, Apr. 2023, Art. no. 105964.

[35] H. Xu, Y. Wang, S. Jian, Q. Liao, Y. Wang, and G. Pang, "Calibrated one-class classification for unsupervised time series anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 11, pp. 5723–5736, Nov. 2024.

[36] C. U. Carmona, F.-X. Aubet, V. Flunkert, and J. Gasthaus, "Neural contextual anomaly detection for time series," in *Proc. Thirty-First Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 2843–2851.

[37] H. Zhou, K. Yu, X. Zhang, G. Wu, and A. Yazidi, "Contrastive autoencoder for anomaly detection in multivariate time series," *Inf. Sci.*, vol. 610, pp. 266–280, Sep. 2022.

[38] T. Pranavan, T. Sim, A. Ambikapathi, and S. Ramasamy, "Contrastive predictive coding for anomaly detection in multi-variate time series data," 2022, *arXiv:2202.03639*.

[39] C. Wang et al., "Drift doesn't matter: Dynamic decomposition with diffusion reconstruction for unstable multivariate time series anomaly detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–17.

[40] Y. Chen et al., "ImDiffusion: Imputed diffusion models for multivariate time series anomaly detection," *Proc. VLDB Endowment*, vol. 17, no. 3, pp. 359–372, Nov. 2023.

[41] T. Zhou, P. Niu, X. Wang, L. Sun, and R. Jin, "One fits all: Power general time series analysis by pretrained LM," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2023, pp. 43322–43355.

[42] S. Mauceri, J. Sweeney, M. Nicolau, and J. McDermott, "Dissimilarity-preserving representation learning for one-class time series classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 10, pp. 13951–13962, Oct. 2024.

[43] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–25.

[44] C. Chang, W.-Y. Wang, W.-C. Peng, and T.-F. Chen, "LLM4TS: Aligning pre-trained LLMs as data-efficient time-series forecasters," 2023, *arXiv:2308.08469*.

[45] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998–6008.

[46] M. Jin et al., "Time-LLM: Time series forecasting by reprogramming large language models," 2023, *arXiv:2310.01728*.

[47] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 1096–1103.

[48] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[49] X. Chen and K. He, "Exploring simple Siamese representation learning," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, Jun. 2021, pp. 15750–15758.
[50] A. Huet, J. M. Navarro, and D. Rossi, "Local evaluation of time series anomaly detection algorithms," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 635–645.
[51] H. A. Dau et al., "The UCR time series archive," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1293–1305, Nov. 2019.
[52] 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," *Proc. VLDB Endowment*, vol. 15, no. 11, pp. 2774–2787, 2022.
[53] C.-Y. Lai, F.-K. Sun, Z. Gao, J. H. Lang, and D. S. Boning, "Nominality score conditioned time series anomaly detection by point/sequential reconstruction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 76637–76655.
[54] D. Kim, S. Park, and J. Choo, "When model meets new normals: Test-time adaptation for unsupervised time-series anomaly detection," in *Proc. AAAI Conf. Artif. Intell.*, Mar. 2023, pp. 13113–13121.

Zhijie Zhong received the B.S. degree from Harbin Engineering University, Harbin, China, in 2022. He is currently pursuing the Ph.D. degree with South China University of Technology, Guangzhou, China.

His research interests include data mining, machine learning, time-series analyses, anomaly detection, and large language models (LLMs).

Zhiwen Yu (Senior Member, IEEE) received the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2008.

He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. He has authored or co-authored more than 200 refereed journal articles and international conference papers, including more than 70 articles in journals of IEEE TRANSACTIONS. His Google citation is more than 10 000 and H-index is 44.

Dr. Yu is a Senior Member of ACM and a member of the Council of China Computer Federation (CCF). He is an Associate Editor of IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS.

Xing Xi received the B.S. degree from Guangdong Baiyun University, Guangzhou, China, in 2020, and the M.A. degree from Guangdong University of Technology, Guangzhou, in 2023. He is currently pursuing the D.E. degree with South China University of Technology, Guangzhou.

His research interests include object detection, open world, and vocabulary object detection.

Yue Xu received the B.S. degree from South China University of Technology, Guangzhou, China, in 2025, where he is currently pursuing the M.A. degree.

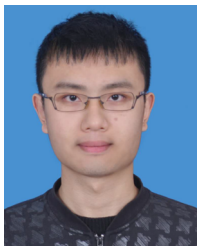
His research interests include adversarial attack, continual learning, and anomaly detection.

Wenming Cao (Member, IEEE) received M.Sc. degree from the School of Automation, Huazhong University of Science and Technology (HUST), Wuhan, China, in 2015, and the Ph.D. degree from the Department of Computer Science, City University of Hong Kong, Hong Kong, in 2019.

He was a Post-Doctoral Fellow with The University of Hong Kong, Hong Kong, from 2019 to 2021. He is currently an Associate Professor with the Department of Information and Computing Science, Chongqing Jiaotong University, Chongqing, China. His research interests include data mining and machine learning.

Yiyuan Yang is currently pursuing the D.Phil. degree with the Department of Computer Science, University of Oxford, Oxford, U.K.

He previously studied at the Department of Automation, Tsinghua University, Beijing, China, and interned at the Alibaba DAMO Academy, Hangzhou, China, and Noah's Ark Laboratory, Huawei, Shenzhen, China. He focuses on the field of intelligent sensing systems, time series, spatiotemporal data mining, anomaly detection, and generative models.



Kaixiang Yang (Member, IEEE) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2012, the M.S. degree from Harbin Institute of Technology, Harbin, China, in 2015, and the Ph.D. degree from the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China, in 2020.

He was a Research Engineer with the 7th Research Institute, China Electronics Technology Group Corporation, Guangzhou, from 2015 to 2017, and a Post-Doctoral Researcher with Zhejiang University, Hangzhou, China, from 2020 to 2021. He is currently with the School of Computer Science and Engineering, South China University of Technology. His research interests include pattern recognition, machine learning, and industrial data intelligence.

Jane You received the B.Eng. degree in electronics engineering from Xi'an Jiaotong University, Xi'an, China, in 1986, and the Ph.D. degree in computer science from La Trobe University, Melbourne, VIC, Australia, in 1992.

She was a Lecturer with the University of South Australia, Adelaide, SA, Australia, and a Senior Lecturer with Griffith University, Nathan, QLD, Australia, from 1993 to 2002. She is currently a Full Professor with The Hong Kong Polytechnic University, Hong Kong. Her current research interests include image processing, pattern recognition, medical imaging, biometrics computing, multimedia systems, and data mining.