Expert Systems With Applications 245 (2024) 123010

Contents lists available at ScienceDirect

Expert Systems With Applications

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

# An adversarial contrastive autoencoder for robust multivariate time series anomaly detection

Jiahao Yu [a], Xin Gao [a,*], Feng Zhai [b,c], Baofeng Li [c], Bing Xue [a], Shiyuan Fu [a], Lingli Chen [a], Zhihang Meng [a]

[a] *School of Artificial intelligence, Beijing University of Posts and Telecommunications, Beijing, 100876, China*
[b] *School of Electrical and Information Engineering, Tianjin University, Tianjin, 300072, China*
[c] *China Electric Power Research Institute Company Limited, Beijing, 100192, China*

ARTICLE INFO

*Keywords:*
Multivariate time series
Anomaly detection
Autoencoder
Contrastive learning
Adversarial training
Feature combination and decomposition

ABSTRACT

Multivariate time series (MTS), whose patterns change dynamically, often have complex temporal and dimensional dependence. Most existing reconstruction-based MTS anomaly detection methods only learn the point-wise information while ignoring the overall trend of time series, resulting in their incompetence in extracting high-level semantic information. Although a few contrastive learning-based approaches have been proposed recently to solve this problem, they forcibly increase the difference between the features of normal data, leading to the loss of useful information. This paper proposes an adversarial contrastive autoencoder (ACAE) for MTS anomaly detection. ACAE conducts feature combination and decomposition as the contrastive learning proxy task, which introduces adversarial training to learn the transformation-invariant representation of data, achieving a robust representation of MTS. Firstly, ACAE constructs positive and negative sample pairs through the multi-scale timestamp mask and random sampling. Secondly, the features of the original samples are combined with those of the positive and negative samples to generate the positive and negative composite features. Finally, ACAE trains the encoder and discriminator to decompose the negative composite features cooperatively to decrease the similarity between the features of negative pairs. In contrast, it adversarially decomposes the positive composite features to increase the similarity between the features of positive pairs. Experimental results show that ACAE outperforms 14 state-of-the-art baselines on five real-world datasets from different fields.

## 1. Introduction

Anomaly detection is a basic data mining task aiming to find samples that are notably different from most other data (Chandola, Banerjee, & Kumar, 2009). It has been developed in various applications, including network intrusion detection, wire fraud detection, and medical diagnosis (Boukerche, Zheng, & Alfandi, 2020; Mokoena, Celik, & Marivate, 2022). With the progress of technology, mass data are generated daily in medical, financial, industrial, and other research fields. Some observation results are recorded orderly and related in time, forming the time series data (Blázquez-García, Conde, Mori, & Lozano, 2021). Time series anomaly detection is one of the main tasks of time series analysis (Blázquez-García et al., 2021; Choi, Yi, Park, & Yoon, 2021) that focuses on extracting valuable information from time series data. Since time series data are usually related to key processes, monitoring them to detect anomalies is significant and has gained widespread interest over the past few years (Zhou, Yu, Zhang, Wu, & Yazidi, 2022).

Some studies concentrate on identifying anomalies in univariate time series (Muhr & Affenzeller, 2022; Paparrizos et al., 2022), but in many real-world applications, time series data are multivariate. For example, the equipment in industrial systems is usually monitored by multiple sensors, each of which measures a different index (Cook, Mısırlı, & Fan, 2019). Multivariate time series (MTS) can better represent the global state of the system, and it is more challenging to detect anomalies in MTS, so it has received more attention recently (Li & Jung, 2022). Firstly, MTS data usually have complex temporal and dimensional dependence. Meanwhile, the normal pattern of MTS may change dynamically over time. Traditional anomaly detection algorithms cannot model the complicated structure of MTS (Breunig, Kriegel, Ng, & Sander, 2000; Gao, Yu, et al., 2022; Liu, Ting, & Zhou, 2012).

[*] Corresponding author.
*E-mail addresses:* yujiahao@bupt.edu.cn (J. Yu), xlhhh74@bupt.edu.cn (X. Gao), zhaifeng@epri.sgcc.com.cn (F. Zhai), libaofeng@epri.sgcc.com.cn (B. Li), xuebing@bupt.edu.cn (B. Xue), ShiyuanFu@bupt.edu.cn (S. Fu), cll0922@bupt.edu.cn (L. Chen), mengzhihang@bupt.edu.cn (Z. Meng).

https://doi.org/10.1016/j.eswa.2023.123010
Received 8 March 2023; Received in revised form 13 October 2023; Accepted 20 December 2023
Available online 29 December 2023

Deep learning is widely used in MTS anomaly detection to attempt to capture the patterns of MTS data (Li & Jung, 2022; Schmidl, Wenig, & Papenbrock, 2022). Secondly, finding and labeling anomalies in time series data is time-consuming and expensive in practice, so some supervised methods (Ismail Fawaz et al., 2020; Ji et al., 2022) are limited in practical applications due to the absence of available prior information.

The prediction-based (Chen, Chen, Zhang, Yuan, & Cheng, 2021; Deng & Hooi, 2021; Wang, Du, Lu, Duan, & Wu, 2022; Zhou, Song, & Qian, 2021) and reconstruction-based (Davari et al., 2022; Gao, Qiu, et al., 2022; Kieu et al., 2022; Tayeh, Aburakhia, Myers, & Shami, 2022) methods are the primary unsupervised MTS anomaly detection methods. They both train models to capture the normal pattern of MTS, assuming that the training data only comprise normal time series. The prediction-based technique forecasts the future observation values according to the input time series. After that, it identifies anomalies according to the discrepancy between the model's predicted and actual values. The reconstruction-based method trains the autoencoder to reconstruct the normal data and uses the reconstruction errors of the test data to detect anomalies. MTS data generally have complex pattern changes, and some variables and external factors may not be recorded. It is challenging to predict MTS, resulting in the poor performance of the prediction-based method in practical applications. Existing reconstruction-based MTS anomaly detection methods provide state-of-the-art results on numerous challenging real-world data.

However, the reconstruction-based method is insufficient for learning the high-level semantic information of MTS data. It cannot construct an accurate outline of the normal pattern of MTS, limiting its performance to improve further. For the existing reconstruction-based methods, the difficulties of MTS anomaly detection lie in the following aspects:

(1) The reconstruction-based method uses mean square error (MSE) to constrain the reconstruction, focusing on decreasing the low-level pixel errors rather than learning the high-level semantic information of data. Since time series has complex temporal dependence and its normal pattern may change dynamically over time, it is challenging for the single point-wise context information to describe its normal pattern well (Zhou et al., 2022). Fig. 1 visualizes several fragments of PSM (Abdulaal, Liu, & Lancewicki, 2021), SMD (Su et al., 2019), and SWAT (Mathur & Tippenhauer, 2016) that are the real-world datasets commonly employed for evaluating the performance of MTS anomaly detection benchmarks. There are various anomalies with complex patterns in real-world datasets, including the subsequence anomaly, context anomaly, and other complex and difficult-to-explain patterns of anomalies (Shaukat et al., 2021). Compared to global anomalies different from all normal data, it is more difficult for existing reconstruction-based methods to detect contextual anomalies shown in Fig. 1(g) - Fig. 1(l). The values of the contextual anomalies are within the range of normal data but do not match the context, making it difficult to detect these anomalies based on point-wise contextual information.

(2) It is difficult for the model to focus on the patterns of the normal data. First, the size of the information bottleneck of the autoencoder in the reconstruction-based method is difficult to determine. The too-small setting of the bottleneck will lead to the insufficient learning ability of the model. In contrast, the too-large setting will make the noise well reconstructed and overfitting. At the same time, noise in the data is more likely to obtain large reconstruction errors. Furthermore, anomalies could contaminate the training data under the unsupervised setting. Fig. 2 visualizes a segment of real-world training data in PSM, which contains significant noise and is contaminated by the anomaly (Wu & Keogh, 2021). The existing reconstruction-based methods are susceptible to interference by the noise and anomalies in the training data when extracting the latent representation of MTS (Gao, Qiu, et al., 2022). If no proper regularization is applied, it is difficult for the model to focus on the patterns of the normal data.

(3) The reconstruction-based method does not have a unified learning objective. For MTS data, at the pixel level, the difference between normal samples may be very large, resulting in the autoencoder not taking account of the multiple patterns of normal data simultaneously. The inlier priority is seriously weakened (Wang et al., 2019).

Contrastive learning has been shown to successfully learn the transformation-invariant representations in different fields, including machine vision (Chen, Kornblith, Norouzi, & Hinton, 2020), audio analysis (Oord, Li, & Vinyals, 2018), and language comprehension (Fang, Wang, Zhou, Ding, & Xie, 2020), showing a powerful ability in unsupervised representation learning. It learns similarities and differences between samples by comparing positive and negative pairs. In this way, the latent representation contains complete high-level semantic information of the data and is effective for various downstream tasks. Therefore, integrating contrastive learning into the autoencoder is a natural idea to enhance the model's capacity to acquire high-level semantic information. However, existing contrastive learning frameworks are unsuitable for MTS anomaly detection. First, using proper data augmentation measures to create different views for each original sample is vital for contrastive learning (Chen et al., 2020). Some current transformations, such as clipping and flipping, may change the patterns of the original MTS data. However, some previous contrastive learning strategies for MTS data (Eldele et al., 2021) believe that the original and the augmented samples are similar. Some studies obtain the augmented samples by adding noise to the data (Zhou et al., 2022). However, the noise scale is difficult to control. Assuming that the noise scale is small, although the augmented samples obtained by adding noise share high-level semantic information with the original samples, their effect on learning the transformation-invariant representation is weak. Because the difference between the original and augmented data is too small, and their latent variables are close to each other. Conversely, too much additional information will be introduced if the noise scale is too large, destroying the original data pattern. The augmented sample, which differs from the original sample at the pixel level but shares high-level semantic information with the original sample, is more conducive to learning the overall trend of MTS. Secondly, the instance discrimination (Wu, Xiong, Yu, & Lin, 2018) proxy task with InfoNCE or its variant loss, which is commonly used at present, encourages the projections of the samples' features onto the unit hypersphere and separates them from each other. Since the time series anomaly detection only utilizes normal data to train the model, many samples contain the same trend. The training data may contain many false negative samples. Using the InfoNCE loss may cause the model to focus on increasing the differences between the features of normal samples, resulting in the features being too scattered and losing some useful information. Such representations are not effective in the downstream task.

The existing reconstruction-based MTS anomaly detection approaches are insufficient for learning high-level semantic information of data, and the existing contrastive learning frameworks are unsuitable for MTS anomaly detection. This paper proposes an adversarial contrastive autoencoder (ACAE) framework, which integrates contrastive learning and adversarial training into the reconstruction framework to capture the normal pattern of MTS robustly. The contributions of this paper are as follows:

- An adversarial contrastive autoencoder framework for robust multivariate time series anomaly detection is proposed. It integrates the contrastive learning constraint into the autoencoder to capture the normal pattern of MTS, simultaneously considering the time series' point-wise information and overall trend. Moreover, it introduces adversarial training to identify the similarity

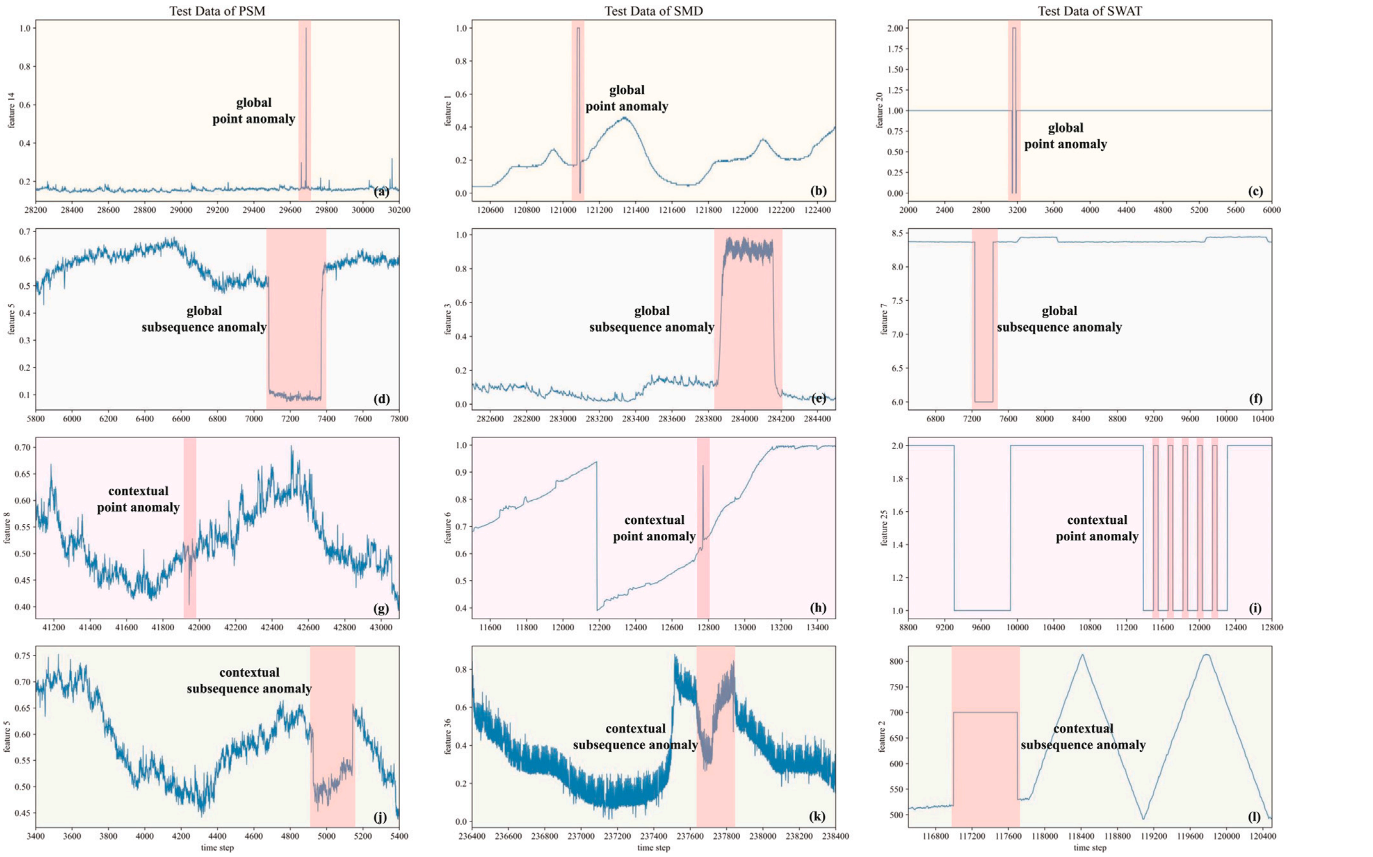


**Fig. 1.** Visualization of multiple anomaly patterns in the real-world datasets.

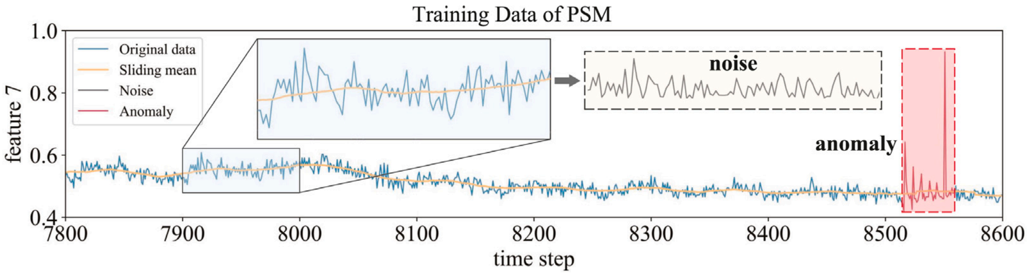


**Fig. 2.** Visualization of the noise and anomaly in the training data of PSM.

between features, which does not assume the prior distribution of features and avoids the problem of information loss, making detecting anomalies more accurate.

- A proxy task based on the feature combination and decomposition for MTS contrastive learning is proposed. Through executing this well-designed proxy task, the model learns the commonalities and subtle differences between the norm MTS data and obtains a robust representation of MTS.
- A multi-scale timestamp mask-based MTS data augmentation method is proposed. It encourages the model to learn the overall trend of MTS through multi-scale information, which is more conducive to learning the transformation-invariant representation and the high-level semantic information of MTS.
- A carefully designed and comprehensive experimental study is carried out. Except for the AUC, to avoid the overestimation caused by the point adjustment, two recent evaluation measures (Garg, Zhang, Samaran, Savitha, & Foo, 2021; Kim, Choi, Choi, Lee, & Yoon, 2022) are adopted in this paper, which can better distinguish the excellent and ordinary models. Experiments empirically prove that ACAE outperforms 14 state-of-the-art baselines on five real-world datasets from different fields.

The rest of this paper is organized as follows: Section 2 discusses the existing reconstruction-based time series anomaly detection methods and the research status of contrastive learning; Section 3 describes the adversarial contrastive autoencoder framework and the implementation details of its main modules; Section 4 presents a series of experimental settings, reported results, and analyses on public datasets; Section 5 gives the conclusions and future work.

## 2. Related work

### *2.1. Reconstruction-based methods*

MTS anomaly detection has attracted wide attention as a hot study field recently. The reconstruction model based on depth autoencoder is the most popular method. It uses an encoder to extract the low-dimensional data representation and a decoder to reconstruct MTS.

Since anomalies are more difficult to reconstruct than normal data, the reconstruction errors of test data can be used to detect anomalies.

*Deep Autoencoding Gaussian Mixture Model* (DAGMM) (Zong et al., 2018) is a classic algorithm for anomaly detection based on deep autoencoder. It first combines the latent vector and reconstruction error of the sample obtained by the autoencoder and then uses an estimation network to detect anomalies. However, the autoencoder designed in DAGMM does not consider the temporal dependence in time series. Hsieh, Chou, and Ho (2019) used LSTM as the encoder and decoder to learn the temporal dependence between time series and conducted anomaly detection on the early factory production line. *Multi-Scale Convolutional Recurrent Encoder–Decoder* (MSCRED) (Zhang et al., 2019) obtains the feature graphs of the sample at different scales based on the convolutional network. It uses the attention-based convolutional long short-term memory network to reconstruct the sample at multiple scales. BeatGAN (Zhou, Liu, Hooi, Cheng, & Ye, 2019) adds a discriminator to the original autoencoder structure to improve the authenticity of reconstruction through adversarial training. The USAD model (Audibert, Michiardi, Guyard, Marti, & Zuluaga, 2020) trains two autoencoders adversarially and obtains the anomaly score for each sample according to the two autoencoders' reconstruction errors. Li et al. (2021) proposed the InterFusion, which is based on the hierarchical variational autoencoders and adopts the dual-view embedding strategy. It explicitly considers the correlation between the time steps and dimensions of the time series. Shen, Yu, Ma, and Kwok (2021) designed a multi-resolution LSTM autoencoder network. It uses multiple LSTM with jump connections to obtain the multi-scale features of data and reconstruct the input. In addition, methods for extracting MTS features based on the Transformer model have received wide attention. For example, TranAD (Tuli, Casale, & Jennings, 2022) suggests a Transformer-based MTS anomaly detection algorithm and uses adversarial training and self-regulation techniques to detect anomalies more accurately. *Anomaly Transformer* (AT) (Xu, Wu, Wang, & Long, 2021) explores the self-attention weights of abnormal time steps and proposes a minimax training technique to enhance the distinction between inliers and outliers.

Most existing reconstruction-based approaches focus on extracting the complex patterns of MTS by designing different autoencoder networks. However, the autoencoder is poor in learning high-level semantic information of data, which limits the reconstruction-based method to model the MTS normal patterns.

### 2.2. Contrastive learning

Self-supervised learning can extract features from unlabeled data by proxy tasks and apply the learned data representations to various downstream tasks. Among them, contrastive learning shows a powerful ability for unsupervised representation learning. By maximizing the distance between the features of different views of the same sample while minimizing the distance between the features of different samples, the model learns to extract the transformation-invariant representations of data. Currently, the model pre-trained by contrastive learning approaches or even exceeds the model with supervised training in many machine vision downstream tasks (Caron et al., 2020; Chen et al., 2020; Grill et al., 2020; He, Fan, Wu, Xie, & Girshick, 2020). CPC (Oord et al., 2018) uses the autoregressive model to predict the latent vectors of the following data and proposes InfoNCE loss to constrain the contrastive learning. MoCo (He et al., 2020) further considers the issue of how to obtain more negative pairs for comparison and constructs a dynamic dictionary to solve this problem. SimCLR (Chen et al., 2020) discusses the vital status of data augmentation and proposes a learnable data augmentation module to promote the effect of contrastive learning. Besides, some subsequent studies (Caron et al., 2020; Grill et al., 2020) can realize contrastive learning and obtain robust data representations without using negative sample pairs.

Recently, using contrastive learning to obtain robust representations of MTS is attracting more and more attention. Eldele et al. (2021) used weak and strong augmentations to convert the original time series into two different but related views and conducted contrastive learning through the designed double-view cross-prediction task. Yue et al. (2022) proposed a time series representation framework based on contrastive learning. It can obtain time series representations at all semantic levels by introducing instance-level and fine-grained contrastive learning tasks on multiple time scales. TimeCLR (Yang, Zhang, & Cui, 2022) obtains two different views of the original time series through a learnable transformation inspired by *Dynamic Time Warping* (DTW) and uses the time series classification model InceptionTime (Ismail Fawaz et al., 2020) to extract features for contrastive learning and data representation. The most relevant method of this paper is *Contrastive Autoencoder for Anomaly Detection* (CAE_AD) (Zhou et al., 2022), which combines the contrastive learning framework with the autoencoder-based MTS anomaly detection framework. It uses contrastive learning at the window and pixel levels to extract the normal patterns of MTS, then performs anomaly detection based on reconstruction errors.

In general, most existing contrastive learning-based time series analysis algorithms are inspired by the classical frameworks in machine vision and natural language processing. However, their data augmentation methods and proxy tasks are unsuitable for time series data, so it is difficult to clearly outline the normal patterns of MTS data.

## 3. Method

Existing methods are incompetent at constructing an accurate profile for the normal pattern of MTS and obtaining robust data representation, resulting in poor performance in practical applications. This paper proposes an adversarial contrastive autoencoder (ACAE) for robust multivariate time series anomaly detection. In this section, the problem description of multivariate time series anomaly detection is first given, and the framework of ACAE is outlined. Then, the main modules of ACAE and the anomaly detection process are described in detail.

### 3.1. Problem description

Let $\mathcal{T} \in \mathbb{R}^{M\times T}$ be a time series data, where $T$ and $M$ represent the length and dimension of $\mathcal{T}$, which is an observation sequence of $T$ time steps collected on $M$ observation objects. If $M = 1$, the time series is a *univariate time series*; if $M > 1$, the time series is a *multivariate time series* (MTS). This paper focuses on detecting anomalies in MTS. Let $s_t \in \mathbb{R}^M$ represent the observed vector of $\mathcal{T}$ at time step $t$, then $\mathcal{T}$ can be expressed as $\mathcal{T} = \{s_1, s_2, s_3, \dots, s_T\}$. MTS anomaly detection aims to detect the abnormal time steps in $\mathcal{T}$. Specifically, an anomaly score is calculated for each time step of $\mathcal{T}$. Then, the time step whose anomaly score exceeds the given threshold is detected as an anomaly.

### 3.2. Framework overview

The schematic diagram of the overall flow of ACAE is shown in Fig. 3. The main modules of the proposed framework include data preprocessing, sample pairs construction, feature extraction, feature combination and decomposition, and reconstruction. First, the original time series is standardized and divided by the sliding window. Secondly, multiple augmented views of original samples are generated through the multi-scale timestamp mask as positive samples, while negative samples are obtained through random sampling. After that, the encoder is used to extract the features of all samples. After that, the feature combination and decomposition proxy task is carried out for contrastive learning. The main process is to combine the features of sample pairs and then adversarially train the discriminator to predict the category of composite features and the proportion of original samples' features in composite features. Finally, the decoder is trained to reconstruct the time series, and the anomaly score of each time step is calculated based on the reconstruction error.

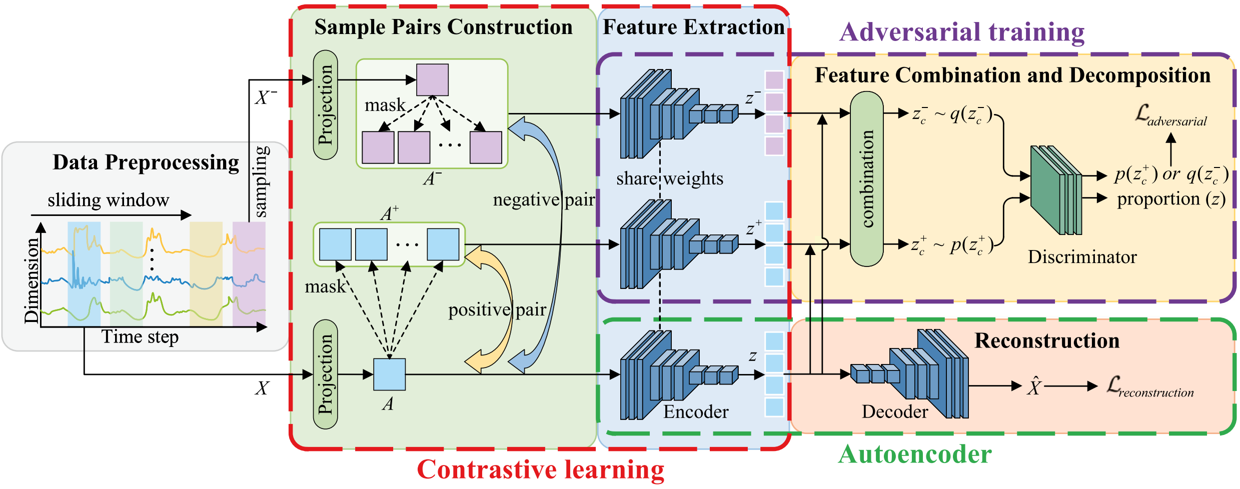


**Fig. 3.** The diagram of the proposed framework ACAE.

### 3.3. Data preprocessing

Firstly, standardization is applied to scale each dimension of multivariate time series data to eliminate large-scale values and accelerate the convergence of the training process. In addition, since MTS data usually contains numerous time points, it is impossible to input the complete time series into the model simultaneously. In order to capture the temporal dependence between time steps, a sliding window with the width of $w$ and the stride of $d$ is used to obtain continuous data segments as the model's input. It means a time window contains $w$ time steps, and the sliding window moves $d$ time steps on the time axis of the original long time series each time. Let $\boldsymbol{X}_i$ represent a time window sample, and $\boldsymbol{x}_{it}$ represents the observation vector of $\boldsymbol{X}_i$ at time step $t$. $\boldsymbol{X}_i$ can be expressed as $\boldsymbol{X}_i = \{\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \boldsymbol{x}_{i3}, \dots, \boldsymbol{x}_{iw}\}$, $\boldsymbol{X}_i \in \mathbb{R}^{M\times w}$, where $M$ is the dimension of the MTS, and $w$ is the window size.

### 3.4. Sample pairs construction

Using data augmentation methods to create different views of the same sample is essential for contrastive learning (Chen et al., 2020). However, some current transformations used for time series augmentation may change the original sample's pattern, making it difficult for the model to learn a reliable data representation. Mask learning has been successfully used in machine vision (He et al., 2022) and natural language processing (Devlin, Chang, Lee, & Toutanova, 2018). The previous works applying the mask have shown that models will obtain a strong representation ability by inferring the invisible part according to the visible part of the data. Therefore, the augmented samples are obtained by adding masks to the original time series in this paper. Although the masked sample may differ from the original sample significantly in pixel level, it retains the high-level semantic feature of the original time series. Besides, masking does not introduce extra noise. By maximizing the similarity between the features of the masked sample and the original sample, the encoder learns the overall trend of the time series by inferring the invisible time steps, which is significant for extracting the high-level semantic features of MTS.

Although the masks can be directly added to the original time series, it is difficult to find a unique marker in the range of values of the original time series. Generally, the value 0 is usually used as the marker. However, the original time series may contain many time steps with a value of 0, so the model cannot distinguish which time steps are masked. In order to enable the network to distinguish between the visible part and the masked part of the time series, a projection layer is first applied to project each time step of original data into a higher dimension and obtain the projection space representation $\boldsymbol{A}_i$ of each time window sample $\boldsymbol{X}_i$:

$$\boldsymbol{a}_{it} = \boldsymbol{W}\boldsymbol{x}_{it} + \boldsymbol{b} \tag{1}$$

where $\boldsymbol{x}_{it} \in \mathbb{R}^{M}$ and $\boldsymbol{a}_{it} \in \mathbb{R}^{M'}$ are the vector of the original time window $\boldsymbol{X}_i$ and the projection space representation $\boldsymbol{A}_i \in \mathbb{R}^{M\times w}$ at the time step $t$, respectively, where $M' > M$. The projection layer with parameters $\{\boldsymbol{W}, \boldsymbol{b}\}$ will be optimized with the encoder. After the projection layer, the timestamp mask is applied to obtain the different views of the original sample in the projection space. Specifically, $w \times p$ time steps along the time axis are sampled and set to $\mathbf{0}$, where $p$ is the mask rate. Random sampling is carried out independently of each time window data in each forward process. It can be proved that there away exists a set of parameters $\{\boldsymbol{W}, \boldsymbol{b}\}$ such that any time step of the original time series is not $\mathbf{0}$ in the projection space (Yue et al., 2022), so the network can distinguish which time steps are masked.

The schematic diagram of the construction of positive and negative sample pairs is shown in Fig. 4. This paper proposes the MTS augmentation method based on the multi-scale timestamp mask. Like most reconstruction-based MTS anomaly detection methods, the proposed method assumes that the training data contains only normal data, and $N$ represents the number of time window samples used for training. A series of augmented samples is obtained by adjusting the mask rate $p$ for each original sample. The augmented samples with different mask rates can help the model extract multi-scale information of time series and enhance the effect of contrastive learning. When the mask rate is low, the invisible part of the data is small, making it easier for the model to learn the detailed features of MTS. On the contrary, when the mask rate is large, the visible part of the data is sparse. The model can learn more high-level semantic information about the data according to the overall trend of the MTS. For a projection space representation $\boldsymbol{A}_i$ from the original data, this paper obtains $m$ augmented view $\{\boldsymbol{A}_i^{(1)+}, \boldsymbol{A}_i^{(2)+}, \boldsymbol{A}_i^{(3)+}, \dots, \boldsymbol{A}_i^{(m)+}\}$ by setting different $p$, so each sample has $m$ positive sample pairs. For each original sample, other samples from the same minibatch and their augmented samples can be regarded as the candidate negative sample pool for that sample. There may be many false negative samples in the time series dataset, and using all negative samples for contrastive learning will lead to much computational consumption. Given the hyperparameter $n$, this paper samples $n$ original samples within the minibatch and $n$ augmented views from each class as negative samples. Each sample has $n \times (m+1)$ negative sample pairs.

### 3.5. Feature extraction

After constructing the positive and negative pairs, all original samples and their positive and negative samples are sent to the Siamese

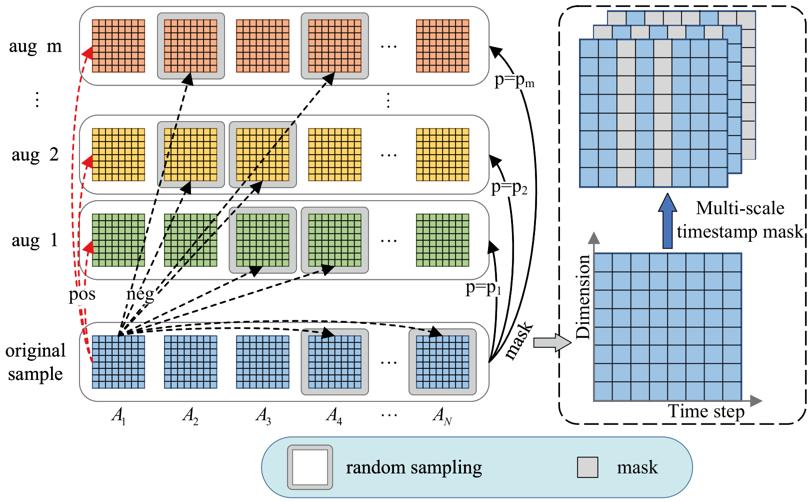


**Fig. 4.** Construction of positive and negative sample pairs.

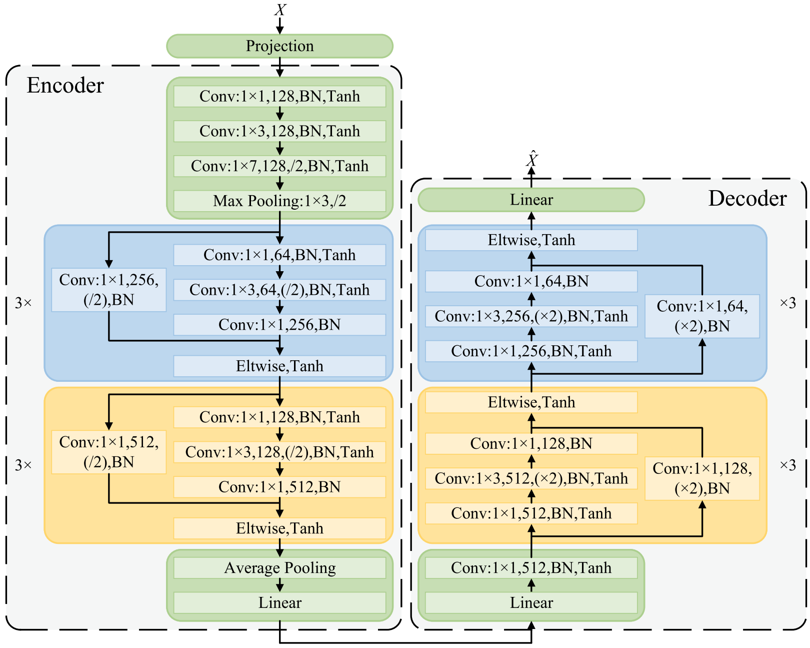


**Fig. 5.** Structure of encoder and decoder.

encoders with shared weights to obtain low-dimensional latent vectors. Let $E(\cdot)$ represent the encoder. This process can be expressed as follows:

$$\boldsymbol{z}_i = E(\boldsymbol{A}_i) \tag{2}$$

where $\boldsymbol{A}_i$ is the projection space representation of $\boldsymbol{X}_i$, $\boldsymbol{z}_i \in \mathbb{R}^h$ is the latent variable of $\boldsymbol{A}_i$, and $h$ is the length of the latent vector.

1D convolutional neural network (1D-CNN) is one of the most widely used neural networks for time series data analysis. ACAE also uses the 1D-CNN network as the backbone network for feature extraction. Since ResNet (He, Zhang, Ren, & Sun, 2016) has shown excellent feature extraction performance in many past studies, the encoder of ACAE is implemented by referring to the first two residual blocks of ResNet50, in which 1D-CNN replaces 2D-CNN, as shown in Fig. 5. In some convolutional layers, (/2) indicates that when multiple residual modules with the same structure are stacked, only the first residual module halves the time dimension of input data. The subsequent residuals do not change the length of the input data.

### 3.6. Feature combination and decomposition

The instance discrimination with InfoNCE or its variant loss is the mainstream proxy task in contrastive learning. It directly calculates the similarity between features, maximizes the similarity between different views of the same sample, and minimizes the similarity between different samples. Time series anomaly detection only uses normal data to train the model. The InfoNCE loss may cause the features of the samples to be too scattered in the latent space, resulting in the loss of some useful information. Therefore, it is difficult for the existing contrastive learning frameworks to obtain robust data representations for time series data, which is not conducive to the downstream anomaly detection task.

This paper proposes the feature combination and decomposition as the proxy task for contrastive learning. It includes two parts: feature combination and feature decomposition.

**Feature combination.** After obtaining the features of the original samples and their positive and negative samples, ACAE combines the features of the positive and negative sample pairs with a proportion,

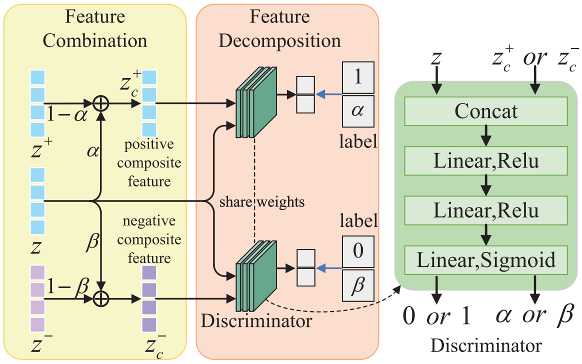


**Fig. 6.** Feature combination and decomposition.

respectively, to obtain the positive and negative composite features. As shown in Fig. 6, $\boldsymbol{z}$ is a latent space representation of an original sample, while $\boldsymbol{z}^+$ and $\boldsymbol{z}^-$ are the latent variables of one of its positive and negative samples. $\boldsymbol{z}$ is combined with $\boldsymbol{z}^+$ and $\boldsymbol{z}^-$ to construct positive and negative composite features. The process is shown as follows:

$$\begin{aligned} \boldsymbol{z}_c^+ &= \alpha \boldsymbol{z} + (1-\alpha)\boldsymbol{z}^+ \\ \boldsymbol{z}_c^- &= \beta \boldsymbol{z} + (1-\beta)\boldsymbol{z}^- \end{aligned} \tag{3}$$

where $\alpha$ and $\beta$ are the combination parameters that determine the contribution of each feature in the composite feature. In each forward process, when each composite feature is generated, $\alpha$ and $\beta$ are sampled independently from uniform distributions with minimum and maximum values of 0 and 1, that is, $\alpha \sim U(0,1)$, $\beta \sim U(0,1)$.

**Feature decomposition.** ACAE implements a discriminator by the Multilayer Perceptron (MLP) to decompose the composite features, as shown in Fig. 6. The objects of the discriminator are to correctly classify the composite features and predict the proportion of the original features in the composite features, that is, the combination parameters $\alpha$ or $\beta$. Due to time series anomaly detection only using normal data to train the model, the differences between the training samples are small. The proposed proxy task is more helpful for the model to learn the differences between the data. In the proxy task, it is difficult to correctly predict the proportion of the original samples' features from the composite features. If the model can correctly decompose the composite features, it has learned the differences between the normal data. Specifically, the feature of the original sample and one of its composite features are fed into the discriminator. A vector of length two can be obtained as the output of feature decomposition. The first digit of the vector represents the discriminator's prediction of the class of the composite feature. Label 1 is used to represent the positive composite feature, and 0 is used to represent the negative composite feature. The second digit of the output represents the discriminator's prediction of the proportion of the original sample's feature, and the ground truth is $\alpha$ or $\beta$. ACAE uses MSE loss to constrain the difference between the prediction vector and the ground truth, so the loss of discriminator is as follows:

$$\mathcal{L}_d = \frac{1}{N}\frac{1}{m+n+n\times m}\sum_{i=1}^{N}(\sum_{j=1}^{m}(\boldsymbol{p}_i^{(j)+} - \boldsymbol{l}_d^+)^2 + \sum_{k=1}^{n+n\times m}(\boldsymbol{p}_i^{(k)-} - \boldsymbol{l}^-)^2) \tag{4}$$

where $N$ is the number of training samples, and $\boldsymbol{p}_i^{(j)+}$ and $\boldsymbol{p}_i^{(k)-}$, represent the decomposition results of the discriminator on the $j_{th}$ positive composite feature and the $k_{th}$ negative composite feature of the $i_{th}$ sample. $\boldsymbol{l}_d^+$ and $\boldsymbol{l}^-$ are the ground truth, where $\boldsymbol{l}_d^+ = [1,\alpha]^T$ and $\boldsymbol{l}^- = [0,\beta]^T$.

If the discriminator can decompose a composite feature correctly, the two components of the composite feature are at least somewhat different. According to contrastive learning, the similarity between the features of the positive sample pairs should be maximized. Suppose the discriminator can correctly decompose the negative composite features while unable to correctly decompose the positive composite features. In that case, it indicates that the features of the original samples are significantly different from those of their negative samples and similar to those of their positive samples. Suppose an original sample's feature $\boldsymbol{z}$ is the same as its augmented sample's feature $\boldsymbol{z}^+$. The positive composite feature $\boldsymbol{z}_c^+$ constructed by any $\alpha$ is the same as $\boldsymbol{z}$. The discriminator will identify that the composite feature only comes from the original sample and predict the value of $\alpha$ to be 1. According to this, generative adversarial antagonism is introduced in predicting the combination parameter $\alpha$ of positive composite features. The encoder is trained under the discriminator's guidance, which tries to fool the discriminator when predicting the value of $\alpha$. The loss of the encoder is as follows:

$$\mathcal{L}_E = \frac{1}{N}\frac{1}{m+n+n\times m}\sum_{i=1}^{N}(\sum_{j=1}^{m}(\boldsymbol{p}_i^{(j)+} - \boldsymbol{l}_E^+)^2 + \sum_{k=1}^{n+n\times m}(\boldsymbol{p}_i^{(k)-} - \boldsymbol{l}^-)^2) \tag{5}$$

where $\boldsymbol{l}_E^+ = [1,1]^T$ is the difference between the loss functions of the encoder and discriminator. The contrastive learning encourages the model to learn the transformation-invariant representation and the high-level semantic information of MTS, improving the robustness of the model.

By training the encoder and discriminator to decompose the negative composite features cooperatively and decompose the positive composite features adversarially, the similarity between the features of negative pairs decreases while that of positive pairs increases. The model learns the commonalities or differences between the positive or negative pairs and obtains robust representations of MTS. Moreover, ACAE designs the discriminator to identify the similarity between features instead of directly calculating it. It avoids the problem of information loss caused by InfoNCE loss that forces features to be uniformly projected onto the hyper-sphere in the latent space, which is more suitable for time series anomaly detection.

### *3.7. Reconstruction and anomaly detection*

Contrastive learning helps the encoder learn the time series' overall trend and high-level semantic features. Meanwhile, the reconstruction task enables the model to focus more on fine-grained features. In addition, two-stage tasks may result in suboptimal performance. Therefore, ACAE carries out the contrastive learning and reconstruction tasks simultaneously. This paper implements a decoder shown in Fig. 5, whose structure is approximately symmetric with the encoder. Its objective is to reconstruct the input data according to the sample's latent variable, which can be expressed as:

$$\hat{\boldsymbol{X}}_i = D(\boldsymbol{z}_i) \tag{6}$$

where $\hat{\boldsymbol{X}}_i \in \mathbb{R}^{M\times w}$ is the reconstructed sample of $\boldsymbol{X}_i$, $\boldsymbol{z}_i$ is the latent variable of $\boldsymbol{X}_i$, and $D(\cdot)$ represents the encoder. The loss of reconstruction calculated by MSE can be expressed as:

$$\mathcal{L}_{recon} = \frac{1}{Nw}\sum_{i=1}^{N}\sum_{t=1}^{w}(\hat{\boldsymbol{x}}_{it} - \boldsymbol{x}_{it})^2 \tag{7}$$

where $N$ is the number of training Windows, $w$ is the sliding window size, and $\hat{\boldsymbol{x}}_{it}$ and $\boldsymbol{x}_{it}$ represent the $i_{th}$ reconstructed and original data at time step $t$, respectively. The total training loss of ACAE can be expressed as the sum of reconstruction loss and contrastive loss:

$$\mathcal{L} = \mathcal{L}_{recon} + \lambda_1\mathcal{L}_d + \lambda_2\mathcal{L}_E \tag{8}$$

where $\lambda_1$ and $\lambda_2$ are the hyperparameters that control the relative importance of each loss. In this paper, alternate optimization is adopted in the model optimization stage, and Algorithm 1 summarizes the training process of ACAE.

ACAE is optimized using a training set of mostly normal data to learn the normal patterns of the MTS. After completing the training

**Algorithm 1** The training stage of ACAE

```
Input: Pre-processed dataset $\mathcal{X}$, initialized encoder $E(\cdot)$, discriminator $d(\cdot)$,
    and decoder $D(\cdot)$;
Output: Trained encoder $E(\cdot)$ and decoder $D(\cdot)$;
 1: repeat
 2:     Input $\mathcal{X}$ into the projection layer;
 3:     Obtain positive and negative samples pairs;
 4:     Input all samples into $E(\cdot)$ to obtain latent variables;
 5:     Construct the composite features according to Eq. (3);
 6:     Execute feature decomposition task;
 7:     $\mathcal{L}_d \leftarrow \lambda_1 \frac{1}{N} \frac{1}{m+n+n\times m} \sum_{i=1}^{N} (\sum_{j=1}^{m} (\boldsymbol{p}_i^{(j)+} - \boldsymbol{l_d}^{+})^2 + \sum_{k=1}^{n+n\times m} (\boldsymbol{p}_i^{(k)-} - \boldsymbol{l}^{-})^2)$;
 8:     Optimize $d(\cdot)$ to minimize $\mathcal{L}_d$;
 9:     Execute (6) again;
10:     $\mathcal{L}_E \leftarrow \lambda_2 \frac{1}{N} \frac{1}{m+n+n\times m} \sum_{i=1}^{N} (\sum_{j=1}^{m} (\boldsymbol{p}_i^{(j)+} - \boldsymbol{l_E}^{+})^2 + \sum_{k=1}^{n+n\times m} (\boldsymbol{p}_i^{(k)-} - \boldsymbol{l}^{-})^2)$;
11:     Optimize $E(\cdot)$ to minimize $\mathcal{L}_E$;
12:     Reconstruct $\mathcal{X}$ using $E(\cdot)$ and $D(\cdot)$;
13:     $\mathcal{L}_{recon} \leftarrow \frac{1}{Nw} \sum_{i=1}^{N} \sum_{t=1}^{w} (\hat{\boldsymbol{x}}_{it} - \boldsymbol{x}_{it})^2$;
14:     Optimize $E(\cdot)$ and $D(\cdot)$ to minimize $\mathcal{L}_{recon}$;
15: until convergence
16: return $E(\cdot)$, $D(\cdot)$
```

stage, ACAE keeps the encoder and decoder and discards the discriminator module. In the test stage, ACAE reconstructs the test sample using the encoder and decoder and calculates the anomaly scores of each time step based on the reconstruction errors, which are defined as follows:

$$S_{it} = (\hat{\boldsymbol{x}}_{it} - \boldsymbol{x}_{it})^2 \tag{9}$$

where $S_{it}$ is the anomaly score of the time step $t$ in $i_{th}$ test data. The threshold selection depends on the practical application scenario, and many studies (Hundman, Constantinou, Laporte, Colwell, & Soderstrom, 2018; Su et al., 2019) on dynamic thresholds have been conducted. This paper focuses on designing a framework for obtaining robust representations of MTS data to conduct anomaly detection more accurately. Therefore, the experimental results reported in this paper are based on the approximate optimal threshold, as in the previous studies (Gao, Qiu, et al., 2022; Zhou et al., 2022).

## 4. Experiment

In this section, several experiments are conducted to evaluate the performance of the proposed method. First, ACAE is compared with 14 baselines on five real-world datasets. Subsequently, parameter sensitivity experiments are carried out to study the response of ACAE under different settings. In addition, ablation experiments are conducted to construct five variants to verify the effectiveness of each module in ACAE. Finally, a case is given to illustrate the distribution of the latent space and anomaly scores of ACAE.

### 4.1. Experimental setup

#### 4.1.1. Datasets

Five real-world datasets from three application fields are used in the experiments in Section 4. Table 1 summarizes the properties of these datasets.

**Secure Water Treatment (SWaT)** (Mathur & Tippenhauer, 2016). SWaT is collected by 51 sensors in a continuously operating water treatment system that records abnormal events caused by cyber and physical attacks.

**Server Machine Dataset (SMD)** (Su et al., 2019). SMD is a five-week dataset collected and publicly released by a large internet company from a server machine with 38 monitoring metrics.

**PSM (Pooled Server Metrics)** (Abdulaal et al., 2021). PSM is a dataset collected from eBay's multiple application server nodes with 26 dimensions.

**Mars Science Laboratory (MSL)** dataset and **Soil Moisture Active Passive (SMAP)** dataset (Hundman et al., 2018). The MSL and SMAP datasets are real-world datasets from NASA with 55 and 25 dimensions, respectively. They contain telemetry anomaly data from the spacecraft monitoring system's Incident Surprise Anomaly (ISA) report.

#### 4.1.2. Evaluation measures

In this paper, AUC, $\text{Fc}_1$, and PA%K are selected as evaluation measures to evaluate the performance of the proposed method and baselines.

**Area Under Curve (AUC).** AUC is one of the most popular measures for evaluating unsupervised anomaly detection tasks. AUC is the area formed by the Receiver Operating Characteristic Curve (ROC) and the axes, which ranges from 0 to 1. It can directly reflect the sorting quality of the algorithm on the test set and exclude the influence of thresholds. A perfect ranking would result in a value of 1, while a random ranking would produce a value approaching 0.5.

**Composite F-score (Fc1)** (Garg et al., 2021). $\text{Fc}_1$ is a metric proposed in the recent literature for time series anomaly detection, focusing on the algorithm's ability to detect abnormal events. It avoids the overestimation caused by the point adjustment (Xu et al., 2018) strategy by calculating event-wise recall and time-wise precision in the F1-score calculation. The model with the higher recall of abnormal segments and fewer false positives of normal time steps will obtain a higher $\text{Fc}_1$ score.

**Point Adjustment%K (PA%K)** (Kim et al., 2022). Similarly, PA%K is proposed to solve the overestimating of the model performance caused by the point adjustment. It calculates the point-wise F1-score but conducts the point adjustment when the proportion of anomalies detected by the model in a continuous abnormal segment exceeds $K$ percent. To reduce the dependence of $K$, $K$ is set to a series of values, and the area under the curve of PA%K is calculated and used as the final score.

#### 4.1.3. Compared methods

This paper compares ACAE with 14 baseline methods, as shown below. LOF, OCSVM, and iForest are classic anomaly detection methods based on machine learning, while others are popular deep learning-based algorithms for time series anomaly detection.

**LOF** (Breunig et al., 2000). It is an anomaly detection method by calculating the local density deviation of the given sample relative to its neighborhoods.

**OCSVM** (Schölkopf, Williamson, Smola, Shawe-Taylor, & Platt, 1999). It maps samples to a high dimensional space by kernel function and divides the boundary between normal samples and anomalies.

**iForest** (Liu et al., 2012). It is an ensemble model that isolates anomalies by randomly selecting features and dividing observations.

**MSCRED** (Zhang et al., 2019). It reconstructs the time series at multiple scales using the attention-based convolutional long short-term memory network.

**BeatGAN** (Zhou et al., 2019). It is a model based on the adversarial autoencoder, which adds a discriminator to the original autoencoder to improve the authenticity of reconstruction.

**USAD** (Audibert et al., 2020). It trains two auto-encoders adversarially and obtains the anomaly score for each sample according to the reconstruction errors of the two auto-encoders.

**UAE** (Garg et al., 2021). It is a lightweight model based on the fully connected autoencoder proposed with Fc1 and obtains good results on Fc1.

**InterFusion** (Li et al., 2021). It is a reconstruction model based on the hierarchical variational autoencoders to learn the temporal and dimensional dependence.

**Table 1**
Descriptions of the real-world datasets.

| Dataset | Application | Dimension | Train | Test | Anomalies (%) |
|---|---|---|---|---|---|
| SWaT | Water | 51 | 495 000 | 449 919 | 12.14 |
| SMD | Server | 38 | 708 405 | 708 420 | 4.16 |
| PSM | Server | 25 | 132 481 | 87 841 | 27.76 |
| MSL | Space | 55 | 58 317 | 73 729 | 10.53 |
| SMAP | Space | 25 | 135 183 | 427 617 | 12.79 |

**GDN** (Deng & Hooi, 2021). It is a model based on the attention mechanism and graph neural network to learn the structure of MTS. It uses prediction errors to detect anomalies.

**GTA** (Chen et al., 2021). It is a prediction-based method that combines the graph neural network and Transformer to model the norm pattern of MTS.

**TranAD** (Tuli et al., 2022). It is a deep Transformer-based model with self-regulation and adversarial training to amplify reconstruction errors.

**AT** (Xu et al., 2021). It is a Transformer-based model detecting anomalies through the correlation differences between sequences except for reconstruction errors.

**TimeCLR** (Yang et al., 2022). It is a contrastive learning framework for time series representation, which uses dynamic time warping for data augmentation and InceptionTime for feature extraction.

**CAE_AD** (Zhou et al., 2022). It is an end-to-end autoencoder combining context-level and instance-level contrastive learning.

#### *4.1.4. Implementation details*

This paper implements ACAE based on PyTorch. In the data augmentation stage, four augmented views for each training sample are generated with mask rates of 0.05, 0.15, 0.3, and 0.5. The random sampling parameter $n$ is set to 4. For the implementation of networks, the dimension $M'$ of the projection layer and the dimension $h$ of the latent vector are set to 256. The number of residual modules in the autoencoder is set to 3, and the dropout of 0.5 is applied in the first two layers of the discriminator. $\lambda_1$ and $\lambda_2$ are set to 1. Furthermore, the sliding window size $w$ is set to 64, the interval $d$ is set to 2, and the batch size is set to 128. The original training data are divided into the training set and validation set according to the ratio of $8 : 2$. The maximum epoch is set at 200. However, when the reconstruction loss of the validation set does not decrease for five consecutive epochs, the training process is stopped in advance, and the model with the lowest reconstruction loss is retained. Adam optimizer with the learning rate and weight decay of 0.0001 is used to train ACAE. All experiments are repeated five times under different random seeds, and average results are reported.

### *4.2. Results evaluation*

Table 2 reports the AUC, $\mathrm{Fc}_1$, and PA%K scores for ACAE and 14 compared methods. The best score on each dataset is indicated in bold font, and the second score on each is indicated in underlining. Overall, ACAE achieves the optimal scores in all three metrics, indicating that ACAE outperforms the compared models. CAE_AD obtains the sub-optimal $\mathrm{Fc}_1$ score and PA%K score, indicating that contrastive learning could help the reconstruction-based model better extract the normal patterns of MTS. iForest gets the sub-optimal AUC score, indicating the good sorting quality for anomaly scores of time steps. However, it may be incapable of detecting continuous anomalies in the abnormal segments because it cannot consider the temporal dependence of time series. Specifically, on the SWaT dataset, the $\mathrm{Fc}_1$ score of ACAE is much higher than other baselines, while the AUC and PA%K scores are slightly below the optimal. Although the AUC scores of ACAE are in the top two on SMD and PSM, the $\mathrm{Fc}_1$ and PA%K of ACAE are below the optimal. The AUC of ACAE is much higher than the compared methods on MSL, while the $\mathrm{Fc}_1$ and PA%K scores are slightly below the optimal. On the SMAP dataset, ACAE is average on three measures. Although ACAE does not obtain the optimal scores on each dataset, it models the normal patterns of MTS data by simultaneously considering the point-wise information and high-level semantic information of time series, making the anomalies more easily detected. Therefore, it achieves the optimal average results, which shows its robustness.

### *4.3. Parameter sensitivity*

This section conducts experiments on real-world datasets and explores the effects of different settings on ACAE performance. The key hyper-parameters of ACAE include the sampling parameter $n$ of the negative samples, the sliding window interval $d$, the sliding window size $w$, and the latent space dimension $h$. Each parameter is fixed to conduct the experiments, and the combination of other parameters is randomly selected to study the proposed algorithm's performance under each parameter's setting. The experiment is repeated 20 times for each parameter setting, and the average results are reported as final. The results are shown in the stacked column diagram in Fig. 7.

The first parameter studied in this section is the sampling number $n$ of the negative samples. $n$ can indicate that the negative samples of each sample are multiples of the positive samples for contrastive learning. The larger $n$ is, the more negative pairs each sample will get. As shown in Fig. 7(a), the performance of ACAE on the SWAT and SMAP datasets decreases and then increases with increasing $n$. No significant trend is seen in the other datasets. From the perspective of average results, the best results can be obtained by a smaller $n$ because the training time series are all normal data similar to each other. Increasing the number of negative samples may form many false negative pairs, which is not conducive to modeling the normal pattern of MTS. Thus, $n = 4$ is set as the default setting for the proposed method.

The second parameter analyzed in this paper is the interval $d$ of the sliding window. The smaller $d$ is, the more training samples will be generated, and the similarity between original samples will increase. Fig. 7(b) illustrates the results in the different settings of the sliding window intervals. As shown in the figure, the performance of ACAE decreases with the increase of $d$ on MSL and the average results. No significant trend is seen in the other datasets. From the perspective of average results, $d = 2$ is set as the default setting for the proposed method to use the original time series information fully.

Then, the influence of window size $w$ is discussed. As the window size increases, the model will learn the longer temporal dependence of MTS. The different window sizes are used in the experiments. Fig. 7(c) shows that the best average results are obtained when the window size $w$ is set to 64. In general, the optimal setting for $w$ is related to the periodicity of each dataset. Since searching for an optimal setting for each dataset in practical applications is difficult, the empirical parameter setting w=64 is used as the default setting for the proposed method.

Finally, this paper explores how the latent variable dimension $h$ affects the performance of ACAE. The sample's representation retains more information when the dimension $h$ increases. As shown in Fig. 7(d), the performance of ACAE increases and then decreases with increasing $h$, while the tendency on SMAP is inverse. No significant trend is seen in the other datasets. Based on the average results, the dimension of latent space is set to 256 as the default setting.

In general, the performance of the proposed method is less affected by the parameters, especially on large datasets, which ensures its accuracy and stability in detecting anomalies in practice.

**Table 2**
AUC, Fc1 and PA%K for ACAE and baselines.

| Methods | SWaT | | | SMD | | | PSM | | | MSL | | | SMAP | | | Ave | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K |
| LOF | 0.785 | 0.356 | 0.787 | 0.620 | 0.131 | 0.163 | 0.729 | 0.580 | 0.606 | 0.561 | 0.274 | 0.296 | **0.630** | **0.362** | **0.389** | 0.665 | 0.341 | 0.448 |
| OCSVM | 0.793 | 0.334 | 0.790 | 0.607 | 0.241 | 0.245 | 0.638 | 0.506 | 0.506 | 0.527 | **0.458** | **0.317** | 0.407 | 0.237 | 0.197 | 0.594 | 0.355 | 0.411 |
| iForest | **0.837** | 0.597 | 0.787 | 0.670 | 0.317 | 0.262 | 0.699 | 0.618 | 0.590 | 0.598 | 0.307 | 0.299 | 0.589 | 0.275 | 0.306 | 0.679 | 0.423 | 0.449 |
| MSCRED | 0.484 | 0.146 | 0.193 | 0.694 | 0.275 | **0.325** | 0.737 | 0.566 | 0.618 | 0.624 | 0.268 | 0.308 | 0.382 | 0.344 | 0.180 | 0.584 | 0.320 | 0.325 |
| BeatGAN | 0.788 | 0.436 | 0.703 | 0.717 | 0.460 | 0.306 | 0.737 | 0.587 | 0.608 | 0.622 | 0.366 | 0.295 | 0.523 | 0.259 | 0.282 | 0.677 | 0.421 | 0.439 |
| USAD | 0.797 | 0.390 | 0.789 | 0.592 | 0.210 | 0.227 | 0.644 | 0.514 | 0.500 | 0.586 | 0.359 | 0.315 | 0.485 | 0.247 | 0.245 | 0.621 | 0.344 | 0.415 |
| UAE | 0.529 | 0.253 | 0.306 | 0.716 | **0.600** | 0.269 | 0.635 | **0.746** | 0.530 | 0.524 | 0.433 | 0.269 | 0.478 | 0.228 | 0.277 | 0.576 | 0.452 | 0.330 |
| InterFusion | 0.704 | 0.470 | 0.521 | 0.683 | 0.153 | 0.174 | 0.695 | 0.542 | 0.570 | 0.572 | 0.442 | 0.279 | 0.443 | 0.271 | 0.238 | 0.619 | 0.376 | 0.356 |
| GDN | 0.686 | 0.361 | 0.597 | 0.637 | 0.251 | 0.194 | 0.693 | 0.537 | 0.570 | 0.545 | 0.296 | 0.271 | 0.471 | 0.275 | 0.261 | 0.606 | 0.344 | 0.378 |
| GTA | 0.596 | 0.288 | 0.333 | 0.719 | 0.527 | 0.280 | **0.773** | 0.679 | **0.643** | 0.603 | 0.376 | 0.285 | 0.489 | 0.233 | 0.275 | 0.636 | 0.421 | 0.363 |
| TranAD | 0.481 | 0.202 | 0.290 | 0.615 | 0.375 | 0.209 | 0.684 | 0.540 | 0.558 | 0.587 | 0.361 | 0.314 | 0.571 | 0.299 | 0.330 | 0.587 | 0.355 | 0.340 |
| AT | 0.555 | 0.492 | 0.274 | 0.497 | 0.314 | 0.080 | 0.415 | 0.406 | 0.401 | 0.462 | 0.248 | 0.196 | 0.509 | 0.269 | 0.198 | 0.488 | 0.346 | 0.230 |
| TimeCLR | 0.817 | 0.560 | **0.813** | 0.671 | 0.472 | 0.262 | 0.706 | 0.623 | 0.557 | 0.564 | 0.440 | 0.315 | 0.433 | 0.202 | 0.203 | 0.638 | 0.459 | 0.430 |
| CAE_AD | 0.818 | 0.571 | 0.786 | 0.737 | 0.488 | 0.290 | 0.747 | 0.624 | 0.630 | 0.566 | 0.440 | 0.294 | 0.448 | 0.201 | 0.247 | 0.663 | 0.465 | 0.450 |
| ACAE | 0.829 | **0.647** | 0.805 | **0.755** | 0.487 | 0.301 | 0.752 | 0.615 | 0.603 | **0.628** | 0.442 | 0.309 | 0.515 | 0.241 | 0.292 | **0.696** | **0.486** | **0.462** |

**Table 3**
AUC, Fc1 and PA%K for ACAE and variants.

| Variants | SWaT | | | SMD | | | PSM | | | MSL | | | SMAP | | | Ave | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K | AUC | $Fc_1$ | PA%K |
| ACAE | 0.829 | 0.647 | 0.805 | 0.755 | 0.487 | 0.301 | 0.752 | 0.615 | 0.603 | 0.628 | 0.442 | 0.309 | 0.515 | 0.241 | 0.292 | **0.696** | **0.486** | **0.462** |
| ACAE_WO_PRO | 0.820 | 0.613 | 0.796 | 0.747 | 0.487 | 0.298 | 0.742 | 0.618 | 0.582 | 0.612 | 0.443 | 0.297 | 0.514 | 0.240 | 0.292 | 0.687 | 0.480 | 0.453 |
| ACAE_ONE_MASK | 0.818 | 0.604 | 0.795 | 0.755 | 0.490 | 0.292 | 0.759 | 0.623 | 0.610 | 0.623 | 0.442 | 0.300 | 0.491 | 0.231 | 0.278 | 0.689 | 0.478 | 0.455 |
| ACAE_NOISE | 0.824 | 0.629 | 0.800 | 0.751 | 0.489 | 0.290 | 0.747 | 0.623 | 0.590 | 0.622 | 0.445 | 0.309 | 0.515 | 0.244 | 0.294 | 0.692 | 0.486 | 0.456 |
| ACAE_GRU | 0.832 | 0.664 | 0.809 | 0.763 | 0.486 | 0.292 | 0.757 | 0.622 | 0.608 | 0.636 | 0.449 | 0.315 | 0.409 | 0.197 | 0.212 | 0.679 | 0.484 | 0.447 |
| ACAE_AE | 0.817 | 0.577 | 0.768 | 0.752 | 0.445 | 0.308 | 0.752 | 0.616 | 0.602 | 0.604 | 0.438 | 0.286 | 0.482 | 0.221 | 0.270 | 0.681 | 0.460 | 0.447 |

### 4.4. Ablation study

Experiments with/without key modules are conducted to study the effects of each module of ACAE in this section. Table 3 shows a comparison between ACAE and its five variants. The projection layer is removed in ACAE_WO_PRO, and the augmented samples are generated by directly applying the multi-scale timestamp mask to the original data. ACAE_ONE_MASK only uses the timestamp mask with the mask rate of 0.3 to generate a positive sample for each sample; ACAE_NOISE obtains the augmented views by adding the Gaussian noise with mean 0 and standard deviations 0.05, 0.15, 0.3, and 0.5 to the original data. ACAE_GRU replaces the well-designed convolutional encoder and decoder of ACAE with the GRU. ACAE_AE does not use the proposed contrastive learning framework and only trains the autoencoder to reconstruct the time series and detect anomalies.

As shown in Table 3, ACAE obtains the highest average scores on the three measures, achieving the best performance. The results show that the performance of ACAE will degrade when the key modules are removed or replaced. Although ACAE_GRU outperforms ACAE on some datasets, it is invalid on the SMAP dataset when the encoder and decoder are replaced with the GRU. When detecting anomalies only by the autoencoder, the performance of ACAE_AE deteriorates dramatically compared to the ACAE. Without the contrastive learning constraint, the autoencoder only learns the point-wise context information, making it difficult to obtain a robust representation. The ablation experiments indicate that each module in the proposed method is useful and necessary.

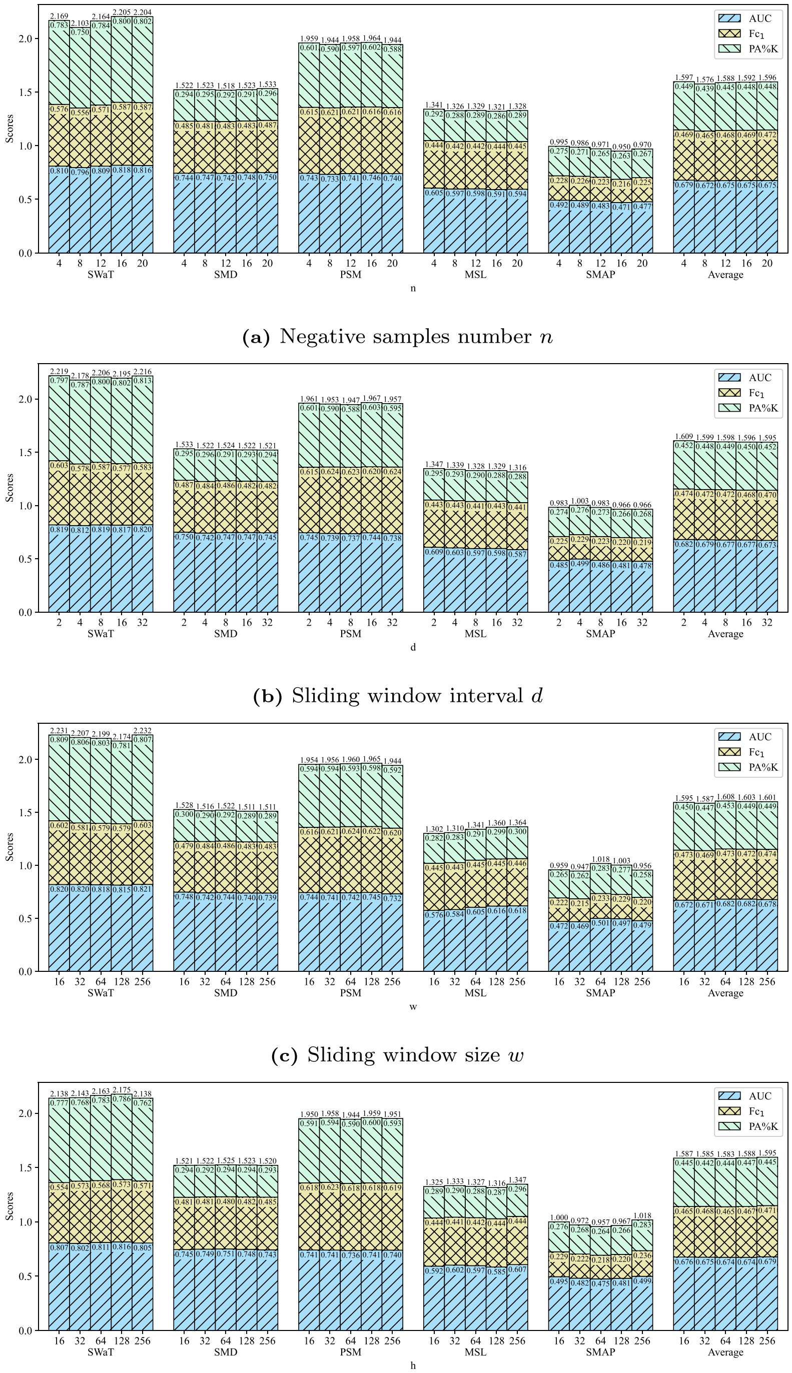
### 4.5. Case analysis

The ACAE proposed in this paper is a reconstruction-based method that learns the normal patterns of MTS and detects anomalies through anomaly scores. In order to illustrate that the proposed method ACAE learns more high-level semantic information and overall trend of MTS data compared to only using reconstruction loss, additional experiments are conducted to visualize the distributions of the latent variables and anomaly scores of ACAE and ACAE_AE. ACAE_AE is the variant of ACAE introduced in Section 4.4, which does not use the proposed contrastive learning framework and only trains the autoencoder to reconstruct the time series and detect anomalies. Specifically, the PSM, SMD, and SWaT dataset are selected as the study cases.

The latent vectors of PSM, SMD, and SWAT generated by ACAE and ACAE_AE are visualized in Fig. 8. This paper uses Stochastic Neighbor Embedding (SNE) to project the data features into a two-dimensional space to visualize the latent vectors. Each point in Fig. 8 represents a time window data latent variable. If at least one time step in the window is an anomaly, it is marked as an abnormal window. As shown in Figs. 8(a) – 8(c), the latent variables generated by ACAE of inliers present multiple clusters, among which each may represent a pattern of normal MTS data. The distribution of the abnormal data's latent variables presents two situations. First, some are similar to normal patterns, so these anomalies will be reconstructed into normal time series, resulting in large reconstruction errors. At the same time, the latent variables of the other abnormal data gather into a banded cluster and away from the normal data. Since ACAE only uses normal time series for training, faced with the inexistent pattern in the training data, the reconstructed samples will deviate from the actual situation, resulting in the anomalies being more easily detected. The results indicate that the latent variables of ACAE contain more high-level semantic information by introducing contrastive learning.

Specifically, the PSM dataset's latent variables generated by ACAE present a circular distribution. Since the PSM dataset only records the MTS data generated by a single device, the normal pattern of the PSM dataset is relatively simple. The proposed method achieves similar results for this dataset with the contrastive learning methods based on the InfoNCE loss. The latent variables of PSM generated by ACAE_AE are distributed in clusters. Compared with ACAE_AE, ACAE achieves more even distributions of latent variables in the latent space by learning the similarities and differences between normal samples. The reconstructed data of abnormal data will be closer to the normal data, making detecting anomalies easier. The SMD and SWAT datasets contain MTS data collected from multiple devices, so the latent variables of normal data present multiple clusters. For the muti-pattern datasets, traditional contrastive learning methods still map their latent variables evenly onto the hyper-sphere, which may result in the loss of useful information in the data. The proposed contrastive learning framework avoids the loss of useful information while learning the similarities and differences between normal data. Compared with ACAE_AE, ACAE achieves more obvious cluster distribution, proving that it learns more high-level semantic information of PSM data.

In addition, to validate the robustness of the anomaly scores of ACAE, the distribution of anomaly scores calculated by ACAE and



**(a)** Negative samples number $n$

**(b)** Sliding window interval $d$

**(c)** Sliding window size $w$

**(d)** Latent space dimension $h$

**Fig. 7.** Results of parameter sensitivity on the real-world datasets.

ACAE_AE of the segments shown in Figs. 1(g) – 1(i) are visualized, which contain some contextual anomalies. As shown in Fig. 9, the anomaly scores of the anomalies are significantly higher than that of the normal time steps for ACAE and ACAE_AE, illustrating that both methods can detect these anomalies. However, compared with ACAE_AE, ACAE achieves more different anomaly scores between anomalies and normal data, making detecting anomalies easier. It indicates that ACAE can robustly model the normal patterns of MTS and is less dependent on the point adjustment or threshold strategy through contrastive learning.

## 5. Conclusion

This paper proposes an adversarial contrastive autoencoder (ACAE) framework for robust MTS anomaly detection, which learns high-level semantic features of MTS better than existing methods. It introduces the contrastive learning constraint in latent space to capture the normal patterns of MTS robustly. Specifically, it obtains the augmented samples through the multi-scale timestamp mask to learn the overall trend of MTS. Moreover, it conducts feature combination and decomposition as the proxy task, which utilizes adversarial training to avoid the loss of information caused by existing frameworks.

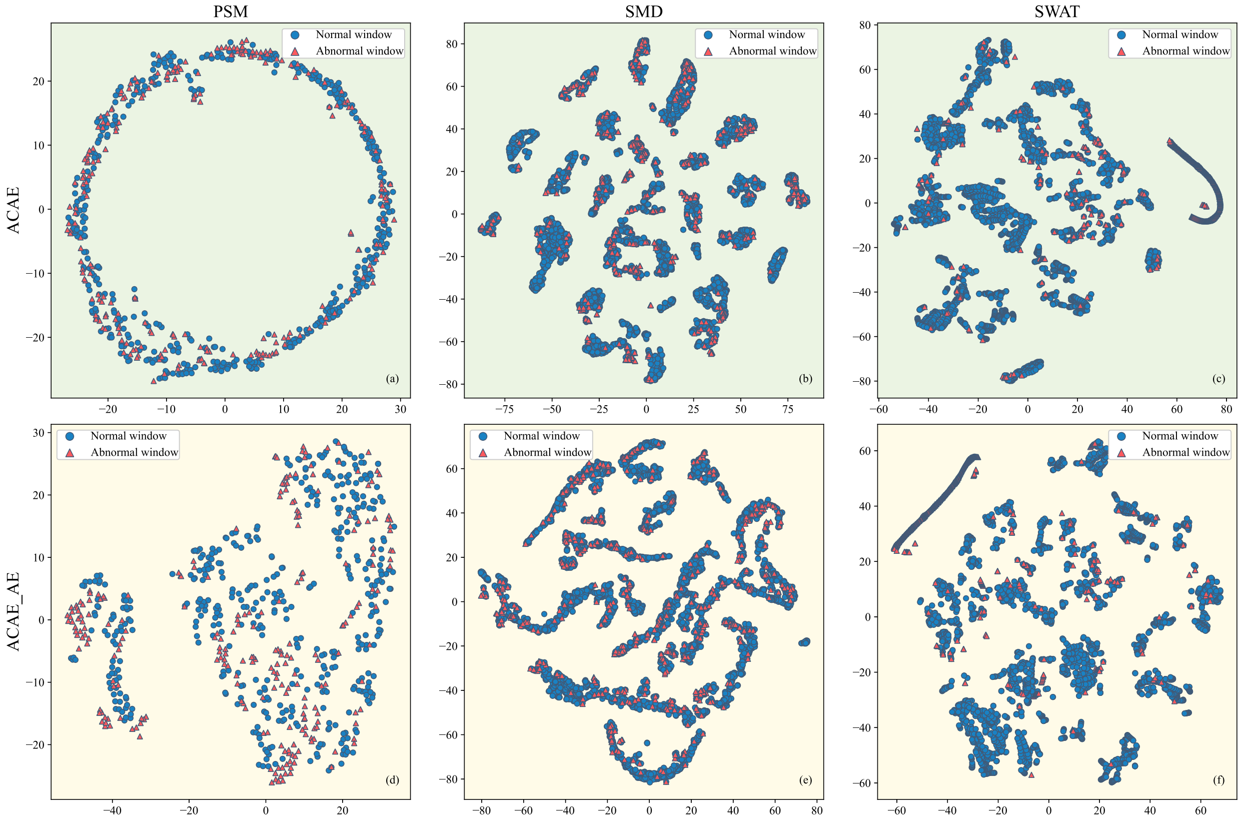


**Fig. 8.** Visualization of the latent variables of the real-world datasets.

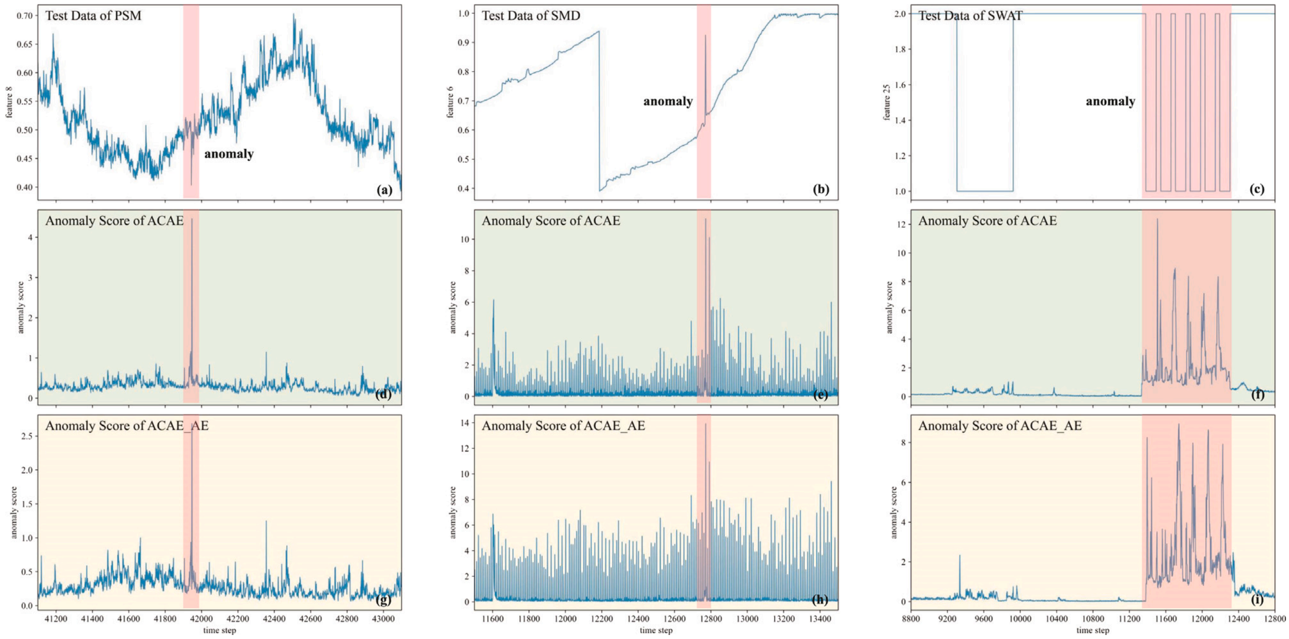


**Fig. 9.** Visualization of the anomaly scores of the real-world datasets.

Experiments were conducted on five real-world datasets from different fields, and the results empirically prove that ACAE outperforms 14 state-of-the-art baselines. In addition, experiments were carried out under different parameter settings. The results indicate that the proposed method is less affected by the hyper-parameters, ensuring its stability in practice. Besides, the results of ablation studies and case analysis also show the effectiveness and robustness of ACAE.

For future work, experiments will be conducted on more real-world data from different fields to verify ACAE's robustness further. The characteristics of MTS data will be analyzed to study the improved methods for the long tail problems in various fields and explore the dynamic threshold for anomaly detection. In addition, the proposed framework can be applied in pre-training and transfer learning, just as contrastive learning in machine vision and natural language.

## CRediT authorship contribution statement

**Jiahao Yu:** Methodology, Formal analysis, Software, Writing – original draft, Writing – review & editing. **Xin Gao:** Conceptualization, Methodology, Formal analysis, Resources, Supervision, Funding acquisition, Writing – original draft, Writing – review & editing. **Feng Zhai:** Conceptualization, Resources, Funding acquisition. **Baofeng Li:** Conceptualization, Resources, Funding acquisition. **Bing Xue:** Software, Validation. **Shiyuan Fu:** Software, Validation. **Lingli Chen:** Writing – review & editing. **Zhihang Meng:** Writing – review & editing.

## Declaration of competing interest

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

## Data availability

Data will be made available on request.

## Acknowledgments

The authors would like to thank their colleagues from the machine learning group for discussions on this paper. This work was supported by the Science & Technology Project of State Grid Corporation of China (5400-202355230A-1-1-ZN) and the High-Performance Computing Platform of BUPT.

## References

Abdulaal, A., Liu, Z., & Lancewicki, T. (2021). Practical approach to asynchronous multivariate time series anomaly detection and localization. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining* (pp. 2485–2494). http://dx.doi.org/10.1145/3447548.3467174.

Audibert, J., Michiardi, P., Guyard, F., Marti, S., & Zuluaga, M. A. (2020). Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 3395–3404). http://dx.doi.org/10.1145/3394486.3403392.

Blázquez-García, A., Conde, A., Mori, U., & Lozano, J. A. (2021). A review on outlier/anomaly detection in time series data. *ACM Computing Surveys*, *54*(3), 1–33, https://dl.acm.org/doi/10.1145/3444690.

Boukerche, A., Zheng, L., & Alfandi, O. (2020). Outlier detection: Methods, models, and classification. *ACM Computing Surveys*, *53*(3), 1–37. http://dx.doi.org/10.1145/3381028.

Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on management of data* (pp. 93–104). http://dx.doi.org/10.1145/342009.335388.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., & Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, *33*, 9912–9924.

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

Chen, Z., Chen, D., Zhang, X., Yuan, Z., & Cheng, X. (2021). Learning graph structures with transformer for multivariate time-series anomaly detection in IoT. *IEEE Internet of Things Journal*, *9*(12), 9179–9189. http://dx.doi.org/10.1109/JIOT.2021.3100509.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International conference on machine learning* (pp. 1597–1607). PMLR.

Choi, K., Yi, J., Park, C., & Yoon, S. (2021). Deep learning for anomaly detection in time-series data: review, analysis, and guidelines. *IEEE Access*, *9*, 120043–120065. http://dx.doi.org/10.1109/ACCESS.2021.3107975.

Cook, A. A., Mısırlı, G., & Fan, Z. (2019). Anomaly detection for IoT time-series data: A survey. *IEEE Internet of Things Journal*, *7*(7), 6481–6494. http://dx.doi.org/10.1109/JIOT.2019.2958185.

Davari, N., Pashami, S., Veloso, B., Nowaczyk, S., Fan, Y., Pereira, P. M., et al. (2022). A fault detection framework based on LSTM autoencoder: A case study for volvo bus data set. In *Advances in intelligent data analysis XX: 20th international symposium on intelligent data analysis, IDA 2022, Rennes, France, April 20–22, 2022, Proceedings* (pp. 39–52). Springer, http://dx.doi.org/10.1007/978-3-031-01333-1_4.

Deng, A., & Hooi, B. (2021). Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence, vol. 35, no. 5* (pp. 4027–4035). http://dx.doi.org/10.1609/aaai.v35i5.16523.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. http://dx.doi.org/10.48550/arXiv.1810.04805, arXiv preprint arXiv:1810.04805.

Eldele, E., Ragab, M., Chen, Z., Wu, M., Kwoh, C. K., Li, X., et al. (2021). Time-series representation learning via temporal and contextual contrasting. http://dx.doi.org/10.48550/arXiv.2106.14112, arXiv preprint arXiv:2106.14112.

Fang, H., Wang, S., Zhou, M., Ding, J., & Xie, P. (2020). Cert: Contrastive self-supervised learning for language understanding. http://dx.doi.org/10.48550/arXiv.2005.12766, arXiv preprint arXiv:2005.12766.

Gao, H., Qiu, B., Barroso, R. J. D., Hussain, W., Xu, Y., & Wang, X. (2022). Tsmae: a novel anomaly detection approach for internet of things time series data using memory-augmented autoencoder. *IEEE Transactions on Network Science and Engineering*, http://dx.doi.org/10.1109/TNSE.2022.3163144.

Gao, X., Yu, J., Zha, S., Fu, S., Xue, B., Ye, P., et al. (2022). An ensemble-based outlier detection method for clustered and local outliers with differential potential spread loss. *Knowledge-Based Systems*, *258*, Article 110003. http://dx.doi.org/10.1016/j.knosys.2022.110003.

Garg, A., Zhang, W., Samaran, J., Savitha, R., & Foo, C. S. (2021). An evaluation of anomaly detection and diagnosis in multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(6), 2508–2517. http://dx.doi.org/10.1109/TNNLS.2021.3105827.

Grill, J. B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, *33*, 21271–21284.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., & Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 16000–16009). http://dx.doi.org/10.1109/cvpr52688.2022.01553.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9729–9738). http://dx.doi.org/10.1109/CVPR42600.2020.00975.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hsieh, R. J., Chou, J., & Ho, C. H. (2019). Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing. In *2019 IEEE 12th conference on service-oriented computing and applications* (pp. 90–97). IEEE, http://dx.doi.org/10.1109/SOCA.2019.00021.

Hundman, K., Constantinou, V., Laporte, C., Colwell, I., & Soderstrom, T. (2018). Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 387–395). http://dx.doi.org/10.1145/3219819.3219845.

Ismail Fawaz, H., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D. F., Weber, J., et al. (2020). Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, *34*(6), 1936–1962. http://dx.doi.org/10.1007/s10618-020-00710-y.

Ji, C., Du, M., Hu, Y., Liu, S., Pan, L., & Zheng, X. (2022). Time series classification based on temporal features. *Applied Soft Computing*, *128*, Article 109494. http://dx.doi.org/10.1016/j.asoc.2022.109494.

Kieu, T., Yang, B., Guo, C., Jensen, C. S., Zhao, Y., Huang, F., et al. (2022). Robust and explainable autoencoders for unsupervised time series outlier detection—Extended version. http://dx.doi.org/10.48550/arXiv.2204.03341, arXiv preprint arXiv:2204.03341.

Kim, S., Choi, K., Choi, H. S., Lee, B., & Yoon, S. (2022). Towards a rigorous evaluation of time-series anomaly detection. In *Proceedings of the AAAI conference on artificial intelligence, vol. 36, no. 7* (pp. 7194–7201). http://dx.doi.org/10.1609/aaai.v36i7.20680.

Li, G., & Jung, J. J. (2022). Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges. *Information Fusion*, http://dx.doi.org/10.1016/j.inffus.2022.10.008.

Li, Z., Zhao, Y., Han, J., Su, Y., Jiao, R., Wen, X., et al. (2021). Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining* (pp. 3220–3230). http://dx.doi.org/10.1145/3447548.3467075.

Liu, F. T., Ting, K. M., & Zhou, Z. H. (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *6*(1), 1–39. http://dx.doi.org/10.1145/2133360.2133363.

Mathur, A. P., & Tippenhauer, N. O. (2016). SWaT: A water treatment testbed for research and training on ics security. In *2016 international workshop on cyber-physical systems for smart water networks* (pp. 31–36). IEEE, http://dx.doi.org/10.1109/CySWater.2016.7469060.

Mokoena, T., Celik, T., & Marivate, V. (2022). Why is this an anomaly? Explaining anomalies using sequential explanations. *Pattern Recognition*, *121*, Article 108227. http://dx.doi.org/10.1016/j.patcog.2021.108227.

Muhr, D., & Affenzeller, M. (2022). Outlier/anomaly detection of univariate time series: A dataset collection and benchmark. In *Big data analytics and knowledge discovery: 24th international conference, DaWaK 2022, Vienna, Austria, August 22–24, 2022, proceedings* (pp. 163–169). Springer, http://dx.doi.org/10.1007/978-3-031-12670-3_14.

Oord, A. v. d., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. http://dx.doi.org/10.48550/arXiv.1807.03748, arXiv preprint arXiv:1807.03748.

Paparrizos, J., Kang, Y., Boniol, P., Tsay, R. S., Palpanas, T., & Franklin, M. J. (2022). TSB-UAD: an end-to-end benchmark suite for univariate time-series anomaly detection. *Proceedings of the VLDB Endowment*, *15*(8), 1697–1711. http://dx.doi.org/10.14778/3529337.3529354.

Schmidl, S., Wenig, P., & Papenbrock, T. (2022). Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, *15*(9), 1779–1797. http://dx.doi.org/10.14778/3538598.3538602.

Schölkopf, B., Williamson, R. C., Smola, A., Shawe-Taylor, J., & Platt, J. (1999). Support vector method for novelty detection. *Advances in Neural Information Processing Systems*, *12*.

Shaukat, K., Alam, T. M., Luo, S., Shabbir, S., Hameed, I. A., Li, J., et al. (2021). A review of time-series anomaly detection techniques: A step to future perspectives. In *Advances in information and communication: Proceedings of the 2021 future of information and communication conference, vol. 1* (pp. 865–877). Springer, http://dx.doi.org/10.1007/978-3-030-73100-7_60.

Shen, L., Yu, Z., Ma, Q., & Kwok, J. T. (2021). Time series anomaly detection with multiresolution ensemble decoding. In *Proceedings of the AAAI conference on artificial intelligence, vol. 35, no. 11* (pp. 9567–9575). http://dx.doi.org/10.1609/aaai.v35i11.17152.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., & Pei, D. (2019). Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2828–2837). http://dx.doi.org/10.1145/3292500.3330672.

Tayeh, T., Aburakhia, S., Myers, R., & Shami, A. (2022). An attention-based ConvLSTM autoencoder with dynamic thresholding for unsupervised anomaly detection in multivariate time series. *Machine Learning and Knowledge Extraction*, *4*(2), 350–370. http://dx.doi.org/10.3390/make4020015.

Tuli, S., Casale, G., & Jennings, N. R. (2022). Tranad: Deep transformer networks for anomaly detection in multivariate time series data. http://dx.doi.org/10.48550/arXiv.2201.07284, arXiv preprint arXiv:2201.07284.

Wang, Y., Du, X., Lu, Z., Duan, Q., & Wu, J. (2022). Improved lstm-based time-series anomaly detection in rail transit operation environments. *IEEE Transactions on Industrial Informatics*, *18*(12), 9027–9036. http://dx.doi.org/10.1109/TII.2022.3164087.

Wang, S., Zeng, Y., Liu, X., Zhu, E., Yin, J., Xu, C., et al. (2019). Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. *Advances in Neural Information Processing Systems*, *32*.

Wu, R., & Keogh, E. (2021). Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*, http://dx.doi.org/10.1109/TKDE.2021.3112126.

Wu, Z., Xiong, Y., Yu, S. X., & Lin, D. (2018). Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3733–3742). http://dx.doi.org/10.1109/cvpr.2018.00393.

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., et al. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference* (pp. 187–196). http://dx.doi.org/10.1145/3178876.3185996.

Xu, J., Wu, H., Wang, J., & Long, M. (2021). Anomaly transformer: Time series anomaly detection with association discrepancy. http://dx.doi.org/10.48550/arXiv.2110.02642, arXiv preprint arXiv:2110.02642.

Yang, X., Zhang, Z., & Cui, R. (2022). TimeCLR: A self-supervised contrastive learning framework for univariate time series representation. *Knowledge-Based Systems*, *245*, Article 108606. http://dx.doi.org/10.1016/j.knosys.2022.108606.

Yue, Z., Wang, Y., Duan, J., Yang, T., Huang, C., Tong, Y., et al. (2022). Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence, vol. 36, no. 8* (pp. 8980–8987). http://dx.doi.org/10.1609/aaai.v36i8.20881.

Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., et al. (2019). A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI conference on artificial intelligence, vol. 33, no. 01* (pp. 1409–1416). http://dx.doi.org/10.1609/aaai.v33i01.33011409.

Zhou, B., Liu, S., Hooi, B., Cheng, X., & Ye, J. (2019). BeatGAN: Anomalous rhythm detection using adversarially generated time series.. In *IJCAI, vol. 2019* (pp. 4433–4439). http://dx.doi.org/10.24963/ijcai.2019/616.

Zhou, Y., Song, Y., & Qian, M. (2021). Unsupervised anomaly detection approach for multivariate time series. In *2021 IEEE 21st international conference on software quality, reliability and security companion* (pp. 229–235). IEEE, http://dx.doi.org/10.1109/QRS-C55045.2021.00042.

Zhou, H., Yu, K., Zhang, X., Wu, G., & Yazidi, A. (2022). Contrastive autoencoder for anomaly detection in multivariate time series. *Information Sciences*, *610*, 266–280. http://dx.doi.org/10.1016/j.ins.2022.07.179.

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., et al. (2018). Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*.