applied sciences

Article

TGAN-AD: Transformer-Based GAN for Anomaly Detection of Time Series Data

Liyan Xu [1,2], Kang Xu [1,2,3,*], Yinchuan Qin [3], Yixuan Li [3], Xingting Huang [4], Zhicheng Lin [3], Ning Ye [3] and Xuechun Ji [1,2]

1 State Key Laboratory of Smart Grid Protection and Control, Nanjing 211106, China
2 Nari Group Corporation, Nanjing 211106, China
3 School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
4 Bell Honros School, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
* Correspondence: kxu@njupt.edu.cn; Tel.: +86-158-5185-0652

Abstract: Anomaly detection on time series data has been successfully used in power grid operation and maintenance, flow detection, fault diagnosis, and other applications. However, anomalies in time series often lack strict definitions and labels, and existing methods often suffer from the need for rigid hypotheses, the inability to handle high-dimensional data, and highly time-consuming calculation costs. Generative Adversarial Networks (GANs) can learn the distribution pattern of normal data, detecting anomalies by comparing the reconstructed normal data with the original data. However, it is difficult for GANs to extract contextual information from time series data. In this paper, we propose a new method, **T**ransformer-based **GAN** for **A**nomaly **D**etection of Time Series Data (TGAN-AD), The transformer-based generators of TGAN-AD can extract contextual features of time series data to prompt the performance. TGAN-AD's discriminator can also assist in determining abnormal data. Anomaly scores are calculated through both the generator and the discriminator. We have conducted comprehensive experiments on three public datasets. Experimental results show that our TGAN-AD has better performance in anomaly detection than the state-of-the-art anomaly detection techniques, with the highest *Recall* and *F*1 values on all datasets. Our experiments also demonstrate the high efficiency of the model and the optimal choice of hyperparameters.

Keywords: anomaly detection; transformer; generative adversarial network; time series data

check for updates

Citation: Xu, L.; Xu, K.; Qin, Y.; Li, Y.; Huang, X.; Lin, Z.; Ye, N.; Ji, X. TGAN-AD: Transformer-Based GAN for Anomaly Detection of Time Series Data. *Appl. Sci.* **2022**, *12*, 8085. https://doi.org/10.3390/app12168085

Academic Editor: Antonio Sarasa Cabezuelo

Received: 26 June 2022
Accepted: 11 August 2022
Published: 12 August 2022

1. Introduction

Anomaly detection aims to detect data points or fragments that do not conform to expected behavior patterns in rapidly changing data. In different application fields, these non-compliant patterns are also called anomalies, outliers, discordant observations, and exceptions. The technology for anomaly detection is commonly used for commercial applications by major industries, such as power, finance, network security, industrial fields, system health monitoring, e-commerce fields, and ecological disaster monitoring [1]. Moreover, anomaly detection plays an elementary and important role in the Artificial Intelligence for IT Operations [2] (AIOps) system, which provides the basis for decision making in the subsequent alarms, automatic stop loss, and root cause analysis.

Time series data are the typical data type in the system of IT operations, especially in the scenario of anomaly detection in AIOps [3]. Common anomalous data in time series data can be divided into three categories [4]: point anomalies, contextual anomalies, and collective anomalies. Due to the multivariate heterogeneity of operation and maintenance data, it is a challenge to improve the accuracy of detection by automatically analyzing and summarizing abnormal patterns in the data. In the existing methods, statistical methods based on traditional thresholds need to assume that the data must follow a certain form of distribution and cannot work well with increasingly dynamic and diverse data. In addition, the statistical methods have difficulty identifying anomalies in the time series since they have limited ability to extract contextual information from the time series data [5]. Due to the increasing volume and complexity of data streams, researchers try to use machine learning methods to process large amounts of data. However, data annotation and abnormal data collection are time-consuming. Unsupervised machine learning methods for anomaly detection provide a solution that can get rid of the heavy manual cost. Most unsupervised anomaly detection methods [6,7] monitor time series data by predicting and reconstructing the time series and calculating the deviation between the true value and the predicted value. Our model also detects anomalies by reconstruction and comparison and can better learn the distribution of normal data. In addition, we use discriminators to determine anomalies directly.

Deep learning methods of anomaly detection can automatically learn the complex correlation of time series without complicated feature engineering. Hence, deep learning methods are commonly used in the task of anomaly detection for time series data. Generative Adversarial Networks (GANs) [8] are a type of typical deep learning model that has achieved great success in image processing tasks. Moreover, GANs have also been proven to be very successful in anomaly detection [9]. However, the classical GANs are weak to capture the complex contextual features of the time series data with the existing generators and discriminators. The deep learning models for sequential data, such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) and Gate Recurrent Unit (GRU) [10], can be used in the generators and discriminators to capture the implicit relationship of the time series data, but they cannot work in a parallel way because of their sequential dependence. Transformer-based models [11] have supreme performance in the forecasting of time series data, since they can learn complex patterns from time series data in a parallel way using the self-attention mechanism.

Hence, in this work, we propose a novel model, Transformer-based GAN for Anomaly Detection of time series data(TGAN-AD), which learns the anomaly patterns of time series data with Transformer in an adversarial way. Our method involves three main components: (1) a generator component, which uses the Transformer to simulate the normal patterns of time series data; (2) a discriminator component, which can capture the intrinsic characteristics of the real-time series data to learn the boundary between normal patterns and anomalous patterns; and (3) an anomaly detection, which can identify the anomaly data with the trained Generator and Discriminator components.

The main contributions of our work are summarized as follows:

- We propose a new model, TGAN-AD, that incorporates Transformer into the GAN framework. TGAN-AD makes use of Transformer to capture the contextual information of the time series data for the subsequent GAN framework.
- We use three public datasets and six baseline methods to comprehensively evaluate TGAN-AD anomaly detection performance. Compared with the baseline methods, TGAN-AD showed the best performance. We further provide some insights on the use of GANs for anomaly detection on time series data.

2. Related Work

In most practical scenarios, the labels for anomaly detection are missing, or the labels for anomalous data and normal data are unreliable. As we know, the model cannot function properly without accurate labeled data, and high-quality annotation is time-consuming. Hence, the intractable situation of the task of anomaly detection prevents us from using supervised machine learning methods, and the unsupervised method of anomaly detection provides a solution to alleviate the problems. Unsupervised anomaly detection methods can be divided into three categories: statistic-based methods, clustering-based methods, and deep learning methods.

2.1. Statistic-Based Methods

Statistical anomaly detection techniques are based on the following key assumptions. Normal data instances occur in high probability regions of a stochastic model, while anomalies occur in low probability regions of the stochastic model [12]. Based on the assumed distribution, the statistical methods can be further classified as follows:

Gaussian-based models: They assume that the data are generated by a Gaussian distribution. The distance between a data instance and the expected value of a Gaussian distribution is the anomaly score of the given instance. A threshold is set over the anomaly score to discriminate between normal and anomalous data. For example, a simple anomaly detection technique, three-sigma rule of thumb is to declare all data instances that are more than 3σ distance away from the distribution mean μ, where σ is the standard deviation for the distribution. The $\mu \pm 3\sigma$ region contains 99.7% of the data instances. This technique is mostly applicable to univariate and multivariate continuous data. A box plot is a way of summarizing data measured on an interval scale and is often used for exploratory data analysis. The box plot rule has been applied to detect outliers in univariate and multivariate medical data by Laurikkala et al. [13]. However, these methods are mostly used for monovariate time series and can generally only handle local outliers with simple distributions and large anomalous deviations.

Regression-based Models: There are two main steps in anomaly detection technology: Firstly, the regression model is fitted to the data. Then, for each test case, the residual is used to determine the exception score. For example, Eduardo et al. [14] proposed the method of traffic characterization and detection of traffic anomalies using sFlow analysis by incorporating two different models, Autoregressive Integrated Moving Average model (ARIMA) and Holt-Winters, into a behavior-based system. Another variant that detects anomalies in multivariate time series data generated by an Autoregressive Moving Average (ARMA) model was proposed by Galeano et al. [15]. In this technique, the authors transform the multivariate time series into a univariate time series by linearly combining the components of the multivariate time series.

The disadvantages of statistical techniques are that they rely on the assumption that the data are generated from a particular distribution. This assumption often does not hold true, especially for high-dimensional real datasets.

2.2. Clustering-Based Methods

Clustering is mainly an unsupervised technology. Although clustering and anomaly detection seem to be fundamentally different, several anomaly detection technologies based on clustering have been developed. These methods are based on the following assumption. Normal data instances are closer to the nearest cluster core, while abnormal data instances are farther from the nearest cluster core. When using clustering algorithms to cluster data, for each data instance, the distance from the nearest cluster centroid is calculated as its anomaly score. For example, Smith et al. [16] proposed Self-Organizing Maps (SOM), K-means Clustering, and Expectation Maximization to cluster training data and then use the clusters to classify test data.

However, if the exceptions in the data themselves form a cluster, the abovementioned methods will not be able to detect such exceptions. To solve this problem, people propose a density-based anomaly detection method. For example, Breunig et al. [17] assigned an anomaly score to a given data instance, known as the Local Outlier Factor (LOF). For any given data instance, the LOF score is equal to the ratio of the average local density of the k nearest neighbors of the instance and the local density of the data instance itself. The anomalous instance will obtain a higher LOF score. Mahoney et al. [18] proposed the CLAD algorithm, which obtains the width from the data by randomly sampling and calculating the average distance between the nearest points. All those clusters whose density is lower than a threshold are declared as “local” outliers, while all those clusters that are far away from other clusters are declared as “global” outliers. He et al. [19] proposed the FindCBLOF algorithm, which assigns an anomaly score to each data instance, called the clustering-based local outlier factor (CBLOF). The CBLOF score captures the size of the cluster to which the data instance belongs and the distance between the data instance and its cluster centroid.

Such techniques can often be adapted to other complex data types by simply plugging in a clustering algorithm that can handle the particular data type. However, these methods are unable to capture temporal correlations.

2.3. Deep Learning Methods

Among them, the method based on deep learning can better represent the hidden information in the dataset and has a better effect on anomaly detection, which attracts more researchers to study the research topic [20,21]. For example, the most commonly used deep learning method for anomaly detection is the AutoEncoder [22]. However, the model does not have a strong regularization method, which makes it easy to overfit. When there are many abnormal points, it will learn abnormal patterns. There is also an optimized variational autoencoder (VAE) [23] proposed by An et al. Unlike AutoEncoder, VAE can learn the distribution of hidden variables generated in the data, so it can have a function similar to regularization to prevent overfitting. They both use the reconstruction error between the encoder and the decoder for anomaly detection. However, they still cannot capture the time correlation and some hidden behaviors of the time series from the multivariate time series. Our model uses the more powerful GAN-based model as the reconstruction framework.

In deep-learning-based methods, generative adversarial networks have great potential applications, and they have shown excellent results for images, text, and time series. Time series anomaly detection methods based on Generative Adversarial Networks currently have some research, such as MAD-GAN, TAno-GAN, Tad-GAN [24–26], and so on. Among the three methods mentioned above, MAD-GAN, TAnoGAN, and TadGAN are all reconstruction-based deep learning methods. Their core idea is to learn a model that can encode data points and then decode the encoded vector (that is, to reconstruct the sequence for a period of time). The effective model after training cannot reconstruct the anomaly; the anomaly will lose information in the encoding process for its low frequency of occurrence. Moreover, to capture the time correlation and some timing-hiding behaviors in the time series, MAD-GAN, TAnoGAN, and TadGAN all use LSTM, which cannot run in a parallel way, as a generator and discriminator model to process time series data. Our model enhances the performance of GANs to learn sequence-to-context correlations more efficiently.

Recently, Transformer [11] has been successfully applied to the NLP field, and its major success in the NLP field demonstrates its powerful modeling capabilities for time series data. Some related works on the construction of time series data on the basis of Transformer have been published. Shaw et al. [27] proposed the concept of relative position coding, so that Transformer can adapt to sequences of different lengths. Dai et al. [28] proposed Transformer-XL, introducing the segment-level recurrence mechanism to establish a connection between each text segment (segments) so that the model can capture more distant dependencies. Dehghani et al. [29] proposed the Universal-Transformer, which introduced time step, time and position coding, and replaced the feed-forward layer with the Transition function, which improved the versatility of the Transformer. Wu et al. [30] proposed the use of a self-attention mechanism to learn complex patterns and dynamics from time series data, which can be applied to univariate and multivariate time series data. Our model is very close to the way this method learns. Wu et al. [31] proposed a new time series prediction model—Adversarial Sparse Transformer (AST). AST uses the Sparse Transformer as a generator to learn sparse attention maps for time series prediction and uses a discriminator to improve prediction performance at the sequence level. Our model also uses discriminators to aid in anomaly detection. Zhou et al. [32] proposed a way of designing an efficient structure suitable for long-term time series forecasting (LSTF) based on Transformer. Li et al. [33] proposed the LogSparse Self Attention structure to reduce the amount of calculation to $\mathrm{O}\left(\mathrm{L}(\log \mathrm{L})^2\right)$.

In short, Transformer has shown obvious advantages in tasks such as prediction of time series data, which provides a practical basis for the incorporation of Transformer into anomaly detection in our work.

3. Background

3.1. Transformer

Transformer [11] can effectively capture the long-term correlation within the sequence and realize the end-to-end generation of input to output. It is formed by stacking an encoder and a decoder based on the self-attention mechanism. The encoder compresses the input sequence to generate semantic coding and self-attention matrix. Based on the self-attention matrix, the decoder focuses attention on the effective information of the input sequence, and finally generates the target vector through the residual network and the feed-forward neural network. The input to the transformer is a multidimensional matrix, which can be a multivariate time series under a sliding window. The self-attention mechanism can analyze the context of the sequence, and the encoding–decoding framework uses the encoder output from one modality as the input of the other modality decoder, realizing the conversion between modalities, as shown in Figure 1.

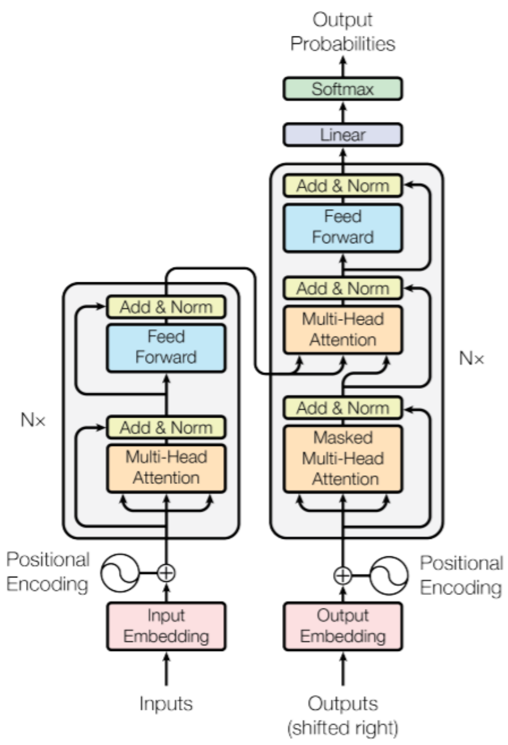


Figure 1. Transformer model proposed in [11].

The attention calculation for all tokens can be expressed as one large matrix calculation using the softmax function, which is useful for training due to computational matrix operation optimizations that quickly compute matrix operations. The matrices Q, K, and V are defined as the matrices where the i-th rows are vectors q_i, k_i, and v_i respectively. These three variable matrices are obtained by projecting the input data through three matrix calculations.

$$\text{Attention}\ (Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

Intuitively, the Q and K matrices are the operators used to calculate the similarity of the current element in the sequence to the other elements. The V matrix contains the information contained in the elements themselves. By calculating between these three variables, each element in the sequence interacts with elements in other positions, and the output can make full use of contextual information. The parallel operation of the self-attention mechanism makes the distance between any word pairs in a sequence as one and can capture long-range contextual information. It can improve the computational efficiency of the model and solve the problems of long-term dependence and excessive network capacity consumption caused by the serial operation mechanism, where each hidden coding of a token must depend on the coding output of the previous token.

3.2. Generative Adversarial Networks

The method of anomaly detection based on the GAN [8], as shown in Figure 2, is to learn the normal behavior in the data through the idea of adversarial training and learn the patterns of the normal data from the training data with the discriminator and generator of GAN. If the test data are consistent or similar to the generated data, it indicates normal data, otherwise they are abnormal data. The generator G generates fake samples, and then the discriminator D is used to distinguish true samples from fake samples. During the training process, the generator G and the discriminator D perform adversarial training, and finally, the generator G can generate fake samples, which can deceive the discriminator.

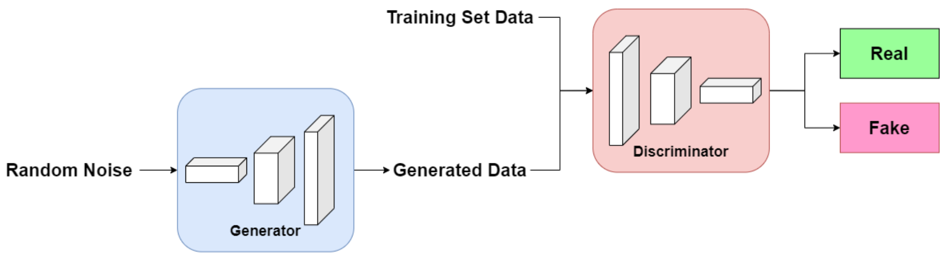


Figure 2. Generative adversarial networks.

4. Proposed Model

4.1. Problem Definition

Anomaly detection of time series data is the process of identifying abnormal events or behaviors from normal time series. Time series data are divided into the training data $x_{\text{train}}^{n\times m}$ and test data $x_{\text{test}}^{k\times m}$, where n and k are the maximum length of the timestamp, and m is the feature dimension of each time series data. Our goal is to build an unsupervised model that can efficiently capture the non-linear pattern and multivariate distribution of multivariate time series, accurately discriminate the abnormal data in the test data $x_{\text{test}}^{k\times m}$, and generate a probability vector $y \subseteq R^k$, where $y_i \in \{0, 1\}$ indicates whether the i-th timestamp is abnormal or not. Our model is based on a generative adversarial network, which can conduct anomaly detection with its discriminator and generator, and Transformer [11], which uses a multi-head attention mechanism to capture feature correlation and time dependence in time series.

4.2. TGAN-AD Architecture

As shown in Figure 3, TGAN-AD uses the Transformer as the generator and discriminator of the GAN framework to capture the temporal correlation and other hidden behaviors in the multivariate time series. To improve the efficiency of our framework, the multivariate time series data are divided into sub-sequences using sliding windows during the pre-processing, as shown in Figure 4, and the window size t_w is set empirically.

The real time series X and the random hidden variables Z are both input into the generator (G) to generate a fake time series $G(X, Z)$: $X, Z \rightarrow G(X, Z)$. Then the fake time series and the real time series are input into the discriminator (D), where the discriminator learns the parameters to distinguish real and fake time series data. The parameters of G and D are both updated according to the output of D, and G generates fake samples approaching the existing normal samples X. Moreover, D can improve its discrimination ability and distinguish fake time series $G(X, Z)$ and normal time series X. By iterative training, D's discrimination ability can accurately distinguish normal sequence X and fake sequence $G(X, Z)$. Then G can capture the hidden time correlation in the normal time series X and generate a fake time series $G(X, Z)$ that can deceive D.

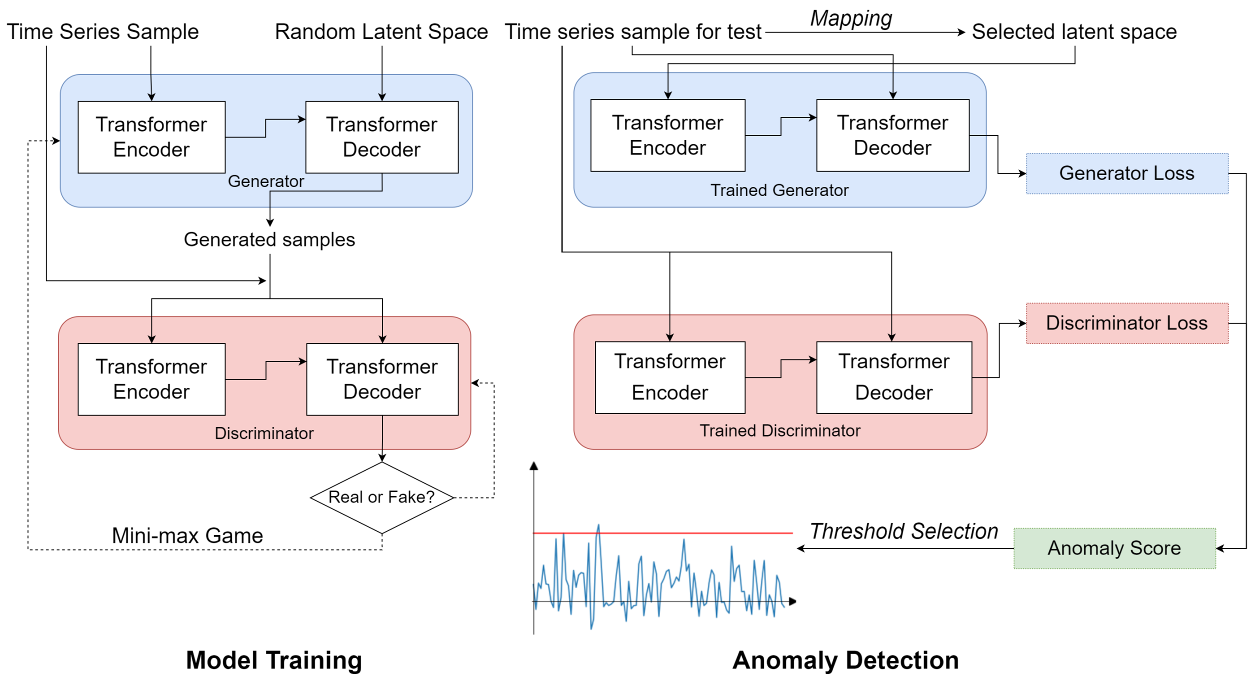
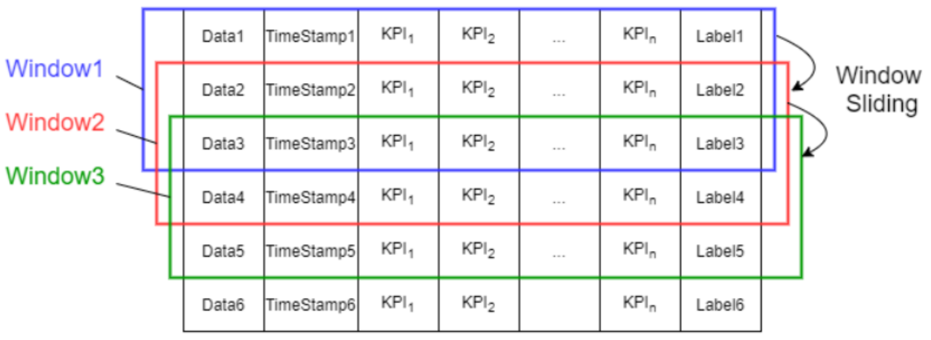


Figure 3. The overall framework of TGAN-AD. The left part is the offline training process of TGAN-AD, and the right part is the process of online anomaly detection. Both the generator and discriminator are constructed based on Transformer.



Data1	TimeStamp1	KPI_1	KPI_2	...	KPI_n	Label1
Data2	TimeStamp2	KPI_1	KPI_2	...	KPI_n	Label2
Data3	TimeStamp3	KPI_1	KPI_2	...	KPI_n	Label3
Data4	TimeStamp4	KPI_1	KPI_2	...	KPI_n	Label4
Data5	TimeStamp5	KPI_1	KPI_2	...	KPI_n	Label5
Data6	TimeStamp6	KPI_1	KPI_2		KPI_n	Label6

Figure 4. Multivariate time series sliding window.

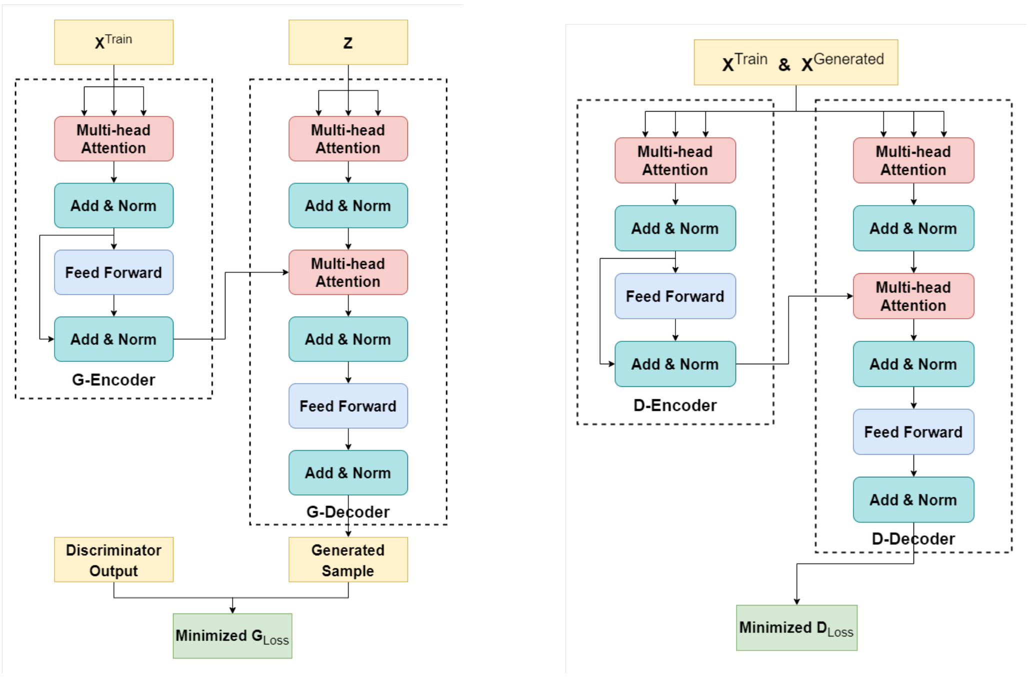
After the offline training, all the parameters of G and D are fixed. The test time series X^{test} is encoded as the hidden space. The optimal Z^{test} is trained by gradient descent and is input into G to reconstruct the test sample: $X^{\text{test}}, Z^{\text{test}} \rightarrow G(X^{\text{test}}, Z^{\text{test}})$. The reconstruction loss G_{loss} is calculated by the deviation of X^{test} and $G(X^{\text{test}}, Z^{\text{test}})$. Meanwhile, the test time series X^{test} is input into the trained D to calculate the discrimination loss D_{loss}. Finally, the anomaly score $(AD - Score)$ of the test data is calculated with two kinds of losses: $(G_{loss}, D_{loss}) \rightarrow AD - Score$. Then the anomaly state of the test data is determined according to the anomaly score.

4.3. Transformer Components for TGAN-AD

The generator and discriminator of our model are trained in an adversarial way, and the core part is the encoder–decoder module based on Transformer. TGAN-AD contains two core components:

4.3.1. Generator Training Process

As shown in Figure 5a, X^{train} is input into the Transformer encoder to learn the hidden representation of the normal time series, which can help the generators generate the data approaching the real time series data. The hidden representation of X^{train} is fed into the Transformer decoder. To generate rich similar samples, the hidden space Z is the input of the Transformer decoder. The fake samples are generated based on the hidden representation of X^{train} and Z with Transformer.



(a) Generator (b) Discriminator

Figure 5. Generator and discriminator of TGAN-AD.

The goal of the generator is to continuously train the transformer to generate time series data approaching the real time series data, which makes it difficult for the discriminator to distinguish the generated data from the training data.

4.3.2. Transformer-Based Discriminator

As shown in Figure 5b, a sample, from the training data or generated data, is input into the Transformer encoder to obtain the hidden representation of the sample; then the sample and its representation are sent to the decoder. The class distribution of the sample is output by the Transformer decoder.

The goal of the discriminator is to accurately distinguish the generated fake data from the training data so that the model can be well trained by the *mini-max* game.

4.4. Detection of Anomalies

The training dataset $x_{\text{train}}^{A\times C}$, test time series $x_{\text{test}}^{B\times C}$, A and B are the maximum length of the timestamp, and C is the feature dimension. To better capture the hidden information in the time series, a sliding window is used to divide the multivariate time series into sub-sequences. That is, $X^{\text{train}} = \{\text{x}_1, \text{x}_2, \text{x}_3, \ldots, \text{x}_\text{a}\}$, $a = (A - t_w) + 1$. $X^{\text{test}} = \{\text{x}_1, \text{x}_2, \text{x}_3, \ldots, \text{x}_\text{b}\}$, $b = (B - t_w) + 1$. In the same way, Z is the random hidden space,

$Z = \{z_1, z_2, z_3, \ldots, z_a\}$. Then put X and Z into TGAN-AD; the TGAN-AD model trains the generator and discriminator in *mini-max* games:

$$\min_G \max_D V(G, D) = \varepsilon_{z \sim p_z(x,z)}[\log(1 - D(G(x,z)))] + \varepsilon_{x \sim p_x(x)}[\log D(x)] \tag{2}$$

After many iterations, the fake samples generated by the TGAN-AD generator can fool the discriminator, indicating that the training is completed. Then the reconstruction loss and discrimination loss of the test sample are calculated to obtain the AD-Score by using the sum of the training.

4.4.1. Discrimination Loss

The discriminator has the ability to identify whether the input sample is anomalous data so discrimination loss can be used as part of the AD-Score. For the discrimination loss, the test time series are input into the trained discriminator, and TGAN-AD directly outputs the discrimination loss. Intuitively, the discrimination loss represents the probability that the input sample is anomalous.

$$D_{\text{loss}} = D_{\text{Trans}}\left(X^{\text{test}}\right) \tag{3}$$

4.4.2. Reconstruction Loss

For reconstruction loss, we firstly search the optimal Z^{test} of the test dataset in the latent space, which can generate the most similar generated sequence in G_{Trans}, i.e., $G_{\text{Trans}}\left(X^{\text{test}}, Z^{\text{test}}\right)$. TGAN-AD uses covariance as a reference to update $G_{\text{Trans}}\left(X^{\text{test}}, Z^{\text{test}}\right)$, that is, the similarity between the generated sequence and the test sequence, and the gradient descent method can also be used to find the optimal sequence.

$$\operatorname{cov}\left(X^{\text{test}}, G_{\text{Trans}}\left(X^{\text{test}}, Z^{\text{test}}\right)\right) = \frac{\sum_{i=1}^{b}(x_i - \bar{x})(G_i - \bar{G})}{b-1} \tag{4}$$

$$\min_{z^{\text{test}}} E\left(X^{\text{test}}, G_{\text{Trans}}\left(X^{\text{test}}, Z^{\text{test}}\right)\right) = 1 - \operatorname{cov}\left(X^{\text{test}}, G_{\text{Trans}}\left(X^{\text{test}}, Z^{\text{test}}\right)\right) \tag{5}$$

After finding an optimal Z^{test}, we calculate the reconstruction loss:

$$G_{\text{loss}} = \sum_{i=1}^{c}\sum_{j=1}^{t_w}\left|X_{i,j}^{\text{test}} - G_{\text{Trans}}\left(X^{\text{test}}, Z^{\text{test}}\right)_{i,j}\right| \tag{6}$$

Since the generator learns the distribution pattern of the normal data, the degree of dissimilarity between the two samples can be calculated by comparing the original sample with the reconstructed sample. That is, the difference between the abnormal data and the normal data can be obtained.

4.4.3. Anomaly Detection Score

The anomaly detection is determined by the two losses above. Calculate the anomaly detection loss based on D_{loss} and G_{loss}, AD-Score:

$$ADS = \alpha G_{\text{loss}} + (1 - \alpha) D_{\text{loss}} \tag{7}$$

Moreover, α represents the weight of reconstruction loss and discrimination loss, and α can be set empirically. Then mark the X^{test} timestamp according to the obtained ADS: if $ADS > \tau$, $\text{Label}^{\text{test}} = 1$ which represents abnormal data, otherwise $\text{Label}^{\text{test}} = 0$ which represents normal data.

The overall algorithm is summarized in Algorithm 1.

Algorithm 1 Transformer-GAN Anomaly Detection Algorithm

```
epoch $\Leftarrow 0$, $G_{Trans}, D_{Trans} \Leftarrow$ initialize network parameters
if epoch within number of training iterations then
  for the $k^{th}$ epoch do
    Generate samples from the random space:
    $X = \{x_i, i = 1, 2, ..., n\}$
    $Z = \{z_i, i = 1, 2, ..., n\}$
    $\Rightarrow G_{Trans}(X, Z)$
    discrimination:
    $X = \{x_i, i = 1, 2, ..., n\} \Rightarrow D_{Trans}(X)$
    $G_{Trans}(X, Z) \Rightarrow D_{Trans}(G_{Trans}(X, Z))$
    Update Transformer discriminator parameters:
    $\min \frac{1}{n} \sum_{i=1}^{n} [-\log D_{Trans}(x_i) - \log(1 - D_{Trans}(G_{Trans}(x_i, z_i)))]$
    Update Transformer generator parameters:
    $\min \sum_{i=1}^{n} \log(-D_{Trans}(G_{Trans}(x_i, z_i)))$
    Record parameters in the current iteration
  end for
end if
for number of iterations do
  Find the best generated sample:
  $Z^k = \min_z \mathrm{E}(X^{\text{test}}, G_{Trans}(x^{text}, z^i))$
end for
Reconstruction loss:
$G_{loss} = \left| X^{test} - G_{\text{Trans}} \left( X^{test}, Z^k \right) \right|$
Discriminate loss:
$D_{loss} = D_{Trans}(X^{test})$
Anomaly Detection Score:
$ADS = \alpha G_{loss} + (1 - \alpha) D_{loss}$
```

5. Datasets

To evaluate the performance of TGAN-AD, we evaluated it on three commonly used multivariate time series datasets, i.e., two cyber-attacks datasets, SWaT (https://mlad.kaspersky.com/swat-testbed/, accessed on 10 August 2022) and WADI (https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/, accessed on 10 August 2022), and a network intrusion detection dataset KDDCup99 (http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, accessed on 10 August 2022) based on data mining. Each dataset is public and of high quality.

5.1. Secure Water Treatment (SWaT)

Secure Water Treatment (SWaT) is a water treatment tested for research in the area of cybersecurity. The Swat dataset contains a total of 264 h of numerical data and network traffic data collected from 51 sensors and processors for 11 consecutive days. It includes 7 days of normal data obtained when the system was under normal operation and operation without being attacked, and 4 days of abnormal data obtained when the system was attacked in different scenarios.

5.2. Water Distribution (WADI)

As an extension of SWaT, Water Distribution (WADI) is a distribution system comprising a larger number of water distribution pipelines. WADI is more vulnerable than the SWaT dataset and has more features than SWaT. It collects data for 16 consecutive days from networks, sensors, and actuators. WADI collected 14 days of normal data and 2 days of abnormal data. The abnormal data contains 15 attacks from the same attack model. The abnormal ratio is also lower than other datasets so that the WADI is more unbalanced.

5.3. KDD Cup 1999

The KDD99 dataset is the dataset from the Third International Knowledge Discovery and Data Mining Tools Competition in 1999. The requirement of the competition was to design a network intrusion detector to detect if the network connection was under attack or intrusion. Each network connection is marked as "normal" or "attack". There are 39 types of abnormalities, 22 of which occur in the training set, and the remaining 17 occur only in the test set. Table 1 shows the information about the datasets.

Table 1. Statistics of the three datasets used in experiments.

Name	Type	Number	Features	Number of Abnormalities	Abnormal Ratio
SWaT	Train	496,800	51	0	0.00%
	Test	449,919	51	54,621	12.14%
WADI	Train	784,571	123	0	0.00%
	Test	172,801	123	9977	5.77%
KDDCUP99	Train	562,387	34	0	0.00%
	Test	494,021	34	97,311	19.69%

6. Experiment

To demonstrate the performance of the proposed model, the following two problems need to be experimentally verified:

- **Q1**: Does the proposed model perform better than the baseline methods in key metrics, especially *Recall* and *F1-Score*?
- **Q2**: How can we determine the most appropriate hyperparameter settings for the model in real-world engineering?

6.1. Data Preprocessing

To better capture the time correlation and other hidden behaviors of the time series, the dataset was divided into sub-sequences. To determine the optimal sub-sequence length, experiments were carried out with different window sizes. The initial subsequence length was set at 10 empirically, i.e., $t_w \in \{10 \times \mathrm{i}, \mathrm{i} = 1, 2, 3 \ldots, 10\}$.

6.2. Baselines

We compared the performance of TGAN-AD with four popular anomaly detection methods, including: **PCA**: The method is based on Principal Component Analysis [34]; **Random Forest**: The method is based on a completely random forest [35]; **LSTM**:The method is based on a Long Short-Term Memory Neural Network [36]; **FNN**:The method is based on a Feed-forward Neural Network [37]; **MAD-GAN**: The method is based on Generative Adversarial Networks [24]; **GDN**: The method is based on Graph Neural Networks [38].

6.3. Evaluation Metric

Five kinds of metrics were used in our work: *Precision*, *Recall*, *F1-Score*, *AUC*, and *PRAUC*, to evaluate the anomaly detection performance. *Precision* describes the proportion of positive examples predicted by the classifier that are real positive examples. *Recall* describes the proportion of real positive examples in the test set that have been selected by the classifier. The *Precision* and *Recall* can provide a diagnostic tool for binary classification models. Moreover, the *ROC* curve (*AUC*) and the *Precision-Recall* curve (*PRAUC*) are used to evaluate the performance of anomaly detection. *AUC* and *PRAUC* can be optimistic about severely imbalanced classification problems with few samples of the minority class, i.e., anomaly classification. The *AUC* and *PRAUC* are obtained by varying the hyperparameters. *Precision*, *Recall*, and *F*1-*Score* are the model performance under the best set of parameters selected.

6.4. Performance and Analysis

To answer **Q1**, the proposed model was used in an experiment with four baseline methods on three publicly available real-world datasets with ground truth labeled anomalies. Their performance on the metrics was recorded. The goal of anomaly detection is to detect anomalies as completely as possible, and we place more emphasis on the performance of the model on the recall metric when conducting experiments and model evaluation. Therefore, while ensuring higher F1 scores, this paper primarily uses recall as a performance measure for the model. Table 2 shows the performance of anomaly detection for three datasets by five methods, including six baseline methods. Bolded items in the table indicate the highest values of metrics in each dataset.

- For the SWaT dataset, TGAN-AD has excellent performance in *PRAUC*, *Recall*, and *F*1-*Score*. The *AUC* and *Precision* are second only to FNN. However, in conditions where *Recall* reaches an impressive 99%, the comprehensive indicator *F*1-*Score* has a significant improvement compared to other methods. It scored 10.6% higher than the second best baseline method. Apart from the *AUC* of 0.896, the other four metrics all reached more than 90%, of which the *Recall* reached 99.0%, which was higher than the MAD-GAN by 8.1%.
- For the WADI dataset, most of the metrics are the best among all methods. This dataset, as we know, is unbalanced and has higher dimensionality than other datasets. However, TGAN-AD performs as well on this dataset as on any other dataset. In addition, the anomaly detection performance of TGAN-AD is significantly higher than that of FNN, LSTM, MAD-GAN, and GDN, all of which are deep learning methods.
- For the KDDCUP99 dataset, TGAN-AD has excellent performance in all five metrics. All five metric values of TGAN-AD reached the highest among all methods. The *F*1-*Score* is 7% higher than the second best baseline method. Except for the recall rate of 87.5, the other four values all reached 90%. Moreover, *PRAUC*, *Recall*, and *F*1-*Score* maintain the highest levels across all datasets.

Table 2. Anomaly detection metrics for different datasets.

Datasets	Methods	AUC	PRAUC	Precision	Recall	F1
SWaT	PCA	0.771	0.716	57.1	80.0	0.667
	Random Forests	0.808	0.792	57.1	96.6	0.718
	LSTM	0.866	0.640	41.1	80.8	0.545
	FNN	**0.969**	0.963	**96.2**	75.8	0.847
	MAD-GAN	0.750	0.917	19.6	90.9	0.323
	GDN	0.898	0.696	94.9	64.2	0.766
	TGAN-AD	0.896	**0.992**	91.8	**99.0**	**0.953**
WADI	PCA	0.772	0.930	**92.0**	60.5	0.730
	Random Forests	0.808	0.792	57.1	96.6	0.718
	LSTM	0.842	0.497	72.4	27.9	0.404
	FNN	0.824	0.395	53.3	94.1	0.681
	MAD-GAN	0.506	0.526	13.6	72.7	0.229
	GDN	0.866	0.581	86.5	35.1	0.507
	TGAN-AD	**0.924**	**0.985**	88.1	**96.7**	**0.922**
KDDCUP99	PCA	0.740	0.870	92.3	46.2	0.615
	Random Forests	0.595	0.808	88.9	18.6	0.308
	LSTM	0.292	0.192	68.9	81.9	0.749
	FNN	0.800	0.808	93.3	66.7	0.778
	MAD-GAN	0.881	0.552	96.3	71.1	0.818
	GDN	0.669	0.224	85.9	85.2	0.855
	TGAN-AD	**0.959**	**0.971**	**98.0**	**87.5**	**0.925**

The given result fully demonstrates the usage of Transformer which can well represent the data of Generator G and Discriminator D and paves the most direct way to the optimal representation of the testing sample which can filter the anomaly data from normal data. We implement our method and its variants on an NVIDIA Tesla T4 graphics card. The models are trained for up to 50 epochs and use early stopping with patience of 10. We recorded the training time of the model. In particular, our model took 53 s to train on the SWaT dataset, 1 min 11 s on the more dimensional WADI dataset, and converged in 41 s on the KDDCUP99 dataset. Our model requires less training time than classical deep learning framework LSTM (1 min 9 s/1 min 41 s/51 s) and the latest multivariate anomaly detection model GDN (2 min 25 s/6 min 42 s/1 min 45 s).

6.5. Model Variations

For **Q2**, to evaluate the importance of different hyperparameters in the TGAN-AD, we also tried the different settings of the hyperparameters in our model, i.e., the α, the layers of Transformer, and the sliding window length. All other settings remained unchanged when the comparative experiments were conducted.

6.5.1. The Effect of Hyperparameter α on the Model

In our experiment, we chose the representative dataset, SWAT, to analyze the impact of α in the model. $\alpha \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. In Equation (7), α is the proportion of reconstruction loss G_{loss} and discrimination loss D_{loss} in the anomaly score. For this experiment, we kept the other hyperparameters constant, while setting the number of transformer layers to four and the sliding window length to 60. In SWAT, when α is set as 0.2, all the metrics showed the best performance. With the increase of α, *Precison*, *Recall*, and *F*1 decreased dramatically, and *AUC* and *PRAUC* also decreased.

As show in Figure 6, with $\alpha = 0$, the anomaly score only depended on D_{loss}. Almost all the results with $\alpha = 0$ were higher than the average performance of all the experiments and ranked only second to the results with $\alpha = 0.2$, and only the *AUC* value was slightly higher with $\alpha = 0.2$. It indicates that the anomaly score depends more on the discrimination loss, and a small quantity of the reconstruction loss can improve the overall performance. With $\alpha = 1$, it indicates that the anomaly score depends only on G_{loss}. The results with $\alpha = 1$ were much lower than the results with other α settings. It shows that reconstruction loss has a weak impact on the anomaly score of TGAN-AD.

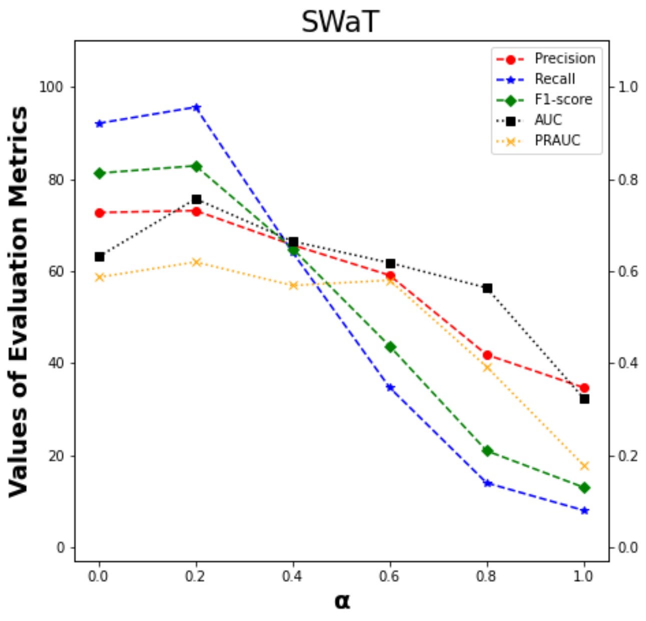


Figure 6. These line graphs show the effect of setting different hyperparameter α on each metric, $\alpha \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$.

6.5.2. Role of Transformer Layers

Different numbers, 2, 4, 6, and 8, of Transformer layers in the generator and discriminator were set. We set the α to 0.2 and the sliding window length to 60. Figure 7 shows our experiment using two datasets, SWaT and KDDCUP99. The performance of anomaly detection outperformed in the SWaT dataset with four layers of encoder–decoder, in the WADI dataset with four and in the KDDCUP99 dataset with four and six layers.

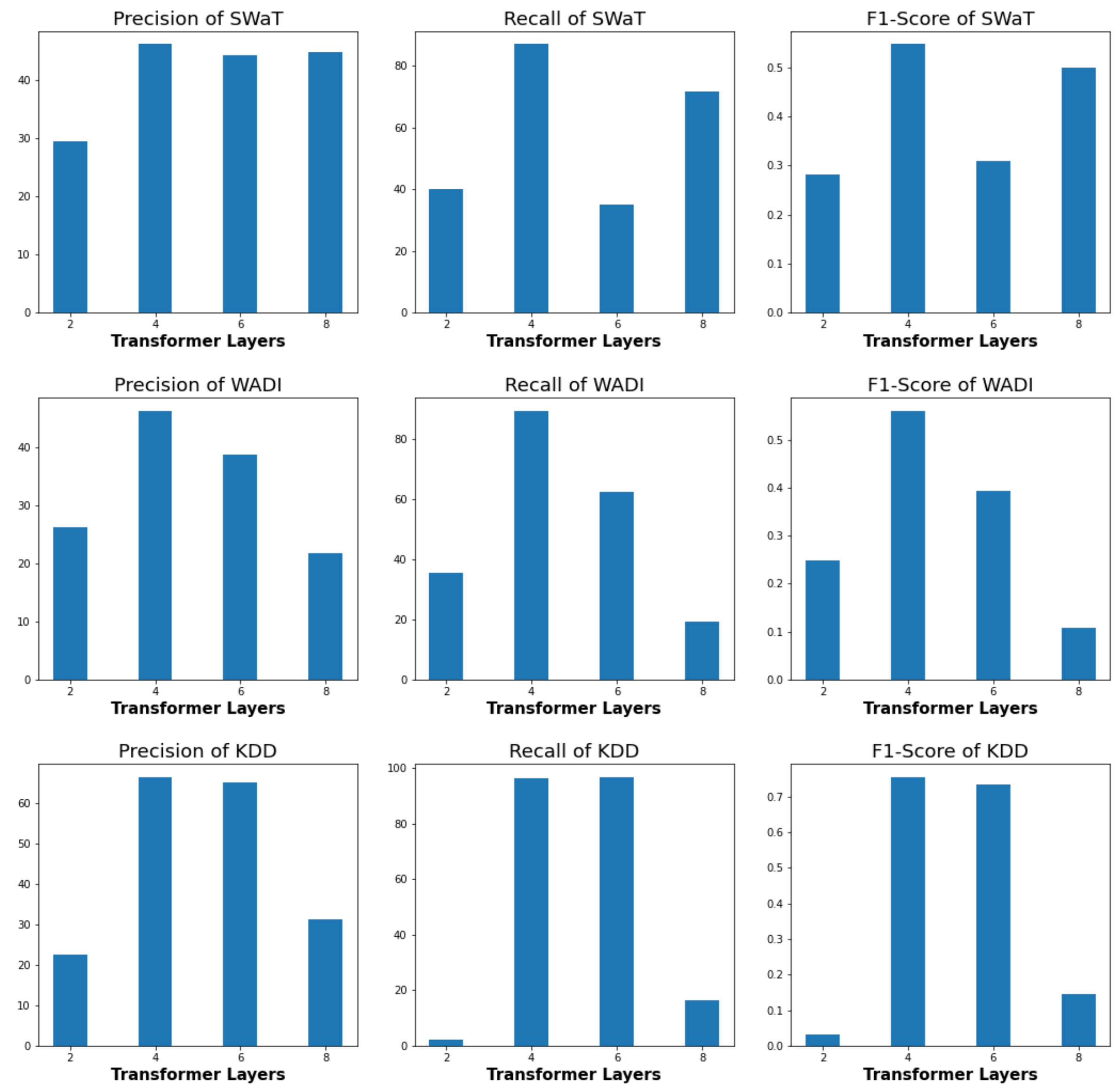


Figure 7. The variation of *Precision*, *Recall*, and *F*1 with different Transformer Layers of two representative datasets, i.e., SWaT, WADI and KDDCUP99. Note that all other parameters are fixed, with the exception of the number of encoder and decoder layers, which are empirically set at 2, 4, 6, and 8.

6.5.3. The influence of Different Sliding Window Widths

The width of the sliding window (i.e., length of a sub-sequence) is sensitive for the model to capture the hidden information in time series. In this experiment, different window widths, $t_w \in \{10 \times \mathrm{i}, \mathrm{i} = 1, 2, 3, 4, 5, 6 \ldots, 10\}$, are used to observe the influence of different sliding window widths on the performance of anomaly detection. Here, α is set to 0.2, and the number of Transformer layers is set to four. For each sub-sequence length, the TGAN-AD model was trained recursively for 100 iterations. We depict the box-plots of the metrics values of TGAN-AD at each of the training iterations over different sub-sequence lengths in Figure 8.

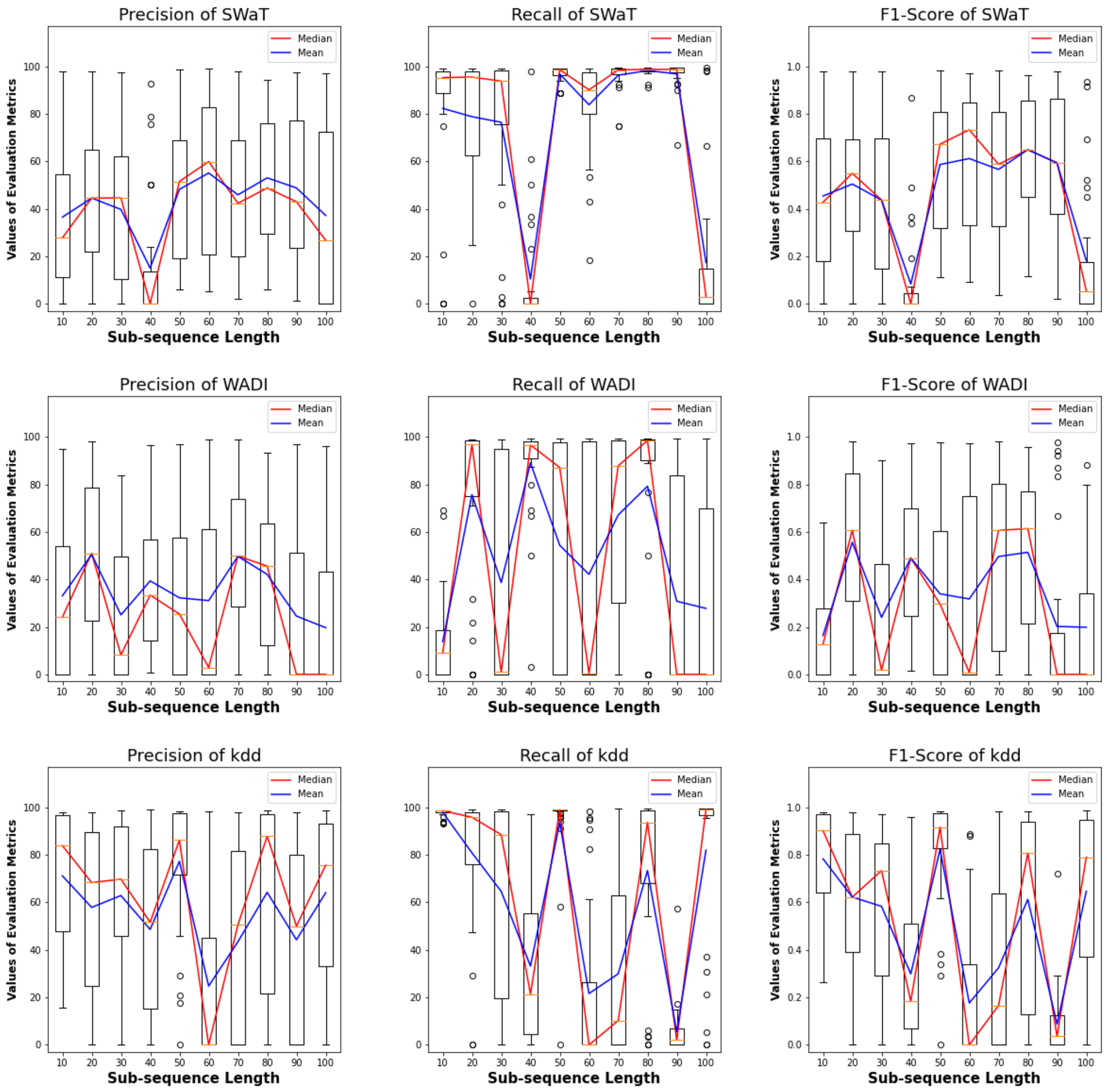


Figure 8. The variation of *Precision*, *Recall*, and *F*1 with different sequence lengths in the representative dataset, i.e., SWaT. The experiments are conducted with 100 iterations. The median values of each box are shown linked together with a red line, and the mean values of each box are shown linked together with a blue line.

As shown in Figure 8, the impact of sequence length of our model is given as follows:

1. **SWaT dataset:** When the sequence length was set at 60, our TGAN-AD achieved the best performance in *Precision* and *F*1, while the value of *Recall* was close to 0.9. When the sequence length was 100 and 40, the model showed relatively bad performance.
2. **WADI dataset:** In the experiment, *F*1 and *Precsion* showed bad performance, since the model predicted some false positive results. However, in the scenario of anomaly detection, the wrong alarm of non-anomalous samples is permissible. When the length was 20, the model performed best.

3. **KDDCUP99 dataset:** TGAN-AD was not stable in testing on the KDD dataset but had good overall average metric values. When the sequence length was 50, it had an excellent metric value ($Precision$, $Recall$, and $F1$) close to 0.9.

6.5.4. Discussion

In summary, the hyperparameters, i.e., α, the layers of Transformer, and the sliding window length, are important for learning the optimal parameters to detect anomalous data from large-scale online time series data. The method of automatic parameter selection is still a challenge for real-world application scenarios. Our model has the stable performance of $Recall$ with different settings, which is significant for the applications of AIOps and other similar scenarios. In real-world applications, time series data frequently lack contextual data, which is critical for time series anomaly detection.

7. Conclusions

Anomaly detection is one of the most popular applications of time series data analysis, and it is also an important research branch in the field of AIOps. In this paper, we propose a novel framework called TGAN-AD for anomaly detection of multivariate time series. We use Transformer to train the generator and discriminator of GANs and finally use reconstruction loss and discrimination loss to measure the anomaly. We tested TGAN-AD on three public datasets and compared it with the state-of-the-art methods of time series anomaly detection. TGAN-AD showed the best performance.

The performance of anomaly detection is sensitive to the sliding window length. Hence, in future work, automatic selection of the optimal sliding window length of the model is a good direction for the further improvement of anomaly detection. In our work, only the elementary Transformer was used. The variants of Transformer can also be used in the model of time series data.

Author Contributions: Conceptualization, L.X.; methodology, K.X.; software, X.H. and Z.L.; formal analysis, N.Y.; resources, X.J.; writing—original draft preparation, Y.Q.; writing—review and editing, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was in part by the State Key Laboratory of Smart Grid Protection and Control under Grant SGNR0000KJJS2007626, in part by the Research Foundation for Advanced Talents of Nanjing University of Posts and Telecommunications under Grants NY218118 and NY219104, and in part by the Jiangsu Project of Social Development under Grant BE2020713.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We would like to thank the reviewers for their comments, which helped improve this paper considerably.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cook, A.A.; Mısırlı, G.; Fan, Z. Anomaly detection for IoT time series data: A survey. *IEEE Internet Things J.* **2019**, *7*, 6481–6494. [CrossRef]
2. Liu, D.; Zhao, Y.; Xu, H.; Sun, Y.; Pei, D.; Luo, J.; Jing, X.; Feng, M. Opprentice: Towards practical and automatic anomaly detection through machine learning. In Proceedings of the 2015 Internet Measurement Conference, Tokyo, Japan, 28–30 October 2015; pp. 211–224.
3. Shang, Z.; Zhang, Y.; Zhang, X.; Zhao, Y.; Cao, Z.; Wang, X. Time Series Anomaly Detection for KPIs Based on Correlation Analysis and HMM. *Appl. Sci.* **2021**, *11*, 11353. [CrossRef]
4. Lai, K.H.; Zha, D.; Xu, J.; Zhao, Y.; Wang, G.; Hu, X. Revisiting Time Series Outlier Detection: Definitions and Benchmarks. 2021. Available online: https://github.com/datamllab/tods/tree/benchmark (accessed on 10 August 2022).
5. Karadayı, Y.; Aydin, M.N.; Öğrenci, A.S. A hybrid deep learning framework for unsupervised anomaly detection in multivariate spatio-temporal data. *Appl. Sci.* **2020**, *10*, 5191. [CrossRef]

6. Li, Z.; Zhao, Y.; Han, J.; Su, Y.; Jiao, R.; Wen, X.; Pei, D. 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, Singapore, 14–18 August 2021; pp. 3220–3230.
7. Wu, J.; Lee, P.P.; Li, Q.; Pan, L.; Zhang, J. CellPAD: Detecting performance anomalies in cellular networks via regression analysis. In Proceedings of the 2018 IFIP Networking Conference (IFIP Networking) and Workshops, Zurich, Switzerland, 14–16 May 2018; pp. 1–9.
8. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2014; Volume 27.
9. Chen, L.; Li, Y.; Deng, X.; Liu, Z.; Lv, M.; Zhang, H. Dual Auto-Encoder GAN-Based Anomaly Detection for Industrial Control System. *Appl. Sci.* **2022**, *12*, 4986. [CrossRef]
10. Fu, R.; Zhang, Z.; Li, L. Using LSTM and GRU neural network methods for traffic flow prediction. In Proceedings of the 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), Wuhan, China, 11–13 November 2016; pp. 324–328.
11. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 5998–6008.
12. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 1–58. [CrossRef]
13. Laurikkala, J.; Juhola, M.; Kentala, E.; Lavrac, N.; Miksch, S.; Kavsek, B. Informal identification of outliers in medical data. In Proceedings of the Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology, Berlin, Germany, 22 August 2000; Volume 1, pp. 20–24.
14. Pena, E.H.; de Assis, M.V.; Proença, M.L. Anomaly detection using forecasting methods arima and hwds. In Proceedings of the 2013 32nd International Conference of the Chilean Computer Science Society (SCCC), Temuco, Chile, 11–15 November 2013; pp. 63–66.
15. Galeano, P.; Peña, D.; Tsay, R.S. *Outlier Detection in Multivariate Time Series via Projection Pursuit*; Universidad Carlos III de Madrid: Madrid, Spain, 2004.
16. Smith, R.; Bivens, A.; Embrechts, M.; Palagiri, C.; Szymanski, B. Clustering approaches for anomaly based intrusion detection. In *Intelligent Engineering Systems through Artificial Neural Networks*; ASME: New York, NY, USA, 2002; Volume 9.
17. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 15–18 May 2000; pp. 93–104.
18. Chan, P.K.; Mahoney, M.V.; Arshad, M.H. *A Machine Learning Approach to Anomaly Detection*; Technical Report; 2003. Available online: https://repository.lib.fit.edu/bitstream/handle/11141/114/cs-2003-06.pdf?sequence=1 (accessed on 10 August 2022).
19. He, Z.; Xu, X.; Deng, S. Discovering cluster-based local outliers. *Pattern Recognit. Lett.* **2003**, *24*, 1641–1650. [CrossRef]
20. Jiang, J.R.; Kao, J.B.; Li, Y.L. Semi-supervised time series anomaly detection based on statistics and deep learning. *Appl. Sci.* **2021**, *11*, 6698. [CrossRef]
21. Serras, J.L.; Vinga, S.; Carvalho, A.M. Outlier Detection for Multivariate Time Series Using Dynamic Bayesian Networks. *Appl. Sci.* **2021**, *11*, 1955. [CrossRef]
22. Pumsirirat, A.; Yan, L. Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 18–25. [CrossRef]
23. An, J.; Cho, S. Variational autoencoder based anomaly detection using reconstruction probability. *Spec. Lect. IE* **2015**, *2*, 1–18.
24. Li, D.; Chen, D.; Jin, B.; Shi, L.; Goh, J.; Ng, S.K. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In Proceedings of the International Conference on Artificial Neural Networks, Munich, Germany, 17–19 September 2019; Springer: Cham, Switzerland, 2019; pp. 703–716.
25. Bashar, M.A.; Nayak, R. TAnoGAN: Time Series Anomaly Detection with Generative Adversarial Networks. *arXiv* **2020**, arXiv: 2008.09567.
26. Geiger, A.; Liu, D.; Alnegheimish, S.; Cuesta-Infante, A.; Veeramachaneni, K. TadGAN: Time series anomaly detection using generative adversarial networks. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 10–13 December 2020; pp. 33–43.
27. Shaw, P.; Uszkoreit, J.; Vaswani, A. Self-attention with relative position representations. *arXiv* **2018**, arXiv:1803.02155.
28. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.V.; Salakhutdinov, R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv* **2019**, arXiv:1901.02860.
29. Dehghani, M.; Gouws, S.; Vinyals, O.; Uszkoreit, J.; Kaiser, Ł. Universal transformers. *arXiv* **2018**, arXiv:1807.03819.
30. Wu, N.; Green, B.; Ben, X.; O'Banion, S. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv* **2020**, arXiv:2001.08317.
31. Wu, S.; Xiao, X.; Ding, Q.; Zhao, P.; Ying, W.; Huang, J. Adversarial Sparse Transformer for Time Series Forecasting. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, BC, Canada, 6–12 December 2020.
32. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time series forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-21), Vancouver, BC, Canada, 2–9 February 2021, Volume 35, pp. 11106–11115.
33. Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.X.; Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5243–5253.

34. Shyu, M.L. A Novel Anomaly Detection Scheme Based on Principal Component Classifier. In Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, In Conjunction with the Third IEEE International Conference on Data Mining (ICDM03), Melbourne, FL, USA, 19–20 November 2003; pp. 172–179.
35. Xu, Y.X.; Pang, M.; Feng, J.; Ting, K.M.; Jiang, Y.; Zhou, Z.H. Reconstruction-based Anomaly Detection with Completely Random Forest. In Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), Virtual Event, 29 April–1 May 2021; pp. 127–135.
36. Malhotra, P.; Vig, L.; Shroff, G.; Agarwal, P. Long short term memory networks for anomaly detection in time series. *Proceedings* **2015**, *89*, 89–94.
37. Rong, Z.; Shandong, D.; Xin, N.; Shiguang, X. Feedforward Neural Network for Time Series Anomaly Detection. *arXiv* **2018**, arXiv: 1812.08389.
38. Deng, A.; Hooi, B. Graph neural network-based anomaly detection in multivariate time series. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 4027–4035.