Multivariate Time Series Anomaly Detection Method Based on mTranAD

Chuanlei Zhang [*1], Yicong Li [1], Jie Li [2], Guixi Li [1] and Hui Ma [3]

[1] Tianjin University of Science and Technology, Tianjin, China
[2] China Unicom Research Institute, Beijing, China
[3] Yunsheng Intelligent Technology Co., Ltd Tianjin, China
a17647@gmail.com

Abstract. Multivariate time series anomaly detection is a crucial area of research in several domains, including finance, logistics, and manufacturing. Successfully identifying abnormal behaviors or events can help prevent disruptions, but the high false positive rate in this field is a significant challenge that affects detection accuracy. In this paper, we propose a novel method, mTranAD, which improves upon the TranAD algorithm by leveraging the benefits of Transformer and variational autoencoder (VAE) in multivariate unsupervised anomaly detection. Specifically, mTranAD replaces TranAD's autoencoder structure with a VAE and trains it using the VAE's loss function. The incorporation of latent variables in the VAE model enables accurate reconstruction of data and mapping of data to a lower dimensional latent space, allowing for a more efficient description of input data complexity with fewer parameters. By utilizing these latent variables, the model can effectively handle high-dimensional, complex data and exhibit greater flexibility when generating new data. We conduct experiments on four public datasets (NAB, MBA, SMAP and WADI) and compare mTranAD's performance with 11 other state-of-the-art methods, including TranAD, MERLIN, LSTM-NDT, OmniAnomaly, USAD, and DAGMM. The experimental results demonstrate that mTranAD outperforms these methods in terms of performance, accuracy, and reliability. The primary purpose of this paper is to improve the TranAD algorithm and enhance the accuracy of multivariate time series anomaly detection by reducing the false positive rate.

Keywords: Anomaly detection, Transformer, Variational autoencoder, mTranAD.

1 Introduction

Anomaly detection is an important and challenging problem in many disciplines, and anomaly detection methods have been widely applied in fields such as fault diagnosis, network intrusion, and monitoring detection. Time series data is a widely used data type in various fields, covering finance, healthcare, network monitoring, and many other domains. It can reflect the dynamic changes of a system or process over time, providing

valuable information for decision-making and problem-solving. However, interpreting time series data can sometimes be challenging, particularly when there are outliers in the data. Outliers in time series data refer to patterns that are clearly deviant from normal behavior, which may indicate problems such as errors, malfunctions, or security threats. Therefore, accurately identifying and handling outliers in time series data is crucial for effective utilization [1]. Deep learning-based methods, especially those based on generative adversarial networks, have become hotspots in anomaly detection. However, adversarial training has problems such as mode collapse and non-convergence that require more research [2]. Unsupervised anomaly detection for multi-sensor time series data is important in machine learning [20].

Traditional time series anomaly detection methods rely on statistical theory, but the emergence of deep learning has led to increased exploration of its application in this field. Multi-layer neural networks, such as Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN), can mine feature information more effectively, improving accuracy and robustness. Different deep learning models and algorithms have been proposed for various application scenarios, including Generative Adversarial Network (GAN) and Variational Autoencoder (VAE)-based methods. Reconstruction-based deep learning algorithms, particularly VAEs, have become popular due to their ability to detect anomalous data that violates temporal dependencies. RNNs and LSTMs have also shown promising performance in handling time series data tasks.

Developing a system that can efficiently and accurately identify anomalies is challenging. While the TranAD model addresses high data volatility and ultra-low inference time, its autoencoder structure lacks strong adversarial noise and missing value capabilities, as well as generalization abilities. To address these limitations, we replaced the AE structure with VAE due to its robustness to noise and missing values caused by its randomness.

2 Related work

Time series anomaly detection is a complex task and has been extensively studied. Since supervised training algorithms lack labeled anomalous data, most anomaly detection methods are based on unsupervised methods. We can categorize unsupervised methods into four types: linear model-based methods, distance-based methods, probability and density estimation-based methods, and deep learning-based methods.

Unsupervised anomaly detection methods can use Principal Component Analysis (PCA) [12] or Partial Least Squares (PLS) to analyze process measurements and identify anomalies. However, these methods only work well with highly correlated data that follows a multivariate Gaussian distribution. Distance-based methods like k-Nearest Neighbor (KNN) and Cluster-based Local Outlier Factor (CBLOF) are also used for anomaly detection, but they perform better when there is prior knowledge about the anomaly duration and number. Probabilistic and density estimation-based methods like Angle-based Outlier Detection (ABOD) and Feature Bagging (FB) are improvements on distance-based methods because they focus more on data distribution,

but they are not suitable for analyzing short-term temporal correlations in multivariate time series data [13].

Deep learning-based unsupervised anomaly detection methods are important in many fields. The Transformer model has gained popularity for time series analysis, as it can leverage self-attention to discover long-term temporal dependencies. It is also used for modeling and detecting anomalies based on reconstruction criteria. Traditional techniques like PCA, k-means, and OCSVM cannot capture temporal dependencies due to their inability to pass time memory states. However, standard Autoencoders may not generalize well due to the presence of noise and anomalies in normal samples, and the separation of feature extraction and prediction model building, which can lead to local optima [13]. Autoencoders comprise an encoder that learns data representation and a decoder that reconstructs the input.

MAD-GAN [16] and MTAD-GAT [17] are GAN models that use LSTM and graph attention networks respectively to model time series. CAE_M [18] uses a convolutional autoencoder memory network with bidirectional LSTM to capture temporal trends, while USAD [15] employs an autoencoder with two decoders for quick training. GDN [19] utilizes graph-biased network w/attention-based prediction, and MERLIN [20] compares sub-sequences. DAGMM [21] compresses input data into latent space for Gaussian mixture modeling, while LSTM-NDT [22] uses LSTM w/dynamic threshold methods. However, recurrent models like LSTM are computationally expensive. MTAD-GAT and GDN use deep neural networks w/window inputs, but small fixed-size windows limit performance. A fast model is needed to capture high-level temporal trends with minimal overheads.

3 Methods

The structure diagram of the mTranAD model is shown in Fig. 1.

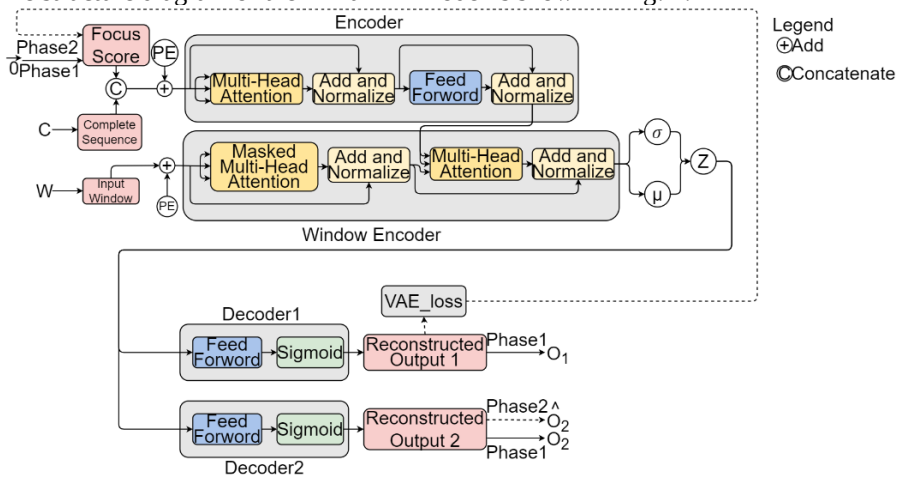

Fig. 1: The mTranAD Model.

We deeply re-architected the Transformer for anomaly detection in time-series data, incorporating attention-based transformations and a Variational Autoencoder.
VAE is a type of generative model that is specifically designed to encode datasets with high dimensionality into a lower-dimensional space called the latent space. This encoded information is then utilized to generate novel samples that resemble the original data by means of decoding the latent representation. VAE uses an encoder network $q_\varphi(z|x)$ to map the input data x to a lower-dimensional latent representation z, and a decoder network $p_\theta\,(\,x \mid z)$ to map the latent representation back to the original data space.

$$z = \mu + \varepsilon * \sigma \tag{1}$$

The loss function of VAE consists of two parts. The first part is the Kullback-Leibler divergence (KL divergence) between the trained generative model distribution $q_\varphi(z|x)$ and the Gaussian distribution p(z) that the data latent variable z is assumed to follow. The KL divergence $D_{KL}[q_\varphi(z|x)||p(z)]$ is used to quantify the difference between the two distributions. The second part is the reconstruction loss, which measures the error between the generated data and the input data, given the latent variable z. It is defined as $E_q[\log 10 p_\theta(x|z)]$. We use the MSE Loss instead of cross-entropy as the reconstruction loss, $\| \hat{O}_2 - W_d \|_2$. The overall loss function of VAE is a linear combination of the two parts, with a trade-off parameter β balancing the importance of each part. β equals 1. It can be expressed as:

$$L\,(\,\theta,\ \varphi,\ x) = -D_{KL}\,[\,q_\varphi(\,z \mid x) \,||\, p\,(\,z\,)] + E_q\,[\,log 10\, p_\theta\,(\,x \mid z)] \tag{2}$$

We have combined TranAD and VAE with the Transformer architecture for processing time-series data. By adversarially training the discriminator to detect anomalies in real samples, we can simplify downstream neural network inference operations using softmax compression. Our approach uses scaled dot-product attention to reduce weight variance and positional encoding for the input matrix [6]. F is matched to W dimensions, followed by concatenation via zero addition. The H_1 input is obtained using positional encoding and performs the following operations during the first encoder pass.

$$H_1^1 = LayerNorm\,(\,H_1 + MultiHeadAtt\,(\,H_1,\ H_1,\ H_1)\,) \tag{3}$$

$$H_1^2 = LayerNorm\,(\,H_1^1 + FeedForward\,(\,H_1^1)\,) \tag{4}$$

Apply multi-head self-attention to matrix H_1 and add it to itself via matrix addition. In the window encoder, position encoding is used to generate input matrix H_2, and the self-attention mechanism in the encoder is adjusted to ensure the decoder cannot access future timestamps in the data during training.

$$H_2^1 = Mask\,(\,MultiHeadAtt\,(\,H_2,\ H_2,\ H_2)\,) \tag{5}$$

$$H_2^2 = LayerNorm\,(\,H_2^1,\ H_2) \tag{6}$$

$$H_2^3 = LayerNorm\,(\,MultiHeadAtt\,(\,H_1^2,\ H_1^2,\ H_1^2) + H_2^2) \tag{7}$$

Operations (5)(6)(7) are similar to operations (3)(4), where the complete time series H_1^2 encoding is used as values and keys by the window encoder, while the encoded input window is used as the query matrix.

$$O_d = Sigmoid\ (\ FeedForward\ (H_2^3)\ ) \tag{8}$$

Here, d∈{1,2} refers to the first and second decoder, respectively. We use the sigmoid activation function to constrain the output within the range of [0,1], and normalize it to match the input window.

The Transformer model can be used for reconstructive prediction of input time series windows. In this model, each timestamp is utilized to act as an encoder-decoder network for prediction while introducing VAE latent variables between the encoder and decoder. The encoder transforms the time series window W of input into a compressed representation using an attention mechanism based on the context, similar to a transformation model. Then, an activation function is used to transform this compressed representation into outputs O_1 and O_2.

We have introduced the concept of adversarial training, using two independent decoders. The second decoder is used to distinguish between the input window and the candidate reconstruction generated by the first decoder in phase 1, by maximizing the difference of $\| \hat{O}_2 - W_d \|_2$ The aim of the first decoder is to deceive the second decoder, while minimizing the reconstruction error of the self-conditioned output. On the other hand, the aim of the second encoder is to maximize the reconstruction error of the self-conditioned output.

Algorithm 1 The mTranAD training algorithm

Require:

Encoder E with parameters φ, Decoder D_1,D_2 with parameters θ

A Gaussian distribution $p(z) \sim \mathcal{N}(0,1)$

Dataset W

Hyperparameter ϵ

1:Initialize weights of E,D_1,D_2

2:n←0

3:do

4: for(d=1 to D)

5: $z_d \leftarrow E_\varphi(W_d, 0)$

6: O_1,$O_2 \leftarrow D_{\theta 1}(z_d), D_{\theta 2}(z_d)$

7: $\hat{z}_d \leftarrow E(W_d, \| O_1 - W_d \|_2)$

8: $\hat{O}_2 \leftarrow D_2(\hat{z}_d)$

10: $L_1 = -D_{KL}\left[E_\varphi(z_d|W_d, 0) \| p(z)\right] + \| O_1 - W_d \|_2 + \epsilon \| \hat{O}_2 - W_d \|_2$

11: $L_2 = -D_{KL}\left[E_\varphi(z_d|W_d, 0) \| p(z)\right] + \| O_2 - W_d \|_2 - \epsilon \| \hat{O}_2 - W_d \|_2$

12: Update the parameters θ and φ using $L1$ and $L2$

13: n←n+1

14:while n<N

We preprocess the raw data, then we use the reconstruction loss generated by the first decoder as the focus score and use it to construct the focus matrix. Then, we re-run the model inference to obtain the output $\hat{O}_2$ of the second decoder. The initially generated

focus scores represent the deviation between the reconstructed output and the given input. These focus scores are used as priors to modify the attention weights, and to give higher neural network activations to specific input subsequences for the extraction of short-term time trends. We call this method "self-attentional regulation". This two-stage self-regressive inference style has three benefits. First, it amplifies the deviation, as the reconstruction error activates the attention part of the Encoder, resulting in anomalous scores that simplify error tag tasks. Second, it prevents false alarms by capturing short-term time trends in the Window Encoder. Finally, we use VAE latent variables for data reconstruction, and two independent decoders for reconstruction. The goal of the first decoder is to create low-quality focus scores by perfectly reconstructing the input, thereby fooling the second decoder. The goal of the second decoder is to maximize these focus scores in its output.

4 Experiment

In this section, we conducted extensive experiments to evaluate the anomaly detection performance of the mTranAD method on several real-world datasets, by comparing it with state-of-the-art multivariate time series anomaly detection models.

4.1 Baseline

In the experiment section, we'll compare our multivariate time series anomaly detection method with several classic baseline models. These include MAD-GAN [16], MTAD-GAT [17], CAE_M [18], USAD [15], and GDN [19]. MAD-GAN is an LSTM-based GAN model that models the distribution of time series data through a generator. MTAD-GAT uses graph attention neural networks to model feature and temporal correlations, then applies a lightweight GRU network for detection to reduce computational costs and improve scalability. CAE_M converts time series into feature vectors using CNN and bidirectional LSTM to capture long-term trends. USAD uses an autoencoder with two decoders in an adversarial training framework for quick training. GDN learns relationship graphs between data patterns using graph-biased network methods and outputs anomaly scores using attention-based prediction and bias scoring. The MERLIN [20] method detects anomalies in time series data by comparing different length sub-sequences with their neighbors. DAGMM [21] uses a deep autoencoder Gaussian mixture model for feature reduction and time modeling using recurrent networks. LSTM-NDT [22] predicts data using LSTM-based neural networks and non-parametric dynamic threshold methods for anomaly detection, but it's slow. MTAD-GAT and GDN use deep neural networks with a time series window as input, but the limited local context information restricts their performance. Our proposed method aims to quickly capture high-level temporal trends with minimal overhead and will be compared against baseline methods for effectiveness.

4.2 Datasets

We used four public datasets: NAB, MBA, SMAP, and WADI. These datasets utilize multivariate time series for anomaly detection.

The NAB dataset stands for Numenta Anomaly Benchmark and is a new benchmark for evaluating anomaly detection algorithms in streaming real-time applications. It contains over 50 marked time series data files, both real-world and artificial, with labeled anomaly behavior periods.
The MBA dataset stands for the MIT-BIH Supraventricular Arrhythmia Database, which is a collection of electrocardiogram (ECG) recordings from four patients that includes multiple abnormal instances of supraventricular contractions and premature beats.
The SMAP dataset stands for the Soil Moisture Active Passive Dataset, which is a dataset of soil samples and telemetry information used by NASA's Mars explorer. The dataset consists of 55 entities, each with 25 dimensions.
The WADI dataset stands for the Water Distribution Dataset. It consists of data from 123 sensors and actuators, totaling 127 features, collected continuously over a period of 16 days, including 14 days of normal operation and 2 days of attack scenarios, with a total of 15 attacks during the two days.
We selected the four aforementioned publicly available real-world time series datasets. First, we normalized the data in each of the four datasets, and then split them into training and testing sets using an 80/20 ratio. The processed data files were saved separately as train.npy, test.npy, and labels.npy. We compared several commonly used efficient models for time series anomaly detection.

4.3 Data preprocessing

This experiment involves multivariate time series, which consist of a timestamp sequence of size D, where each data point x_d is collected at a specific timestamp d and $x_d \in R^n$,∀d. Here, when n=1, it represents the univariate case. Based on the training input time series of D, we need to predict the anomaly status of the testing time series of length D', with the same mode as the training sequence, Y= $(y_1,...,y_{D'})$. We use a binary variable $y_d \in \{0,1\}$ to indicate whether the data point at the d-th timestamp in the test set is anomalous (1 for anomalous data point).
To train and test the models, we normalized the data and converted it into time series windows. Specifically, we normalized the time series data in the following manner:

$$x_d \leftarrow \frac{x_d - \min(D)}{\max(D) - \min(D) + \varepsilon'} \tag{9}$$

Where min(D) represents the minimum vector of the mode in the training data time series, max(D) represents the maximum vector of the mode in the training data time series, and ε' is a constant vector used to prevent division by zero errors. We normalize the data so that it falls within the range of [0,1).

$$W_d = \{x_{d-M+1}, \ldots, x_d\}.$$

During the experiment, we used the replication padding method to pad the first d data points of the original time series and generate a sliding window sequence W of length M. For each d<M, we appended a constant vector $\{x_d,\ldots,x_d\}$ of length M-d to W_d, so that the length of each window for d is M. In the training process of the model, we did not use a single data point d as the training input, but used the sliding window sequence W for training, and used W (corresponding to d) as the test sequence. At the same time,

we also considered the time slices before the current timestamp d of the sequence d, and labeled it as Q_d.

Our goal now is not to directly predict the anomaly label y_d for each window W_d, but to first predict the anomaly score s_t for that window. To do this, we use the anomaly scores of past input windows and calculate a threshold P. If s_t exceeds this threshold, we label the input window as anomalous, so y_d=1(s_t>=P). To calculate the anomaly score s_t, we reconstruct the input window as O_d and calculate the deviation between W_d and O_d. For each window W_d, we use an autoencoder to map it to O_d. Then, we calculate the reconstruction error (i.e., the deviation between W_d and O_d) and use it as the anomaly score s_t for window W_d.

4.4 Evaluation indicators

To evaluate the overall performance of the mTranAD model in detecting anomalies, we use precision, recall, receiver operating characteristic curve area (ROC/AUC), and F1 score as evaluation metrics:

$$Precesion = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{12}$$

Where TP is the number of actually detected abnormal samples, FP is the number of samples that are detected as abnormal but are actually normal, FN is the number of abnormal samples that are not detected, and TN is the number of samples that are detected as normal.

4.5 Experimental results

Table 1 presents the precision, recall, ROC/AUC, and F1 score of mTranAD and baseline models for all datasets. mTranAD outperforms the baseline in terms of AUC and F1 score on all four datasets. mTranAD achieves the highest anomaly detection accuracy in terms of F1 score, which is the decisive evaluation metric, on the NAB, MBA, SMAP, and WADI datasets. On average, the mTranAD model has an F1 score of 0.8802 and an AUC of 0.9084. On the SMAP dataset, the TranAD model achieved the highest AUC score (0.9921), but mTranAD had the highest F1 score among all datasets. Compared to the TranAD model, mTranAD achieves an improvement of 0.5126%, 0.0920%, 1.144%, and 41.18% respectively on the above-mentioned datasets. Table 1 reveals poor performance of several methods on the WADI dataset due to its large sequence lengths and diverse data formats. The POT technique, used in mTranAD, TranAD, and related models, considers local peaks in data to set more precise thresholds, widely used in anomaly detection to reduce false alarms and improve identification of significant anomalous datapoints, leading to deeper data analysis and insights. It is important to note that the use of the POT technique requires parameter tuning based on the actual situation to achieve optimal results. The mTranAD model we propose is based on the TranAD model, which introduces the latent variables and loss calculation of VAE. It transforms the features of the encoded data into a

multivariate latent distribution, and then reconstructs it through the decoder. Compared with the TranAD model, our model can reduce the risk of overfitting and lower the false positive rate to improve the accuracy of multivariate time series anomaly detection.

Table 1. Performance comparison of the mTranAD and baseline methods on the following dataset. P: Precision, R: Recall rate, AUC: Area under ROC curve, F1: F1 score of training data

Method	NAB			
	P	R	AUC	F1
MERLIN	0.8013	0.7262	0.8414	0.7619
LSTM-NDT	0.6400	0.6667	0.8322	0.6531
DAGMM	0.7622	0.7292	0.8572	0.7453
OmniAnomaly	0.8421	0.6667	0.8330	0.7442
MSCRED	0.8522	0.6700	0.8401	0.7502
MAD-GAN	0.8666	0.7012	0.8478	0.7752
USAD	0.8421	0.6667	0.8330	0.7442
MTAD-GAT	0.8421	0.7272	0.8221	0.7804
CAE_M	0.7918	0.8019	0.8019	0.7968
GDN	0.8129	0.7872	0.8542	0.7998
TranAD	0.8889	0.9892	0.9541	0.9364
mTranAD	0.8889	1.0000	**0.9996**	**0.9412**
Method	MBA			
	P	R	AUC	F1
MERLIN	0.9846	0.4913	0.7828	0.6555
LSTM-NDT	0.9207	0.9718	0.9780	0.9456
DAGMM	0.9475	0.9900	0.9858	0.9683
OmniAnomaly	0.8561	1.0000	0.9570	0.9225
MSCRED	0.9272	1.0000	0.9799	0.9623
MAD-GAN	0.9396	1.0000	0.9836	0.9689
USAD	0.8953	0.9989	0.9701	0.9443
MTAD-GAT	0.9018	1.0000	0.9721	0.9484
CAE_M	0.8442	0.9997	0.9661	0.9154
GDN	0.8832	0.9892	0.9528	0.9332
TranAD	0.9569	1.0000	0.9885	0.9780
mTranAD	0.9587	1.0000	**0.9890**	**0.9789**
Method	SMAP			
	P	R	AUC	F1
MERLIN	0.1577	0.9999	0.7426	0.2725
LSTM-NDT	0.8523	0.7326	0.8602	0.7879
DAGMM	0.8069	0.9891	0.9885	0.8888

OmniAnomaly	0.8130	0.9419	0.9889	0.8728
MSCRED	0.8175	0.9216	0.9821	0.8664
MAD-GAN	0.8157	0.9216	0.9891	0.8654
USAD	0.7480	0.9627	0.9890	0.8419
MTAD-GAT	0.7991	0.9991	0.9844	0.8880
CAE_M	0.8193	0.9567	0.9901	0.8827
GDN	0.7480	0.9891	0.9864	0.8518
TranAD	0.8043	0.9999	**0.9921**	0.8915
mTranAD	0.8211	1.0000	0.9895	**0.9017**

Method	WADI			
	P	R	AUC	F1
MERLIN	0.0636	0.7669	0.5912	0.1174
LSTM-NDT	0.0138	0.7823	0.6721	0.0271
DAGMM	0.0760	0.9981	0.8563	0.1412
OmniAnomaly	0.3158	0.6541	0.8198	0.4260
MSCRED	0.2513	0.7319	0.8412	0.3741
MAD-GAN	0.2233	0.9124	0.8026	0.3588
USAD	0.1873	0.8296	0.8723	0.3056
MTAD-GAT	0.2818	0.8012	0.8821	0.4169
CAE_M	0.2782	0.7918	0.8728	0.4117
GDN	0.2912	0.7931	0.8777	0.4260
TranAD	0.3829	0.8296	0.8968	0.4951
mTranAD	0.6040	0.8296	**0.9084**	**0.6990**

The effectiveness of anomaly detection is shown in Fig. 2, which presents the predicted and true labels for the NAB test set using the mTranAD model. Purple represents the true anomaly values, green represents the anomaly score, blue represents the true values, and orange represents the predicted values. We can observe that the timestamps with high anomaly scores also have true anomaly values. In other words, when there is an anomalous fluctuation in the green line, there is a corresponding fluctuation in the blue line, and this fluctuation is also within the purple area.

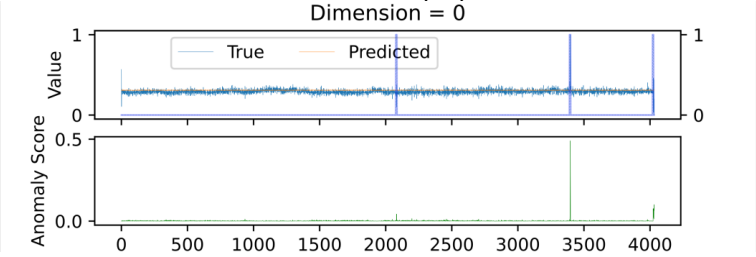

Fig. 2. Real Prediction Tags for NAB Datasets in the mTranAD Model.

5 Summary

This paper proposes a method for detecting abnormal data by combining the Transformer with the VAE model based on the idea of adversarial training. The method first normalizes and handles missing values of the raw data, and then trains the data through the transformer encoder-VAE resampling-decoder. After the model is fitted, the test data is inputted, and the reconstruction error is maximized and minimized to ensure training stability. We verify the method using four publicly available datasets: NAB, MBA, SMAP, and WADI. These datasets use multi-sensor time series for anomaly detection. The experimental results show that the mTranAD model has better detection performance for multivariate time series data. The source code will be made available at https://github.com/Liyicong98/mTranAD.git.

References

1. Tuli, Shreshth.: TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. VLDB (2022).
2. Martín, Léon.: Towards Principled Methods for Training Generative Adversarial Networks. In 5th International Conference on Learning Representations, ICLR (2017).
3. Qiang Yang.: Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) 10, 2 (2019), 1–19 (2019).
4. Osada, G., Omote, K.: Network Intrusion Detection Based on Semi-supervised Variational Auto-Encoder. In: Foley, S., Gollmann, D., Snekkenes, E. (eds) Computer Security – ESORICS 2017. ESORICS 2017. Lecture Notes in Computer Science, vol 10493. Springer, Cham (2017).
5. Ashish Vaswani.: Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems. 6000–6010 (2017).
6. Y. Wang, N. Masoud. : Real-Time Sensor Anomaly Detection and Recovery in Connected Automated Vehicle Sensors. IEEE Transactions on Intelligent Transportation Systems.vol. 22, no. 3, pp. 1411-1421 (2021).
7. Shreshth Tuli, Giuliano Casale.: PreGAN: Preemptive Migration Prediction Network for Proactive Fault-Tolerant Edge Computing. IEEE Conference on Computer Communications (INFOCOM). IEEE (2022).
8. An J, Cho S. : Variational Autoencoder based Anomaly Detection using Reconstruction Probability (2015).
9. B. J. Beula Rani.: Survey on Applying GAN for Anomaly Detection. 2020 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, pp. 1-5 (2020).
10. Junfu Chen, Dechang Pi.: Imbalanced satellite telemetry data anomaly detection model based on Bayesian LSTM. Acta Astronautica, Volume 180, pp. 232-242, ISSN 0094-5765 (2021).
11. Peter Rousseeuw, Domenico Perrotta.: Robust Monitoring of Time Series with Application to Fraud Detection. Econometrics and Statistics, Volume 9, pp. 108-121, ISSN 2452-3062 (2019).
12. Meimei Ding, H. Tian.: PCA-based network Traffic anomaly detection. In Tsinghua Science and Technology, vol. 21, no. 5, pp. 500-509 (2016).

13. Min Hu.: A novel computational approach for discord search with local recurrence rates in multivariate time series. Information Sciences, Volume 477, pp. 220-233, ISSN 0020-0255 (2018).
14. Hossein Abbasimehr.: An optimized model using LSTM network for demand forecasting. Computers & industrial engineering 143, 106435 (2020).
15. Julien Audibert, Pietro Michiardi.: USAD: UnSupervised Anomaly Detection on Multivariate Time Series. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 3395–3404 (2020).
16. Dan Li.: MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In International Conference on Artificial Neural Networks. Springer, 703–716 (2019).
17. Hang Zhao, Yujing Wang.: Multivariate time-series anomaly detection via graph attention network. International Conference on Data Mining (2020)
18. Yuxin Zhang.: Unsupervised Deep Anomaly Detection for Multi-Sensor Time-Series Signals. IEEE Transactions on Knowledge and Data Engineering (2021).
19. Ailin Deng.: Graph neural network-based anomaly detection in multivariate time series. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35. 4027–4035 (2021).
20. Takaaki Nakamura, Makoto Imamura.: MERLIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives. In 2020 IEEE International Conference on Data Mining (ICDM). IEEE, 1190–1195 (2020).
21. Bo Zong, Qi Song.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In International Conference on Learning Representations (2018).
22. Kyle Hundman.: Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. 387–395 (2018).
23. Shaohan Huang. : HitAnomaly: Hierarchical Transformers for Anomaly Detection in System Log. IEEE Transactions on Network and Service Management 17, 4 (2020), 2064–2076 (2020).
24. A. A. Cook.: Anomaly Detection for IoT Time-Series Data: A Survey. In IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6481-6494 (2020).
25. Ahmad.: Unsupervised real-time anomaly detection for streaming data. Neurocomputing 262, 134-147(2017).
26. Dai, Enyan, Chen, Jie.: Graph-Augmented Normalizing Flows for Anomaly Detection of Multiple Time Series. United States. ICLR (2022).
27. Shukla, Satya Narayan.: Heteroscedastic Temporal Variational Autoencoder For Irregularly Sampled Time Series. ICLR (2021).
28. Tang, Wensi, Guodong Long.: Omni-Scale CNNs: a simple and effective kernel size configuration for time series classification. International Conference on Learning Representations, ICLR (2022).
29. Shin, Yooju, Susik Yoon.: Coherence-based Label Propagation over Time Series for Accelerated Active Learning. International Conference on Learning Representation, ICLR (2022).
30. T. Kieu.: Outlier Detection for Multidimensional Time Series Using Deep Neural Networks. 19th IEEE International Conference on Mobile Data Management (MDM), Aalborg, Denmark, pp. 125-134 (2018).