GDN-based Time Series Anomaly Detection using Time Series Clustering

Mingyu Lee , Seungho Jeon and Jung Take Seo[*]

Gachon University, Seongnam-daero 1342, Seongnam-si, Republic of Korea
cpsmgyu@gmail.com, {shjeon90, seojt}@gachon.ac.kr

Abstract

Cyberattacks targeting industrial control systems(ICS) have been steadily increasing and becoming more sophisticated. To detect such cyber-attacks, research has increased that trains AI models using datasets collected from low-level field devices to detect anomalies. However, datasets collected from cyber physical system(CPS) are large, making efficient learning difficult through general data preprocessing. Moreover, since the dataset collected from ICS possesses time series characteristics, an appropriate algorithm selection that considers this is essential. In this paper, we propose a method that employs time series clustering to lighten and preprocess the dataset, enabling cost-efficient learning. Additionally, we utilize the graph deviation network (GDN) algorithm to train on time series data and detect anomalies. In the experiments, we evaluate the anomaly detection performance using the WADI dataset, which relates to a water distribution system, using both time series clustering and GDN. We then compare this with existing research. The experimental results indicate that our method achieves higher performance in terms of precision and F1-score compared to prior studies.

Keywords: Time series clustering, anomaly detection, graph deviation network.

1 Introduction

A cyber physical system(CPS) is a system that processes and manages data from various complex tasks, processes, and information in the real world through network-connected devices. An industrial control system is a type of CPS that controls, monitors, and manages critical infrastructure such as water treatment, power plant, and smart grid. Recently, cyberattacks targeting industrial control systems have continuously increased, and as information technology advances, the attack techniques

[*] Corresponding author: Gachon University, Seongnam-si, 13120, Republic of Korea, Email: seojt@gachon.ac.kr

have become more sophisticated. For instance, Industroyer 2, a malware discovered by a Ukrainian energy company in April 2022, consists of wiper and ICS-enabled malware for Windows, Linux, and Solaris and uses the IEC-104 protocol to communicate with industrial devices [1]. Such malware can cause physical damage by controlling and malfunctioning industrial devices. Recently, in order to detect cyberattacks targeting CPS, research has been conducted to detect anomaly by learning to AI models using datasets collected from field devices such as sensors. However, numerous field devices exist in CPS, like smart factories, water treatment, and power plants. The sheer volume of data collected from these devices leads to significant costs during the preprocessing stage, making it challenging to train models efficiently. Additionally, existing graph-based deep learning algorithms don't consider sequential characteristics, limiting their effectiveness in handling time-series data.

To address these challenges, we use time series clustering and GDN [2]. In this paper, we employ time series clustering to lighten large sensor datasets and then use the GDN algorithm to construct an anomaly detection model. After performing time series clustering through the proposed method and subsequently training the model, the results showed a precision of 0.99, a recall of 0.54, and an F1-score of 0.70. Our approach demonstrated improved results when compared to the performance of the anomaly detection model of existing research using GDN on multivariate time-series datasets.

The contributions of this paper are as follows:

• We preprocessed datasets using time series clustering to train a model efficiently.
• We verified the improvement in model performance due to time series clustering.
• We overcame the limitations of graph-based algorithms using the GDN algorithm.
• We performed anomaly detection on time-series data using time series clustering and GDN, achieving higher performance than other models of existing research.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 provides the design of a GDN-based time series anomaly detection framework. Section 4 conducts experiments to address two research questions (RQs). Conclusion and future work are presented in Section 5.

2 Related work

2.1 Time-series

Time series refers to data measured sequentially at regular time intervals, and a typical example of time series collected in industrial control systems is sensor measurements. Time series can be divided into univariate time series and multivariate time series based on the number of variables. Univariate time series monitors and records a single variable, such as temperature, pressure, or flow data collected from a specific sensor. On the other hand, multivariate time series simultaneously monitors and records multiple variables, for instance, temperature, pressure, and flow data collected from multiple sensors. Additionally, since multivariate time series possess correlations among multiple variables, they are utilized in analyses for process stability monitoring, anomaly detection, and energy efficiency improvement, considering these correlations.

2.2 MTS anomaly detection

MTS anomaly detection methods learn sequential features of MTS. [3] proposes GRN, an interpretable MTS anomaly detection method based on neural graph networks and gated recurrent units. [4] proposes GLUE, a graph deviation network with local uncertainty estimation that is based on the recently proposed GDN. [5] proposes an unsupervised anomaly detection method for MTS using long short-term memory network(LSTM) and graph convolutional network(GCN). [6] proposes an end-to-end physics-informed gated recurrent graph attention unit network (PGRGAT) to detect anomalies of interconnected sensors and actuators in CPS. [7] proposes a self-supervised framework for MTS anomaly detection that considers each univariate time series(UTS) as an individual feature and includes two graph attention layers in parallel to learn the complex dependencies of MTS in both temporal and feature dimensions. [8] proposes OmniAnomaly, a stochastic recurrent neural network(RNN) for MTS anomaly detection that captures the normal patterns of MTS by learning robust representations.

3 Design of GDN-based Time Series Anomaly Detection Framework

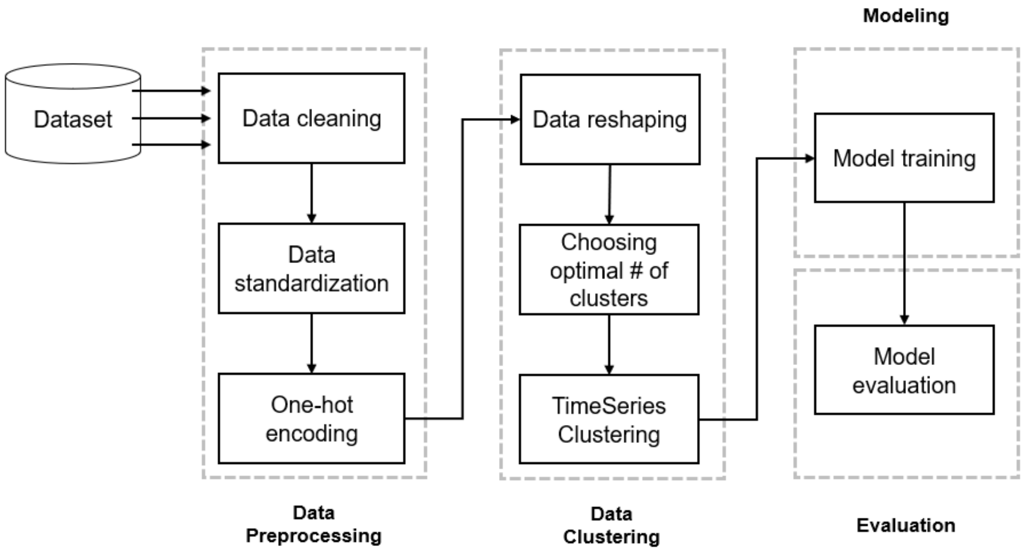

Figure 1: GDN-based time series anomaly detection framework

In this section, we will describe the GDN-based time series anomaly detection framework as well as the method of multivariate time series clustering.

3.1 Overview

Figure 1 provides the overall architecture of the GDN-based time series anomaly detection framework. It consists of four parts as follows: (1) Data Preprocessing. (2) Data Clustering. (3) Model Training and Testing. (4) Evaluation. The details of each part are explained in Section 3.2 ~ Section 3.5.

3.2 Data Preprocessing

A multivariate time series dataset collected from various field devices in industrial control systems (e.g., controllers, sensors) can contain missing or immutable values. Additionally, for development convenience and to facilitate advantageous model training, it is essential to preprocess the dataset using data cleaning, standardization, and one-hot encoding.

Data cleaning is the process of removing incomplete, erroneous, or missing values to enhance the accuracy of the analysis and enable proper model training. In this process, rows or columns filled with missing values, as well as columns that have only a single unique value, are deleted.

Data standardization is a technique that adjusts the range or distribution of data to a uniform scale, making it easier for computers to understand and utilize. In the proposed framework, float data is standardized using equation (3). Standardization is carried out column-wise, where x_i represents the data at index i, μ is the mean of column x, σ is the standard deviation of column x, and z_i is the standardized value of x_i.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1}$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{2}$$

$$z_i = \frac{x_i - \mu}{\sigma} \tag{3}$$

One-hot encoding is a method that converts categorical variables into binary vectors composed of 0 and 1, allowing computers to effectively process categorical data.

3.3 Data Clustering

When clustering general datasets, one typically defines data groups based on their characteristics and identifies the central points of these groups. However, sequential information can be lost when clustering time series data using general clustering algorithms. Therefore, a suitable clustering method that considers correlations between time series samples is needed to handle these data. In this section, the preprocessed dataset is reshaped for clustering. Subsequently, time series clustering is iteratively performed to determine the optimal number of clusters that best classify the data. After clustering the voluminous time series data through time series clustering, data points located far from the centroid beyond a threshold in each cluster are removed, as described in Algorithm 1, to lighten the dataset.

Algorithm 1. Algorithm to lighten dataset after time series clustering

```
Input:
  $\mathcal{J}$: data point in cluster
  $\mathcal{L}$: centroid in cluster

1. Repeat
2.   $d \leftarrow$ get a difference $\mathcal{J}$ and $\mathcal{L}$
3.   if $d \geq$ threshold then
4.     drop $\mathcal{J}$ from cluster
5.   end if
6. Until performed for each $\mathcal{J}$ of the clusters
```

3.4 Modeling

In this section, the lightened dataset is split into training and testing sets and then modeled using the GDN algorithm. GDN, a modified approach of unsupervised learning based GNN, learns a graph of relationships and detects deviations. This algorithm is suitable for training ICS time series datasets collected from various field devices since it applies unique model parameters for each sensor. GDN employs a predictive approach, where it anticipates the expected behavior of each sensor based on the learned graph or pattern. An anomaly detection model is generated after performing model training on the training dataset using the GDN algorithm. This model is then validated using a test dataset labeled with attacks.

3.5 Evaluation

In the evaluation part, the trained model is validated using the test dataset. A model trained solely on normal data predicts attacks using a test dataset including attack data and labels. The success of the prediction is verified based on the attack labels. Performance is derived based on the confusion matrix results.

4 Experiments

In this section, we conduct two experiments to evaluate the proposed framework. Specifically, we aim to answer the following questions:

- RQ1: How does the threshold, used as a criterion for dropping data points after time series clustering, influence the performance of anomaly detection?
- RQ2: What performance difference exists between the anomaly detection model trained with the optimal threshold and models from existing research?

4.1 Experimental Setup

We implement the proposed method and model in PyTorch version 1.5.1 with CUDA 10.2. We conduct experiments on a server equipped with Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz and Tesla V100S GPU. Table 1 describes the parameters for model training.

parameter	Basic value	Rerun value
Batch size	128	256
Epoch	100	50
Sliding window size	15	15

Table 1: Parameters for anomaly detection model training

4.2 Datasets

As research using data collected from ICS has increased, various ICS datasets like BATADAL [9], SWaT [10], WADI [11], and HAI [12] have been released. We use the WADI dataset from iTrust, which was collected from actual sensors of a Water Distribution Infrastructure for security research. This dataset consists of a multivariate time-series dataset composed of normal operation data, WADI_14days.csv, collected over 14 days and abnormal operation data, WADI_attackdata.csv, collected through 15 attacks over 2 days, as detailed in Table 2. We preprocessed the WADI dataset using four data processing techniques. The Preprocessed.csv is the output obtained by concatenating

WADI_14days.csv and WADI_attackdata.csv and then processing them. After conducting time series clustering, we divided it into train.csv and test.csv.

WADI dataset	# of columns	# of rows
WADI_14days.csv	127	1,209,600
WADI_attackdata.csv	127	172,800
Preprocessed.csv	139	1,382,350
├ train.csv	139	943,960
└ test.csv	139	438,390

Table 2: Statistics of the WADI and preprocessed dataset

- Missing Value Removal: We used the pandas library to drop columns composed solely of missing values and then dropped rows with missing values.
- Standardization: We implemented standardization and applied it to columns with a data type of float64.
- One-hot encoding: We implemented one-hot encoding and applied it to columns with a data type of int64.
- Time series clustering: Using TimeSeriesKMeans from tslearn, we conducted clustering on train.csv. To find the optimal number of clusters, we measured the sum of distances between data points and centroids for each cluster, starting from 2 clusters up to 20 clusters. The result was as depicted in Figure 2. We determined the optimal number of clusters to be 11, where the rate of decrease was at its smallest. The clustering results when the number of clusters is 11 (n_clusters = 11) are shown in Figure 3.

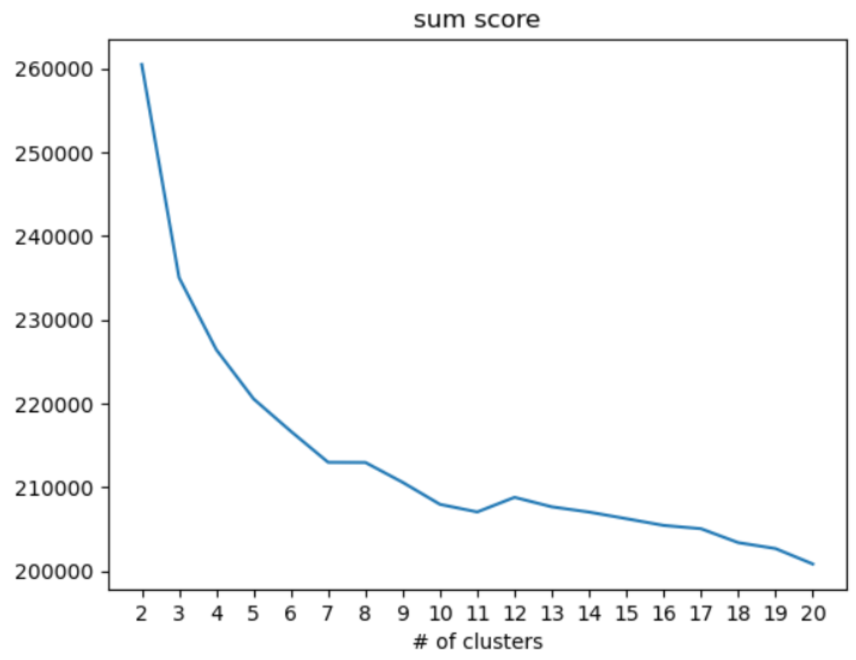

Figure 2: Sum of distances between data points and centroids based on the number of clusters

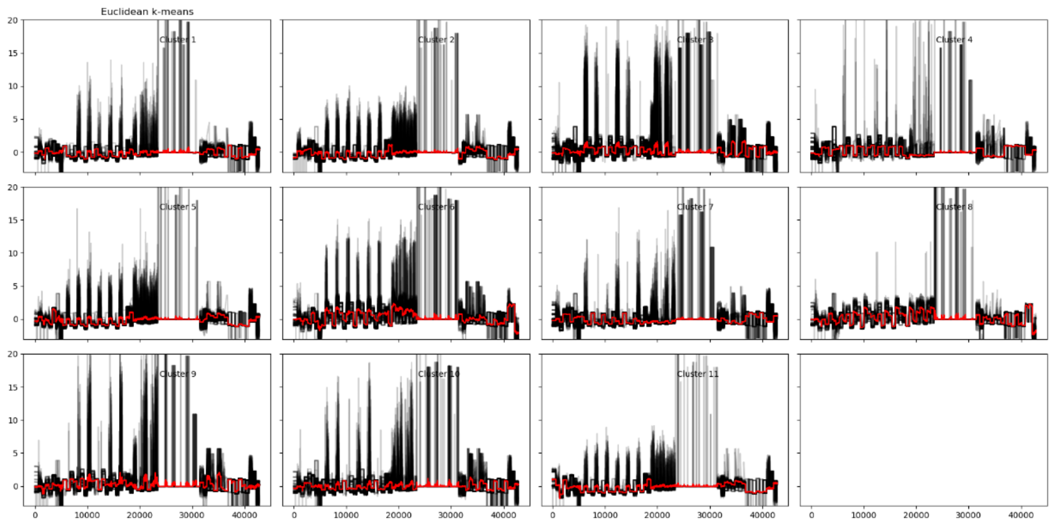

Figure 3: Clustering results of the WADI train dataset (n_clusters=11)

4.3 Evaluation Metrics

The proposed framework employs time series clustering to lighten the dataset followed by anomaly detection using the GDN-based model. Therefore, we use compression ratio, precision, recall, and F1-score to evaluate the performance of the proposed method and model.

The compression ratio is calculated based on the number of rows in the dataset before and after time series clustering, as shown in equation (4). A higher compression ratio indicates that the dataset has been reduced in size through preprocessing.

$$Compression\ ratio = \left(1 - \frac{\text{preprocessed rows}}{\text{original rows}}\right) * 100 \tag{4}$$

Precision, as defined by equation (5), is the ratio of instances correctly predicted as anomalies to all instances predicted as anomalies. Recall, as described in equation (6), is the ratio of instances correctly predicted as anomalies to all actual anomalies. The harmonic mean of these two metrics is the F1-score, as given by equation (7). These are the primary metrics used to evaluate the performance of an anomaly detection model.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{5}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{6}$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

4.4 Experimental Results

- RQ1: How does the threshold, used as a criterion for dropping data points after time series clustering, influence the performance of anomaly detection?

We adjusted the explanatory variable, threshold, from 100 to 60 and measured performance based on the confusion matrix for the following test cases:

- Test case 1: Using a dataset of 943,960 rows without performing clustering, trained with GDN.
- Test case 2: From 11 clusters, we removed the farthest 10% of data points from each cluster's centroid, resulting in a dataset of 846,955 rows.
- Test case 3: From 11 clusters, we removed the farthest 20% of data points from each cluster's centroid, resulting in a dataset of 757,978 rows.
- Test case 4: From 11 clusters, we removed the farthest 30% of data points from each cluster's centroid, resulting in a dataset of 663,649 rows.
- Test case 5: From 11 clusters, we removed the farthest 40% of data points from each cluster's centroid, resulting in a dataset of 568,650 rows.

Table 3 displays the performance based on the confusion matrix for the five test cases. After applying time series clustering, precision improved in test cases 2 and 3, while recall decreased; however, the f1-score remained consistent with the results prior to clustering. Additionally, it was observed that surpassing a certain threshold, as seen in test case 5, led to a significant drop in performance.

Test case	Compression ratio	Precision	Recall	F1-score	Threshold
1	0%	0.93	0.56	0.70	100
2	10.3%	0.99	0.54	0.70	90
3	19.8%	0.97	0.55	0.70	80
4	29.7%	0.89	0.58	0.70	70
5	39.8%	0.53	0.72	0.61	60

Table 3: Performance based on compression ratio and confusion matrix according to threshold

- RQ2: What performance difference exists between the anomaly detection model trained with the optimal threshold and the models from existing related studies?

To make the most comparable comparison, we gathered studies that trained anomaly detection models based on GDN using the WADI dataset. Table 4 shows the results comparing confusion matrix-based performance evaluation targeting our proposed model and models from the collected existing studies. In the case of our proposal, precision and f1-score were measured the highest.

Works	Precision	Recall	F1-score
C. Tang et al. (2023)	0.29	0.79	0.42
S. Ray et al. (2022)	0.63	0.66	0.64
W. Wu et al. (2023)	0.32	0.33	0.27
D. Ailin et al. (2021)	0.97	0.40	0.56
Our proposal (test case 2)	0.99	0.54	0.70

Table 4: The performance of the GDN method based on the WADI dataset compared to other methods in related works

5 Conclusion and Future Work

In this paper, we have proposed a GDN-based anomaly detection method using time series clustering. The proposed method has been proven to provide better performance for time series dataset in comparison with other methods. Additionally, we found that the anomaly detection model, when applied with time series clustering, can yield high performance with a lighter dataset.

However, the generally lower recall performance of the proposed method needs to be improved for real-world applications. As part of future work, we plan to apply time series clustering to other datasets and algorithms other than GDN. Through this, we aim to demonstrate the effectiveness of time series clustering by conducting experiments to confirm dataset reduction and model performance enhancement.

Acknowledgements

This research was supported by the "Development of Cyberattack Detection Technology for New and Renewable Energy Control System Using AI (Artificial Intelligence)" of the Korea WESTERN POWER(KOWEPO) (2022-Commissioned research-11).

References

[1] Welivesecurity by eset (2022, April). *Industroyer2: Industroyer reloaded.* Retrieved from https://www.welivesecurity.com/2022/04/12/industroyer2-industroyer-reloaded

[2] D. Ailin and H. Bryan. (2021). Graph Neural Network-Based Anomaly Detection in Multivariate Time Series. In *Proceedings of the 21th Association for the Advancement of Artificial Intelligence(AAAI)* (pp. 4027-4035).

[3] C. Tang, L. Xu, B. Yang, Y. Tang and D. Zhao. (2023). GRU-Based Interpretable Multivariate Time Series Anomaly Detection in Industrial Control System. Computers&Security, 127(103094).

[4] S. Ray, S. Lakdawala, M. Goswami and C. Gao. (2021). Learning graph neural networks for multivariate time series anomaly detection. arXiv preprint arXiv:2111.08082.

[5] R. Qi, D. Li and S. -K. Ng. (2022). MAD-SGCN: Multivariate Anomaly Detection with Self-learning Graph Convolutional Networks. In *Proceedings of the 38th International Conference on Data Engineering(ICDE)* (pp. 1232-1244).

[6] W. Wu, C. Song, J. Zhao and Z. Xu. (2023). Physics-informed gated recurrent graph attention unit network for anomaly detection in industrial cyber-physical systems. Information Sciences, 629, pp. 618-633.

[7] H. Zhao, Y. Wang, J. Duan, C. Huang, D. Cao, Y. Tong, B. Xu, J. Bai, J. Tong and Q. Zhang. (2020). Multivariate time-series anomaly detection via graph attention network. In *Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM)* (pp. 841-850). IEEE.

[8] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun and D. Pei. (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).

[9] iTrust. (2022). BATtle of Attack Detection Algorithms (BATADAL). Retrieved from https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/

[10] iTrust. (2022). SecureWater Treatment (SWaT). Retrieved from https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_swat/

[11] iTrust. (2022). Water Distribution (WADI). Retrieved from https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_wadi/

[12] H.K. Shin, W. Lee, J.H. Yun and H.C. Kim. (2020). HAI 1.0: HIL-Based Augmented ICS Security Dataset. In *Proceedings of the 13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20)*