Autoencoder-based Anomaly Detection for Time Series Data in Complex Systems

Xundong Gong
State Grid Jiangsu Electric Power Company
Nanjing, China
gxd@js.sgcc.com.cn

Shibo Liao
Hangzhou Dianzi University
Hangzhou, China
211080010@hdu.edu.cn

Fei Hu
State Grid Wuxi Power Supply Company
Wuxi, Chinay
hufei@js.sgcc.com.cnD

Xiaoqing Hu
State Grid Wuxi Power Supply Company
Wuxi, China
huxq.wx@js.sgcc.com.cn

Chunshan Liu
Hangzhou Dianzi University
Hangzhou, China
chunshan.liu@hdu.edu.cn

2022 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS) | 978-1-6654-5073-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/APCCAS55924.2022.10090260

***Abstract*—In this paper, we present a new anomaly detection method for time-series data in complex systems such as power grid and cellular networks. The proposed anomaly detection method is developed following unsupervised learning, where an AutoEncoder based on Gated Recurrent Units (GRU-AE) is trained to reconstruct a time-series of interest, and anomalies are detected via detecting exceptionally large reconstruction errors. A multi-timestamp stacking method is adopted to reduce the number of time steps in the GRU-AE to facilitate the training of the model and a new training scheme with random shuffling is proposed to prevent overfitting. The proposed GRU-AE based detector is applied in multiple time scales to detect different types of anomalies. Numerical results obtained via time-series data from real cellular network demonstrate the performance of the proposed method.**

***Index Terms*—time series, anomaly detection, AutoEncoder, multi-timestamp stacking, random shuffling.**

I. Introduction

An anomaly in time-series often refers to an unexpected change in pattern in one or a collection of data points. Anomalies in time-series such as outliers can make adverse effect on data analysis which are desirable to be detected when they occur. Anomalies themselves may contain useful information on the underlying systems, which, upon detection, can be used for fault diagnosis [1] and early intervention of developing faults [2]. For instance, time-series anomaly detection has been applied in smart grid to detect cyber attacks [3].

Depending on the anomalous characteristics relative to the time-series, time-series anomalies can generally be divided into three categories [4]: 1) Point Anomaly, 2) Contextual Anomaly and 3) Collective Anomalies. A point anomaly is a single anomalous data point that does not conform to the normal behaviours of the entire time-series, e.g., an extremely large/small outlier. A contextual anomaly is anomalous in a specific context, which may not be anomalous in a different context. Collective anomalies are collections of anomalous data points, i.e., change point [5]. In this work, we focus on the anomaly detection problem for time-series data in large-scale complex systems. Examples of such system include cellular networks, wireless sensor networks, power grids and etc.

Among the existing anomaly detection techniques for time-series [6], deep-learning based methods have gained more and more attention [7]. However, as anomalies are ofter rare, it is difficult to obtain enough number of anomalous instances to train a deep-learning model in a supervised manner. Additionally, it is difficult if not impossible to be exhaustive on the types of anomalies. This makes it more difficult to adopt supervised learning for anomaly detection. In this context, unsupervised learning models building on AutoEncoders (AE) have become popular choices. The main principle of AE-based anomaly detection is to train an AE to reconstruct normal time-series, and to detect anomalies once the reconstruction errors become exceptionally large [8], [9].

In order to detect different types of anomalies in large-scale complex systems such as cellular networks and power grids, the AE may need to take a relatively long sequence as input such that seasonal characteristics can be learnt by the AE. However, when the number of time steps requires to input is large, it may become difficult to train the AE [8], [9] and consequently may limit the detection capability of different anomalies. Inspired by this observation, we in this work proposed a new Gated Recurrent Units based AutoEncoder (GRU-AE) with multiple time steps per node and an improved training method for time-series anomaly detection. The proposed anomaly detector operates at two different time scales to detect different types of anomalies. We validate the proposed method using data collected in large scale cellular networks.

II. Preliminaries

Before presenting our proposed GRU-AE anomaly detector, we first briefly introduce some essential elements of the proposed model.

This work was supported by the Science and Technology Project of State Grid Corporation of China (SGJSWX00KJJS220847).

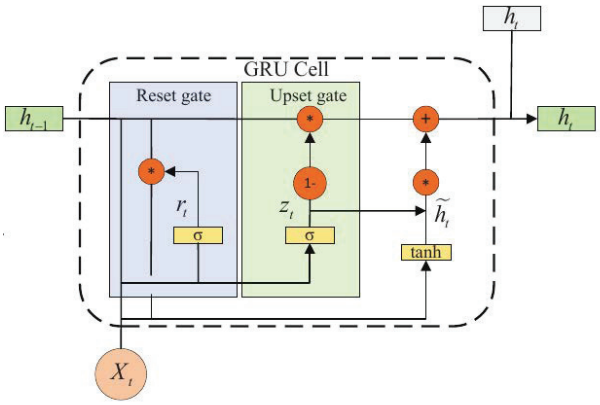


Fig. 1. An illustration of the structure of a GRU unit.

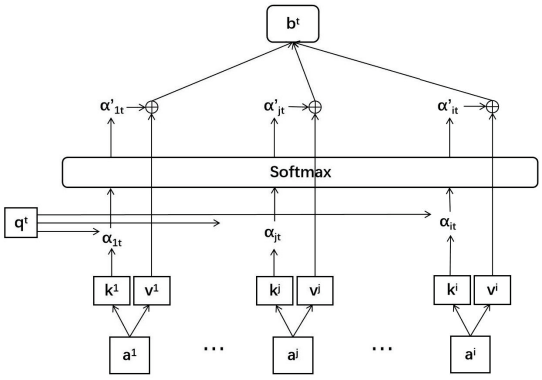


Fig. 2. An illustration of the computation flow of the attention unit.

A. Autoencoder

An AE [10] consists of two modules, the *encoder* and the *decoder*. Suppose the input of the encoder is $\mathbf{X} = \{x_1, x_2, \ldots, x_m\}$. The encoder maps the m-dimensional input vector $\mathbf{X}$ to an n-dimensional vector $\mathbf{H}$, where n is typically smaller than m. The decoder takes vector $\mathbf{H}$ as input and maps it back to the m-dimensional space to reconstruct $\mathbf{X}$. Denote the output of the decoder as $\mathbf{X}' = \{x'_1, x'_2, \ldots, x'_m\}$, the mapping at the encoder as $\mathbf{H} = f(\mathbf{X})$ and the mapping at the decoder as $\mathbf{X}' = g(\mathbf{H})$. Then the goal of the AE is to train two appropriate functions $f(\cdot)$ and $g(\cdot)$ approximated by deep-learning models to minimise the reconstruction error $\mathbb{E}_{\mathbf{X}} |\mathbf{X} - \mathbf{X}'|^2$.

B. GRU-autoencoder

GRU is a variant of recurrent neural networks (RNN) [11]. Each GRU unit has two gates, the update gate and the reset gate. The role of the update gate is to control the amount of information from the previous states to be retained in the current state while the reset gate controls the amount of information to be forgotten. Fig. 1 illustrates the structure of a GRU unit, where the information flow can be described as follows:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r), \tag{1}$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z), \tag{2}$$

$$\widetilde{h_t} = tanh(W \cdot [r_t * h_{t-1}, x_t] + b_h), \tag{3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t}. \tag{4}$$

Here σ denotes logistic sigmoid function, X_t the input at the current moment, h_t the state, W the weight matrix (e.g. W_r is the reset gate weight matrix), b the bias vector (e.g. b_r is the reset gate bias).

GRU is commonly used for time-series modeling. A GRU-AE uses GRU at both the encoder and the decoder. Compared to other AE, the GRU-AE shows better results in modeling time-series data.

C. Attention Mechanism

In order to improve the efficiency of GRU-AE [12], the attention mechanism is applied to GRU-AE here. The core idea of the attention mechanism is to let the decoder give different levels of attention to each hidden state of the encoder at each time step. Suppose the input to the attention unit are $\{a^1, \ldots, a^j, \ldots, a^i\}$, then the output are the same number of samples $\{b^1, \ldots, b^t, \ldots, b^i\}$. Fig. 2 illustrates the process of generating the output samples $\{b^1, \ldots, b^t, \ldots, b^i\}$, where is summarised as follows:

$$k^j = W^k a^j, \tag{5}$$

$$v^j = W^v a^j, \tag{6}$$

$$\alpha_{jt} = V^T tanh(W^s \cdot [q^t, k^j]), \tag{7}$$

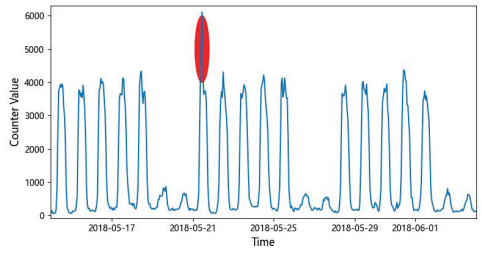
$$\alpha'_{jt} = \frac{e^{\alpha_{jt}}}{\sum_{l=1}^{i} e^{\alpha_{lt}}}, \tag{8}$$

$$b^t = \sum_j \alpha'_{jt} \cdot v^j. \tag{9}$$

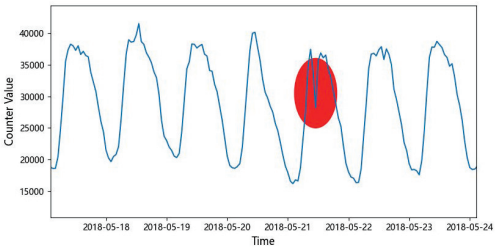
Here q^t is the current matching query, W and V are the weight matrix. The attention mechanism is to calculate the attention score α_{jt} from the query value q^t and the input a^j, and then obtain the weight coefficients α'_{jt} by the softmax function. Finally the output b^t, $\forall t \in \{1, \ldots, i\}$ is calculated based on the weight coefficient α'_{jt} and all the inputs.

III. AutoEncoder (AE)-based Anomaly Detector

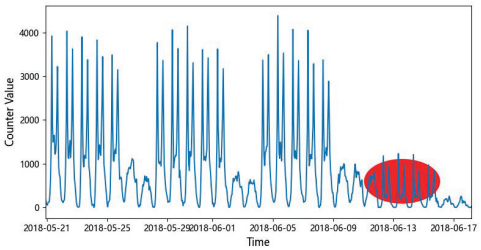
Our goal is to detect anomalies from a dataset that contains massive traffic load time-series in a large scale cellular network. Fig. 3 gives three representative examples of the time-series and the anomalies that are commonly found in the dataset. As can be seen from Fig. 3, the traffic load has clear periodicity of 24 hours or 1 week, due to the underlying force driven by human activities. Fig. 3 (a) shows an example of a point anomaly (marked with red color) which has exceptionally large values compared to other parts of the sequence. Fig. 3 (b) has a contextual anomaly which violates



(a)



(b)



(c)

Fig. 3. Examples of time series and anomalies: (a) Point Anomaly. (b) Contextual Anomalies. (c) Collective Anomalies.

the regular fluctuation pattern during a day while Fig. 3 (c) shows a collective anomaly with a change point.

In this section, we detail our proposed GRU-AE based anomaly detection method.

A. GRU-AE based anomaly detector

By using the GRU to build the AE and adding the attention mechanism, our proposed GRU-AE based model is shown in Fig. 4. Denote the input of length i as x_t, $t \in \{1, \ldots, i\}$, and the output as x'_t. As illustrated in the figure, at the encoder stage, the hidden state $N_t^{en} (2 \leq t \leq i)$ is related to the input x_t at moment t and the hidden state N_{t-1}^{en} at moment $t-1$. The hidden state N_1^{en} is related only to the input at moment $t = 1$. The attention mechanism takes the hidden states of the encoder as input, whose output is then fed to the decoder. It is noted that the final hidden state of the encoder, i.e., N_i^{en}, is directly fed to the first GRU unit of the decoder as its initial hidden state. In the decoder, the decoding is performed in the reverse order. That is, the decoder outputs x'_i first but x'_1 the last.

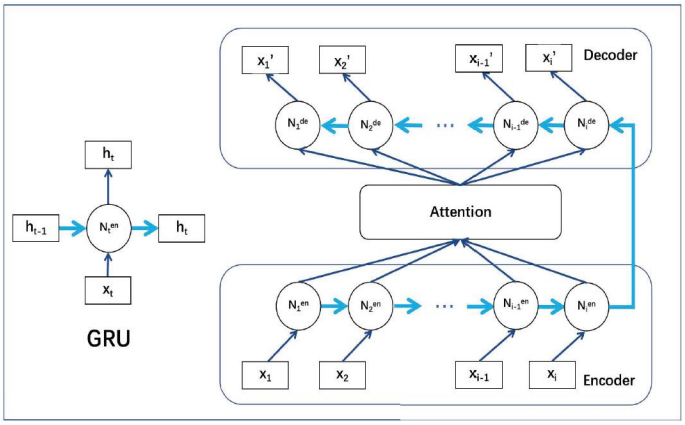


Fig. 4. The architecture of the proposed GRU-AE model.

The GRU-AE based anomaly detector works as follows. Once the GRU-AE model is trained, it takes x_t, $t \in \{1, \ldots, i\}$ as the input and outputs x'_t, $t \in \{1, \ldots, i\}$. The reconstruction errors $e_t = x'_t - x_t$ are then calculated. Denote ϵ_{low} and ϵ_{up} as two detection thresholds. If $e_t < \epsilon_{low}$ or $e_t > \epsilon_{up}$, an alarm of anomaly is raised and the data sample t is labeled as an anomalous point. The thresholds ϵ_{low} and ϵ_{up} are determined according to the reconstruction errors in the training data, which need to be chosen carefully to strike a balance between low false alarm rate and low false missing rate.

To detect different types of anomalies as illustrated in Fig. 3, the proposed GRU-AE based detector is operated at two different time scales, namely, on a single-day and multi-day basis, respectively. Specifically, in the single-day operation, the model takes consecutive data points of a single day as input. The purpose is to detect point anomalies (e.g., Fig. 3(a)) as well as short-term contextual anomalies (e.g., Fig. 3(b)). The multi-day detector takes multiple weeks of data as the input, with the aim to detect collective anomalies (e.g., Fig. 3(c)) that affect for one day or longer periods. Since the training and application of the single-day model are straightforward, we focus on the multi-day model in the remaining parts of this section.

B. Training of the multi-day GRU-AE anomaly detector

For the multi-day GRU-AE model, a relatively long period of data is needed so as to capture the seasonal variations of the sequences. In conventional approaches, the data is fed into the network directly in its original time scale. For instance, if the data is sampled hourly, then each time step in the GRU network is 1 hour. However, such an approach may not be able to capture the long-term characteristics of the data when the input period is too long. Motivated by this observation, we instead stack the data points of the same day into a vector and feed the vector into one time step of the GRU network. In other words, the time step of the GRU becomes one day, rather than the original time granularity of the sequence. With this modification, the GRU can learn the fluctuation pattern within a day as well as the seasonality across days.

When training the multi-day GRU-AE model, the following modification is applied to prevent overfitting: the output

sequence of vectors are shuffled with a probability of ξ. Specifically, the random shuffle operation is applied independently with a probability of ξ to each batch in each epoch. With this operation, it is expected that a fraction of $1-\xi$ training samples use the original order to calculate the gradient, while the other ξ samples use a random order. It is noted that such a method works only when the data on different days share similar characteristics. In other words, in order for the above mentioned multi-day GRU-AE model to work, the time-series data requires to be pre-examined and partitioned. Days sharing similar characteristics, e.g., working days, are fed together to the model.

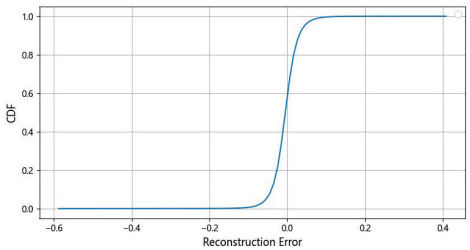
IV. Numerical Results

In this section, we evaluate the performance of the proposed GRU-AE anomaly detector using a dataset collected by a cellular service provider. The dataset used in this work contains time-series of a single counter collected at over 2,700 locations from January 15, 2018 to July 8, 2018. The original data has only 0.071% of missing points, thus linear interpolation is adopted to fill the missing points. We randomly select data from 200 locations to train the GRU-AE models and to determine the test thresholds ϵ_{low} and ϵ_{up}, and use the rest to test the performance of the anomaly detector.

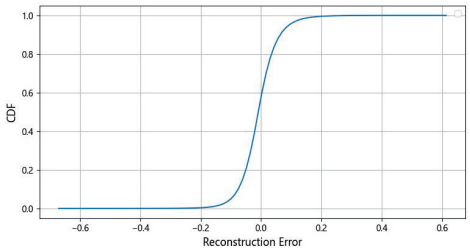
As mentioned in Sec. III-A, the proposed GRU-AE is applied at two time-scales, including the single-day and multi-day. In the multi-day operations, since it is commonly found that the data exhibits different characteristics on weekdays and on weekends, the mutli-day GRU-AE is trained to work for weekdays and for weekends separately. Specifically, for the GRU-AE operating on weekdays, the number of consecutive weekdays is set to $t_{work} = 10$ days, i.e., two weeks of weekdays. For the GRU-AE operating on weekends, the number of consecutive weekends is set to $t_{weekend} = 6$ days, i.e., three weeks of weekends. For the attention unit in the GRU-AE model, the weight matrices W^v and W^s are set to the identity matrix. Other model parameters are given in Table I for the single-day and multi-day (weekday and weekend) of the GRU-AE models. The models are trained with the Adam optimiser with the learning rate set to 0.003, the batch size set to 32 and epoch set to 50. The platform used to build the model is tensorflow-2.5 and python-3.9. The data are standadised before being fed to the model for training.

Fig. 5 presents the cumulative distribution function (CDF) of the reconstruction errors of the models on the training dataset. As can be seen from the CDF plots, the trained models yield relatively low reconstruction errors for most of the training samples. The test thresholds ϵ_{up} and ϵ_{low} are picked from the right and left tail of the distributions of the errors, respectively. The corresponding choices, measured as the percentile of the reconstruction errors, are listed in Table II.

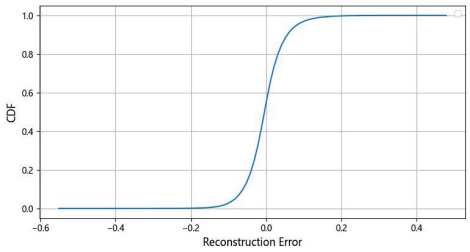
Fig. 6 presents three examples on the results of our anomaly detection, where the red dots indicate the anomalies being detected. From the detection results, we can see that our model can accurately detect anomalies in the data.



(a)



(b)



(c)

Fig. 5. CDF of the reconstruction errors on the training data: (a) Single-day. (b) Multi-day (weekday). (c) Multi-day (weekend).

Since anomalies account for only a small fraction of the data, using Precision alone is not sufficient to evaluate the performance of the proposed method. Hence, Recall and F-scores are also included as performance metrics. The three performance metrics are calculated as:

$$Precision = \frac{TP}{TP + FP}, \tag{10}$$

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

$$F - score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{12}$$

where TP denotes true positives, FP false positives, and FN false negatives. To evaluate the performance metrics, we establish the ground truth and estimate the value of TP, FP and FN by randomly visual inspection of over 2,500 time-series in the test dataset. Our estimates of the Precision, Recall and the F-score are 96.5%, 93.1% and 94.8%, respectively.

V. Conclusions

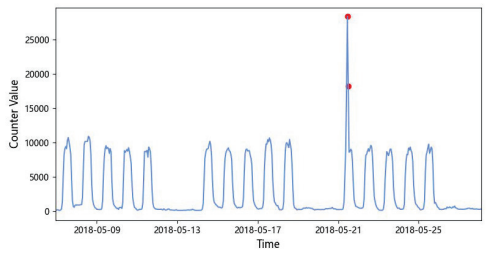
In this work, we have proposed an GRU-AE based anomaly detector to detect anomalies from time-series of complex

TABLE I
PARAMETER SETUP FOR THE GRU-AE MODELS

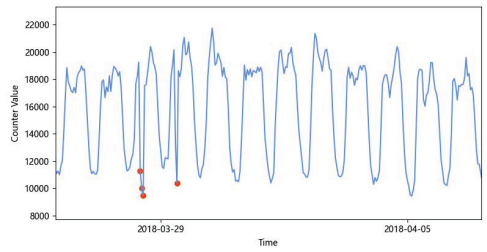
Time-scale	Dimension of k^i	Dimension of α_i	No. Hidden nodes (GRU)	Random shuffle prob. (ξ)
Single-day	5	1	3	0
Multi-day(weekday)	4	1	4	0.2
Multi-day(weekend)	8	1	6	0.3

TABLE II
DETECTOR THRESHOLD ON THE RECONSTRUCTION ERROR

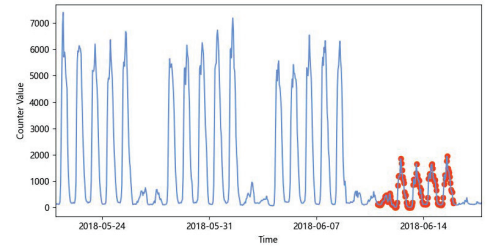
	Single-day		Multi-day(weekday)		Multi-day(weekend)	
Threshold	ϵ_{up}	ϵ_{low}	ϵ_{up}	ϵ_{low}	ϵ_{up}	ϵ_{low}
Percentile	0.20%	0.10%	0.40%	0.10%	0.20%	0.10%



(a)



(b)



(c)

Fig. 6. Examples of anomaly detection: (a) Point Anomaly. (b) Contextual Anomalies. (c) Collective Anomalies. The detected anomalies are marked with red color.

networks that exhibit regular daily and weekly fluctuation patterns. A multi-timestamp stacking has been adopted to reduce the number of time steps in the GRU to better learn the daily and long-term variation pattern of the time-series. A training algorithm based on random shuffling has also been developed to prevent overfitting. The proposed GRU-AE detector has been applied at both single-day and multi-day time scales to detect diverse anomalies. Numerical results obtained from real dataset of contemporary cellular networks have verified the effectiveness of the proposed method.

REFERENCES

[1] W. Li, W. Zhou, Y. M. Wang, C. Shen, X. Zhang, and X. Li, "Meteorological radar fault diagnosis based on deep learning," in *2019 International Conference on Meteorology Observations (ICMO)*. IEEE, 2019, pp. 1–4.

[2] M. Wadi and W. Elmasry, "An anomaly-based technique for fault detection in power system networks," in *2021 International Conference on Electric Power Engineering–Palestine (ICEPE-P)*. IEEE, 2021, pp. 1–6.

[3] L. Zhang, X. Shen, F. Zhang, M. Ren, B. Ge, and B. Li, "Anomaly detection for power grid based on time series model," in *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*. IEEE, 2019, pp. 188–192.

[4] W. Cui and H. Wang, "Anomaly detection and visualization of school electricity consumption data," in *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*. IEEE, 2017, pp. 606–611.

[5] T. Zhu, P. Li, L. Yu, K. Chen, and Y. Chen, "Change point detection in dynamic networks based on community identification," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 2067–2077, 2020.

[6] P. Zhao, X. Chang, and M. Wang, "A novel multivariate time-series anomaly detection approach using an unsupervised deep neural network," *IEEE Access*, vol. 9, pp. 109 025–109 041, 2021.

[7] H. Dhole, M. Sutaone, and V. Vyas, "Anomaly detection using convolutional spatiotemporal autoencoder," in *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. IEEE, 2019, pp. 1–5.

[8] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," in *2018 19th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2018, pp. 125–134.

[9] J. Guo, G. Liu, Y. Zuo, and J. Wu, "An anomaly detection framework based on autoencoder and nearest neighbor," in *2018 15th International Conference on Service Systems and Service Management (ICSSSM)*. IEEE, 2018, pp. 1–6.

[10] Q. Fournier and D. Aloise, "Empirical comparison between autoencoders and traditional dimensionality reduction methods," in *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. IEEE, 2019, pp. 211–214.

[11] D.-J. Choi, J.-H. Han, S.-U. Park, and S.-K. Hong, "Comparative study of cnn and rnn for motor fault diagnosis using deep learning," in *2020 IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA)*. IEEE, 2020, pp. 693–696.

[12] F. Meng, P. Lu, J. Li, T. Hu, M. Yin, and F. Lou, "Gru and multi-autoencoder based insider threat detection for cyber security," in *2021 IEEE Sixth International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2021, pp. 203–210.