China Instrument and Control Society
Instrumentation
INSTRUMENTATION, Vol. 11, No. 2, June 2024 https://doi.org/10.15878/j.instr.202400165

Article

Hybrid Gaussian Network Intrusion Detection Method Based on CGAN and E-GraphSAGE

Xinyi Liang, Hongyan Xing*, Wei Gu, Tianhao Hou, Zhiwei Ni, Xinyi Wang

School of Electronics Information Engineering, Nanjing University of Information Science & Technology, Nanjing 210044

* Corresponding author email: xinghy@nuist.edu.cn

Abstract: The rapid development of the Internet of Things (IoT) and modern information technology has led to the emergence of new types of cyber-attacks. It poses a great potential danger to network security. Consequently, protecting against network attacks has become a pressing issue that requires urgent attention. It is crucial to find practical solutions to combat such malicious behavior. A network intrusion detection (NID) method, known as GMCE-GraphSAGE, was proposed to meet the detection demands of the current intricate network environment. Traffic data is mapped into gaussian distribution, which helps to ensure that subsequent models can effectively learn the features of traffic samples. The conditional generative adversarial network (CGAN) can generate attack samples based on specified labels to create balanced traffic datasets. In addition, we constructed a communication interaction graph based on the connection patterns of traffic nodes. The E-GraphSAGE is designed to capture both the topology and edge features of the traffic graph. From it, global behavioral information is combined with traffic features, providing a solid foundation for classifying and detecting. Experiments on the UNSW-NB15 dataset demonstrate the great detection advantage of the proposed method. Its binary and multi-classification F1-score can achieve 99.36% and 89.29%, respectively. The GMCE-GraphSAGE effectively improves the detection rate of minority class samples in the NID task.

Keywords: network intrusion detection; IoT; deep learning

Citation: Xinyi Liang, Hongyan Xing, Wei Gu, Tianhao Hou, Zhiwei Ni, and Xinyi Wang. "Hybrid Gaussian Network Intrusion Detection Method Based on CGAN and E-GraphSAGE." Instrumentation 11, no. 2 (2024). https://doi.org/10.15878/j.instr.202400165.

0 Introduction

With its widespread implementation, IoT has become a hot spot in the current development of network technology[1]. The applications and services of the IoT cover various fields, such as logistics, healthcare, transportation, and environmental protection. This provides new intelligent solutions for the production, life, and social development of people[2]. However, it is typical for sensor nodes in IoT settings to have limited storage, computing power, and energy, making them easy targets for attacks[3]. This issue causes a notable decline in the performance and quality of IoT service, which poses major security threats and financial losses to users and organizations alike[4,5]. As the risk of network attacks and data breaches continues to grow, it is crucial to detect abnormal traffic quickly and accurately in intricate network environments.

Network intrusion detection system (NIDS) plays an important role in the current network environment with its excellent ability to recognize malicious network traffic. There are two main types of NIDSs: signature-based detection and anomaly-based detection[6]. Since signature-based detection is challenging to detect zero-day attacks, anomaly-based detection has become the focus of current academic research[7]. Anomaly-based intrusion detection methods mainly consist of statistical learning[8,9], machine learning (ML)[10,11], and deep learning (DL)[12,13]. Although intrusion detection research has advanced significantly due to the development of ML and DL, the current detection models still have some

limitations.

At this stage, most DL-based NID models analyze a single link or a local network and only use traffic features for classification and detection. Due to the lack of grasping the global network information, their detection accuracy is difficult to improve. Graph neural network (GNN) extends DL methods to non-euclidean structure data (Graph). With its strong graph representation and feature extraction abilities, GNNs have emerged in network security tasks such as traffic prediction and intrusion detection in recent years, showing great detection potential. Network traffic is the detection object of NID, consisting of IP addresses, port numbers, and other traffic features. A traffic communication graph can be constructed by mapping IP addresses and ports to nodes and traffic features to edges[15]. Therefore, using graphs to explore the hidden spatial information in the topology of traffic communication networks is highly compatible with NID tasks and deserves in-depth research[16]. Lo et al. introduced GraphSAGE into intrusion detection and proposed E-GraphSAGE. The model can sample and aggregate edge data of the graph, demonstrating its potential for categorizing network traffic. Lan et al. [17] improved the E-GraphSAGE model with their E-miniBatch GraphSAGE. Mini-Batch training enables the model to better adapt to complex network environments. However, it did not perform any undersampling and directly used the full UNSW-NB15 dataset. The high level of imbalance in the data can significantly impact the classifier's learning bias, leading to limited classification performance. Chang et al. [15] proposed a NID scheme called E-ResGAT, which integrated residual learning into GNN. Although E-ResGAT has excellent detection performance, it consumes much more computational resources than the E-GraphSAGE model due to the residual network.

By fully using the graph structural features, the above method improves the performance of intrusion detection to some extent. In particular, GNN performs quite well in the binary classification task of determining whether the traffic is abnormal, and its detection accuracy is as high as 99%. However, in the multi-classification task of distinguishing attack categories, GNN does not perform as well as it should. Detecting anomalous traffic is the basic requirement for NID models. An advanced NIDS should also be able to execute appropriate defenses against different attack types. This places a higher demand on accurately recognizing the kind of attack.

In applications of intrusion detection, network security, etc., attack samples are usually much less than normal samples. In cases where data is imbalanced, the classification results of the NID model may show favor towards the majority class. This can have a detrimental effect on the model's accuracy. As a result, the data imbalance problem has become one of the most common challenges in intrusion detection. To address this problem, JIANG[18] developed a hybrid sampling method combining OSS and SMOTE. With data balancing, this approach improves the detection accuracy of deep hierarchical networks in the NID task. While sampling can be an effective solution for addressing data imbalance, simple oversampling may generate samples that do not conform to the original data distribution. This can result in overfitting of the data and limit the effectiveness of the NID task.

The fast progress of deep learning has brought DGMs like the variational autoencoder (VAE) and the generative adversarial network (GAN) into view. They overcome the limitations of data sampling and have become a common means of solving the data imbalance problem. Xu[19] proposed an LCVAE model based on the log hyperbolic cosine (log-cosh) function to improve the loss function of the conditional variational autoencoder (CVAE). It can better model intrusion data's discrete nature and outperform the oversampling methods such as ROS, SMOTE, and ADASYN. The VAE and its variants can learn the data distribution by minimizing the KL divergence between the latent and prior distributions. However, due to the limitations of the distribution assumptions, it may produce low-quality samples. In contrast, GAN can generate high-quality data directly without coding, which is not limited by data distribution, making it a mainstream generative method in the field of NID. Lee[20] utilized GAN to generate samples of rare categories in attack traffic. Its classification performance on a random forest classifier was better than that of SMOTE sampling. Dlamini[21] designed a conditional generative adversarial network (CGAN) that is well-suited for anomaly detection. The method was proven effective through experiments conducted on the NSL-KDD and UNSW-NB15 datasets. CGAN is more advanced than GAN because it can generate traffic samples of specific classes, which shows that CGAN has a significant potential for generating traffic data.

GNN excels at extracting global traffic features, while CGAN is highly effective in generating traffic data. Inspired by this, this paper conducts further research and proposes the GMCE-GraphSAGE NID model, in which CGAN is combined with E-GraphSAGE, a graph neural network algorithm, to enhance the detection ability. Traffic features are mapped into gaussian distribution so that subsequent models can effectively learn the features of traffic samples. In addition to solving the data imbalance problem, we investigate NIDs using graphs. From it, global behavioral information is combined with traffic features, providing a solid foundation for classifying and detecting. Our approach performs well in binary and multi-classification with an advanced nature.

In summary, the main contributions of this paper are as follows:

1) We proposed a NID method known as GMCE-GraphSAGE, in which the conditional generative adversarial network (CGAN) is combined with E-GraphSAGE, a graph neural network (GNN) algorithm, to enhance the detection ability.

2) We mapped traffic features into gaussian distribution so that subsequent models can effectively learn the features of traffic samples. On the basis of

CGAN, a DGM is designed for NID tasks, which can generate samples that closely resemble the spatial distribution of original data.

3) We constructed a traffic communication interaction graph on the basis of connection patterns of traffic nodes. In addition, E-GraphSAGE, as a classification model, is designed to capture both the topology and edge features of the traffic graph. This provides a solid foundation for classifying and detecting.

4) We compared the performance of GMCE-GraphSAGE with several other state-of-the-art models. The experimental results show our model outperforms other methods in terms of accuracy, precision, and F1-score. This indicates that GMCE-GraphSAGE provides a practical approach for NID.

The remaining sections of this paper are organized as follows. Section II describes the proposed model in detail. Section III introduces the experimental environment and evaluation basis. Section IV shows the experimental results. Section V provides the conclusion of the work.

1 GMCE-GraphSAGE Model

As a proven NID method, the flowchart of GMCE-GraphSAGE is shown in Fig.1 and consists of 3 steps:

Step 1: Traffic data preprocessing. In the data preprocessing paradigm, the traffic features were mapped into gaussian distribution by estimating the optimal parameters for stabilizing the variance and minimizing the skewness through maximum likelihood.

Step 2: Data Generation. To fully learn the distributional characteristics of gaussian traffic data, a CGAN model was constructed. It can generate attack samples of the minority class to balance the traffic dataset.

Step 3: Edge feature extraction and classification. We constructed a traffic communication interaction graph and the E-GraphSAGE algorithm was used to capture the edge features and topology information of the traffic graph. From it, the global behavioral information and features of the traffic are fused, which provides a solid foundation for classifying and detecting.

1.1 Data preprocessing

Data preprocessing consists of the following two steps:

Numerical processing:

In addition to the source/destination IP address and port, three symbolic features are included in the UNSW-NB15 dataset: proto, state, and service. In order to make model training easier, we use the label encoder method to convert the seven symbolic-based features into digital representations.

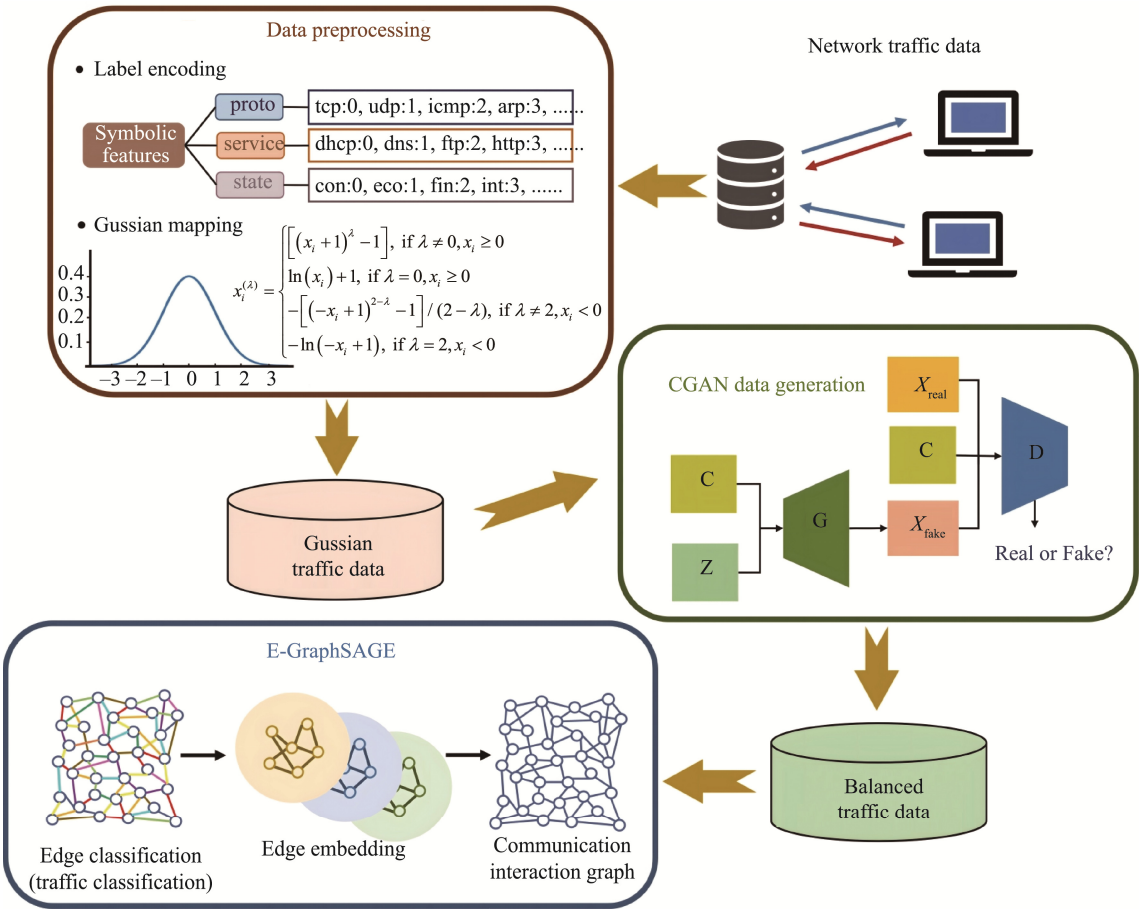


Fig.1 Flow chart of the GMCE-GraphSAGE method

Normalization processing:

Due to the different dimension criteria between the features of the raw traffic, training directly on these features can affect the classification prediction results of the model. Therefore, we use the Yeo-Johnson mapping transform to normalize the traffic features into a gaussian distribution of the same dimension, as is shown in Eq.1. gaussian mapping makes model training results more dependent on the properties of the data itself. In addition, it lessens the computational complexity of subsequent models, leading to quicker model convergence.

$$x_i^{(\lambda)} = \begin{cases} \left[(x_i+1)^{\lambda}-1\right], \text{ if } \lambda \neq 0, x_i \geq 0 \\ \ln(x_i)+1, \text{ if } \lambda = 0, x_i \geq 0 \\ -\left[(-x_i+1)^{2-\lambda}-1\right]/(2-\lambda), \text{ if } \lambda \neq 2, x_i < 0 \\ -\ln(-x_i+1), \text{ if } \lambda = 2, x_i < 0 \end{cases} \quad (1)$$

1.2 CGAN data generation

1.2.1 GAN model

The structure of the classical GAN model is shown in Fig.2, which contains two deep neural networks, the generator G and the discriminator D, respectively. The task of the generator is to continuously learn the distribution of real samples and generate sample data that can deceive the discriminator. The task of the discriminator is to distinguish whether the generated data are real samples. The two are trained in alternating turns to compete with each other. Over time, the generator reaches a Nash equilibrium, enabling it to generate samples that closely resemble the actual distribution.

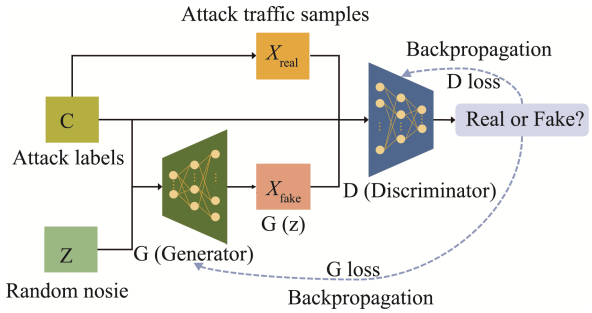


Fig.2 Model structure diagram of GAN

GAN is a prerequisite for understanding the CGAN mechanism. We assume that the real sample is x, the distribution of the real sample is $P_{data}(x)$, and an arbitrary noise vector z obeying the a priori noise distribution $P_z(z)$ is fed into the generator G. G generates the fake sample $G(z)$ by learning the mapping relation between $P_z(z)$ and $P_{data}(x)$. The discriminator D evaluates the real sample x and the generated sample $G(z)$. The outputting, $D(x)$ and $D(G(z))$, is the probability of whether they are real samples.

$$\min_G \max_D V(D,G) = E_{x\sim P_{data}(x)}[\log D(x)] + E_{z\sim P_{z(z)}}[\log(1-D(G(z)))] \quad (2)$$

The objective function of GAN is shown in Eq.2, and its optimization process can be summarized as a "Twoplayer minimax game" problem. G and D are trained in synchronized alternation: (1) During the training of G, we expect D to generate new samples closest to the actual distribution. Therefore, the probability that D determines the fake sample as the real sample should be as large as possible, which means $D(G(z))$ and $1-D(G(z))$ should approach 1 and 0, respectively. (2) During the training of D, we aim for the likelihood of D identifying the generated samples as real to be minimized. This means that $D(G(z))$ should approach 0 while $1-D(G(z))$ should approach 1. Meanwhile, the probability that D determines the real sample correctly should be as large as possible, converging to 1. Thus the training goal of the discriminator is to make the objective function as large as possible.

1.2.2 CGAN model

The training method of GAN is unsupervised learning, generating samples that are hard to control and unpredictable. In contrast, CGAN inputs conditional information into G and D and can realize the control of data generation patterns. The structure of its model is shown in Fig.3. In the CGAN model, we use the category label y as conditional information. CGAN is trained to generate virtual samples based on specified labels. Specifically, G first takes arbitrary noise vector z and category label y as inputs. G then generates fake samples $G(z)$ by learning the mapping relation between the noisy prior distribution $P_z(z)$ and the conditional distribution $P(x|y)$. Finally, D discriminates between the true sample x and the generated sample $G(z)$ and outputs the probability that they are real samples, i.e., $D(x)$ and $D(G(z|y))$.

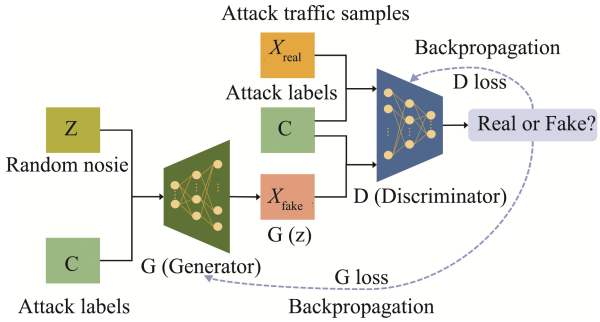


Fig.3 Model structure diagram of CGAN

As is shown in Eq.3, the objective function of CGAN is very similar to that of GAN, with the difference that the optimization process of CGAN is a “Twoplayer minimax game” problem with conditional probabilities. The

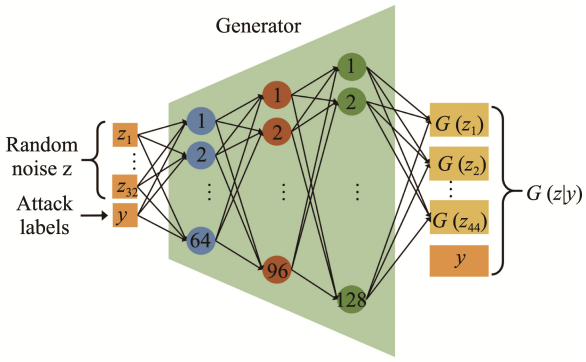
labeling condition allows G to generate samples with specific labels from the noise. Only the generated sample is realistic enough and matches the label can it pass the discriminator.

$$\min_G \max_D V(D,G) = E_{x\sim P_{data}(x)}[\log D(x \mid y)] + E_{z\sim P_{z(z)}}[\log(1 - D(G(z \mid y)))] \quad (3)$$

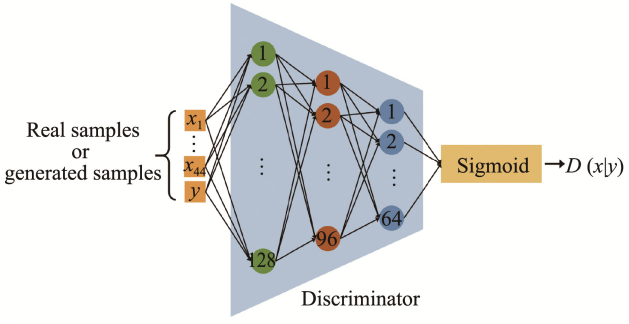
In the proposed CGAN, both the generator and discriminator use a 3-layer feed-forward neural network structure, as shown in Fig.4. We chose ReLU and SGD as the activation function and the optimizer. The initial random noise dimension is set to 32, the learning rate is set to 0.0005, and the batch size is 128, as shown in Table 1.

Tabel 1 Hyparameters of CGAN

Parameter	G	D
Activation function	ReLU	ReLU
Optimizer	SGD	SGD
Initial random noise dimension	32	/
Learning rate	0.0005	0.0005
Batch size	128	128
Layers	3	3
Layer 1 neurons	64	128
Layer 2 neurons	96	96
Layer 3 neurons	128	64



(a) Structure of the generator



(b) Structure of the discriminator

Fig.4 Structural parameters of the generator and the discriminator: (a) Structure of the generator, (b) Structure of the discriminator

1.3 E-GraphSAGE classification

The flow of the E-GraphSAGE model is shown in Fig.5, which mainly includes three parts: the construction of communication interaction graph, edge embedding, and edge classification.

1.3.1 Traffic graph construction

Currently, intrusion detection models rely heavily on network data traffic. This type of data includes information such as source/destination IP address and port number, as well as packet, protocol, byte, and other traffic features. Source/destination IP addresses and port numbers, as location features, are well-suited for defining nodes for traffic graphs. As a result, during the construction of the traffic graph, we use the combination of source IP and source port and the combination of destination IP and destination port as the source and destination nodes of the graph. Meanwhile, the traffic feature information serves as edge features between the source and destination nodes. For example, the source node (149.171.126.18:47439) and the destination node (175.45.176.1:53) communicate, and the traffic information generated from the interaction is the edge feature.

Moreover, only some destination IP addresses are used as attack nodes in common NID datasets. Consequently, in order to avoid the IP address becoming a critical feature for model training, we mapped source IP addresses to a random address in the range of 172.16.0.1–172.31.0.1.

1.3.2 Edge embedding

Traffic features in the NID dataset are only available as edge features in the graph, not node features. Therefore, the node features are initialized as all-1 vectors with $x_v = \{1,1,\cdots,1\}$, and the vector dimensions are the same as the number of edge features. The neighborhood aggregation function in edge embedding is shown in Eq.4. It aggregates edge features of sampled neighborhoods at layer $k-1$ to nodes v creating aggregated embeddings of the sampled neighborhood at the k-th layer $h^k_{N(v)}$. From Eq.5, the aggregated embedding of the k-th layer is merged with the node embeddings of layer $k-1$ and multiplied with the trainable weight matrix. The node embedding of the k-th layer is finally obtained through the nonlinear activation function. At depth K, the final edge embedding can be obtained by connecting the node embedding z^K_u of node u and the node embedding z^K_v of node v, as shown in Eq.6.

$$h^k_{N(v)} \leftarrow \mathrm{AGG}_k(\{e^{k-1}_{uv}, \forall u \in N(v), uv \in \mathcal{E}\}) \quad (4)$$

$$h^k_v \leftarrow \sigma(W^k \cdot \mathrm{CONCAT}(h^{k-1}_v, h^k_{N(v)})) \quad (5)$$

$$z^K_{uv} = CONCAT\left(z^K_u, z^K_v\right), uv \in \mathcal{E} \quad (6)$$

In Eq.4, e^{k-1}_{uv} are the features of edge from $N(v)$, the sampled neighborhood of node v, at layer $\{\forall u \in \mathcal{N}(v), uv \in \mathcal{E}\}$ denotes the set of sampled edges in the neighborhood.

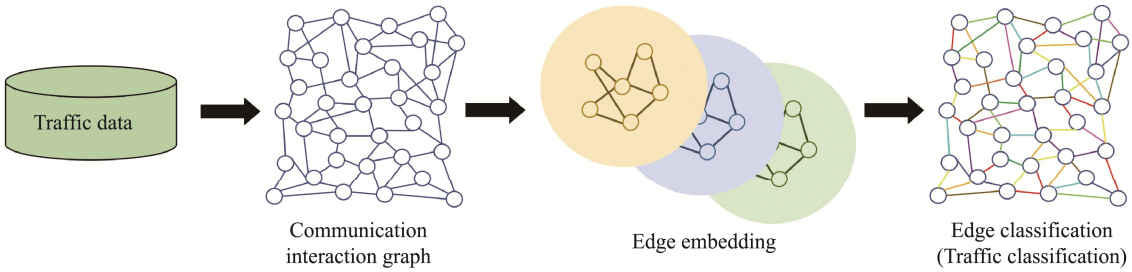


Fig.5 Flow chart of E-GraphSAGE

Specifically, the model uses a typical two-layer convolutional structure (K=2) and mean function as the aggregation function, as shown in Eq.7. The sampled neighborhood size is set to 8, and the node embedding dimension is set to 64. We chose the ReLU and Adam as the nonlinear activation function and optimizer, respectively, with the learning rate set to 0.01, as shown in Tabel 2.

$$h_{\mathcal{N}(v)}^{k} = \sum_{u\in\mathcal{N}(v),uv\in\mathcal{E}} \frac{e_{uv}^{k-1}}{|N(v)|_e} \tag{7}$$

In Eq.7, $|N(v)|_e$ is the number of edges in the sampled neighborhood, and e_{uv}^{k-1} is the edge features of the layer $k-1$.

Tabel 2 Hyparameters of E-GraphSAGE

Parameter	E-GraphSAGE
Convolutional layers	2
Aggregation function	Mean function
Sampled neighborhood size	8
Node embedding dimension	64
Activation function	ReLU
Optimizer	Adam
Learning rate	0.01

2 Experimental environment and evaluation basis

The study was conducted on a personal computer platform with the configuration shown in Table 3.

Tabel 3 Experimental platform

Project	Configuration
Operation system	Windows 11
CPU	Intel (R) Core (TM) i5-12600KF 3.70 GHz
GPU	NVIDIA 3070 8G
Memory	32G
Frame	Pytorch

2.1 The benchmark datasets

The UNSW-NB15 dataset[22] is an open dataset released by the Network Security Laboratory (NSL) at the University of New South Wales (UNSW), Australia, for network intrusion detection research. The dataset includes nine types of attack traffic and one type of normal traffic.

It is hard to construct traffic interaction graphs because the officially partitioned training and testing set do not contain location features. Therefore, we sampled four files, UNSW-NB15_1.csv, UNSW-NB15_2.csv, UNSW-NB15_3.csv, and UNSW-NB15_4.csv, containing

Tabel 4 Partition of the training and testing sets

Category		Description	Data	
			Train	Test
Attack	Analysis	Different attacks of port scan, spam and html files penetrations.	2,000	677
	Backdoor	Access bypassing system security mechanisms.	1,746	583
	Dos	Temporary service interruption of a host connected to the Internet.	12,264	4,089
	Exploits	Vulnerability attacks in operating systems or software.	33,393	11,132
	Fuzzers	Application suspension due to massive random data feeds.	18,184	6,062
	Generic	Attack against all blockciphers.	40,000	18,871
	Reconnaissance	Strikes that can simulate attacks that gather information.	10,491	3,496
	Shell Code	The payload code in the exploitation of software vulnerability.	1,133	378
	Worms	Attacker replicates itself and spreads to other computers.	130	44
Normal		Normal connection.	56,000	37,000
Total			175,341	82,332

the complete traffic data and partitioned the training set and testing set according to the ratio of 7:3. The specific partition is shown in Table 4.

2.2 Evaluation metrics

We used four metrics: accuracy, precision, recall, and F1-score, to evaluate the binary classification performance of the method. These metrics are calculated based on a confusion matrix, as shown in Table 5. In the NID task, TP denotes the number of attack samples the model correctly predicts as attacks. FP denotes the number of normal samples the model incorrectly predicts as attacks. TN denotes the number of normal samples correctly predicted as normal. FN denotes the number of attack samples incorrectly predicted as normal. The sum of TP, TN, FP, and FN is the total sample size.

Tabel 5 Definition of confusion matrix

	Predicted Attack	Predicted Normal
Actual Attack	TN	FP
Actual Normal	FN	TP

Accuracy is defined as the percentage of attack samples correctly classified out of the total samples, as shown in Eq.8. It provides an overall performance assessment of the model. However, in the case of sample imbalance, accuracy is not the best measure of model performance. It must be combined with other metrics such as precision, recall, and F1-score to be evaluated comprehensively.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

Precision is defined as attack samples correctly predicted out of the total predicted samples, as shown in Eq.9. It can measure the ability of models to identify attacks.

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

Recall is defined as the percentage of attack samples correctly predicted out of all attack samples, as shown in Eq.10. It can measure the ability of models to find all attacks.

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

The F1-score is defined as the reconciled mean of precision and recall, as shown in Eq.11. It is a comprehensive metric to evaluate the performance of the intrusion detection model.

$$F1 = \frac{2 \times R \times P}{R + P} \tag{11}$$

In Eq.11, R is Recall and P is Precision.

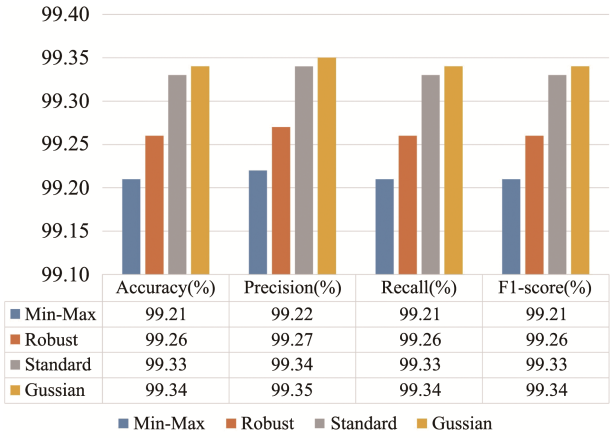
Weighted metrics are more effective in evaluating NID performance when there is a sample imbalance. As a result, the four metrics: accuracy, weighted precision, weighted recall, and weighted F1-score, are used to evaluate the multi-classification performance of the method. Weighting factors are set to category proportions.

3 Experimental validation and analysis of GMCE-GraphSAGE performance

The main purpose of binary classification is to determine whether the traffic is abnormal, which is the first task of NID. Once abnormal traffic is detected, the second task of NID is to identify the attack type using multi-classification. From it, NID can perform the appropriate defenses against different attacks. Therefore, we verify the performance of GMCE-GraphSAGE in both binary and multi-classification tasks with the help of two tools: t-distributed stochastic neighbor embedding (T-SNE) and ablation experiments.

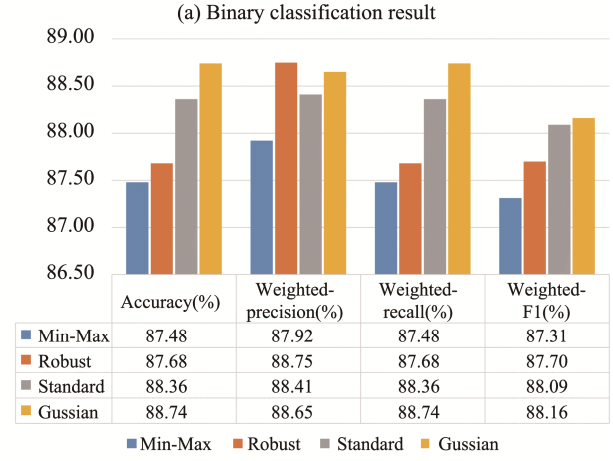
3.1 Normalization performance analysis

Min-Max, Robust, and Standard are common normalization methods in NID. We use E-GraphSAGE as a benchmark model to compare the performance of gaussian mapping normalization with the above three methods in the classification task on the UNSW-NB15 dataset. As can be seen in Fig.6(a), normalization did not have a significant effect on the binary classification. However, CE-GraphSAGE with gaussian normalization performs best in all four methods, as shown in Fig.6(b). Its accuracy, precision, recall, and F1-score achieved 88.74%, 88.65%, 88.74%, and 88.16% respectively.



	Accuracy(%)	Precision(%)	Recall(%)	F1-score(%)
Min-Max	99.21	99.22	99.21	99.21
Robust	99.26	99.27	99.26	99.26
Standard	99.33	99.34	99.33	99.33
Gussian	99.34	99.35	99.34	99.34

(a) Binary classification result

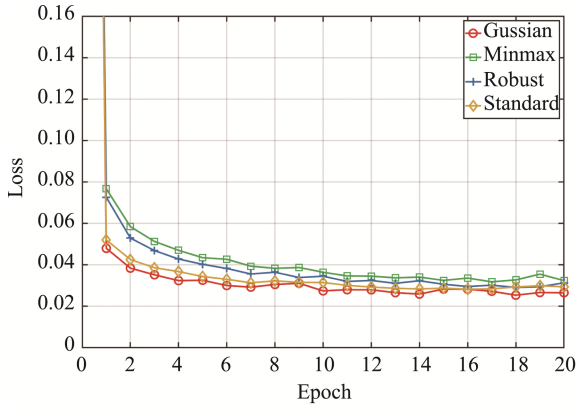


	Accuracy(%)	Weighted-precision(%)	Weighted-recall(%)	Weighted-F1(%)
Min-Max	87.48	87.92	87.48	87.31
Robust	87.68	88.75	87.68	87.70
Standard	88.36	88.41	88.36	88.09
Gussian	88.74	88.65	88.74	88.16

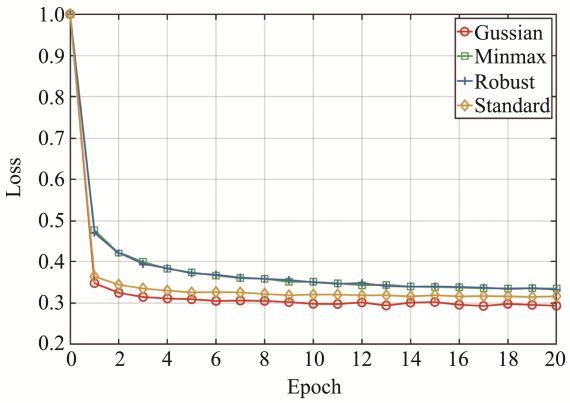
(b) Multi-classification result

Fig.6 Classification results by different normalization methods: (a) Binary classification result, (b) Multi-classification result

We plotted the loss curves of the E-GraphSAGE model under different normalization methods, as shown in Fig.7. In the binary classification loss curve plot shown in Fig. 7(a), the GME-GraphSAGE model combined with Gaussian mapping normalization still achieves the lowest loss value despite the fact that the difference in the model loss convergence values for different preprocessing methods is only 0.005%. This indicates that the predictions of the model are closest to the true values.



(a) Binary loss curve



(b) Multi-loss curve

Fig.7 Loss curves of E-GraphSAGE model: (a) Binary loss curve, (b) Multi-loss curve

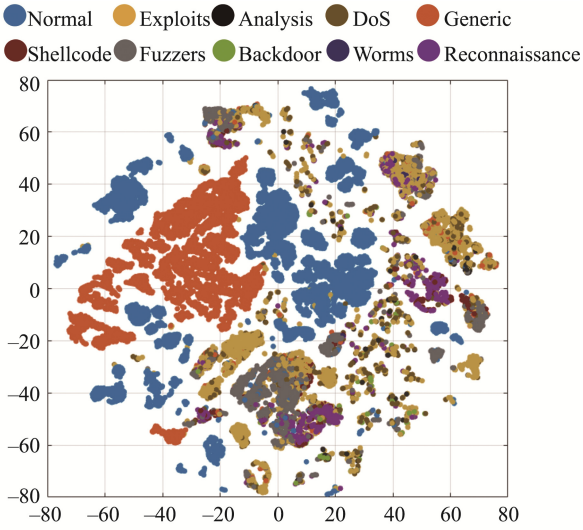
In the multi-classification loss curve plot shown in Fig.7(b), the loss of the model combining gaussian normalization has converged to 0.347 at the first epoch. It draws a significant gap with the other three methods. It can be seen that the computational complexity of the model is reduced by gaussian mapping, which allows the model to converge quickly.

From Fig.6, we can see that the E-GraphSAGE model has achieved a relatively excellent performance in the binary classification task, with all four evaluation metrics reaching over 99%. Multi-classification, as a second line of defense, also plays an important role in NID. Unfortunately, E-GraphSAGE does not perform well in such cases. As a result, we improved the model to enhance its accuracy in detecting multiple classes. In the subsequent analysis, we will mainly focus on the performance of our model in multi-classification.

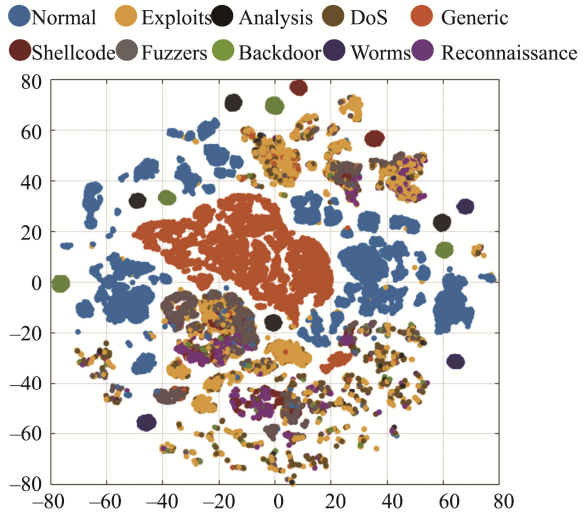
3.2 Data generation performance

As can be seen in Table 4, UNSW-NB15 is a highly unbalanced dataset. The Worms, Shell Code, Backdoor, and Analysis categories have significantly fewer samples in the training set. In particular, the Worms category contains only 130 samples. In order to make the E-GraphSAGE model effective in extracting the critical features of the traffic data, we generated 19,000 minority class samples (among them: Worms: 6,000, Shell Code: 3,000, Backdoor: 5,000, Analysis: 5,000) using the CGAN model proposed in Section 2.2. The balanced training set is then constructed.

To demonstrate the data generation capability of the CGAN model more intuitively, we analyze it with the help of T-SNE, a non-linear dimensionality reduction algorithm. We can map high-dimensional data to a low-dimensional space to make features easier to visualize through this tool. The association degree between features and network behaviors can be observed from it. The clustering results for the original and CGAN-enhanced data are shown in Fig.8.



(a) Original data



(b) CGAN-enhanced data

Fig.8 T-SNE visualization of before and after CGAN enhancing results: (a) T-SNE visualization of original data, (b) T-SNE visualization of CGAN-enhanced data

As shown in Fig.8(a), the distribution of the original dataset is seriously imbalanced. There is no clear boundary between samples of different categories. It can be easily found that many of the Worms, Shell Code, Backdoor, and Analysis samples are mixed with samples from other categories. The number of these four minority-class samples was too small to be detected in the T-SNE, which is detrimental to the subsequent intrusion detection.

There are distinct clusters of Worms, Shell Code, Backdoor, and Analysis in Fig.8(b). The generated minority class samples have similarities and maintain diversity with the original ones, which demonstrates the data generation capabilities of CGAN. However, we can see that the tiny samples from these four minority categories are still randomly distributed in the other categories, demonstrating that the method has some limits.

In addition, from Fig. 8(a) and (b), we can see that the samples of Reconnaissance, Fuzzers, and Exploits categories are partially mixed together. This may be due to the high feature similarity of these three categories, which makes it difficult to distinguish them. Clustering tools alone cannot recognize these categories. At this time, a powerful means of key feature extraction becomes especially important.

3.3 Feature extraction and classification performance

E-GraphSAGE is a classification model which can extract edge features and topology information from the traffic graph. The model is trained with the balanced training set created by CGAN. At the same time, we performed an ablation study of GMCE-GraphSAGE to validate the effectiveness of each module.

(1) E-GraphSAGE Only: We only used E-GraphSAGE module after Min-Max normalization preprocessing for intrusion detection to evaluate the classification performance of this module.

(2) GME-GraphSAGE: The CGAN module is removed from the GMCE-GraphSAGE, which will evaluate the normalization ability of the gaussian mapping.

(3) CE-GraphSAGE: We removed the GM module, and Min-Max normalization was used. Still, we kept the CGAN module and the E-GraphSAGE module, which will evaluate the generation capability of the CGAN module. The results of the ablation experiments for the binary classification task are shown in Table 6, the basic E-GraphSAGE model achieves 99.21% for all four evaluation metrics. Comparing models (1), (2), and (3), it can be seen that the performance metrics are slightly improved after GM and CGAN data enhancement. And GMCE-GraphSAGE achieved the best performance among the four models. This is because the edge features extracted by the E-GraphSAGE model are the key basis for distinguishing abnormal traffic from normal traffic. It has achieved quite excellent performance in binary classification tasks. Therefore the detection performance of the GMCE-GraphSAGE model combining GM and CGAN, although improved, is limited.

The multi-classification results of the ablation study are shown in Table 7. It can be easily found that all the modules of GMCE-GraphSAGE exhibit excellent performance. A comparison of models (2) and (1) shows that GM can maximize the data quality and reduce the interference caused by unbalanced traffic feature weights to the classifier. This helps to improve the stability and detection performance of the classifier. From the comparison between model (3) and model (1), it can be seen that model (3) achieves excellent detection performance despite the lack of GM preprocessing. This is due to the powerful data generation capability of the CGAN model. The generated samples are close enough to the real distribution to improve the minority class detection accuracy effectively.

Table 6 Ablation experiment result of binary classification

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
GMCE-GraphSAGE	99.36	99.37	99.36	99.36
(1) E-GraphSAGE	99.21	99.22	99.21	99.21
(2) GME-GraphSAGE	99.34	99.35	99.34	99.34
(3) CE-GraphSAGE	99.35	99.36	99.35	99.35

Table 7 Ablation experiment result of multi-classification

Model	Accuracy (%)	weighted-Precision (%)	weighted-Recall (%)	weighted-F1 (%)
GMCE-GraphSAGE	89.39	90.28	89.39	89.37
(1) E-GraphSAGE	87.48	87.92	87.48	87.31
(2) GME-GraphSAGE	88.74	88.65	88.74	88.16
(3) CE-GraphSAGE	88.51	89.27	88.51	88.58

The modules work together to form the GMCE-GraphSAGE model, and optimal detection performance is obtained. As can be seen from the ablation experiments, it is because of the data generation and global traffic feature extraction that our method performs so well. This shows that GMCE-GraphSAGE has great detection potential and can guard the second line of defense for intrusion detection.

3.4 Discussion and additional comparison

To validate the detection performance of the GMCE-GraphSAGE model, we compare the detection results with several state-of-the-art NID methods. Among them are ML-based methods (NB, KNN), DL-based methods (DNN, CNN-BiLSTM), DL-based methods combined with data augmentation (ADSYN-CNN, VAEGAN-CNN) and advanced methods (IGRF-RFE, LOGNN, E-GraphSAGE). According to the comparison results in Table 8, our model achieved the best results. Its accuracy, precision, recall, and F1-score achieved 89.39%, 90.28%, 89.39%, and 89.37%, respectively.

From the perspective of classification performance, NB, KNN, and DNN do not perform well on all evaluation metrics (< 70%). It is indicated that traditional ML methods and shallow neural networks are no longer suitable for intrusion detection in such a complex network environment. The hybrid neural network CNN-BiLSTM has certain advantages in extracting temporal and spatial features. However, the features it extracts are of a single link, and the lack of global network information limits its detection performance to a large extent. In contrast, the classification of the proposed GMCE-GraphSAGE model is on the basis of global behavioral information and traffic features. Thus the great detection performance is obtained.

From the perspective of data generation, although ADSYN-CNN achieved the best recall (94.65%), its precision was low (77.76%). This is probably due to the fact that ADSYN is very sensitive to outliers, enabling CNN to accurately find specific attacks. The high sensitivity is also a limitation for the classifier to recognize attacks, resulting in low precision. VAEGAN combines the advantages of VAE and GAN. Although it can learn sample distributions for data generation, it does not perform well with unbalanced data distributions. It is probably because VAEGAN prefers to generate samples more easily accepted by the discriminator during training. It is implied that there is a significant lack of generated samples for the minority class. In contrast, our CGAN model can learn the real distribution of traffic and generate minority class samples based on specified labels. It is an effective solution for data imbalance problems.

In addition, our model is compared with current advanced models. IGRF-RFE selects the optimal subset of features for classification by combining Information Gain and Random Forest methods. However, it is well known that feature selection is an effective strategy to enhance algorithms based on ML. We are skeptical about whether it is effective for DL-based NIDs. The logarithm neuron (LOGN) is designed to improve the capability of the LOGNN model in data feature extraction. However, the model has a severe drawback: If not combined with a specifically designed anti-gradient vanishing loss function (AGLF), the gradient vanishing problem will cause the model to fail and collapse. It is difficult to be widely used in NID. E-GraphSAGE, which utilizes graphs for anomaly detection, is one of the most advanced NID schemes. Based on this, our GMCE-GraphSAGE adds GM and CGAN data enhancement, achieving the highest performance in most metrics.

Table 8 Comparison mult-classification results (%) of different detection models

Model	Accuracy (%)	weighted-Precision (%)	weighted-Recall (%)	weighted-F1 (%)
NB[23]	43.7	57.9	43.7	39.6
KNN[23]	62.2	57.8	62.2	57.6
DNN[23]	64.5	61.4	64.5	58.6
CNN-BiLSTM[24]	83.18	83.18	83.70	81.19
ADSYN-CNN[24]	82.15	77.76	94.65	85.38
VAEGAN-CNN[25]	86.65	87.79	86.65	85.41
IGRF-RFE[26]	84.24	83.60	84.24	82.85
LOGNN[27]	85.70	85.40	85.70	85.50
E-GraphSAGE	87.48	87.92	87.48	87.31
GMCE-GraphSAGE	89.39	90.28	89.39	89.37

4 Conclusion and future work

In this paper, a novel NID method called GMCE-GraphSAGE was proposed, which can effectively improve the imbalance problem with high detection accuracy. In GMCE-GraphSAGE, we designed a suitable data preprocessing approach to map the traffic features into the gaussian domain. In this way, the subsequent model can fully learn the features of minority class samples. Our CGAN model can learn the real distribution of traffic and generate minority class samples based on specified labels. In addition, the proposed E-GraphSAGE model is able to capture topological information and edge features of the traffic graph. It is a solid foundation for classifying and detecting. Thus, excellent detection performance is obtained especially in multi-classification.

To validate the detection performance of GMCE-GraphSAGE, we performed a series of experiments on the UNSW-NB15 dataset with the help of T-SNE and ablation studies. In the binary-classification task, the proposed method achieved 99.36%, 99.37%, 99.36%, and 99.36% in accuracy, precision, recall, and F1-score, respectively. In the multi-classification task, the proposed method achieved 89.39%, 90.28%, 89.39%, and 89.37% in accuracy, weighted-precision, weighted-recall, and weighted-F1-score, respectively. Compared with other methods, GMCE-GraphSAGE has been found to have significant advantages. It is an effective solution for NID.

However, GMCE-GraphSAGE also has the following drawbacks. On the one hand, GMCE-GraphSAGE has limited learning ability for minority class training samples. Although the intrusion detection performance is improved to some extent, the detection accuracy is still poor. Therefore, we will further focus on adding a probabilistic model to DGM to learn the distributional properties of the traffic thoroughly. This helps generate diverse samples and improves the performance and robustness of NID.

On the other hand, even if the data enhancement algorithm is used, it is not effective at improving detection when the minority samples are few. Our next research will focus on exploring the intrinsic physical correlation between intrusion and features. By utilizing DL and existing feature mining techniques, we will further investigate and analyze more significant feature relations based on the network protocol from the source to enhance the detection performance.

Author Contributions:

Xinyi Liang: Conceptualization, Methodology, Software, Writing - Original Draft. Hongyan Xing: Supervision, Funding acquisition. Wei Gu: Data Curation, Software. Tianhao Hou: Validation, Formal analysis. Zhiwei Ni: Writing - review & editing. Xinyi Wang: Investigation, Visualization.

Funding Information:

This research was funded by the National Natural Science Foundation of China (grant number. 62171228), National Key Research and Development Program of China (grant number. 2021YFE0105500).

Data Availability:

The authors declare that the main data supporting the findings of this study are available within the paper and its Supplementary Information files.

Conflict of Interest:

The authors declare no competing interests.

Dates:

Received 12 March 2024; Accepted 21 April 2024; Published online 30 June 2024

References

[1] FAROOQ, M. U., WASEEM, M., MAZHAR, S., KHAIRI, A. & KAMAL, T. (2015). A review on internet of things (IoT). *International journal of computer applications,* 113(1), pp.1-7.

[2] ASHRAF, J., KESHK, M., MOUSTAFA, N., ABDEL-BASSET, M., KHURSHID, H., BAKHSHI, A. D. & MOSTAFA, R. R. (2021). IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities. *Sustainable Cities and Society,* 72, p.103041.

[3] AL-GARADI, M. A., MOHAMED, A., AL-ALI, A. K., DU, X., ALI, I. & GUIZANI, M. (2020). A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Communications Surveys & Tutorials,* 22(3), pp.1646-1685.

[4] SIMON, J., KAPILESWAR, N., POLASI, P. K. & ELAVEINI, M. A. (2022). Hybrid intrusion detection system for wireless IoT networks using deep learning algorithm. *Computers and Electrical Engineering,* 102, p.108190.

[5] GE, M., SYED, N. F., FU, X., BAIG, Z. & ROBLES-KELLY, A. (2021). Towards a deep learning-driven intrusion detection approach for Internet of Things. *Computer Networks,* 186, p.107784.

[6] FU, Y., DU, Y., CAO, Z., LI, Q. & XIANG, W. (2022). A deep learning model for network intrusion detection with imbalanced data. *Electronics,* 11(6), p.898.

[7] SAMRIN, R. & VASUMATHI, D. (2017). Review on anomaly based network intrusion detection system. *2017 international conference on electrical, electronics, communication, computer, and optimization techniques (ICEECCOT)*. IEEE, pp.141-147.

[8] LI, F., SHINDE, A., SHI, Y., YE, J., LI, X.-Y. & SONG, W. (2019). System statistics learning-based IoT security: Feasibility and suitability. *IEEE Internet of Things Journal,*

6(4), pp.6396-6403.

[9] SHA, W., ZHU, Y., CHEN, M. & HUANG, T. (2015). Statistical learning for anomaly detection in cloud server systems: A multi-order Markov chain framework. *IEEE transactions on cloud computing,* 6(2), pp.401-413.

[10] TSAI, C.-F., HSU, Y.-F., LIN, C.-Y. & LIN, W.-Y. (2009). Intrusion detection by machine learning: A review. *expert systems with applications,* 36(10), pp.11994-12000.

[11] HOU, T., XING, H., LIANG, X., SU, X. & WANG, Z. (2022). Network intrusion detection based on DNA spatial information. *Computer Networks,* 217, p.109318.

[12] VARALAKSHMI, S., PREMNATH, S., YOGALAKSHMI, V. & KAVITHA, V. (2021). Design of IOT network using deep learning-based model for anomaly detection. *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE, pp.216-220.

[13] TSIMENIDIS, S., LAGKAS, T. & RANTOS, K. (2022). Deep learning in IoT intrusion detection. *Journal of network and systems management,* 30, pp.1-40.

[14] AHMAD, Z., SHAHID KHAN, A., WAI SHIANG, C., ABDULLAH, J. & AHMAD, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies,* 32(1), p.e4150.

[15] CHANG, L. & BRANCO, P. (2021). Graph-based solutions with residuals for intrusion detection: The modified e-graphsage and e-resgat algorithms. *arXiv preprint arXiv:2111.13597*.

[16] JIANG, W. (2022). Graph-based deep learning for communication networks: A survey. *Computer Communications,* 185, pp.40-54.
LAN, J., LU, J. Z., WAN, G. G., WANG, Y. Y., HUANG, C. Y., ZHANG, S. B., HUANG, Y. Y. & MA, J. N. (2022). E-minBatch GraphSAGE: An Industrial Internet Attack Detection Model. *Security and Communication Networks,* 2022.

[17] JIANG, K., WANG, W., WANG, A. & WU, H. (2020). Network intrusion detection combined hybrid sampling with deep hierarchical network. *IEEE access,* 8, pp.32464-32476.

[18] XU, X., LI, J., YANG, Y. & SHEN, F. (2020). Toward effective intrusion detection using log-cosh conditional variational autoencoder. *IEEE Internet of Things Journal,* 8(8), pp.6187-6196.

[19] LEE, J. & PARK, K. (2021). GAN-based imbalanced data intrusion detection system. *Personal and Ubiquitous Computing,* 25, pp.121-128.

[20] DLAMINI, G. & FAHIM, M. (2021). DGM: a data generative model to improve minority class presence in anomaly detection domain. *Neural Computing and Applications,* 33, pp.13635-13646.

[21] MOUSTAFA, N. & SLAY, J. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *2015 military communications and information systems conference (MilCIS)*. IEEE, pp.1-6.

[22] VINAYAKUMAR, R., ALAZAB, M., SOMAN, K., POORNACHANDRAN, P., AL-NEMRAT, A. & VENKATRAMAN, S. (2019). Deep learning approach for intelligent intrusion detection system. *Ieee Access,* 7, pp.41525-41550.

[23] KOCHER, G. & KUMAR, G. (2022). A hybrid deep learning approach for effective intrusion detection systems using spatial-temporal features. *Adv. Eng. Sci.,* 54(2), pp.1503-1519.

[24] HE, J., WANG, X., SONG, Y., XIANG, Q. & CHEN, C. (2023). Network intrusion detection based on conditional wasserstein variational autoencoder with generative adversarial network and one-dimensional convolutional neural networks. *Applied Intelligence,* 53(10), pp.12416-12436.

[25] YIN, Y., JANG-JACCARD, J., XU, W., SINGH, A., ZHU, J., SABRINA, F. & KWAK, J. (2023). IGRF-RFE: a hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 dataset. *Journal of Big data,* 10(1), p.15.

[26] WANG, Z., XU, Z., HE, D. & CHAN, S. (2021). Deep logarithmic neural network for Internet intrusion detection. *Soft Computing,* 25(15), pp.10129-10152.