IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 September 2022, accepted 5 October 2022, date of publication 6 October 2022, date of current version 13 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3212751

RESEARCH ARTICLE

A Network Security Situation Element Extraction Method Based on Conditional Generative Adversarial Network and Transformer

YU YANG, CHENGPENG YAO, JINWEI YANG, AND KUN YIN
College of Information Engineering, Engineering University of PAP, Xi'an 710086, China

Corresponding author: Yu Yang (miaoyude@163.com)

This work was supported by the Fundamental Research Fund of the School of Information Engineering, Engineering University of PAP, under Grant WJY202130.

ABSTRACT Recently, the massive increase in network users has dramatically increased network traffic, making it more difficult to maintain network security. The task of network security situation element extraction is to detect and classify network traffic. The detection rate of minority class samples is low in existing network traffic feature extraction classification methods, and most of the network threat data have seen extreme sample imbalance, which further affects the detection accuracy of minority class samples. To solve these problems, this paper proposes a network security situation element extraction method using conditional generative adversarial network (CGAN) and Transformer. Here, CGAN is applied to solve the sample imbalance problem in the data and improve the detection accuracy of minority samples. Transformer, as an effective feature learning method in natural language direction, has excellent long-distance feature extraction ability. By combining CGAN with Transformer, the detection accuracy of network traffic can be effectively improved. Also, validation was performed using the UNSW-NB15 and KDDcup99 datasets. Experimental results demonstrate that the method using a combination of CGAN and Transformer improved the detection rate for minority samples compared with other advanced-feature extraction classification methods, thereby improving the overall accuracy, F1-score, and specificity. The results are 89.38 % and 93.07 %, 89.75 % and 93.68 %, 87.65 % and 98.20 %, respectively.

INDEX TERMS Network security, network security situation element extraction, conditional generative adversarial network, transformer.

I. INTRODUCTION

The rapid development of computer technology has seen the application of the network to all places of work, life, and everything around us. To a report [1] released by the China Internet Information Center (CNNIC) in 2021, the number of Chinese internet users reached 1.011 billion as of June 2021, an increase of 21.75 million from December 2020. Also, the number of users using online shopping, online news, online delivery, and other online services rapidly increases yearly. This increased popularity of the network has increased data traffic in the network, thereby creating more difficulty in detecting network threats. For any network that has encountered network threats, both individuals and groups cause varying degrees of damage. Thus, maintaining good network security has become important. The situation of network security issues is complex because of emerging network threats, such as traditional viruses, Trojan horses, and phishing sites. As shown in a report [2] released by the China Computer Network Emergency Technology Processing Coordination Center (CNCERT) in December 2021. During the week of December 13-December 19, the number of malicious computer programs spread in China was about 63,741,000 times, and the number of hosts infected with malicious programs reached 1,019,000. These statistics highlight a disturbing network security problem. Also, the development of the fifth-generation mobile communication network [3] technology has seen network traffic growing

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Saleem.

in explosive form, and several complex network threats are beginning to emerge, all posing a serious challenge to network managers. In the current environment, the means to effectively respond to network threats has become particularly important.

Recently, network security situation awareness [4] has become a hot research topic, comprising three layers: situation element extraction, situation assessment, and situation prediction. Compared with traditional passive defense means, it has the advantage of proactive prediction and advanced defense. It effectively deals with network threats in today's complex and ever-changing network environment. Network security situation element extraction is the first layer of the situation awareness model, primarily concerned with extracting traffic features in the network and classifying traffic categories. Thus, accurate situation element extraction is an important guarantee for the security protection of the network.

The main task of network security situation element extraction is to detect and classify network traffic, so it can also be attributed to the problem of intrusion detection. Traditional detection and classification of network traffic mainly use machine learning (ML) [5] algorithms, such as KNN [6] and random forest (RF) [7]. Also, the ML algorithm is more efficient in network traffic detection and classification, but in dealing with some complex problems, the overall accuracy is still low. With the expansion of data volume and increased computing power of computers, there has been a rapid increase in the number of network traffic detection and classification methods using deep learning (DL) [8]. DL is considered to be an effective algorithm for solving classification and regression problems [9] and has significant advantages when dealing with large complex data. At present, there have been many studies on network traffic detection and classification based on deep learning, such as Long and Short-Term Memory (LSTM) [10], bidirectional LSTM (Bi-LSTM) [11], and Convolutional Neural Network (CNN) [12]. DL algorithm for network traffic detection and classification ability is stronger, and using DL algorithm has gradually become the mainstream.

In the previous research on network traffic detection and classification, the focus is on designing better models, and the problem of the unbalanced sample of network traffic data is rarely considered. However, most of the existing data sets have serious sample imbalance, which leads to the detection accuracy of minority classes being generally low when dealing with minority classes. And in the network traffic data, most of the minority samples correspond to some attack samples, which reflects the importance of minority sample detection. Secondly, there are few applications of Transformer in the current research on network traffic feature extraction. Therefore, based on the shortcomings of the above research, this paper combines CGAN [13] and Transformer [14] to extract network security situation elements and validated the model using the UNSW-NB15 [15] and KDDcup99 [16] datasets. The work has contributed in the following ways:

1. We propose a DL model integrating CGAN and Transformer for network security situation element extraction. The proposed model improves the detection rate of minority samples and the accuracy of overall situation element extraction.

2. The transformer method is applied to feature extraction of network traffic. The network traffic data have complex features and high dimensionality. Hence, using the long-distance feature capturing ability of the transformer, we better captured the complex features contained in the network traffic data and improved the detection accuracy.

3. For the sample imbalance problem of the UNSW-NB15 and KDDcup99 datasets, CGAN is used to oversample two datasets. The CGAN adopts the idea of adversarial data generation. Compared with other machine learning methods to generate new samples, it has great advantages in generating minority sample data. The detection accuracy of minority attacks is improved by oversampling the minority samples in two datasets.

4. The CGAN combined with the Transformer model was compared with other feature extraction models on the UNSW-NB15 and KDDcup99 datasets and evaluated with several evaluation metrics.

The remaining part of this paper is organized as follows: Section 2 reviews the existing works on sample imbalance and feature extraction for classification. Section 3 introduces the methods and datasets used in this paper and gives the evaluation metrics and overall architectural model for the experiments. Section 4 conducts multiple sets of experimental validation and analyzes the experimental results. Section 5 presents the conclusion and provides an outlook on future research.

II. RELATED WORK

In this section, we provide an overview of the sample imbalance problem and network traffic detection and classification.

A. PROCESSING SAMPLE IMBALANCE

The sample imbalance problem has been studied in many aspects. An imbalance in the training sample leads to a bias in the test results toward the majority class, which reduces the detection rate of the minority class. There are three main solutions to the sample imbalance problem: First, the expansion of the dataset for a few classes of samples from the original dataset and some collected data. This method can outperform other algorithms. Second, the resampling of the samples, which includes oversampling [17] and undersampling [18]. Oversampling is the expansion of the minority class of the sample by directly copying or generating new data for the minority class. Its disadvantage is that the copied or generated data do not have obvious feature expressions, which leads to a poor generalization of the model. Undersampling achieves sample balance by reducing the majority class sample to a number similar to the minority class sample, with the disadvantage that reducing the majority class sample may cause loss of important features and make the model underfit. Hence, a combination of oversampling and undersampling

better solves the shortcomings of the above-mentioned individual methods for resampling techniques. Third, setting penalty coefficients, which will have a penalty cost when misclassifying minority class samples. This makes the classifier focus more on minority class samples to improve the detection accuracy for minority class samples. In the field of network security, the most used technique is still resampling, which can effectively and quickly deal with the sample imbalance by directly processing the dataset.

Practical research is mainly focused on the resampling of samples. Seo and Kim [19] oversampled three minority classes, U2R, R2L, and Probe, in the KDDcup99 dataset using the SMOTE method, and obtained the best SMOTE ratio to obtain the best classification results through experiments. Zhang et al. [20] use a combination of SMOTE and Gaussian mixture model (GMM)-based undersampling techniques for unbalanced datasets and prove that this combination of oversampling and undersampling methods works well by validating it on the UNSW-NB15 and CICIDS2017 datasets. Liu, Gao, and Hu [21] used the adaptive synthesis (ADASYN) oversampling technique to process the NSL-KDD, UNSW-NB15, and CICIDS2017 datasets for intrusion detection. The ADASYN oversampling technique generated newer samples near some difficult-to-learn samples to achieve sample balancing. The traditional ML method has achieved certain results in dealing with sample imbalance, but it is not effective when facing samples with complex feature distribution. GAN has developed rapidly in recent years. It is based on the DL method and adopts the idea of adversarial data generation, which has advantages in dealing with complex feature distribution. There have been some studies on generating new samples through GAN. Liu et al. [22] used the generative adversarial network (GAN) to improve the distribution of data sets in vehicle intrusion detection. The GAN achieves the purpose of sample balance by generating pseudo-real data sets. Chui et al. [23] used CGAN to generate low-sample data from student academic datasets to improve the prediction accuracy of the model.

In this paper, CGAN, which is more effective in dealing with the sample imbalance problem, is chosen. CGAN generates new data through the confrontation between two neural networks, and its data is closer to the real data, which has greater advantages than other oversampling methods.

B. NETWORK TRAFFIC DETECTION AND CLASSIFICATION

For feature learning on network data, three main approaches were used, namely, supervised learning [24], unsupervised learning [25], and semi-supervised learning [26]. In supervised learning, each sample has a corresponding label and each input has its corresponding output during training. The classifier learns from the samples by fitting the function and predicts the corresponding label of each sample using its test features. Supervised learning is widely used in ML and DL, with SVM and KNN common in ML and neural networks, such as CNN and recurrent neural network (RNN) in DL. In unsupervised learning, each sample does not have its corresponding label, and the classifier classifies those with similar features into one class using the features of the samples, thereby completing the classification task for each class. Common unsupervised learning classification methods include k-means clustering algorithms, principal component analysis (PCA), and simulated annealing algorithms. Semi-supervised learning is a combination of supervised and unsupervised learning, which allows a portion of samples to have corresponding labels, reduces label collection, and increases model generalization. Common semi-supervised learning classification methods include Transductive Support Vector Machines (TSVM) and Semi-Boost.

For unsupervised learning algorithms. Verkerken et al. [27] evaluated four unsupervised classification models, autoencoder, one-class SVM, isolation forest, and PCA in intrusion detection classification using the CICIDS-2017 and CSE-CIC-IDS-2018 datasets. Each of the four classifiers performed strongly or weakly on the two datasets, indicating the difference in the processing ability of different classifiers for different datasets. For semi-supervised learning algorithm. Yuan et al. [28] used a semi-supervised classification method in network intrusion detection classification for three AdaBoost algorithms as classifiers using the KDDcup99 dataset. Only a small number of samples were labeled during training to improve the model generalization, which had higher classification accuracy compared with the semi-supervised SVM method. Li, Meng, and Au [29] proposed a decision tree algorithm for detection and classification using the semi-supervised model with a lower error rate and higher hit rate in comparison with the KNN, SVM, and random forest algorithms. Unsupervised learning algorithm and semi-supervised learning algorithm have their advantages, but this paper needs to improve the detection rate of minority samples, so it needs to be implemented by the supervised learning algorithm. There have been many studies on network traffic anomaly detection using supervised learning algorithms. Sun et al. [30] used an SVM classifier in network intrusion detection for smart meters. It has a shorter training time compared with neural networks, which enables SVM to update the data quickly and maintain high detection accuracy. Hu, Liu, and Ma [31] used a combination of CNN, attention, and LSTM in network security situation element extraction to extract features, and finally input them to the softmax classifier for classification, which has a higher detection accuracy than Multilayer Perceptron (MLP) and RNN on the KDDcup99 dataset. The Transformer has developed rapidly in recent years. For feature extraction classification tasks, it is more used in text classification. Tezgider et al. [32] used Transformer for text classification and achieved better results than other DL models. Zhang et al. [33] used the improved Transformer for extreme multi-label text classification and achieved better results than the traditional Transformer.

In this paper, we use the supervised learning algorithm Transformer by combining CGAN for network traffic detection and classification. Compared with other supervised learning algorithms, it has better feature learning ability and

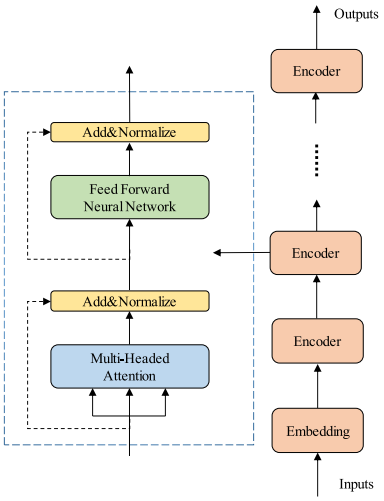

FIGURE 1. Transformer structural framework.

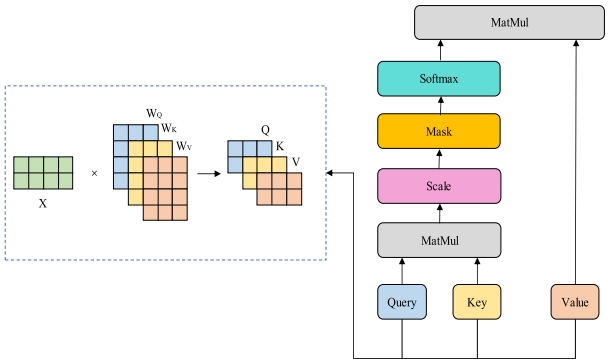

FIGURE 2. Scaled dot-product attention layers structural framework.

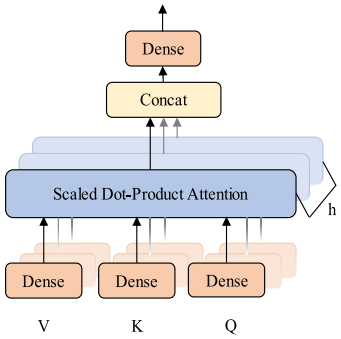

FIGURE 3. Multi-head attention structural framework.

can get better training and testing results by learning to fit the model.

III. METHODOLOGY

In this section, we introduce the Transformer and CGAN and discuss their analysis using the UNSW-NB15 and KDDcup99 datasets and model evaluation metrics. Finally, we derive the overall model architecture.

A. TRANSFORMER

The transformer model was proposed by Google in "Attention Is All You Need" in 2017 as an effective alternative to RNNs [14]. The overall architecture of the transformer is an encoder-decoder structure, i.e., a sequence is an input at the input and the corresponding target sequence is output at the output [34], such that the encoder and decoder structures are approximately the same. Figure 1 shows the coding structure of the feature extraction classification used in this paper. It comprises multiple encoders stacked, each containing the multi-head attention mechanism, addition and normalization, and feedforward neural network. The core of the transformer encoder is the attention mechanism. Although traditional RNNs rely on generating linear sequences in the hidden layer nodes to pass state information forward, the transformer uses its multi-head attention mechanism to capture long-range features of the sequences. This method avoids the gradient disappearance or explosion problem caused by directional propagation. The following describes each encoder's component module separately.

1) MULTIHEAD ATTENTION

The multi-head attention is composed of multiple scaled dot-product attention layers [35]. Figure 2 shows the structure of scaled dot-product attention layers. The scaled dot-product attention layers calculate the correlation between sequence data by computing the three vectors, namely, query, key, and value. After inputting a value X, three matrices, namely, W_Q, W_K, and W_V, are randomly generated (the three matrices are updated by backpropagation), and X is multiplied by W_Q, W_K, and W_V to get Q, K, and V, respectively. After multiplying Q with K generated in this sequence, the scale can be used to obtain the attention of X compared with other values in the sequence, and then multiply V by softmax calculation to obtain the attention vector of this value. Also, query, key, and value can be calculated from the matrix form to speed up the operation. Scaled dot-product attention layers are expressed as follows:

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

As shown in Figure 3, a multiheaded mechanism is used for scaled dot-product attention layers and their structure. The multiheaded mechanism obtains multiple calculations by simultaneously initializing multiple sets of Q, K, and V matrices. The role of "Concat" is to concatenate the multi-head attention matrix into a matrix, and then multiply the concatenate matrix with the weight matrix and input it into the feedforward neural network. This method enables the model to learn relevant information in different representation subspaces [36], expanding the ability of the model to focus on different locations. Its formula is as follows:

$$MultiHead\ (Q, K, V) = Concat(head_1, \ldots, head_h) \tag{2}$$

$$head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{3}$$

2) ADD AND NORMALIZATION

This layer includes the residual network [37] and normalization, which solves the gradient disappearance problem and

weight matrix degradation to make the information transfer deeper due to the deeper layers of the model. The normalization method uses layer normalization [38], which differs from batch normalization [39] which calculates the variance of the sample for each batch and calculates the mean and variance for all neurons in a layer. Although different samples have different means and variances, this method eliminates the effect of batch size on normalization. The formulas of the residual module and Layer normalization are as follows.

$$LayerNorm(x + SubLayer(x)) \tag{4}$$

$$LN\left(x_i\right) = \alpha \times \frac{x_i - \mu_L}{\sqrt{\sigma_L^2 + \epsilon}} + \beta \tag{5}$$

where α and β are the learned parameters, μ_{L} is the mean, σ_{L} is the standard deviation, and ϵ is the increase that prevents the divisor from being 0.

3) FEEDFORWARD NEURAL NETWORK

The feedforward neural network contains a two-layer fully connected layer, whose activation function uses Relu. The previous multi-head attention was performed to learn linear transformations. The feedforward neural network layer is added to learn nonlinear transformations. The nonlinear transformation of Relu is used to enhance partial mappings with large weight values and suppress the partial mappings with small weight values, thus further enhancing the learning ability of the model. The formula is as follows:

$$FFN(x) = max\left(0, xW_1 + b_1\right) W_2 + b_2 \tag{6}$$

B. CGAN

GAN [13] proposed by Goodfellow in 2014, the network consists of a generator and discriminator. The main function of the generator is to generate false data, while the discriminator discriminates between the generated false data and the input real data. Through multiple iterations, the classifier reaches the training requirement when the classification effect of generated data and real data reaches equilibrium, and the generated data can be used as real data for training.

The data generated by traditional GAN is random and cannot distinguish the type of data. The task goal of this paper is to expand the minority class samples and improve the detection accuracy of minority classes. Therefore, CGAN, which can expand the specified class, is selected for oversampling. Compared with GAN, CGAN adds conditional parameters to the generator input, so that new data can be generated according to the specified category. The structure is shown in Figure 4. The input of the generator is random noise Z and label Y, and the false data G is generated after the generator. The discriminator judges the real data X and the false data G input. The generator and the discriminator are composed of multi-layer neural networks. In the training process, the parameters of the generator are adjusted by reverse propagation. Finally, the false data generated by the generator and the real data reach equilibrium in the discriminator. The

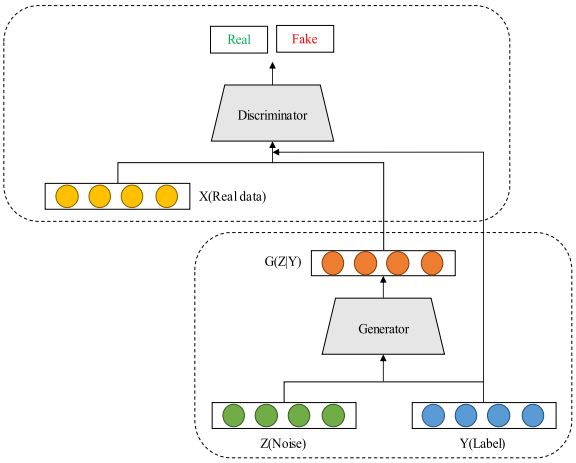

FIGURE 4. The CGAN structure.

TABLE 1. UNSW-NB15 dataset distribution.

	Class	Training data	Percentage	Test data
	Normal	56,000	31.94%	37,000
Attack	Fuzzers	18184	10.37%	6062
	Analysis	2000	1.14%	677
	Backdoors	1746	1.00%	583
	Dos	12264	6.99%	4089
	Exploits	33393	19.04%	11132
	Generic	40000	22.81%	18871
	Reconnaissance	10491	5.98%	3496
	Shellcode	1133	0.65%	378
	Worms	130	0.07%	44
	Total	175341	100%	82332

formula of the loss function is as follows.

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[logD(\boldsymbol{x} \mid \boldsymbol{y})] + E_{\mathbf{z} \sim p_z(\mathbf{z})}[log(1 - D(G(\mathbf{z} \mid \boldsymbol{y})))] \tag{7}$$

C. DATASET DESCRIPTION

1) UNSW-NB15

The UNSW-NB15 dataset was created by the Australian Center for network security in 2015. The dataset consists of real normal traffic and generated attack traffic. It contains most of the modern network traffic, and there is a sample imbalance in the data set, which is suitable for the experimental verification of network anomaly traffic detection in this paper. The dataset contains 42 feature attributes and two class identifiers. One type of identifier contains normal and attack types, and the other type of identifier has a total of 10 categories, of which 9 are attack types and one is a normal type. The data are divided into two parts, namely, training and test sets, containing 175,341 and 82,332, respectively, as shown in Table 1.

TABLE 2. KDDCup99 dataset categories.

Class		Type of attack	
		Training data	Test data
Normal		normal	
Attack	DoS	back, land, pod, smurf, neptune, teardrop	back, land, neptune, smurf, teardrop, pod, mailbomb, udpstorm, apache2, processtable
	Probing	ipsweep, nmap, portsweep, satan	ipsweep, portsweep, satan, mscan, saint, nmap
	R2L	spy, guess-passwd, warezclient, imap, warezmaster, phf, multihop, fpt-write	guess-passwd, named, imap, multihop, phf, fptwrite, xlock, snmpgetattack, warezmaster, worm, spy, xsnoop, snmpguess, sendmail, warezclient
	U2R	buffer-overflow, loadmodule, perl, rootkit	buffer-overflow, perl, rootkit, ps, sqlattack, xterm, loadmodule, httptunnel

2) KDDCUP99

KDDcup99 is a public intrusion detection dataset. It is one of the most widely used data sets in intrusion detection. It contains many traffic characteristics and has a serious sample imbalance. It is suitable for the experimental verification of network abnormal traffic detection in this paper. The original data volume is large, and 10% of each set was selected for experiments in this paper. The dataset contains 41 feature attributes and a class identifier. The class identifier contains five major categories, of which four are attack types. The attack types are further divided into 39 subcategories, of which the training and test sets contain 22 and 39 types, respectively, as shown in Table 2. The test set contains attack types that are not present in the training set as a way to check the generalization of the model. The training and test sets have 494021 and 311029 records, respectively, as shown in Table 3.

D. DATASET PREPROCESSING

We train 42 feature items of the UNSW-NB15 dataset and 41 feature items of the KDDcup99 dataset. In their feature items, three feature items are non-numerical characteristics. Since non-numerical characteristics cannot be used as input to this model, they must be numerically transformed first. Here, the three features are numerically mapped directly by

TABLE 3. Experimental operating environment.

Class	Training data	Percentage	Test data
Normal	97278	19.69%	60593
DoS	391458	79.24%	229853
Probing	4107	0.83%	4166
R2L	1126	0.23%	16189
U2R	52	0.01%	228
Total	494021	100%	311029

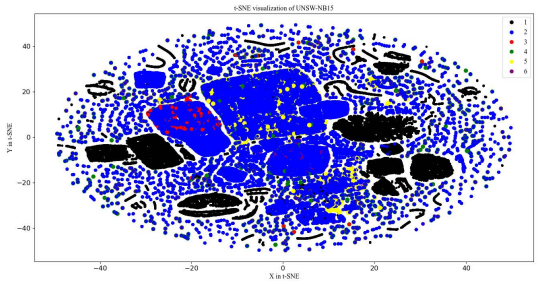

FIGURE 5. Data distribution of UNSW-NB15.

label encoding. The UNSW-NB15 dataset is a mixture of real network traffic and laboratory-generated traffic, with fewer attack samples for some categories. The KDDcup99 dataset is obtained from a simulated local area network, where DoS attacks account for nearly 80 %, and the number of samples for other attack categories is too small. Since both datasets simulate real network traffic, there are too few samples for some low-frequency attacks. Therefore, CGAN oversampling is required to balance the dataset. Figures 5 and 6 represent the data distribution of the UNSW-NB15 dataset before and after oversampling. Among them, 1 represents the normal category, 3,4,5, and 6 represent Analysis, Backdoors, Shellcode, and Worms, respectively, and 2 represents other attack categories. As can be seen from the graph, after CGAN oversampling, the number of samples in the four minority classes of Analysis, backdoor, shellcode code, and Worms increased. Figure 7 and Figure 8 are the data distribution of the KDDcup99 dataset before and after oversampling. Among them, 1 represents the normal class, and 2, 3, 4, and 5 represent DoS, Probing, R2L, and U2R attacks, respectively. Similarly, after CGAN oversampling, the number of Probing, R2L, and U2R minority class samples has been improved. Next, we standardized the data by performing linear transformation data to speed up model convergence and prevent errors caused by certain numerical anomalies. The equation for the standardization is as follows:

$$X_{new} = \frac{X_i - \mu}{\sigma} \tag{8}$$

E. EVALUATION METRICS

To better evaluate the experiment, six metrics, namely, accuracy, recall, precision, F1-score, specificity, and FAR, were used in this paper. Accuracy is measured as the ratio of the number of correct classifications to the total number, focusing on the overall assessment. The recall metric is the ratio of

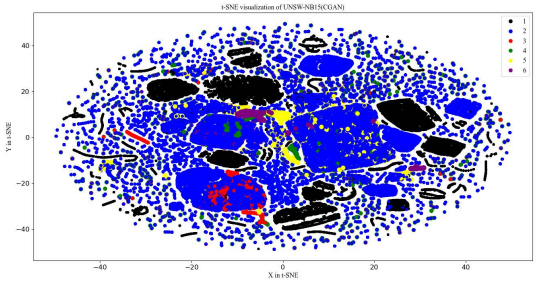

FIGURE 6. Data distribution of UNSW-NB15(CGAN).

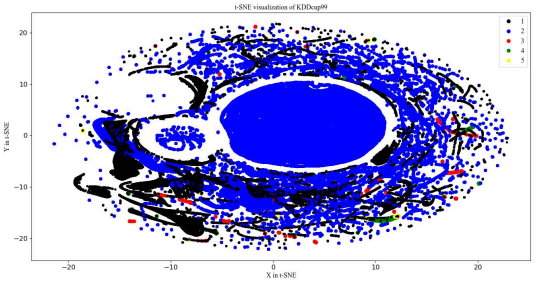

FIGURE 7. Data distribution of KDDcup99.

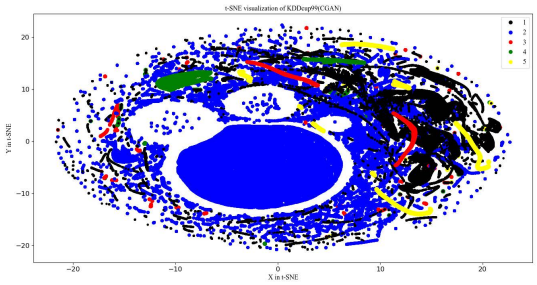

FIGURE 8. Data distribution of KDDcup99(CGAN).

the number of correct classifications for this category to the total number of such categories. Precision is the number of correct classifications for this category to the total number of categories judged as such. F1-score is the summed average of recall and precision. Specificity is the ratio of the number of correctly classified categories outside this category to the total number of other categories. The FAR is the ratio of the number of incorrectly classified categories outside this category to the total number of other categories. The formula for evaluating the indicators is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

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

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

$$F1 - score = \frac{2PR}{P + R} \tag{12}$$

$$Specificity = \frac{TN}{TN + FP} \tag{13}$$

$$FAR = \frac{FP}{TN + FP} \tag{14}$$

F. MODEL STRUCTURE

The overall model structure is constructed from the above method, including the training set, test set, preprocessing module, Transformer, and softmax classifier. Figure 9 shows the diagram of the model structure. The construction of the whole model is divided into two parts. The first part is data preprocessing. This part is to filter and reconstruct the traffic data. Preprocessing is divided into two parts: the training set and the test set. The training set has one more oversampling operation than the test set. Preprocessing Firstly, the data is numerically processed, and some character data is transformed into numerical data suitable for deep learning model learning. Then the training set is oversampled, the minority samples in the data set are expanded by CGAN, and the new data generated by CGAN is added to the original training set. Finally, standardization improves the model's ability to learn the characteristics of data. The second part is feature extraction and classification. After preprocessing, the training data is input into the Transformer model for feature extraction. After learning the feature distribution of the data, the internal weight parameters of the trained neural network are used to predict the test data. Finally, the test results are classified by the softmax function, and the classification results are evaluated by the metrics. Algorithm 1 illustrates the operation process of the model.

IV. EXPERIMENT AND RESULTS

In this section, the model was validated by comparing two sets of experiments. The first is an experimental comparison of different oversampling methods, and the second is feature extraction and classification comparison of the transformer model with other advanced models. The experiments were validated on the UNSW-NB15 and KDDcup99 datasets. The UNSW-NB15 dataset was subjected to binary classification, detecting both normal and attacks, whereas the KDDcup99 dataset was subjected to a multicategory classification, detecting the normal attack and the other four attacks.

A. IMPLEMENTATION

In the oversampling experiment, for UNSW-NB15 and KDDcup99 datasets, three oversampling methods, Random Oversampling, ADASYN, and SMOTE, were compared with the CGAN method.

Two main feature extraction algorithms were used in this study: ML and DL algorithms. Four ML algorithms, namely, AdaBoost, Logistic Regression (LR), Random Forest (RF), and KNN [40], were used for comparison experiments. Similarly, three DL algorithms, namely, CNN, LSTM, and Bi-LSTM, were used for comparison experiments.

For the experimental environment, the CPU is the Intel core i5 10600 KF with an NVIDIA RTX2060 GPU. The model was implemented by TensorFlow, Keras, and Sklearn methods using Python programming language. Table 4 shows the experimental environment.

Determining the most appropriate hyperparameter setting through multiple experiments. In the training of the CGAN

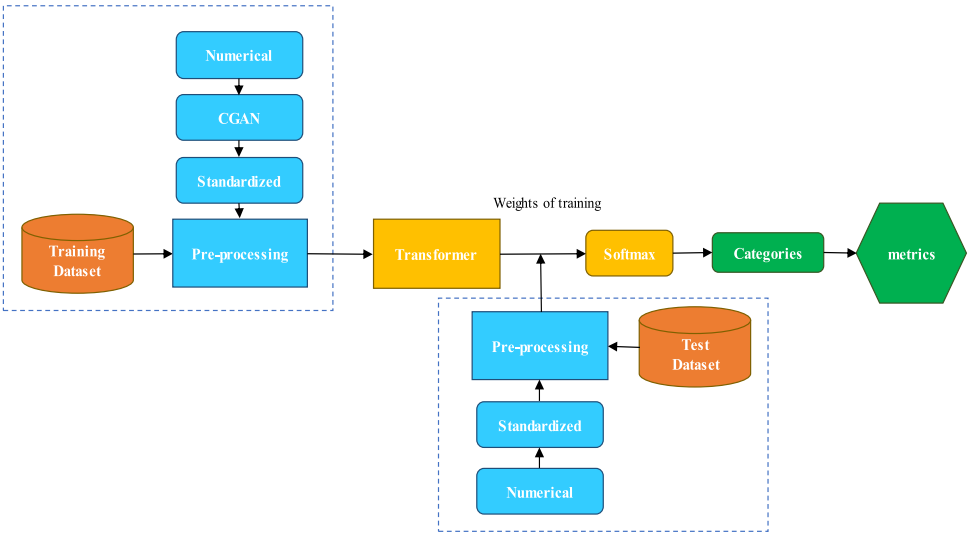

FIGURE 9. The framework for network security situation element extraction based on CGAN and transformer.

Algorithm 1 The Computational Flow of the Model

Input: Training dataset, Test dataset
Output: Classification metrics
Extract Features (x) and Labels (y) from the Training dataset and Test dataset
while data pre − processing **do**
    Perform numerical encoding;
    Oversampling using CGAN;
    Standardization of features;
for i in {1, 2, . . . , n} **do**
    Load Transformer model
    Input Training dataset to Transformer model
    Calculate training loss
    Backpropagation update weight
    Save the updated model
    Repeat until the cycle is complete
Test model with the Test dataset
Calculate Classification metrics
end

model, the generator and discriminator use four layers of the neural network, and the activation function uses tanh and sigmoid respectively. The core parameters are shown in Table 5. The learning ability of the model is enhanced by gradually increasing each layer of neurons, where ''input_dim'' represents the feature dimension of the input data. In the training of Transformer, the Adam optimizer and Relu activation function were used in the feedforward neural network. Because the two datasets have different adaptability to model parameters, they have different parameter combinations for different data sets, where ''num_layers'' represents the total number of encoder layers, ''num_heads'' represents the number of attention layers in the multi-head attention mechanism,

TABLE 4. Experimental operating environment.

Project	Environmental Parameters
CPU	Intel core i5 10600KF
GPU	NVIDIA RTX2060
Python version	3.7
TensorFlow version	2.1
Keras version	2.3.1

''d_model'' represents the input and output dimensions in the feedforward network, and ''dff'' represents the inner dimension of the feedforward network. Table 5 shows the core parameters of the model.

B. OVERSAMPLING

1) OVERSAMPLING PROCESSING

In the UNSW-NB15 dataset, normal samples had the largest proportion, accounting for 31.94% of the training set. Among the attack types, four items, namely, Analysis, Backdoors, Shellcode, and Worms, accounted for relatively small proportions, including 1.14%, 1.00%, 0.65%, and 0.07% of the total training set, respectively. Too few samples will lead to insufficient model training, resulting in a low detection rate for four types of attacks. Therefore, it is necessary to expand the samples of these categories by oversampling. These four items are oversampled by the CGAN method. The training set and test set after oversampling are shown in Figure 10. Train1 is the original training set, and Train2 is the training set after CGAN oversampling. Table 7 shows a summary of the number of each type after resampling and the number of normal and attack terms that were dichotomized.

In the KDDcup99 dataset, DoS attacks had the largest proportion, accounting for 79.24% of the training set, and Probing, R2L, and U2R accounted for 0.83%, 0.23%, and

TABLE 5. CGAN model core parameters.

Generator	Discriminator
Dense (128, LeakyReLU)	Dense (128, LeakyReLU)
BatchNormalization	Dropout (0.2)
Dense (256, LeakyReLU)	Dense (256, LeakyReLU)
BatchNormalization	Dropout (0.2)
Dense (512, LeakyReLU)	Dense (512, LeakyReLU)
BatchNormalization	Dropout (0.2)
Dense (input_dim, tanh)	Dense (1, sigmoid)

TABLE 6. Transformer model core parameters.

Parameters	Parameter value	
	UNSW-NB15	KDDcup99
optimizer	Adam	Adam
activation function	Relu	Relu
num_layers	2	1
num_heads	8	8
d_model	512	512
dff	1024	2048
learning_rate	0.00005	0.00005
batch_size	32	32
dropout	0.25	0.25

0.01%, respectively. The three items are oversampled by the CGAN method. The training set and test set after oversampling are shown in Figure 11, where Train1 is the original training set and Train2 is the training set after oversampling. Table 8 presents the number of categories after oversampling.

2) OVERSAMPLING EXPERIMENT

First, the experiments compared the dataset without sample balancing with the dataset after oversampling using the transformer model for classification experiments. Figures 12–15 show the confusion matrices before and after they were derived. From Figures 12 and 13, the detection rate of the transformer model for the attack class of the UNSW-NB15 dataset has decreased after oversampling; however, the detection rate for the normal class significantly increased. Furthermore, the detection rate of the transformer model for the attack class of the UNSW-NB15 dataset decreased contrary to that of the normal class which significantly increased from 25909 to 29594. From Figures 14 and 15, the transformer model increased the number of correct detections for each KDDcup99 dataset after oversampling, except for a slight decrease in the number of normal detections,

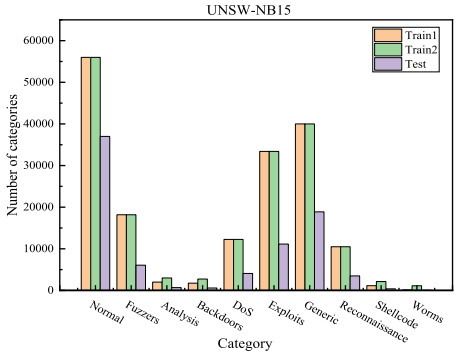

FIGURE 10. Before and after oversampling of UNSW-NB15 dataset.

TABLE 7. Before and after oversampling of the UNSW-NB15 dataset.

	Class	Unsampled Balance	Sample Balance	Total
	Normal	56000	56000	56000
Attack	Fuzzers	18184	18184	123341
	Analysis	2000	3000	
	Backdoors	1746	2746	
	Dos	12264	12264	
	Exploits	33393	33393	
	Generic	40000	40000	
	Reconnaissance	10491	10491	
	Shellcode	1133	2133	
	Worms	130	1130	

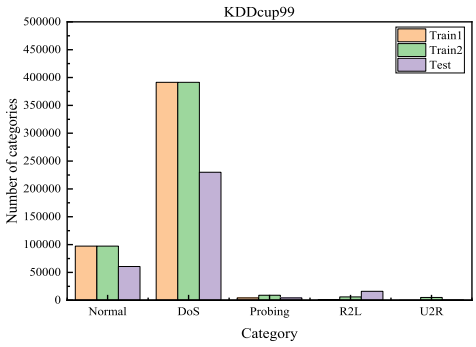

FIGURE 11. Before and after oversampling of the KDDcup99 dataset.

especially for the R2L attack, where the number of correct detections increased from 135 to 2257, indicating that oversampling improves the detection rate of a few class samples. Tables 9 and 10 present the evaluation indicators before and after oversampling. The UNSW-NB15 dataset showed an increase in the F-score for both the Normal and Attacked after oversampling using CGAN. With an increase in overall F-score from 86.9% to 89.8%, overall specificity increased from 82.9% to 87.7%, and overall FAR decreased from 17.1% to 12.3%. After oversampling the KDDcup99 dataset using CGAN, the overall metrics improved with the larger improvement observed in the R2L and U2R attacks,

TABLE 8. Before and after oversampling of the KDDcup99 dataset.

Class		Unsampled Balance	Sample Balance	Total
Normal		56000	56000	56000
Attack	Fuzzers	18184	18184	123341
	Analysis	2000	3000	
	Backdoors	1746	2746	
	Dos	12264	12264	
	Exploits	33393	33393	
	Generic	40000	40000	
	Reconnaissance	10491	10491	
	Shellcode	1133	2133	
	Worms	130	1130	

where the F-score increased from 1.7% to 24.2% and 4.3% to 15.5%, respectively, indicating that the oversampling was essential to correct detection. At the same time, it can also be seen that the Recall of R2L and U2R attacks is still at a low level after oversampling, because there are many differences in the feature distribution of the training set and the test set of these two categories. To further reduce the difference in feature distribution, consider combining other oversampling methods to generate new data to avoid the limitations of data generated by a single model.

To verify the effectiveness of the used oversampling method, we compared other oversampling methods with the proposed method. Random Oversampling, ADASYN, SMOTE, and the three methods were compared with the CGAN method. The number of oversampling was the same as that of CGAN. The experimental results are shown in Table 11 and Table 12. It can be seen from table 11 that all three oversampling methods have improved the performance indicators, but the CGAN method has the greatest improvement. Accuracy and F1-score increased from 85.81 % to 89.38 % and 86.92 % to 89.75 %, respectively. It can be seen from Table 12 that in the three oversampling methods, except ADASYN, the other two indicators have been improved. Among them, CGAN has the best effect. Accuracy and F1-score have increased from 92.40 % to 93.07 % and 93.25 % to 93.68 %, respectively. In summary, the CGAN method has better performance than other oversampling methods on two datasets.

C. CLASSIFICATION RESULTS IN DIFFERENT ALGORITHMS

The classification training started after the sample balancing. The UNSW-NB15 and KDDcup99 datasets were classified into two and five, respectively. Seven currently advanced algorithms (AdaBoost, LR, RF, KNN, CNN, LSTM, and Bi-LSTM) were selected for comparison, all using the resampled dataset. Tables 12 and 13 show the experimental results, and the metrics in the tables are the overall evaluation metrics of each algorithm classification. Table 13 shows that the transformer algorithm outperformed other algorithms in all metrics on the UNSW-NB15 dataset, with accuracy and F1-score of 89.38% and 89.75%, respectively. The CNN, LSTM, and RF algorithms also performed well, with accuracy and F1-score reaching over 87% and 88%, respectively.

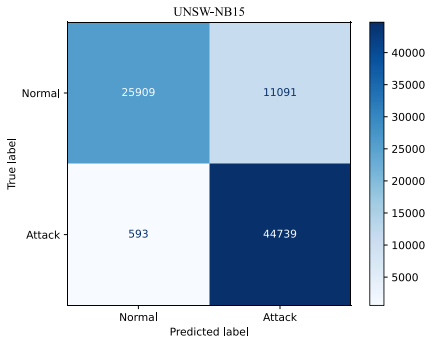

FIGURE 12. Transformer confusion matrix for UNSW-NB15.

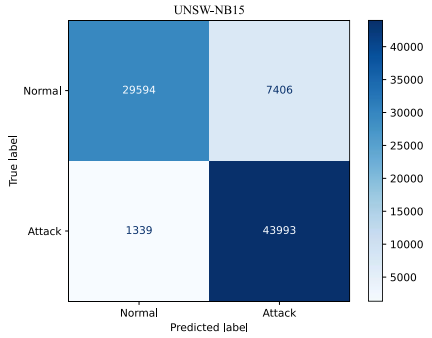

FIGURE 13. CGAN-Transformer confusion matrix for UNSW-NB15.

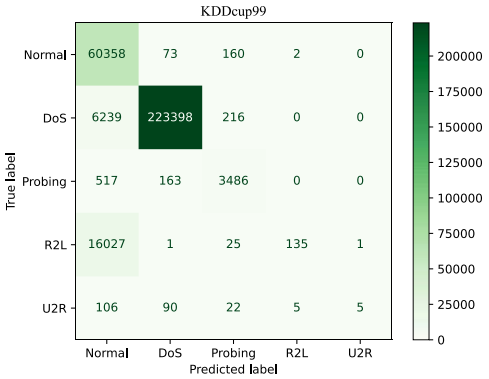

FIGURE 14. Transformer confusion matrix for KDDcup99.

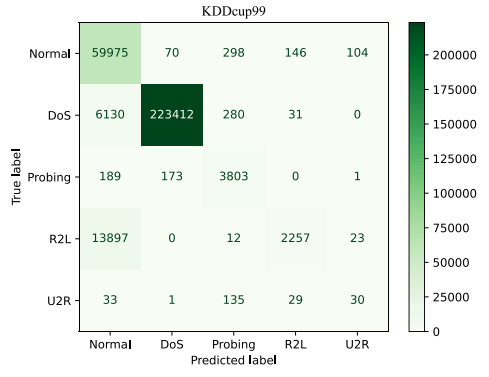

FIGURE 15. CGAN-Transformer confusion matrix for KDDcup99.

TABLE 9. Comparison of classification metrics before and after oversampling of the UNSW-NB15 dataset.

Method	Class	Recall	Precision	F1-score	Specificity	FAR
Unbalance	Normal	70.0%	**97.8%**	81.6%	**98.7%**	1.3%
	Attack	**98.7%**	80.1%	88.5%	70.0%	**30.0%**
	Total	85.8%	88.1%	86.9%	82.9%	17.1%
CGAN	Normal	**80.0%**	95.7%	**87.1%**	97.1%	**3.0%**
	Attack	97.0%	**85.6%**	**91.0%**	**80.0%**	20%
	Total	**89.4%**	**90.1%**	**89.8%**	**87.7%**	**12.3%**

TABLE 10. Comparison of classification metrics before and after oversampling of KDDcup99 dataset.

Method	Class	Recall	Precision	F1-score	Specificity	FAR
Unbalance	Normal	**99.6%**	72.5%	83.9%	90.9%	9.1%
	DoS	97.2%	**99.9%**	**98.5%**	99.6%	0.4%
	Probing	83.7%	**89.2%**	86.3%	**99.9%**	**0.1%**
	R2L	0.8%	**95.1%**	1.7%	**99.9%**	0.1%
	U2R	2.2%	**83.3%**	4.3%	**99.9%**	0.1%
	Total	92.4%	94.1%	93.2%	97.9%	2.1%
CGAN	Normal	99.0%	**74.8%**	**85.2%**	**91.9%**	**8.1%**
	DoS	**97.2%**	**99.9%**	**98.5%**	**99.7%**	**0.3%**
	Probing	**91.3%**	84.0%	**87.5%**	99.8%	0.2%
	R2L	**13.9%**	91.6%	**24.2%**	**99.9%**	**0.1%**
	U2R	**13.2%**	19.0%	**15.5%**	**99.9%**	**0.1%**
	Total	**93.1%**	**94.3%**	**93.7%**	**98.2%**	**1.8%**

TABLE 11. Comparison of different oversampling techniques for the UNSW-NB15 dataset.

Method	Accuracy	Recall	Precision	F1-score	Specificity	FAR
Imbalanced	85.81%	85.81%	88.06%	86.92%	82.90%	17.10%
Random Oversampling	88.98%	88.98%	89.89%	89.43%	87.10%	12.90%
ADASYN	87.99%	87.99%	89.24%	88.61%	85.80%	14.20%
SMOTE	88.37%	88.37%	89.37%	88.87%	86.37%	13.63%
CGAN	**89.38%**	**89.38%**	**90.12%**	**89.75%**	**87.65%**	**12.35%**

TABLE 12. Comparison of different oversampling techniques for the KDDcup99 dataset.

Method	Accuracy	Recall	Precision	F1-score	Specificity	FAR
Imbalanced	92.40%	92.40%	94.12%	93.25%	97.92%	2.08%
Random Oversampling	92.77%	92.77%	94.24%	93.50%	98.19%	1.81%
ADASYN	92.30%	92.30%	92.29%	92.29%	98.12%	1.88%
SMOTE	92.99%	92.99%	94.27%	93.63%	98.18%	1.82%
CGAN	**93.07%**	**93.07%**	**94.29%**	**93.68%**	**98.20%**	**1.80%**

The two traditional machine learning algorithms, AdaBoost and LR, performed poorly with an accuracy of less than 80%. Results in Table 14 show that the transformer algorithm outperformed other algorithms on the KDDcup99 dataset in all metrics with accuracy and F1-score reaching 93.07% and 93.68%, respectively. The four algorithms, KNN, CNN, LSTM, and Bi-LSTM, also performed well in multiclassification, with less than a 1% difference compared with the transformer metrics. Poor results were recorded for RF, LR, and AdaBoost metrics. Next, we will discuss the four algorithms of KNN, CNN, LSTM, and Bi-LSTM and transformer in more detail.

To further verify the performance of the transformer algorithm feature learning, we discuss the evaluation indices of each sample category for the four algorithms, KNN, CNN, LSTM, Bi-LSTM, and transformer algorithm. Figures 16–19

TABLE 13. Classification results of different algorithms on the UNSW-NB15 dataset.

Method	Accuracy	Recall	Precision	F1-score	Specificity	FAR
AdaBoost	78.60%	78.60%	83.84%	81.14%	73.98%	26.02%
LR	79.01%	79.01%	84.39%	81.61%	74.39%	25.61%
RF	87.36%	87.36%	89.07%	88.21%	84.85%	15.15%
KNN	83.91%	83.91%	85.80%	84.85%	81.06%	18.94%
CNN	87.31%	87.31%	88.97%	88.13%	84.83%	15.17%
LSTM	87.73%	87.73%	88.94%	88.33%	85.54%	14.46%
Bi-LSTM	81.56%	81.56%	85.81%	83.63%	77.52%	22.48%
Transformer	**89.38%**	**89.38%**	**90.12%**	**89.75%**	**87.65%**	**12.35%**

TABLE 14. Classification results of different algorithms on the KDDcup99 dataset.

Method	Accuracy	Recall	Precision	F1-score	Specificity	FAR
AdaBoost	74.49%	74.49%	83.95%	78.94%	83.82%	16.18%
LR	92.01%	92.01%	92.07%	92.04%	97.55%	2.45%
RF	91.11%	91.11%	93.44%	92.26%	98.02%	1.98%
KNN	92.62%	92.62%	94.21%	93.41%	98.09%	1.91%
CNN	92.60%	92.60%	94.08%	93.33%	98.10%	1.90%
LSTM	92.70%	92.70%	93.52%	93.11%	98.16%	1.84%
Bi-LSTM	92.63%	92.63%	93.97%	93.30%	98.07%	1.93%
Transformer	**93.07%**	**93.07%**	**94.29%**	**93.68%**	**98.20%**	**1.80%**

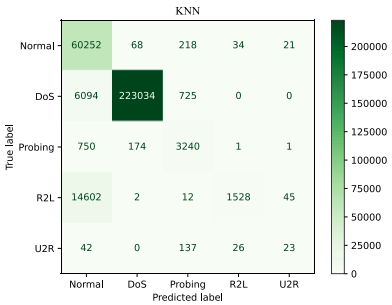

FIGURE 16. KNN confusion matrix for KDDcup99.

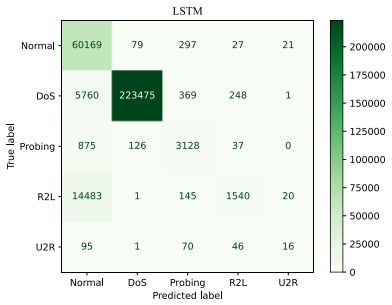

FIGURE 18. LSTM confusion matrix for KDDcup99.

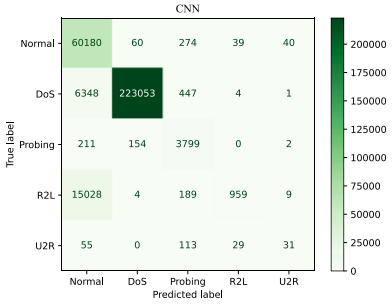

FIGURE 17. CNN confusion matrix for KDDcup99.

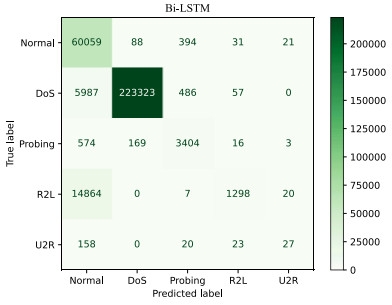

FIGURE 19. Bi-LSTM confusion matrix for KDDcup99.

show their confusion matrices. It is shown that the number of correct detections is highest for DoS, Probing, and R2L using the transformer algorithm. Also, the number of correct detections for both the Normal and U2R categories was slightly less than the maximum level. For attacks such as R2L, the number of correct detections for all five algorithms was low, with the lowest being CNN, which has only 959 correct detections. Table 15 gives the evaluation metrics for each category of the five algorithms, and it is shown that each algorithm has the highest F1-score for the DoS category because of the largest number of DoS categories. The main difference is reflected in the detection of the four minority classes, namely, Normal, Probing, R2L, and U2R. The F1-scores of the transformer model on these four categories are 85.2%, 87.5%, 24.2%, and 15.5%, ranking first, first, first, and third among the five methods, and the specificity is 91.9%, 99.8%, 99.9%, and 99.9%, ranking the highest among the five methods. This indicates that the transformer model is better at feature learning for a few classes and has better feature learning ability than other algorithms.

TABLE 15. Classification results of different algorithms on KDDcup99 dataset.

Method	Class	Recall	Precision	F1-score	Specificity	FAR
KNN	Normal	**99.4%**	73.7%	84.7%	91.4%	8.6%
	DoS	97.0%	**99.9%**	98.4%	99.7%	0.3%
	Probing	77.8%	74.8%	76.3%	99.6%	0.4%
	R2L	9.4%	**96.2%**	17.2%	**99.9%**	**0.1%**
	U2R	10.1%	25.6%	14.5%	**99.9%**	**0.1%**
CNN	Normal	99.3%	73.5%	84.5%	91.4%	8.6%
	DoS	97.0%	**99.9%**	**98.5%**	99.7%	0.3%
	Probing	91.2%	78.8%	84.5%	99.7%	0.3%
	R2L	5.9%	93.0%	11.1%	**99.9%**	**0.1%**
	U2R	**13.6%**	37.3%	**19.9%**	**99.9%**	**0.1%**
LSTM	Normal	99.3%	73.9%	84.8%	91.5%	8.5%
	DoS	**97.2%**	**99.9%**	**98.5%**	**99.8%**	**0.2%**
	Probing	75.1%	78.0%	76.5%	99.7%	0.3%
	R2L	9.5%	81.1%	17.0%	**99.9%**	**0.1%**
	U2R	7.0%	27.6%	11.2%	**99.9%**	**0.1%**
Bi-LSTM	Normal	99.1%	73.6%	84.5%	91.4%	8.6%
	Dos	**97.2%**	**99.9%**	**98.5%**	99.7%	0.3%
	Probing	81.7%	79.0%	80.3%	99.7%	0.3%
	R2L	8.0%	91.1%	14.7%	**99.9%**	**0.1%**
	U2R	11.8%	**38.0%**	18.1%	**99.9%**	**0.1%**
Transformer	Normal	99.0%	**74.8%**	**85.2%**	**91.9%**	**8.1%**
	Dos	**97.2%**	**99.9%**	**98.5%**	99.7%	0.3%
	Probing	**91.3%**	**84.0%**	**87.5%**	**99.8%**	**0.2%**
	R2L	**13.9%**	91.6%	**24.2%**	**99.9%**	**0.1%**
	U2R	13.2%	19.0%	15.5%	**99.9%**	**0.1%**

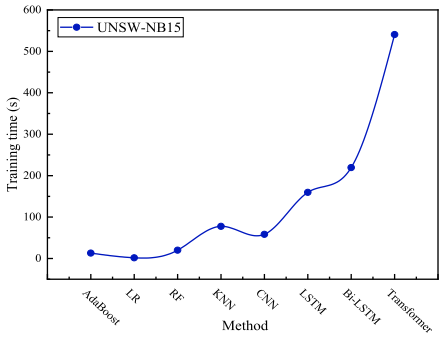

FIGURE 20. Training time for UNSW-NB15.

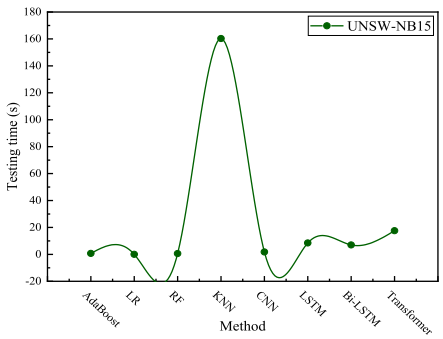

FIGURE 21. Test time for UNSW-NB15.

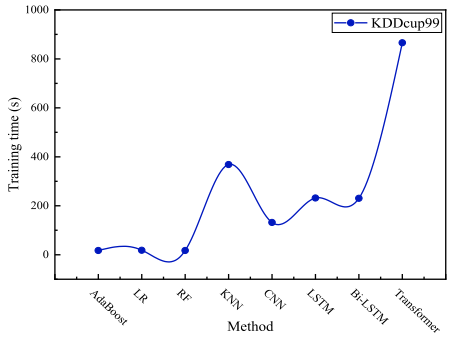

FIGURE 22. Training time for KDDcup99.

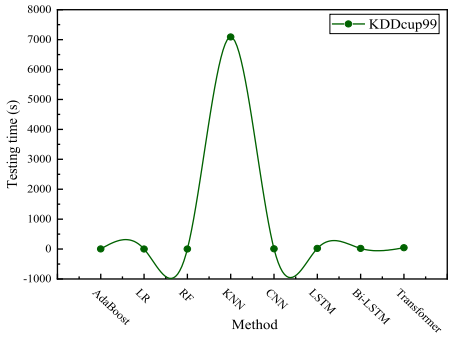

FIGURE 23. Test time for KDDcup99.

To further verify the superiority of the method combining CGAN and Transformer in network traffic anomaly detection, this method is compared with other advanced methods, as shown in Table 16 and Table 17. The methods compared with the UNSW-NB15 dataset are the results of binary classification of the test set. The methods that the KDDcup99 dataset compares are the results of 5 classifications of the test set.

TABLE 16. Comparison with advanced methods on the UNSW-NB15 dataset.

Method	Accuracy	Year
Decision Tree [41]	85.56%	2016
Two-stage Classifier [42]	85.78%	2018
ADASYN-LightGBM [21]	85.89%	2021
Improved SVM [43]	85.99%	2019
CGAN-Transformer	**89.38%**	2022

TABLE 17. Comparison with advanced methods on the KDDcup99 dataset.

Method	Accuracy	Year
ACGANs-CNN [44]	89.59%	2021
SMOTE-RF [45]	92.57%	2019
KNN [46]	92.90%	2018
DNN-3 [46]	93.00%	2018
CGAN-Transformer	**93.07%**	2022

D. RUNTIME ANALYSIS

The training and test time of each algorithm were compared, and the fairness of the experiments was guaranteed by running them on the same host. Figures 20 and 21 show the comparison of training and test time of the UNSW-NB15 dataset. Results show that the training time of the transformer model was the longest, followed by Bi-LSTM and LSTM. For the test time, KNN was the longest, followed by the transformer algorithm. Figures 22 and 23 are the comparison of the training and testing time of the KDDcup99 dataset. As with the UNSW-NB15 dataset, Transformer has the longest training time and KNN has the longest test time. The running time of KNN is related to the distribution of samples in the dataset, and the running time is longer on the samples with complex distribution, so KNN spends more time than other machine learning algorithms. The training time of the transformer model is relatively long because of its complex structure. However, the complex structure makes its feature learning ability stronger, and its excellent feature extraction ability is more important for situation element extraction. Therefore, it is worth taking more time to obtain higher detection accuracy for extracting network security situation elements.

V. CONCLUSION AND FUTURE WORKS

In this paper, we proposed a network security situation element extraction method using the CGAN model and the Transformer model. The sample imbalance problem is handled by CGAN, and feature extraction is performed using the transformer model. We validated the experiments with the UNSW-NB15 and KDDcup99 datasets. Our proposed method has higher metrics performance both on oversampling comparison experiments and comparison tests of feature extraction. In the experiments of feature extraction classification, the Accuracy, F1-score and Specificity are higher on both 2 and 5 classifications than with the AdaBoost, LR, RF, KNN, CNN, LSTM, and Bi-LSTM models. The overall accuracy, F1-score, and specificity were 89.38% and 93.07%, 89.75% and 93.68%, and 87.65% and 98.20% on the two datasets, respectively. On the KDDcup99 dataset, it also has better detection rates on three minority classes, Probing, R2L, and U2R. In future work, we can try to use more data sets to verify the effectiveness of the model. At the same time, it is necessary to consider that the model only performs oversampling and feature learning on the features extracted from the existing data set, and there is no good universality. Therefore, we can try to oversample and learn features directly from packet traces to increase the applicability of the method. And try a more lightweight model to improve training efficiency while maintaining feature extraction capabilities.

REFERENCES

[1] CNNIC. (2022). *Statistical Report on China's Internet Development*. Accessed: Jun. 13, 2022. [Online]. Available: https://www.cnnic.net.cn

[2] CNCERT. (2022). *Network Security Information and Dynamics Weekly Report*. Accessed: Jun. 13, 2022. [Online]. Available: https://www.cert.org.cn

[3] T. Alexander, W. Mazurczyk, A. Mishra, and A. Perotti, "Mobile communications and networks," *IEEE Commun. Mag.*, vol. 57, no. 4, p. 94, Apr. 2019.

[4] T. Bass, "Intrusion detection systems and multisensor data fusion," *Commun. ACM*, vol. 43, no. 4, pp. 99–105, Apr. 2000.

[5] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electron. Markets*, vol. 31, no. 3, pp. 685–695, 2021.

[6] B. SenthilnayakiK, K. Venkatalakshmi, and A. Kannan, "Intrusion detection system using fuzzy rough set feature selection and modified KNN classifier," *Int. Arab J. Inf. Technol.*, vol. 16, no. 4, pp. 746–753, 2019.

[7] G. Li, M. Yin, S. Jing, and B. Guo, "An effective algorithm for intrusion detection using random shapelet forest," *Wireless Commun. Mobile Comput.*, vol. 2021, pp. 1–9, Sep. 2021.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[9] R. Zhao, J. Yin, Z. Xue, G. Gui, B. Adebisi, T. Ohtsuki, H. Gacanin, and H. Sari, "An efficient intrusion detection method based on dynamic autoencoder," *IEEE Wireless Commun. Lett.*, vol. 10, no. 8, pp. 1707–1711, Aug. 2021.

[10] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "LSTM-based intrusion detection system for in-vehicle can bus communications," *IEEE Access*, vol. 8, pp. 185489–185502, 2020.

[11] Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, "A bidirectional LSTM deep learning approach for intrusion detection," *Expert Syst. Appl.*, vol. 185, Dec. 2021, Art. no. 115524.

[12] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics*, vol. 9, no. 6, p. 916, Jun. 2020.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, vol. 2, 2014, pp. 1–9.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 1–11.

[15] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[16] K. Siddique, Z. Akhtar, F. A. Khan, and Y. Kim, "KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research," *Computer*, vol. 52, no. 2, pp. 41–51, Feb. 2019.

[17] V. A. Fajardo, D. Findlay, C. Jaiswal, X. Yin, R. Houmanfar, H. Xie, J. Liang, X. She, and D. B. Emerson, "On oversampling imbalanced data with deep conditional generative models," *Expert Syst. Appl.*, vol. 169, May 2021, Art. no. 114463.

[18] M. A. Arefeen, S. T. Nimi, and M. S. Rahman, "Neural network-based undersampling techniques," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 2, pp. 1111–1120, Feb. 2022.

[19] J.-H. Seo and Y.-H. Kim, "Machine-learning approach to optimize SMOTE ratio in class imbalance dataset for intrusion detection," *Comput. Intell. Neurosci.*, vol. 2018, Nov. 2018, Art. no. 9704672.
[20] H. Zhang, L. Huang, C. Q. Wu, and Z. Li, "An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset," *Comput. Netw.*, vol. 177, Aug. 2020, Art. no. 107315.
[21] J. Liu, Y. Gao, and F. Hu, "A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM," *Comput. Secur.*, vol. 106, Jul. 2021, Art. no. 102289.
[22] Y. J. Liu, J. Wang, and Y. Zhao, "Intrusion detection of vehicle based on generative adversarial networks," *J. Phys., Conf. Ser.*, vol. 1757, no. 1, Jan. 2021, Art. no. 012052.
[23] K. T. Chui, R. W. Liu, M. Zhao, and P. O. De Pablos, "Predicting students' performance with school and family tutoring using generative adversarial network-based deep support vector machine," *IEEE Access*, vol. 8, pp. 86745–86752, 2020.
[24] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection models," in *Proc. IEEE Symp. Secur. Privacy*, May 1999, pp. 120–132.
[25] M. Usama, J. Qadir, A. Raza, H. Arif, K.-L.-A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, "Unsupervised machine learning for networking: Techniques, applications and research challenges," *IEEE Access*, vol. 7, pp. 65579–65615, 2019.
[26] B. M. Shahshahani and D. A. Landgrebe, "The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes phenomenon," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 5, pp. 1087–1095, Sep. 1994.
[27] M. Verkerken, L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, "Towards model generalization for intrusion detection: Unsupervised machine learning techniques," *J. Netw. Syst. Manag.*, vol. 30, no. 1, pp. 1–25, Jan. 2022.
[28] Y. Yuan, L. Huo, Y. Yuan, and Z. Wang, "Semi-supervised tri-AdaBoost algorithm for network intrusion detection," *Int. J. Distrib. Sensor Netw.*, vol. 15, no. 6, Jun. 2019, Art. no. 155014771984605.
[29] W. Li, W. Meng, and M. H. Au, "Enhancing collaborative intrusion detection via disagreement-based semi-supervised learning in IoT environments," *J. Netw. Comput. Appl.*, vol. 161, Jul. 2020, Art. no. 102631.
[30] C.-C. Sun, D. J. S. Cardenas, A. Hahn, and C.-C. Liu, "Intrusion detection for cybersecurity of smart meters," *IEEE Trans. Smart Grid*, vol. 12, no. 1, pp. 612–622, Jan. 2021.
[31] Y. Hu, R. Liu, and Z. Ma, "Identification of cybersecurity elements based on convolutional attention LSTM networks," *J. Phys., Conf. Ser.*, vol. 1757, no. 1, Jan. 2021, Art. no. 012146.
[32] M. Tezgider, B. Yildiz, and G. Aydin, "Text classification using improved bidirectional transformer," *Concurrency Comput., Pract. Exp.*, vol. 34, no. 9, Apr. 2022, Art. no. e6486.
[33] J. Zhang, W.-C. Chang, H.-F. Yu, and I. Dhillon, "Fast multi-resolution transformer fine-tuning for extreme multi-label text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 7267–7280.
[34] M. Bugueño and M. Mendoza, "Learning to combine classifiers outputs with the transformer for text classification," *Intell. Data Anal.*, vol. 24, pp. 15–41, Dec. 2020.
[35] Y. Yuan and L. Lin, "Self-supervised pretraining of transformers for satellite image time series classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 474–487, 2021.
[36] X. Wang and Y. Tong, "Application of an emotional classification model in e-commerce text based on an improved transformer model," *PLoS ONE*, vol. 16, no. 3, Mar. 2021, Art. no. e0247984.
[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[38] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.
[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 448–456.
[40] A. Muaz, M. Jayabalan, and V. Thiruchelvam, "A comparison of data sampling techniques for credit card fraud detection," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 6, pp. 477–485, 2020.
[41] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Secur. J., Global Perspective*, vol. 25, nos. 1–3, pp. 18–31, Apr. 2016.
[42] W. Zong, Y.-W. Chow, and W. Susilo, "A two-stage classifier approach for network intrusion detection," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 11125, 2018, pp. 329–340.
[43] D. Jing and H.-B. Chen, "SVM based network intrusion detection for the UNSW-NB15 dataset," in *Proc. IEEE 13th Int. Conf. ASIC (ASICON)*, Oct. 2019, pp. 1–4.
[44] Q. Zhou, M. Tan, and H. Xi, "ACGANs-CNN: A novel intrusion detection method," *J. Phys., Conf. Ser.*, vol. 1757, no. 1, 2021, Art. no. 012012.
[45] X. Tan, S. Su, Z. Huang, X. Guo, Z. Zuo, X. Sun, and L. Li, "Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm," *Sensors*, vol. 19, no. 1, p. 203, Jan. 2019.
[46] R. K. Vigneswaran, R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Evaluating shallow and deep neural networks for network intrusion detection systems in cyber security," in *Proc. 9th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2018, pp. 1–6.

YU YANG received the Ph.D. degree. He is currently an Associate Professor with the School of Information Engineering, Engineering University of PAP. His research interests include network security and machine learning.

CHENGPENG YAO is currently pursuing the Graduate degree with the School of Information Engineering, Engineering University of PAP. His research interests include network security situation awareness and deep learning.

JINWEI YANG is currently pursuing the Graduate degree with the School of Information Engineering, Engineering University of PAP. Her research interest includes intrusion detection.

KUN YIN is currently pursuing the Graduate degree with the School of Information Engineering, Engineering University of PAP. His research interest includes anomaly detection.