IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 27, 2020, accepted December 27, 2020, date of publication December 30, 2020,
date of current version January 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3048198

Intrusion Detection of Imbalanced Network Traffic Based on Machine Learning and Deep Learning

LAN LIU[1], PENGCHENG WANG[1], JUN LIN[2], AND LANGZHOU LIU[1]
[1]School of Electronic and Information Engineering, Guangdong Polytechnic Normal University, Guangzhou 510655, China
[2]China Electronic Product Reliability and Environmental Testing Research Institute, Guangzhou 510610, China

Corresponding author: Jun Lin (linjun@ceprei.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61972104, in part by the Special Project for Research and Development in Key Areas of Guangdong Province under Grant 2019B010121001, and in part by the Special Fund for Science and Technology Innovation Strategy of Guangdong Province under Grant 2020A0332.

ABSTRACT In imbalanced network traffic, malicious cyber-attacks can often hide in large amounts of normal data. It exhibits a high degree of stealth and obfuscation in cyberspace, making it difficult for Network Intrusion Detection System(NIDS) to ensure the accuracy and timeliness of detection. This paper researches machine learning and deep learning for intrusion detection in imbalanced network traffic. It proposes a novel Difficult Set Sampling Technique(DSSTE) algorithm to tackle the class imbalance problem. First, use the Edited Nearest Neighbor(ENN) algorithm to divide the imbalanced training set into the difficult set and the easy set. Next, use the KMeans algorithm to compress the majority samples in the difficult set to reduce the majority. Zoom in and out the minority samples' continuous attributes in the difficult set synthesize new samples to increase the minority number. Finally, the easy set, the compressed set of majority in the difficult, and the minority in the difficult set are combined with its augmentation samples to make up a new training set. The algorithm reduces the imbalance of the original training set and provides targeted data augment for the minority class that needs to learn. It enables the classifier to learn the differences in the training stage better and improve classification performance. To verify the proposed method, we conduct experiments on the classic intrusion dataset NSL-KDD and the newer and comprehensive intrusion dataset CSE-CIC-IDS2018. We use classical classification models: random forest(RF), Support Vector Machine(SVM), XGBoost, Long and Short-term Memory(LSTM), AlexNet, Mini-VGGNet. We compare the other 24 methods; the experimental results demonstrate that our proposed DSSTE algorithm outperforms the other methods.

INDEX TERMS IDS, imbalanced network traffic, machine learning, deep learning, CSE-CIC-IDS2018.

I. INTRODUCTION

With the rapid development and wide application of 5G, IoT, Cloud Computing, and other technologies, network scale, and real-time traffic become more complex and massive, cyber-attacks have also become complex and diverse, bringing significant challenges to cyberspace security. As the second line of defense behind the firewall, the Network Intrusion Detection System(NIDS) needs to accurately identify malicious network attacks, provide real-time monitoring and dynamic protection measures, and formulate strategies.

James Anderson first proposed the concept of intrusion detection in 1980, and then some scholars applied machine learning methods in intrusion detection [1]. However, due to the limitation of computer storage and computing power at that time, machine learning failed to attract attention. With the rapid development of computers and the emergence and promotion of Artificial Intelligence(AI) and other technologies, many scholars have applied machine learning methods to network security. They have achieved certain results [2]–[4].

In real cyberspace, normal activities occupy the dominant position, so most traffic data are normal traffic; only a few are malicious cyber-attacks, resulting in a high imbalance of categories. In the highly imbalanced and redundant network

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

traffic data, intrusion detection is facing tremendous pressure. Cyber-attacks can hide in a large amount of normal traffic. Therefore, the machine learning algorithm cannot fully learn the distribution of a few categories, and it is easy to misclassify [5].

Since Lecun *et al.* [6] proposed the theory of Deep Learning as an essential subfield of machine learning, deep learning has shown excellent performance in Computer Vision (CV) [7], Natural Language Processing (NLP) [8]. Intrusion detection technology based on deep learning has been widely studied in academia and industry. The method of deep learning is to mine the potential features of high-dimensional data through training models and convert network traffic anomaly detection problems into classification problems [9]. By training a large number of data samples, adaptive learning of the difference between normal behavior and abnormal behavior effectively enhances the real-time performance of intrusion processing. However, in the multi-classification of network traffic, the imbalance of classification still affects.

Faced with imbalanced network traffic data, we propose a novel Difficult Set Sampling Technique(DSSTE) algorithm to tackle the class imbalance problem in network traffic. This method effectively reduces the imbalance and makes the classification model learning difficult samples more effective. We use classic machine learning and deep learning algorithms to verify on two benchmark datasets. The specific contributions are as follows.

(1) We use the classic NSL-KDD and the up-to-date CSE-CIC-IDS2018 as benchmark datasets and conduct detailed analysis and data cleaning.

(2) This work proposes a novel DSSTE algorithm, reducing the majority samples and augmenting the minority samples in the difficult set, tackling the class imbalance problem in intrusion detection so that the classifier learns the differences better in training.

(3) The classification model uses Random Forest(RF), Support Vector Machine(SVM), XGBoost, Long and Short Time Memory(LSTM), AlexNet, Mini-VGGNet. Comparing with other methods, we divide the experiment into 30 methods.

The rest of this article is organized as follows. The second part mainly introduces the related work of intrusion detection and class imbalance research. The third section introduces our proposed DSSTE algorithm, machine learning, and deep learning algorithm. The fourth section analyzes and experiments on the benchmark dataset. Finally, the paper concludes in the fifth section.

II. RELATED WORKS

A. INTRUSION DETECTION SYSTEM(IDS)

In the research of network intrusion detection based on machine learning, scholars mainly distinguish normal network traffic from abnormal network traffic by dimensionality reduction, clustering, and classification, to realize the identification of malicious attacks [10], [11].

Pervez proposed a new method for feature selection and classification merging of multi-class NSL-KDD Cup99 dataset using Support Vector Machine(SVM) and discussed the classification accuracy of classifiers under different dimension features [12]. Shiraz studied some new technologies to improve CANN intrusion detection methods' classification performance and evaluated their performance on the NSL-KDD Cup99 dataset [13]. He used the K Farthest Neighbor(KFN) and the K Nearest Neighbor(KNN) to classify the data and used the Second Nearest Neighbor(SNN) of the data when the nearest and farthest neighbors have the same class label. The result shows the CANN detection rate and reduces the failure the alert rate is improved or provides the same performance. Bhattacharya proposed a machine learning model based on hybrid Principal Component Analysis(PCA)-Firefly [14]. The dataset used was the open dataset collected from Kaggle. Firstly, the model performs one key coding for transforming the IDS dataset, then uses the hybrid PCA-Firefly algorithm to reduce the dimension, and the XGBoost algorithm classifies the reduced dataset.

In recent years, with the powerful ability of automatic feature extraction, deep learning has made remarkable achievements in the fields of Computer Vision(CV), Autonomous driving(AD), Natural Language Processing(NLP). Many scholars apply deep learning to intrusion detection for traffic classification, which has become a hot spot of current research. The method of deep learning is to mine the potential characteristics of high-dimensional data through a training model and transform network traffic anomaly detection into classification problem [15]. Through a large number of sample data training, adaptive learning between normal network traffic and abnormal network traffic effectively enhances real-time intrusion processing.

Torres *et al.* [16] first converted network traffic characteristics into a series of characters and then used Recurrent Neural Network(RNN) to learn their temporal characteristics, which were further used to detect malicious network traffic. Wang *et al.* [17] proposed a malicious software traffic classification algorithm based on Convolutional Neural Network(CNN). By mapping the traffic characteristics to pixels, the network traffic image is generated, and the image is used as the input of the CNN to realize traffic classification. Staudemeyer and Shamsinejad [13] proposed an intrusion detection algorithm based on Long Short-Term Memory(LSTM), which detects DoS attacks and probe attacks with unique time series in the KDD Cup99 dataset. Kwon *et al.* [18] has carried out relevant research on the deep learning model, focusing on data simplification, dimension reduction, classification, and other technologies, and proposes a Fully Convolutional Network(FCN) model. By comparing with the traditional machine learning technology, it is proved that the FCN model is useful for network traffic analysis. Tama *et al.* [19] proposed an anomaly-based IDS based on a two-stage meta-classifier, which uses a hybrid feature selection method to obtain accurate feature representations.

They conducted on the proposed method on the NSL-KDD and UNSW-NB15 intrusion datasets and improved detection rates.

B. CLASS BALANCING METHODS

In the field of machine learning, the problem of category imbalance has always been a challenge. Therefore, intrusion detection also faces enormous challenges in network traffic with extremely imbalanced categories. Therefore, many scholars have begun to study how to improve the intrusion recognition accuracy of imbalanced network traffic data.

Piyasak proposed a method to improve the accuracy of minority classification [20]. This method combines the Synthetic Minority Over-sampling Technique(SMOTE) and Complementary Neural Network(CMTNN) to solve imbalanced data classification. Experiments on the UCI dataset show that the proposed combination technique can improve class imbalance problems. Yan proposed an improved local adaptive composite minority sampling algorithm (LA-SMOTE) to deal with the network traffic imbalance problem and then based on the deep learning GRU neural network to detect the network traffic anomaly [21]. Abdulhammed *et al.* [22] deal with the imbalanced dataset CIDDS-001 using data Upsampling and Downsampling methods, and by Deep Neural Networks, Random Forest, Voting, Variational Autoencoder, and Stacking Machine Learning classifiers to evaluate datasets. In their proposed method, the accuracy can reach 99.99%.

Recently, Chuang and Wu [23] trained the depth automatic encoder to establish a data generation model to generate reasonable data needed to form a balanced dataset. His experiments show that the generation of balanced datasets helps to deal with the problem of over fitting caused by imbalanced data, and it can prevent the training model from misjudging new data types, including those not in the training dataset. Bedi *et al.* [24] proposed a new type of IDS based on Siamese Neural Network(Siamese-NN), the proposed Siam-IDS can detect R2L and U2R attacks without using traditional class balancing techniques, such as over-sampling and random under-sampling. The performance of Siam-IDS was compared with Deep Neural Network(DNN) and CNN, Siam-IDS can achieve a higher recall value for R2L and U2R attack categories compared with similar products.

Most scholars use interpolation, oversampling, encoder synthesis data, and other data augmentation methods, balance the training set, and achieve better experimental performance results. Although their method synthetic close to real data and effectively expand the minority class, the test data distribution may exceed the range. The classifier cannot accurately predict this distribution. We propose the DSSTE algorithm to mine the difficult samples in the imbalanced training set, compress the majority class among them, and zoom in or out the minority class's continuous attributes. This method reduces the imbalance and produces data that conforms to the true distribution.

III. METHOD

Faced with imbalanced network traffic, we propose the Difficult Set Sampling Technique(DSSTE) algorithm to compress the majority samples and augment the number of minority samples in difficult samples, reducing imbalance in the training set that the intrusion detection system can achieve better classification accuracy. We use Random Forest, SVM, XGBoost, LSTM, Mini-VGGNet, and AlexNet as classifiers for classification models.

We proposed the intrusion detection model shown in Figure 1. Data pre-processing first performed in our intrusion detection structure, including duplicate, outlier, and missing value processing. Then, partitioning the test set and the training set, and the training set processed for data balancing using our proposed DSSTE algorithm. Before modeling, to increase the speed of the convergence, we use StandardScaler to standardize the data and digitize the sample labels. Finally, the processed training set is used to train the classification model, and then the model is evaluated by the test set.

A. DSSTE ALGORITHM

In imbalanced network traffic, different traffic data types have similar representations, especially minority attacks can hide among a large amount of normal traffic, making it difficult for the classifier to learn the differences between them during the training process. In the similar samples of the imbalanced training set, the majority class is redundant noise data. The number is much larger than the minority class, making the classifier unable to learn the distribution of the minority class, so we compress the majority class. The minority class discrete attributes remain constant, and there are differences in continuous attributes. Therefore, the minority class's continuous attributes are zoomed to produce data that conforms to the true distribution. Therefore, we propose the DSSTE algorithm to reduce the imbalance.

First, the imbalanced training set to divide into near-neighbor set and far-neighbor set by Edited Nearest Neighbor(ENN) algorithm. The samples in the near-neighbor set are highly similar, making it very difficult for the classifier to learn the differences between the categories, so we refer to the samples in the near-neighbor set as difficult samples and the far-neighbor set as easy samples. Next, we zoom in and out the minority samples in difficult set. Finally, the easy set and minority in difficult set are combined with its augmentation samples to make up a new training set. We use the K neighbors in the ENN algorithm as the scaling factor of the entire algorithm. When scaling factor K increases, the number of difficult samples increases, and the compression rate of the majority of samples and the synthesis rate of the minority of class also increase. The DSSTE algorithm is written as Algorithm 1.

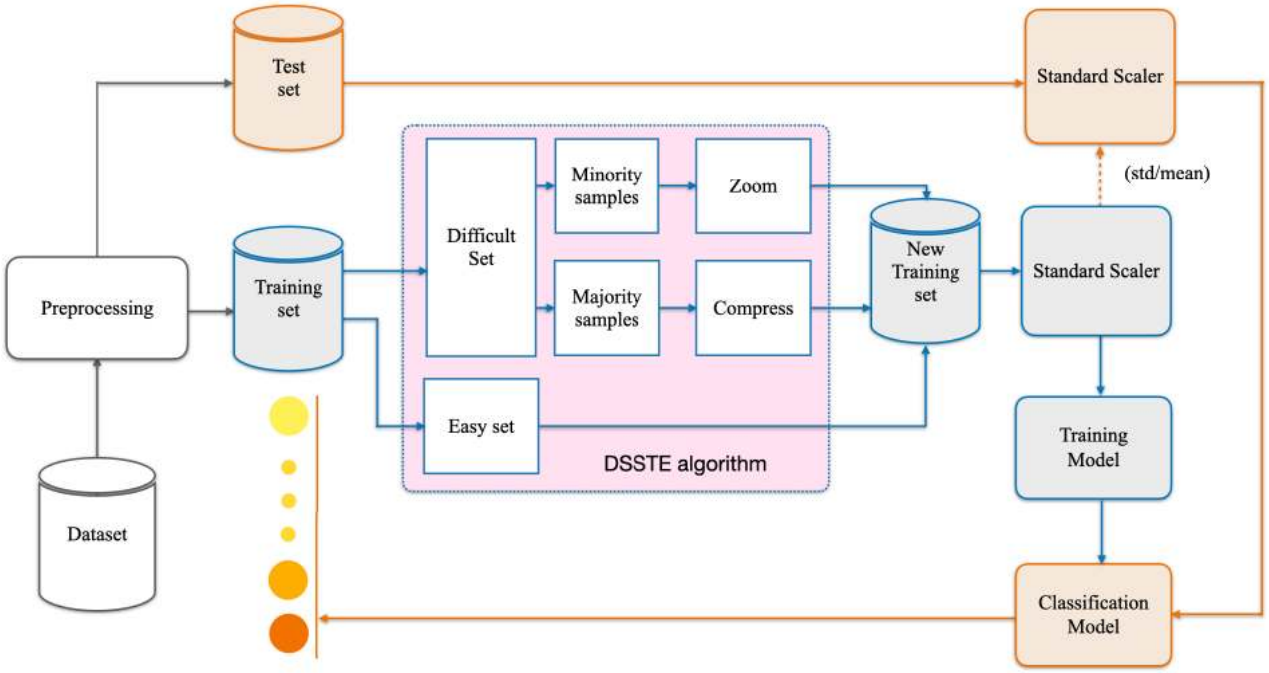


FIGURE 1. The overall framework of network intrusion detection model.

B. MACHINE LEARNING AND DEEP LEARNING ALGORITHMS

In the classifier's design, we use Random Forest, SVM, XGBoost, LSTM, AlexNet, and Mini-VGGNet to train and test, which are detailed in the following part.

1) RANDOM FOREST

Leo Breiman proposed random Forest in 2001 [25]. Random Forest is an excellent supervised learning algorithm that can train a model to predict which classification results in a certain sample type belong to based on a given dataset's characteristic attributes and classification results. Random Forest is based on a decision tree and adopts the Bagging(Bootstrap aggregating) method to create different training sample sets. The random subspace division strategy selects the best attribute from some randomly selected attributes to split internal nodes. The various decision trees formed are used as weak classifiers, and multiple weak classifiers form a robust classifier, and the voting mechanism is used to classify the input samples. After a random forest has established a large number of decision trees according to a certain random rule when a new set of samples is input, each decision tree in the forest makes a prediction on this set of samples separately, and integrates the prediction results of each tree, get a final result.

2) SUPPORT VECTOR MACHINE

Coretes and Vapink first proposed support Vector Machine (SVM) in 1995 [26]. It shows many unique advantages in a small sample, nonlinear, and high-dimensional pattern recognition and can be extended to other functions such as function fitting Machine learning problems [27]. Before the rise of deep learning, SVM was considered the most successful and best-performing machine learning method in recent decades. The SVM method is based on the Vapnik Chervonenkis(VC) dimension theory of statistical learning theory and the principle of structural risk minimization. Its basic idea is to find a separation hyperplane between different categories, so that different category can be better separated. The SVM method believes that when deciding to separate the hyperplane, only the sample point closest to the hyperplane, as long as the support vector is found, the hyperplane can be determined.

3) XGBoost

XGBoost is a parallel regression tree model that combines the idea of Boosting, which is improved based on gradient descent decision tree by Chen and Guestrin [28]. Compared with the GBDT(Gradient Boosting Decision Tree) model, XGBoost overcomes the limited calculation speed and accuracy. XGBoost adds regularization to the original GBDT loss function to prevent the model from overfitting. The traditional GBDT performs a first-order Taylor expansion on the calculated loss function and takes the negative gradient value as the residual value of the current model. In contrast, XGBoost performs a second-order Taylor expansion to ensure the accuracy of the model. Moreover, XGBoost blocks and sorts each feature, making it possible to parallelize the calculation when

Algorithm 1 DSSTE Algorithm

Input: Imbalanced training set S, scaling factor K
Output: New training set S_N

Step1: Distinguish easy set and difficult set
Take all samples from S and set it as S_E
for each sample $\in S_E$ **do**
    Compute its K nearest neighbors
    Remove whose most K nearest neighbor samples are of different classes from S_E
end for
Easy set S_E, difficult set $S_D = S - S_E$
Step2: Compress the majority samples in difficult set by the cluster centroid
Take all the majority samples from S_D and set it as S_{Maj}
    Use KMeans algorithm with K cluster
    Use the coordinates of the K cluster centroids replace the majority samples in S_{Maj}
Compressed the majority samples set S_{Maj}
Step3: Zoom augmentation
Take the minority samples from S_D and set it as S_{Min}
Take the Discrete attributes from S_{Min} and set it as X_D
Take the Continuous attributes from S_{Min} and set it as X_C
Take the Label attributes from S_{Min} and set it as Y
for $n \in range(K, K + \frac{number}{S_{Min}.shape[0]})$ **do** // zoom range is $[1 - \frac{1}{K}, 1 + \frac{1}{K}]$, $S_{Min}.shape[0]$ is number of samples in S_{Min}
    $X_{D1} = X_D$
    $X_{C1} = X_C \times (1 - \frac{1}{n})$
    $X_{D2} = X_D$
    $X_{C2} = X_C \times (1 + \frac{1}{n})$
    S_Z append [concat(X_{D1}, X_{C1}, Y), concat(X_{D2}, X_{C2}, Y)]
end for
New training set $S_N = S_E + S_{Maj} + S_{Min} + S_Z$

looking for the best split point, which significantly accelerates the calculation speed [29].

4) LONG SHORT-TERM MEMORY

The Long Short-Term Memory(LSTM) network is a Recurrent Neural Network(RNN) structure proposed by Hochreiter and Jurgen in 1997 [30]. Like most RNN, the LSTM network is universal because as long as there is a suitable weight matrix, the LSTM network can calculate any network element that can be calculated by any conventional computer. Different from the traditional RNN, the LSTM network is very suitable for learning from experience. When there is a time lag of unknown size and boundary between important events, the time series can be classified, processed, and predicted. LSTM is not sensitive to gap length and has advantages over other RNN and hidden Markov models and other sequence learning methods in many applications [31]. The problem of gradient disappearance and gradient explosion is solved by introducing the gate structure and storage unit.

5) AlexNet

AlexNet is one of the classic basic networks of deep learning. It was proposed by Hinton and his student Alex Krizhevsky in 2012 [32]. Its main structure is an 8-layer deep neural network, including 5-layer convolutional layers and 3-layer fully connected layers, which are not counted in the Activation layer and pooling layer. The ReLU function is used as the activation function in the AlexNet convolutional layer, instead of the Sigmoid function widely used in previous networks. The introduction of the ReLU function solves the problem of gradient dispersion when the neural network is deep. The AlexNet neural network uses the Maxpooling method in the convolutional layer to downsample the feature map output by the convolutional layer, instead of the average pooling commonly used before. Therefore, the AlexNet neural network has better performance than the previous neural network.

6) MINI-VGGNet

In 2014, researchers from the Visual Geometry Group of Oxford University and Google DeepMind jointly developed a new deep convolutional neural network: VGGNet and won second place in the ILSVRC2014 classification project. Their paper ''Very Deep Learning Convolutional Neural Networks for Large-Scale Image Recognition'' mainly focuses on the influence of convolutional neural networks' depth on the recognition accuracy of large-scale image sets [33]. The main contribution is to use a small convolution kernel ($3 \times 33 \times 3$) to construct various depths of convolutional neural network structures. Moreover, it evaluated these network structures and finally proved that the 16-19 layer network

depth could achieve better recognition accuracy. VGG-16 and VGG-19 are commonly used to extract image features. VGG can be regarded as a deepened version of AlexNet. The entire network is superimposed by a convolutional layer and a fully connected layer. Unlike AlexNet, VGGNet uses a small-sized convolution kernel(3×3).

AlexNet is one of the classic basic networks of deep learning. It was proposed by Hinton and his student Alex Krizhevsky in 2012 [32]. Its main structure is an 8-layer deep neural network, including 5-layer convolutional layers and 3-layer fully connected layers, which are not counted in the Activation layer and pooling layer. The ReLU function is used as the activation function in the AlexNet convolutional layer, instead of the Sigmoid function widely used in previous networks. The introduction of the ReLU function solves the problem of gradient dispersion when the neural network is deep. The AlexNet neural network uses the Maxpooling method in the convolutional layer to downsample the feature map output by the convolutional layer, instead of the average pooling commonly used before. Therefore, the AlexNet neural network has better performance than the previous neural network.

In this experiment, because we have fewer traffic characteristics, we used the Mini-VGGNet(miniVGG) network mentioned by Ismail for classification experiments [34]. In general, Mini-VGGNet contains two sets of CONV = > RELU = > CONV = > RELU = > POOL, followed by FC = > RELU = > FC = > SOFTMAX layer. The first two CONV layers will learn 32 3×3 cores. The last two CONV layers will learn 64 cores that are also 3×3. The POOL layer will perform a Maxpooling operation with 2×2 cores and a Stride of 2×2.

IV. EXPERIMENT

In this experiment, we use the classical classification algorithms of machine learning and deep learning, including Random Forest(RF), Support Vector Machine(SVM), XGBoost, Long Short-Term Memory(LSTM), AlexNet, and Mini-VGGNet. And compared with other oversampling methods, including Random Under-sampling(RUS) [35], Random Over-sampling(ROS) [36] and Synthetic Minority Over-sampling TEchnique(SMOTE) [37], it is divided into 30 methods to combine.

A. BENCHMARK DATASET

We choose NSL-KDD and CSE-CIC-IDS2018 as the benchmark dataset for experiments.

NSL-KDD is the most classic dataset in the field of intrusion detection [38]. It is an improvement based on the KDD99 dataset and is reasonably divided into different difficulty levels in the test set. Although it still has some problems and is not a perfect representation of the existing real network, it can still be used as an effective benchmark dataset to help researchers compare different intrusion detection methods. Each sample in NSL-KDD includes 41 features listed in Table 1.

TABLE 1. Description of the NSL-KDD features.

Attributes	Description
1–9	Basic features of network connections
10–22	Content-related traffic features
23–31	Time-related traffic features
32–41	Host-based traffic features

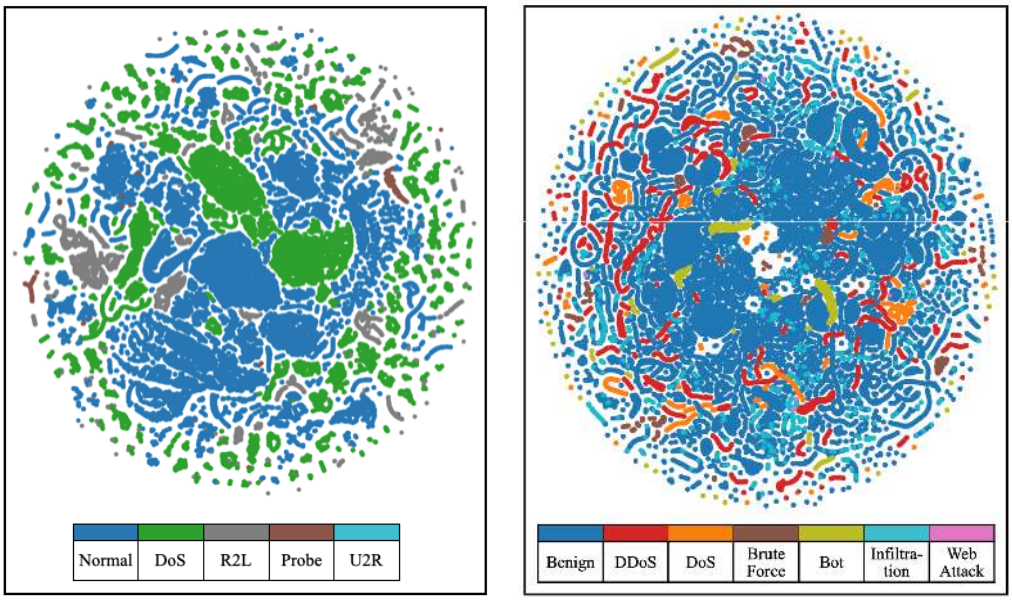
CSE-CIC-IDS2018 is an intrusion detection dataset created by the Canadian Institute of Cyber Security (CIC) on AWS (Amazon Web Services) in 2018. It is also the latest and comprehensive intrusion dataset currently publicly available [39]. CSE-CIC-IDS2018 is a dataset collected for launching real attacks. It is an improvement based on the CSE-CIC-IDS2017 dataset. It contains the necessary standards for the attack dataset and covers various known attack types. The dataset contains six different attack scenarios: Brute Force, Botnet, DoS, DDoS, Web Attacks, and Infiltration. Each sample in CSE-CIC-IDS2018 includes 83 features listed in Table 2.

TABLE 2. Description of the CSE-CIC-IDS2018 features.

Attributes	Description
1–4	Basic features of network connections
5–16	Features of network packets
17–22	Features of network flows
23–45	Statistic of network flows
46–63	Content-related traffic features
64–67	Features of network subflows
68–79	General purpose traffic features
80-83	Basic features of network connections

We use t-SNE to visualize the NSL-KDD and CSE-CIC-IDS2018 by dimensionality reduction [40]. As shown in Figure 2, we can see that the normal samples are much larger than the attack samples, making some attacks easy to hide and confusion among them makes traditional intrusion detection technology increasingly challenging to detect.

In NSL-KDD dataset, we use KDDTrain+ and KDDTest+ as the training set and test set, and it is divided into five categories: Normal, DOS, R2L, Probe, U2R. Since CSE-CIC-IDS2018 is a huge and redundant dataset, there is no official division between training and test sets. In order to ensure the imbalance of traffic data and verify the effectiveness of our proposed method. We randomly selected 40,000 Benign traffic. For the attack traffic data with more than 20,000, we randomly selected 20,000 from them. For the attack traffic data with less than 20,000, we all pick it out. Since DoS



(a) NSL-KDD (b) CSE-CIC-IDS2018

FIGURE 2. Use t-SNE to visualize NSL-KDD(a) and CSE-CIC-IDS2018(b).

attacks-SlowHTTPTest only has three valid data after removing features such as Timestamp, we will not add them to our experiment. Furthermore, divide 80% of the selected data into the training set and 20% into the test set.

NSL-KDD, CSE-CIC-IDS2018 are highly imbalanced datasets, with normal traffic accounting for the vast majority, which conforms to traffic data distribution in the whole network world. We have performed traffic label statistics on tens of millions of samples, and it can be seen that the abnormal traffic is much smaller than the normal traffic. The specific results are shown in Table 3.

B. DATA PREPROCESSING

When the dataset is extracted, part of the data contains some noisy data, duplicate values, missing values, infinity values, etc. due to extraction errors or input errors. Therefore, we first perform data preprocessing. The main work is as follows.

(1) Duplicate values: delete the sample's duplicate value, only keep one valid data.

(2) Outliers: in the sample data, the sample size of missing values(Not a Number, NaN) and Infinite values(Inf) is small, so we delete this.

(3) Features delete and transform: In CSE-CIC-IDS2018, we delete features such as ''Timestamp'', ''Destination Address'', ''Source Address'', ''Source Port'', etc. If features ''Init Bwd Win Byts'' and features ''Init Fwd Win Byts'' have a value of -1, we add two check dimensions. The mark of -1 is 1. Otherwise, it is 0. In NSL-KDD, we use the OneHot encoder to complete this conversion. For example, ''TCP'', ''UDP'' and ''ICMP'' are functions of three protocol types. After OneHot encoding, they become binary vectors (1, 0, 0), (0, 1, 0), (0, 0, 1). The protocol type function can be divided into three categories, including 11 categories for flag function and 70 categories for service function. Therefore, the 41 dimensions initial feature vector becomes 122 dimensions.

(4) Numerical standardization: In order to eliminate the dimensional influence between indicators and accelerate the gradient descent and model convergence, the data is standardized, that is, the method of obtaining Z-Score, so that the average value of each feature becomes 0 and the standard deviation becomes 1, converted to a standard normal distribution, which is related to the overall sample distribution, and each sample point can have an impact on standardization. The standardization formula is as follows, u is the mean of each feature, s is the standard deviation of each feature, and x_i' is the element corresponding to each column's features.

$$u = \sum_{i=1}^{N} x_i \tag{1}$$

$$s = \sum_{i=1}^{N} (x_i - u)^2 \tag{2}$$

$$x_i' = \frac{x_i - u}{s} \tag{3}$$

C. EXPERIMENTAL PARAMETERS

The proposed method uses the Sklearn(machine learning framework) and Tensorflow(deep learning framework) and completes related experiments on the Google Colaboratory platform. The machine learning algorithm uses CPU calculations, and the deep learning algorithm uses TPU for acceleration. The specific parameters are shown in Table 4.

To prevent overfitting, we standardized the data. In machine learning, the integrated learning model uses shallow trees to prevent overfitting. In the deep learning

TABLE 3. Distribution of the benchmark datasets.

Dataset	Type	Total	Imbalance ratio	Training set	Testing set
NSL-KDD	Normal	77054	-	67343	9711
	DoS	53385	1.44	45927	7458
	R2L	3882	19.85	995	2887
	Probe	14077	5.47	11656	2421
	U2R	119	647.51	52	67
CSE-CIC-IDS2018	Benign	40000	-	32000	8000
	Bot	20000	2.00	16000	4000
	DDOS attack-LOIC-UDP	1730	23.12	1384	346
	DDOS attack-HOIC	20000	2.00	16000	4000
	DDoS attacks-LOIC-HTTP	20000	2.00	16000	4000
	DoS attacks-GoldenEye	20000	2.00	16000	4000
	DoS attacks-Hulk	20000	2.00	16000	4000
	DoS attacks-Slowloris	9908	4.04	7926	1982
	SSH-Bruteforce	20000	2.00	16000	4000
	FTP-BruteForce	46	869.57	37	9
	Infilteration	20000	2.00	16000	4000
	Brute Force -Web	550	72.73	440	110
	Brute Force -XSS	227	176.21	181	46
	SQL Injection	82	487.80	66	16

TABLE 4. Development environment.

Project	Properties
OS	Ubuntu 18.04.3 LTS
CPU	Intel(R) Xeon(R) CPU @2.30GHz
TPU	-
Memory	12.7 GiB
Disk	64.0 GiB
Framework	Sklearn 0.22.2.post1 + TensorFlow 2.2.0

TABLE 5. Machine learning model related parameters.

Classfier	Parameters
RandomForestClassifier	n_estimators=200, criterion='gini', min_samples_plit=2, min_samples_leaf=1
svm.LinearSVC	penalty='l2', loss='squared_hinge', dual=True, tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept_scaling=1, max_iter=1000
XGBClassifier	objective='multi:softmax', booster='gbtree', verbosity=0, silent=0, learning_rate=0.1

model, we use TPU for acceleration, so we chose a larger Batch and increased the epochs accordingly. Overfitting was further prevented by observing the accuracy and loss changes during the training phase, using appropriate learning rates, and adding Dropout to the neural network layer.

For the machine learning algorithm, we used the RandomForestClassifier, svm.LinearSVC, XGBClassifier experiments provided in Sklearn. The specific parameters are shown in Table 5.

For deep learning algorithms, LSTM and GRU use the original one-dimensional sequence (144 × 1) of the learning data set. For AlexNet and MiniVGGNet, We added 41 0 dimensions at the end of the feature and then reshaped the single-channel two-dimensional matrix processing (12 × 12 × 1). We uniformly adopt Adam's optimizer (lr = 0.001) and perform 100 epochs in the model training stage, and the batch size is 1024. The parameters are shown in Table 6.

TABLE 6. Deep learning model related parameters.

LSTM	AlexNet	miniVGGNet
Input(144,1)	Input(12,12,1)	Input(12,12,1)
LSTM-64	Conv11-96	Conv3-32
Dropout-0.2	Maxpool-3	Conv3-32
LSTM-64	Conv5-256	Maxpool-2
Dropout-0.2	Conv3-384	Dropout-0.25
LSTM-64	Conv3-384	Conv3-64
Dropout-0.2	Conv3-256	Conv3-64
FC-512	Maxpool-2	Maxpool-2
Dropout-0.5	Flatten	Dropout-0.25
	FC-2048	Flatten
	Dropout-0.5	FC-512
	FC-2048	Dropout-0.25
	Dropout-0.5	
SooftMax-5 / SooftMax-14		

D. EVALUATION METRICS

We use the Accuracy, Prediction, Recall, and F1-Score to evaluate the experimental model's performance. These evaluation criteria reflect the performance of the intrusion detection system's flow recognition accuracy rate, and false alarm rate. The combination of the model prediction results and the true label is divided into four types: False Negative(FN), a positive sample, which is mistakenly judged as a negative sample. False Positive(FP), negative samples are misjudged as positive samples. True Negative(TN), actually negative samples, are correctly judged as negative samples. True Positive(TP), actually positive samples, are judged as the positive sample. These metrics are calculated according to Equations 4-7.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

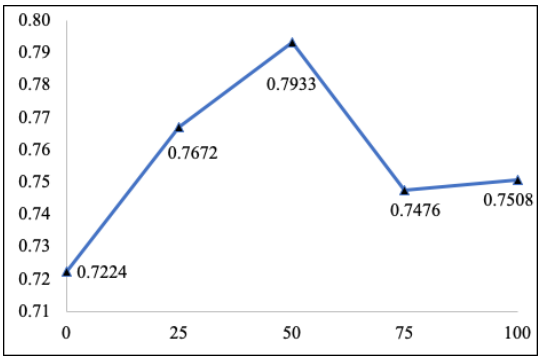
$$Recall = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$F1\_Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7}$$

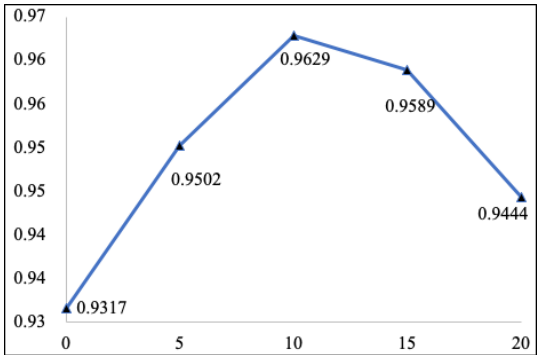
E. EXPERIMENTAL RESULTS

In our experiments, we first explored the classifier's performance on the training set treated with different deflation factors. In the proposed DSSTE algorithm, there is a parameter scaling factor of K. When K increases within a certain range, the number of difficult samples will also increase, but when K exceeds the range, the number of difficult samples will constantly be constant. However, the majority compression and the minority augmentation in the difficult samples will increase with K change. Therefore, to ensure that the data sampling is useful and does not generate excessive noise and that the DSSTE algorithm achieves the best sampling results, we experimented with different scaling factors.

We processed the training set in NSL-KDD and CSE-CIC-IDS2018 using different scaling factors K. We performed experiments on the proposed six classifiers, and performance was evaluated using the average F1-Score of each classifier, as shown in Figure 3.



(a) NSL-KDD KDDTest+



(b) CSE-CIC-IDS2018

FIGURE 3. F1-Score of DSSTE algorithm with different scaling factor K.

In NSL-KDD, the classifiers achieve excellent average performance at $K = 50$. In CSE-CIC-IDS2018, the classifiers achieve excellent average performance at $K = 10$. Therefore, based on the average F1-Score, in NSL-KDD, we used the scaling factor $k = 50$, where the difficult samples in Normal, DoS, and Probe were compressed, and the difficult samples in R2L and U2R were augmented with data. In CSE-CIC-IDS2018, we used the scaling factor $K = 10$ and performed a similar treatment to NSL-KDD for the difficult samples. The new training set after the treatment is shown in Table 7.

Table 8 summarizes the comparison between DSSTE and other sampling methods, and our proposed DSSTE algorithm outperforms other methods in NSL-KDD and CSE-CIC-IDS2018.

In the experimental results for the NSL-KDD dataset, LSTM achieved the highest accuracy of 78.24% and the highest F1-Score of 75.03% in the original training set. After sampling the RUS algorithm's training, XGBoost achieved the highest accuracy rate of 78.79%, and miniVGGNet achieved the highest recall rate of 75.57%. After sampling the ROS

TABLE 7. The new training set class distribution processed by the DSSTE algorithm.

Dataset	Type	Training set	New training set
NSL-KDD (K=50)	Normal	67343	61914
	DoS	45927	40425
	R2L	995	15683
	Probe	11656	7348
	U2R	52	2652
CSE-CIC-IDS2018 (K=10)	Benign	32000	25195
	Bot	16000	15952
	DDOS attack-LOIC-UDP	1384	2891
	DDOS attack-HOIC	16000	15984
	DDoS attacks-LOIC-HTTP	16000	14469
	DoS attacks-GoldenEye	16000	14320
	DoS attacks-Hulk	16000	15496
	DoS attacks-Slowloris	7926	10709
	SSH-Bruteforce	16000	15976
	FTP-BruteForce	37	367
	Infilteration	16000	11728
	Brute Force -Web	440	1947
	Brute Force -XSS	181	1391
	SQL Injection	66	759

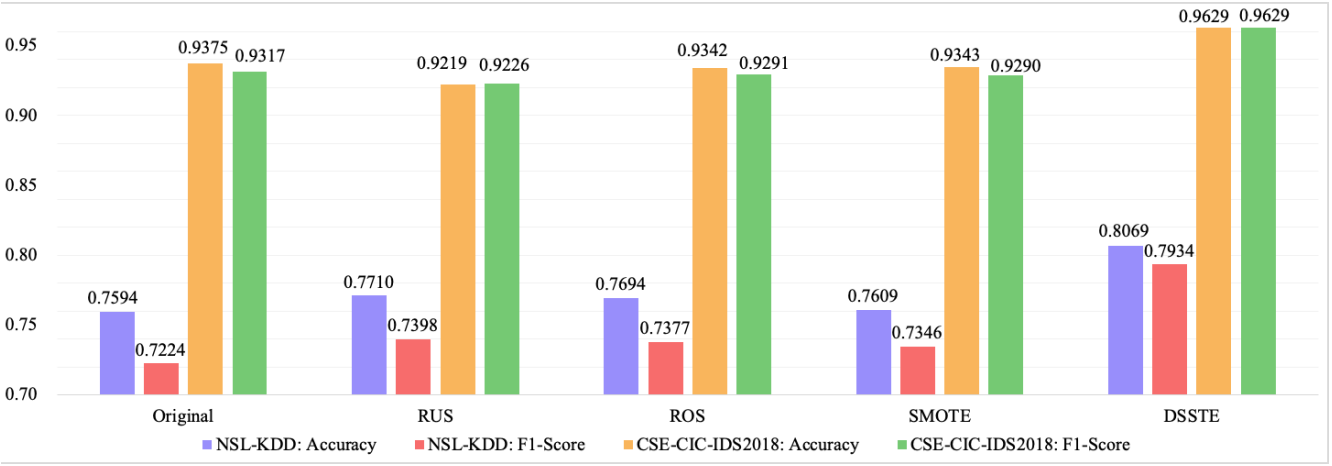


FIGURE 4. Comparison of the performance of different sampling methods(Accuracy and F1-Score are the average of each classifier).

algorithm's training, LSTM achieved the highest accuracy rate of 78.72% and the highest recall rate of 75.82%. After the SMOTE algorithm sampled the training set, AleNet achieved the highest accuracy rate of 78.75% and the highest recall rate of 77.27%. In the training set sampled by DSSTE proposed in this paper, AleNet achieved the highest accuracy rate of 82.84% and the highest recall rate of 81.66%.

In the experimental results of the CSE-CIC-IDS2018 dataset, random forest achieves the highest accuracy of 94.89 % and the highest F1-Score of 94.72 % in the unprocessed training set. After the RUS, ROS, and SMOTE algorithms sampled the training set. The random forest achieved the highest accuracy and F1-Score. However, the performance improvement was very small or even lower than that of the original data set. In the training set sampled by the DSSTE algorithm proposed in this paper, miniVGGNet achieves the highest accuracy of 96.99% and the highest recall of 97.04%. However, the accuracy and recall of random forest are also

TABLE 8. Comparison results between DSSTE and different methods(Acc, Pre, and F1-Score are the average of multiple classes, weighted by the number of samples in each class).

Model	NSL-KDD				CSE-CIC-IDS2018			
	Acc	Pre	Recall	F1	Acc	Pre	Recall	F1
RF	0.7434	**0.8137**	0.7434	0.7015	**0.9489**	**0.9481**	**0.9489**	**0.9481**
SVM	0.7366	0.7384	0.7366	0.6966	0.9225	0.9261	0.9225	0.9126
XGBoost	0.7715	0.8107	0.7715	0.7365	0.9398	0.9449	0.9398	0.9340
LSTM	**0.7824**	0.7838	0.7823	**0.7503**	0.9375	0.9444	0.9370	0.9313
AlexNet	0.7618	0.8050	0.7611	0.7194	0.9376	0.9440	0.9369	0.9313
miniVGGNet	0.7605	0.8066	0.7594	0.7303	0.9388	0.9450	0.9384	0.9326
RUS + RF	0.7655	0.8220	0.7655	0.7304	**0.9419**	**0.9454**	**0.9419**	**0.9428**
RUS + SVM	0.7362	0.7510	0.7362	0.7058	0.8902	0.9087	0.8902	0.8926
RUS + XGBoost	**0.7879**	0.8177	**0.7879**	0.7468	0.9212	0.9362	0.9212	0.9234
RUS + LSTM	0.7705	0.7970	0.7704	0.7462	0.9150	0.9294	0.9139	0.9171
RUS + AlexNet	0.7834	**0.8250**	0.7814	0.7537	0.9294	0.9308	0.9268	0.9280
RUS + miniVGGNet	0.7827	0.8134	0.7826	**0.7557**	0.9337	0.9330	0.9329	0.9319
ROS + RF	0.7515	0.8125	0.7515	0.7066	**0.9492**	**0.9484**	**0.9492**	**0.9483**
ROS + SVM	0.7493	0.8005	0.7493	0.7300	0.9165	0.9311	0.9165	0.9073
ROS + XGBoost	0.7809	0.8196	0.7809	0.7532	0.9385	0.9448	0.9385	0.9320
ROS + LSTM	**0.7872**	**0.8289**	**0.7866**	**0.7582**	0.9348	0.9409	0.9346	0.9293
ROS + AlexNet	0.7850	0.8057	0.7849	0.7451	0.9299	0.9387	0.9295	0.9262
ROS + miniVGGNet	0.7626	0.7893	0.7626	0.7332	0.9362	0.9406	0.9359	0.9316
SMOTE + RF	0.7409	0.8070	0.7409	0.6977	**0.9488**	**0.9481**	**0.9488**	**0.9480**
SMOTE + SVM	0.7467	0.7987	0.7467	0.7275	0.9155	0.9302	0.9155	0.9062
SMOTE + XGBoost	0.7744	0.8142	0.7744	0.7421	0.9381	0.9449	0.9381	0.9318
SMOTE + LSTM	0.7509	0.7976	0.7508	0.7239	0.9345	0.9431	0.9344	0.9278
SMOTE + AlexNet	**0.7875**	**0.8256**	**0.7851**	**0.7727**	0.9324	0.9423	0.9308	0.9287
SMOTE + miniVGGNet	0.7651	0.7698	0.7646	0.7435	0.9366	0.9423	0.9366	0.9317
DSSTE + RF	0.8050	**0.8468**	0.8050	0.7863	0.9692	0.9739	0.9692	0.9698
DSSTE + SVM	0.7759	0.8076	0.7759	0.7658	0.9488	0.9497	0.9488	0.9463
DSSTE + XGBoost	0.8013	0.8349	0.8013	0.7761	0.9602	0.9641	0.9602	0.9611
DSSTE + LSTM	0.8178	0.8271	0.8177	0.8098	0.9638	0.9711	0.9636	0.9650
DSSTE + AlexNet	**0.8284**	0.8394	**0.8278**	**0.8166**	0.9653	0.9709	0.9625	0.9649
DSSTE + miniVGGNet	0.8127	0.8268	0.8132	0.8057	**0.9699**	**0.9746**	**0.9697**	**0.9704**

very close to each other. Random forest exhibits the generalization capability of integrated learning when used in combination with each sampling algorithm, and it requires fewer hardware resources.

As shown in Figure 4, we counted the average accuracy and F1-Score of the classifier for each sampling method. In the NSL-KDD dataset, the sampling algorithms' performance using RUS, ROS, and SMOTE are all improved compared to the original algorithm. In terms of prediction accuracy and F1-Score, the improvement is very slight. The proposed DSSTE algorithm is significantly improved, in which the average accuracy is improved by 4.75%, and the average F1-Score is improved by 7.1%. In the CSE-CIC-IDS2018 dataset, performance gains are very slight or even degraded after using the RUS, ROS, and SMOTE sampling algorithms. After the training set with DSSTE algorithm sampling proposed in this paper, the average accuracy improves by 2.54%, and the average F1-Score improves by 3.13%.

The F1-Score is a harmonic average of the prediction and recall rates, which is a good indicator of a classification model's performance. So we adopt F1-Score and accuracy as the metrics to compare the different methods proposed by other authors in the face of imbalanced network traffic. As shown in Table 9, our proposed data sampling method DSSTE has a higher accuracy than other meth-

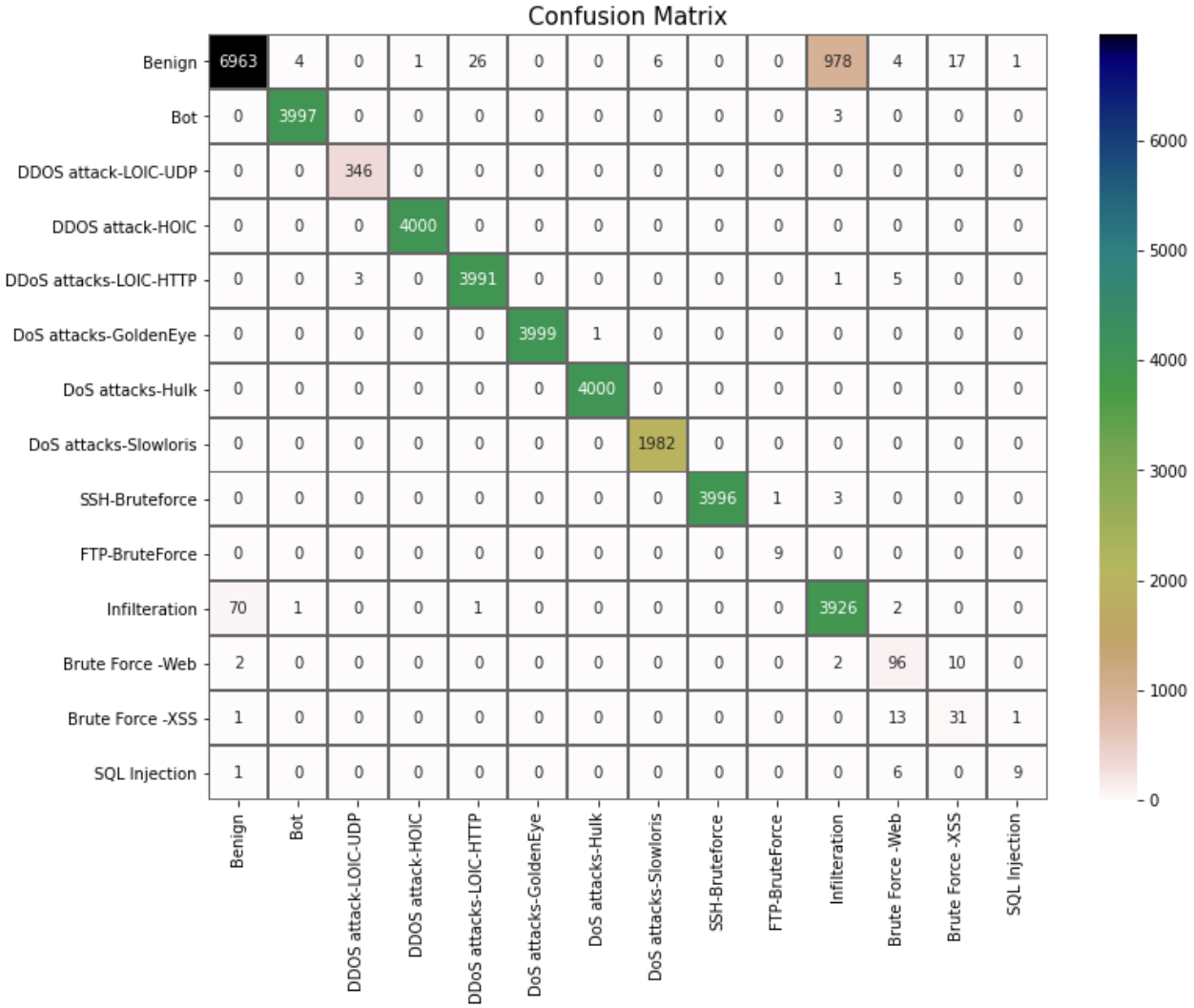


FIGURE 5. Confusion matrix of CIC-IDS-2018 by DSSTE+minVGGNet.

TABLE 9. Comparison results of DSSTE with the existing approaches on NSL-KDD KDDTest+.

Method	Year	Acc	F1-Score
DSSTE+AlexNet	2020	**0.8284**	0.8166
AESMOTE [41]	2020	0.8209	**0.8243**
I-SiamIDS [42]	2020	0.8000	0.6834
AE-RL [43]	2019	0.8016	0.794
ADASYN [44]	2019	0.7897	/
WGAN-GP [45]	2019	0.8080	/
ID-CAVE [46]	2017	0.8010	0.7908
SMOTE-EUS [15]	2016	0.7910	0.7576

ods on KDDTest+. The F1-Score is very close to that of AESMOTE, which exhibits the advantage of reinforcement learning for automatic pairwise sequence learning, but reinforcement learning training requires a lot of time to build the model. Therefore, our proposed method is more generalizable to imbalanced network traffic.

The CIC-IDS-2018 is a large and redundant dataset, and data are selected and processed differently by different scholars. Therefore, we do not compare it with other authors on the CIC-IDS-2018 dataset. In our experiments, we can see that the DSSTE method is significantly better than other sampling algorithms. As shown in Figure 5, DSSTE+AlexNet exhibits excellent performance on the CIC-IDS-2018 dataset. It achieves close to 100% detection rate in some attacks, and also improves the identification of Brute Force and Infilteration attacks.

To sum up, traditional sampling methods reduce the imbalance in the training set and synthesize close to the real data; it does not produce a distribution that matches the real data. RUS algorithm leads to loss of valid information; the ROS algorithm leads to data redundancy and overfitting. At the same time, SMOTE interpolation generates noise traffic and data overlap, increasing the number of difficult samples in the

training set. Our proposed DSSTE algorithm is very targeted to compress and augment difficult data from an imbalanced training set. It enables the classifier to grasp more data distribution, thus improving the classification performance.

V. CONCLUSION

As network intrusion continues to evolve, the pressure on network intrusion detection is also increasing. In particular, the problems caused by imbalanced network traffic make it difficult for intrusion detection systems to predict the distribution of malicious attacks, making cyberspace security face a considerable threat.

This paper proposed a novel Difficult Set Sampling Technique(DSSTE) algorithm, which enables the classification model to strengthen imbalanced network data learning. A targeted increase in the number of minority samples that need to be learned can reduce the imbalance of network traffic and strengthen the minority's learning under challenging samples to improve the classification accuracy. We used six classical classification methods in machine learning and deep learning and combined them with other sampling techniques. Experiments show that our method can accurately determine the samples that need to be expanded in the imbalanced network traffic and improve the attack recognition more effectively.

In the experiment, we found that deep learning performance is better than machine learning after sampling the imbalanced training set samples through the DSSTE algorithm. Although the neural networks strengthen data expression, the current public datasets have already extracted the data features in advance, which is more limited for deep learning to learn the preprocessed features and cannot take advantage of its automatic feature extraction. Therefore, in the next step, we plan to directly use the deep learning model for feature extraction and model training on the original network traffic data, performance the advantages of deep learning in feature extraction, reduce the impact of imbalanced data and achieve more accurate classification.

REFERENCES

[1] D. E. Denning, "An intrusion-detection model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[2] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs decision trees in intrusion detection systems," in *Proc. ACM Symp. Appl. Comput. (SAC)*, 2004, pp. 420–424.

[3] M. Panda and M. R. Patra, "Network intrusion detection using Naive Bayes," *Int. J. Comput. Sci. Netw. Secur.*, vol. 7, no. 12, pp. 258–263, 2007.

[4] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (IDS)," *J. Intell. Learn. Syst. Appl.*, vol. 6, no. 1, pp. 45–52, 2014.

[5] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. Int. Conf. Artif. Intell.*, vol. 56, 2000, pp. 111–117.

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

[7] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, Apr. 2016.

[8] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [review article]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.

[9] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018.

[10] D. A. Cieslak, N. V. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets," in *Proc. IEEE Int. Conf. Granular Comput.*, May 2006, pp. 732–737.

[11] M. Zamani and M. Movahedi, "Machine learning techniques for intrusion detection," 2013, *arXiv:1312.2177*. [Online]. Available: http://arxiv.org/abs/1312.2177

[12] M. S. Pervez and D. M. Farid, "Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs," in *Proc. 8th Int. Conf. Softw., Knowl., Inf. Manage. Appl. (SKIMA)*, Dec. 2014, pp. 1–6.

[13] H. Shapoorifard and P. Shamsinejad, "Intrusion detection using a novel hybrid method incorporating an improved KNN," *Int. J. Comput. Appl.*, vol. 173, no. 1, pp. 5–9, Sep. 2017.

[14] S. Bhattacharya, P. K. R. Maddikunta, R. Kaluri, S. Singh, T. R. Gadekallu, M. Alazab, and U. Tariq, "A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU," *Electronics*, vol. 9, no. 2, p. 219, Jan. 2020.

[15] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proc. 9th EAI Int. Conf. Bio-inspired Inf. Commun. Technol. (Formerly BIONETICS)*, 2016, pp. 21–26.

[16] P. Torres, C. Catania, S. Garcia, and C. G. Garino, "An analysis of recurrent neural networks for botnet detection behavior," in *Proc. IEEE Biennial Congr. Argentina (ARGENCON)*, Jun. 2016, pp. 1–6.

[17] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, 2017, pp. 712–717.

[18] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, "A survey of deep learning-based network anomaly detection," *Cluster Comput.*, vol. 22, pp. 949–961, 2019.

[19] B. A. Tama, M. Comuzzi, and K.-H. Rhee, "TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," *IEEE Access*, vol. 7, pp. 94497–94507, 2019.

[20] P. Jeatrakul, K. W. Wong, and C. C. Fung, "Classification of imbalanced data by combining the complementary neural network and smote algorithm," in *Proc. Int. Conf. Neural Inf. Process.* Springer, 2010, pp. 152–159.

[21] B. Yan and G. Han, "LA-GRU: Building combined intrusion detection model based on imbalanced learning and gated recurrent unit neural network," *Secur. Commun. Netw.*, vol. 2018, pp. 1–13, Aug. 2018.

[22] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, "Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic," *IEEE sensors Lett.*, vol. 3, no. 1, Jan. 2019, Art. no. 7101404.

[23] P.-J. Chuang and D.-Y. Wu, "Applying deep learning to balancing network intrusion detection datasets," in *Proc. IEEE 11th Int. Conf. Adv. Infocomm Technol. (ICAIT)*, Oct. 2019, pp. 213–217.

[24] P. Bedi, N. Gupta, and V. Jindal, "Siam-IDS: Handling class imbalance problem in intrusion detection systems using siamese neural network," *Procedia Comput. Sci.*, vol. 171, pp. 780–789, 2020.

[25] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[26] C. Cortes and V. Vapnik, "Support vector machine," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[27] L. Feng, L. Yu, L. Xueqiang, and L. Zhuo, "Research on query topic classification method," *Data Anal. Knowl. Discovery*, vol. 31, no. 4, pp. 10–17, 2015.

[28] T. Chen and C. Guestrin, "XGBoost: A scalable tree BOOSTING system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 785–794.

[29] X. Lei and Y. Xie, "Improved XGBoost model based on genetic algorithm for hypertension recipe recognition," *Comput. Sci*, vol. 45, pp. 476–481, 2018.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[31] A. Raghavan, F. D. Troia, and M. Stamp, "Hidden Markov models with random restarts versus boosting for malware detection," *J. Comput. Virol. Hacking Techn.*, vol. 15, no. 2, pp. 97–107, Jun. 2019.

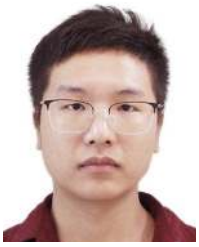
[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556
[34] A. Ismail, S. A. Ahmad, A. C. Soh, K. Hassan, and H. H. Harith, "Improving convolutional neural network (CNN) architecture (miniVGGNet) with batch normalization and learning rate decay factor for image classification," *Int. J. Integr. Eng.*, vol. 11, no. 4, pp. 1–9, 2019.
[35] M. A. Tahir, J. Kittler, and F. Yan, "Inverse random under sampling for class imbalance problem and its application to multi-label classification," *Pattern Recognit.*, vol. 45, no. 10, pp. 3738–3750, Oct. 2012.
[36] A. Liu, J. Ghosh, and C. E. Martin, "Generative oversampling for mining imbalanced datasets," in *Proc. DMIN*, 2007, pp. 66–72.
[37] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.
[38] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.
[39] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, 2018, pp. 108–116.
[40] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
[41] X. Ma and W. Shi, "AESMOTE: Adversarial reinforcement learning with SMOTE for anomaly detection," *IEEE Trans. Netw. Sci. Eng.*, early access, Jun. 24, 2020, doi: 10.1109/TNSE.2020.3004312.
[42] P. Bedi, N. Gupta, and V. Jindal, "I-SiamIDS: An improved Siam-IDS for handling class imbalance in network-based intrusion detection systems," *Appl. Intell.*, pp. 1–19, Sep. 2020.
[43] G. Caminero, M. Lopez-Martin, and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Comput. Netw.*, vol. 159, pp. 96–109, Aug. 2019.
[44] A. K. Verma, P. Kaushik, and G. Shrivastava, "A network intrusion detection approach using variant of convolution neural network," in *Proc. Int. Conf. Commun. Electron. Syst. (ICCES)*, Jul. 2019, pp. 409–416.
[45] J.-T. Wang and C.-H. Wang, "High performance WGAN-GP based multiple-category network anomaly classification system," in *Proc. Int. Conf. Cyber Secur. Emerg. Technol. (CSET)*, Oct. 2019, pp. 1–7.
[46] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in IoT," *Sensors*, vol. 17, no. 9, p. 1967, Aug. 2017.

LAN LIU was born in Yiyang, China, in 1977. She received the B.S., M.S., and Ph.D. degrees in computer architecture from the Huazhong University of Science and Technology, Wuhan, Hubei, China, in 1999, 2002, and 2007, respectively. She joined Guangdong Polytechnic Normal University, as an Assistant, in 2003. She was an Associate Professor and a Master Tutor with the Department of Electronic Information, in 2008 and 2015, respectively. She was a Visiting Professor to research network security with the Computer Science Laboratory, University of Waikato, Hamilton, New Zealand, in 2016. She is the author of more than 30 articles that are published on IEEE conferences and journals (EI indexed or SCI indexed). Her research interests include network security, deep learning, and software defined networks.

PENGCHENG WANG was born in Nanchong, China. He received the B.S. degree in communication engineering from Physical and Electronic Information Engineering, Neijiang Normal University, in 2019. He is currently pursuing the M.S. degree in electronics and communication engineering with Guangdong Polytechnic Normal University. His main research interests include network information security and deep learning.

JUN LIN received the M.S. degree in computer architecture from the Huazhong University of Science and Technology, in 2002. He is currently a Senior Engineer and the Vice Director of the Software & System Research Unit, China Electronic Product Reliability and Environment Testing Research Institute. He is mainly engaged in computer software development and quality engineering technology research and service in mobile communication, computer network, industrial control system, information security, and other relevant fields with solid research foundation and rich experience in engineering, toke charge or mainly participated in the research of dozen important scientific research projects funded by national or provincial government focusing on frontier fields, such as new-generation broadband wireless mobile network, industry internet, IoT, infrastructure software, and so on. He is responsible in technology for IEEE P360 intelligent wearable technology standard establishment. He has published more than ten articles. He holds eight invention patents and seven software copyrights. He received six awards in science and technology.



LANGZHOU LIU was born in Hubei, China. He received the bachelor's degree in engineering. He is currently pursuing the master's degree with Guangdong Polytechnic Normal University. His main research interests include network information security and deep learning.