IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 20, 2020, accepted March 6, 2020, date of publication March 11, 2020, date of current version March 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2980136

An Intrusion Detection Method Using Few-Shot Learning

YINGWEI YU AND NAIZHENG BIAN
College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
Corresponding author: Naizheng Bian (nbian@hnu.edu.cn)

This work was supported by the National Key Research and Development Program of China under Grant 2018YFB1305900.

ABSTRACT Network intrusion detection is an essential means to ensure the security of the network information system. In the real network, abnormal behaviors occur much less frequently than normal behaviors, resulting in scarcity of abnormal samples. We proposed an intrusion detection method based on Few-Shot Learning (FSL), which only used less than 1% of NSL-KDD KDDTrain+ dataset for training, and achieved high accuracy of 92.34% for KDD-Test+ and 85.75% for KDD-Test-21, while other methods, such as J48, Naive Bayes(NB), Random Forest(RF), Support Vector Machine(SVM), recurrent neural network(RNN) and Channel boosted and residual learning based deep convolutional neural network (CBR-CNN), used 20% of KDDTrain+ dataset for training, and achieved relatively low accuracy (less than 89.41% for KDD-Test+ and less than 80.36% for KDD-Test-21). The experiment on dataset of UNSW- NB15 showed a similar result. The detection rates for Dos, U2R, R2L and U2R are improved by our method too, especially for U2R and R2L, which only take up a small proportion of the dataset, the detection rates are increased from 13% to 81.50% and 44.41% to 75.93%, respectively.

INDEX TERMS Few-shot learning, intrusion detection, deep learning, data scarcity.

I. INTRODUCTION

The intrusion detection systems (IDS) have always been a useful tool to protect network and information systems. The IDS can be divided into two types based on detection principles [5]. The first one is misuse detection, which can identify network behaviors by analyzing network data through existing signatures or defined matching patterns and rules. The problem, however, is that misuse detection can only detect known attacks that have identified signatures and is powerless against new attacks. The other is anomaly detection, which identifies abnormal behavior of the system and generates alerts when the actual behavior exceeds the deviation threshold of the normal behavior. Compared with misuse detection, abnormal based detection can detect unknown attacks, so it became the research focus of IDS [19].

The anomaly detection methods based on machine learning have higher detection rates and lower false alarm rates than other methods. Most of the traditional machine learning methods are shallow learning, which tends to emphasize feature engineering and selection. Many deep learning methods have been applied to IDS in recent years, which extract advanced features from the raw data through the search space of the neural network and achieve excellent results. However, the intrusion detection dataset is unbalanced, the normal class data volume is often much larger than the abnormal sample volume, and the proportion of each category in the abnormal sample is very disparate. They prevented deep learning methods from further improving the result.

Few-shot Learning (FSL) is a novel deep learning method that has become popular in the past two years. It aims to learn from a small amount of labeled data. FSL will be an effective method to solve the problem of the small amount of network intrusion detection data. However, the FSL needs a balanced dataset, so we need to select a training set from the original intrusion detection dataset by a balanced sampling method. For the training sets, we chose the same number of samples from each category and sampled N times to get a new dataset. In every training epoch, the model will be trained N times in sequence according to the sampling order. This ensures that the dataset is balanced for each training session. The experimental results show that our method is better than others. The main contributions of this proposed method are as follows:

(1) For the first time, FSL is applied to detecting abnormal network behaviors, and a balanced resampling method is used to maintain the balance of the dataset during training.

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

(2) Two network models, deep neural network and convolutional neural network, are proposed to make FSL suitable for sequence data.

(3) The CenterLoss function used in face recognization is applied to improve the performance of our model.

We evaluated the model on the NSL-KDD and UNSW-NB15 datasets. The experimental results show that our model uses the least training data and outperforms other methods.

II. RELEVANT WORK

With the development of deep learning research, more and more intrusion detection research uses deep learning technology. The research conducted in [26] presented an intrusion detection system that consists of a recurrent neural network. Wu *et al.* [24] used convolutional neural networks to select features and employed a weighted softmax to classify the data.

Furthermore, other researchers have used feature engineering combined with machine learning and deep learning to study intrusion detection systems. Kasongo and Sun [7] proposed a detection system based feature engineering intrusion. Their work used the information gain algorithm to filter the features and employed the deep neural networks to classify data. Salo *et al.* [15] proposed a dimensionality reduction technique that combined the approaches of information gain and principal component analysis with an ensemble classifier for the intrusion detection system.

Although the above work has achieved an excellent overall detection rate, the main problem of these approaches was a low detection rate of minority categories. The reason for this is the unbalanced dataset and the small sample size of minority categories. For example, the samples of U2R are less than 0.05% in the KDDTrain+ dataset, and the detection rate of U2R is less than 15% [24], [26].

FSL can learn features well from very few labeled samples for classification. The Few-shot learning algorithms can be organized into three main categories: initialization based, metric learning based, and hallucination based methods [3]. The distance metric learning method will later be used in this paper. It obtains new representation through an embedding function, which makes samples of the same class close and samples of the different classes far. A distance function is used to calculate the similarity between test samples and known samples when samples are tested.

Matching Networks [21] employs two different embedding functions for support set and query set, and uses cosine distance as the distance function to compute the similarity of two embeddings. This algorithm improves one-shot accuracy on ImageNet from 87.6% to 93.2% and from 88.0% to 93.8% on the Omniglot dataset compared to other approaches. Prototypical Networks employs the same embedding function for support set and query set and use Euclidean distance to calculate the distances of the unlabeled to prototype representations of each class.

Prototypical Networks [17] employs the same embedding function for support set and query set. And it uses Euclidean distance to calculate the distances of the unlabeled embedding to prototype representations of each class.

Instead of using a specific distance function, the relation networks [18] uses the neural network as its distance function to measure the distance between unlabeled embedding and support embedding.

The above methods are all metric learning based, which aims to learn an embedding function to get the represented embeddings, and then uses different distances to classify data.

The loss function plays an important role in the distance metric learning method. The triplet loss [16] was proposed to increase the Euclidean margin for different classes of feature embedding. Wen *et al.* [23] proposed a center loss to find the centers of each of the different classes and used the centers to reduce intra-class distance.

A. FEW-SHOT LEARNING TRAINING STRATEGY

The critical idea of the FSL model is based on prior knowledge to constrain the complexity of hypothesis space $\mathcal{H}$ and reduce its sample complexity $\mathcal{S}$ [22].

We adopted metric learning based FSL in this paper, which is also called the distance metric learning method.

The hypothesis space $\mathcal{H}$ can be treated as an embedding function parameterized with a deep neural network. The learned embedding function f_ϕ maps the data to representation embeddings of each class. The function also makes two representation embeddings of samples in the same class more similar, and samples in different classes more dissimilar.

In FSL, models are usually trained on an N-way-K-shot task. The "way" of the task refers to the number of classes, and the "shot" is the number of instances provide for each class. The set of provided examples for comparing the metric is called the support set S.This support set includes KN labeled samples, $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_K, y_1), \ldots, (\mathbf{x}_K, y_N)\}$. The unlabeled dataset is called the query set Q. In the training stage, the embedding function f_ϕ embeds $x^{(i)} \in \mathcal{X} \subseteq \mathbb{R}^d$ to a new embedding space $z^{(i)} \in \mathcal{Z} \subseteq \mathbb{R}^m$, then in the testing stage, the classfifier compares the distance between query and support embeddings.

There are two main distance functions for the similarity comparison: Euclidean distance and cosine distance. The choice of the similarity measure $d(\cdot)$ is vital in our experiment. We will show the experimental results of different distances in the next chapter.

B. LOSS FUNCTION

We noticed that the FSL based on distance metric is similar to face identification task in the testing stage. They mark unlabeled embeddings as the closest target embedding by comparing them with target embeddings. As the loss function is key for the distance metric learning method, we will introduce the loss function of face identification first.

The face identification task adopts the center loss function, which is an improvement of the traditional softmax loss function. The traditional softmax loss function is presented

as follows. In the Eq.(1), $\boldsymbol{x}_i \in \mathbb{R}^d$ is the i th vector which propagated forward from the deep network, d is the feature dimension. $W \in \mathbb{R}^{d\times n}$ and $b \in \mathbb{R}^n$ are the the weights and bias of the last layer, respectively. Where n is the number of the class, m is the batch size of the network.

$$\mathcal{L}_S = -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}} \tag{1}$$

The data is embedded into embeddings through the deep network, which produce a distribution over classes for a query sample based on a softmax over distances to the support set in the embedding space. The distance function $d(\cdot)$ in our paper are Euclidean distance and cosine distance. Where the $\mathbf{C}_k \in \mathbb{R}^M$ denotes the embedding of the kth class after through an embedding function $f_\phi : \mathbb{R}^D \to \mathbb{R}^M$, and the $\mathbf{C}_k$ is the mean vector of the embedded support points belonging to its class. $\mathbf{x}$ is the embedding feature of the samples in the query set, y is the true label of $\mathbf{x}$.The part of loss funciton $\mathcal{L}_S$ of the model is negative log-probability $\mathcal{L}_S = -\log p_\phi(y = k|\mathbf{x})$.

$$p_\phi(y = k|\mathbf{x}) = \frac{\exp\left(-d\left(f_\phi(\mathbf{x}), \mathbf{c}_k\right)\right)}{\sum_{k'} \exp\left(-d\left(f_\phi(\mathbf{x}), \mathbf{c}_{k'}\right)\right)} \tag{2}$$

The center loss function [23] improved the discriminative power of the deeply learned embeddings. There are two kinds of distance for an embedding, one is the intra-class distance, and another is the inter-class distance. $\mathcal{L}_C$ increases the discriminative power, which means increasing the inter-class distance and minimizing the intra-class distance.

$$\mathcal{L}_C = \sum_{i=1}^{m} \left\|\boldsymbol{x}_i - \boldsymbol{c}_{y_i}\right\|_2^2 \tag{3}$$

where the $c_{y_i} \in \mathbb{R}^d$ denotes the y_ith class center of deep features. The c_{y_i} is not the y_ith center of the whole training set, it will be updated on each iteration. A scalar λ is used for balancing the two loss functions. So the final loss function is shown as follows.

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_S + \lambda\mathcal{L}_C \\ &= -\log p_\phi(y = k|\mathbf{x}) + \lambda \sum_{i=1}^{m} \left\|\boldsymbol{x}_i - \boldsymbol{c}_{y_i}\right\|_2^2 \\ &= d\left(f_\phi(\mathbf{x}), \mathbf{c}_k\right) + \log \sum_{k'} \exp\left(-d\left(f_\phi(\mathbf{x}), \mathbf{c}_{k'}\right)\right) \\ &\quad + \lambda \sum_{i=1}^{m} \left\|\boldsymbol{x}_i - \boldsymbol{c}_{y_i}\right\|_2^2 \end{aligned} \tag{4}$$

The model will minimize such a loss function via gradient-descent(see details in Algorithm 1:Training Episode Procedure). Where C_i is the ith class of dataset, $\mathcal{D}_{C_i}$ and $\mathcal{T}_{C_i}$ are the ith class of training set and testing set,respectively. N_T and N_S denote the number of per class. N_P is the number of sampling.

C. EMBEDDING FUNCTION

The goal of embedding function is to extract essential features, reduce the dimension of data, and retain all the information of the original data to the maximum extent. Various embedding models can be used to extract features. Here, we choose the deep neural networks(DNN) and Convolutional neural network (CNN) as our embedding functions.

Algorithm 1 Training Episode Procedure

Require:
Training set $\mathcal{D}$, N_T, N_P,Testing dataset $\mathcal{T}$, N_S, hyperparameter λ.

Ensure:
The loss J for a training episode.

for i in $\{C_1, \ldots, C_k\}$ **do**
  $S_k \leftarrow$ RandomSample($\mathcal{T}_{C_i}, N_S$) {Select support examples from Testing dataset}
  for j in $\{1, \ldots, N_P\}$ **do**
    $Q_k \leftarrow Q_k +$ RandomSample($\mathcal{D}_{C_i}, N_T$) {Select N_T examples for training dataset}
  end for
  $\mathbf{c}_k \leftarrow \frac{1}{N_S} \sum_{(\mathbf{x}_i, y_i)\in S_k} f_\phi(\mathbf{x}_i)$ {Compute benchmark of embedding features from support examples}
end for
$J \leftarrow 0$ {Initialize loss}
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q}\left[\mathcal{L}_S + \lambda\mathcal{L}_C\right]$ {Update loss}
  end for
end for

1) DNN

Inspired by neuroscience, deep neural networks(DNN) have been widely used in machine learning problems. The deep neural network consists of the input layer, hidden layer and output layer. Furthermore, the number of hidden layers usually determines the width of the neural network. Each unit in the hidden layer is like a neuron, which accepts input from the previous layer, and then calculates the activation value. As linear combinations of vectors end up being linear, the problems of deep learning are usually nonlinear. The activation function ReLu [13] was added in the network, which guarantees the nonlinearity of the network.

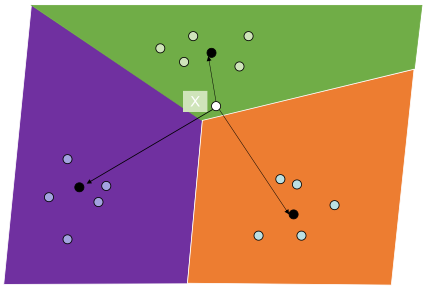


FIGURE 1. Different colored areas represent the feature space of different categories of data, where the black points represent the representation embedding of this category. The white point X represents unlabeled data, and we can classify it by calculating the distance between unlabeled data and the presentation embedding.

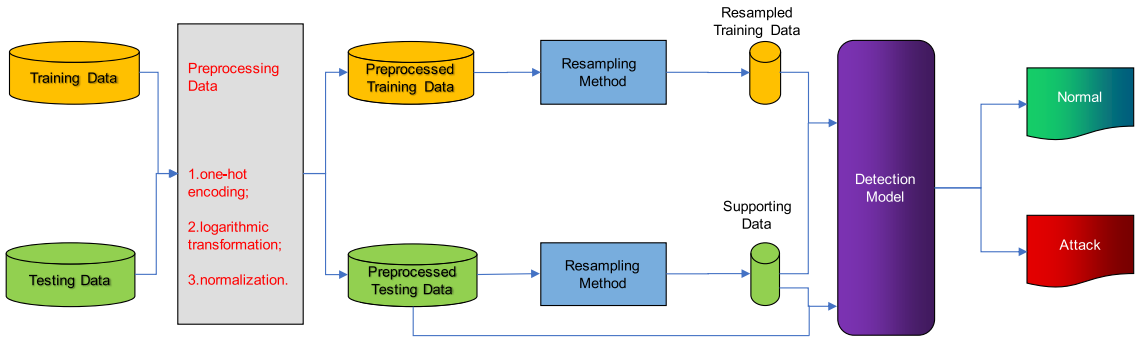


FIGURE 2. The framework of proposed model.

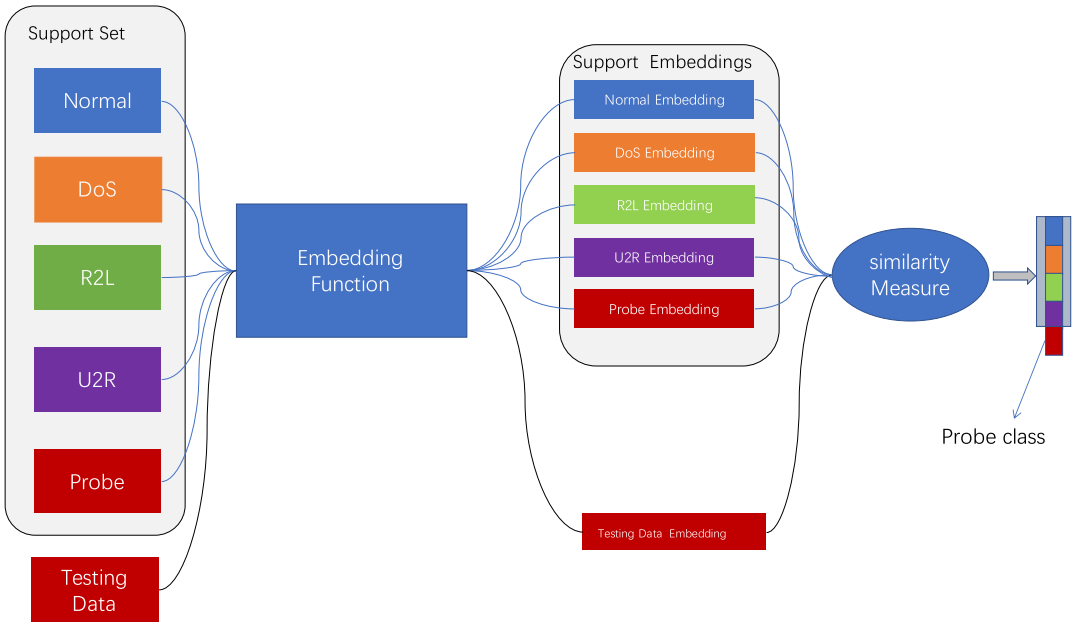


FIGURE 3. The testing phase of the multi-class classification on the NSL-KDD dataset.

2) CNN

The convolutional neural network (CNN) is a classical neural network architecture. Especially in the field of computer vision, CNN has made significant achievements, and almost all network architectures have adopted CNN [9].

Figure 2 shows the architecture of the intrusion detection model architecture based on Few-shot learning. This architecture consists of three phases, including preprocessing of raw data, resampling training data and support data, and attacking recognition with FSL.

Preprocessing raw data is one of the essential works for deep learning, which cleans data and provides object data for the classification algorithm. Data normalization can reduce model search time and enhance model performance. The second step involves resampling the training set and support set to balance the dataset. The third step is model training and the last step is testing. The model can find suitable hyperparameters through the resampled training set and support set, and evaluate it with the support set and testing set. The specific testing process is illustrated in figure 3.

Figure 3 shows the flow of testing phase. Different background colors of the data in the support set represent different categories. Here we assume the testing data is Probe and color it red, the same color as the Probe class. The embeddings of the testing data and all the data in the support set are generated with the embedding function. Similarity measurement is used to evaluate the distance between the testing data embedding and each embedding in the support embeddings. Because the testing embedding is most similar to the Probe embedding, it is classified as the Probe.

III. IMPLEMENTATION

The experiment is performed on a laptop with 8GB of RAM, Intel Core I7-7700HQ, and Nvidia GeForce 1050Ti. The operating system is Ubuntu 18.04, and the software environment is Pytorch 1.10, CUDA 10 and cuDNN 7.5. In order to ensure that the experiment can be reproduced, we also tested in colab.[1]

[1] https://colab.research.google.com

TABLE 1. The number of different categories in the NSL-KDD dataset.

	Normal	DoS	Probe	R2L	U2R	Total
KDDTrain+	67,345	45,926	11,655	995	52	12,5973
KDDTest+	9,711	7,458	2,421	2,754	200	22,544
KDDTest−21	2,152	4,342	2,402	2,754	200	11,850

A. DATASETS

1) NSLKDD DATASET

We used the new NSL-KDD dataset in our experiment. It removes duplicate and redundant records of the KDD Cup 99 dataset, so the data is more imbalanced than the original data.

Based on its proportion and difficulty of prediction, three subsets were generated, KDDTrain$^+$, KDDTest$^+$ and KDDTest-21. KDDTrain$^+$ includes 125,973 records of data and the corresponding KDDTest$^+$ has 22,544 records of test data. KDDTest-21 has 11850 records [20].

All different attacks of NSL-KDD dataset can be categorized into four main classes:

1) DoS: Denial of Service attacks is when intensive access requests are sent to a specific target host, causing the target computer's network resources and system resources to be exhausted, such as Teardrop, Smurf.

2) Probe: Probing attacks is an attempt to collect host information from networks, such as Satan, Portsweep, and Saint.

3) R2L: Remote-to-Login (R2L) Attack is when remote attackers access the computer through the vulnerability, then using the computer's account number and weak password to enter the target host to operate, such as Xsnoop, Httptunnel.

4) U2R: User-to-Root (U2R) Attack is when a user with no permission or low privilege obtains root privileges through a vulnerability or illegal operation, such as Rootkit, Buffer_overflow, and Loadmodule. The initial distribution of these categories is shown in the table 1.

2) UNSW-NB15 DATASET

The UNSW-NB15 dataset was created using three IXIA PerfectStorm as virtual servers by the Australian Centre for Cyber Security (ACCS). It is a mixture of real modern normal activities and synthetic intrusion behaviors [11]. The purpose of this dataset is to solve some inherent problems of the KDD and NSL-KDD dataset. These two datasets are not a comprehensive representation of a modern low foot print attack environment. The UNSW-NB15 dataset aims to reflect a more modern and complex threat environment.

The training set contains a total of 175,341 records, and there are 82,332 records in the testing set. Table 2 shows in detail the class distribution of the UNSW-NB15 dataset.

B. DATA PREPROCESSING

In the NSL-KDD dataset, all data of the feature *num_outbound_cmds* is 0. The feature *times* is an NSL-KDD attached attribute, whose purpose is to measure the difficulty of prediction of the data. Thus we first removed the two features that do not affect the result. Among the remaining 40 attributes, we found three non-numeric features and 37 numeric features.

TABLE 2. The number of different categories in the UNSW-NB15 dataset.

Attack types	Training set	Testing set
Normal	56,000	37,000
Generic	40,000	18,871
DoS	12,264	4,089
Fuzzers	18,184	6,062
Reconnaissance	10,491	3,496
Shellcode	1,133	378
Worms	130	44
Backdoor	1,746	583
Exploits	33,393	11,132
Total	175,341	82,332

In the UNSW-NB15 dataset, we also remove two features of *id* and *label*, and the *attack_cat* attribute represents the classification of data.

The non-numeric features in dataset are encoded by one-hot encoding.For example *protocol_type* in NSL-KDD dataset which has three attributes of *tcp*, *udp*, *icmp* is replace with three-dimensional vectors (1, 0, 0), (0, 1, 0), (0, 0, 1). After conversion,the NSL-KDD finally has 121-dimensional features. And the same transformation method was applied on UNSW-NB15 to get 196 features.

After all the data has been converted to numbers, the number distribution of some features spans multiple orders of magnitude, which is not conducive to the similarity assessment. In order to solve this problem, the logarithmic transformation was applied to these features.

$$x_j = \log(x_i + 1) \tag{5}$$

Eq(5) is the logarithmic transformation equation. Where x_j is the new feature, x_i is the previous feature, and the logarithmic transformation will compress the range of large numbers and extend the range of small numbers [1].

1) DATA NORMALIZATION

Before training the model, we used the Min-max normalization to normalize our data. Normalization scales the data according to certain rules and finally makes them fit a specific interval. It can also improve the convergence speed and accuracy of the model. The transformation function is as following Eq(6):

$$x_{i\_\text{normalized}} = \frac{x_{ij} - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{6}$$

where x_{ij} is the value of the jth records of the ith feature. After normalization, the range of numbers for all our vectors is within the range [0, 1].

2) RESAMPLED TRAINING DATASET

From table 1, we can see that the number of samples varies significantly among the different categories in the NSL-KDD dataset. The number of Normal samples is more than 1000 times that of U2R samples.

The imbalance of dataset can affect the performance of deep learning classifiers. In imbalanced dataset, the most prevalent class is called the majority class, while the rarest class is called the minority class. There are three main approaches to tackle the imbalance problem.

The first approach preprocesses the raw data to get better input data by over-sampling, under-sampling and hybrid-sampling. Over-sampling generates the samples of the minority class, while the under-sampling drops the samples of the majority class. Hybrid-sampling is a combination of the over-sampling method and the under-sampling method.

The second approach adopts ensemble-based algorithms. The ensemble-based classifiers such as bagging or boosting algorithms are helpful to alleviate the influence of the imbalanced class distribution.

The third approach uses various loss functions to handle the imbalanced class samples in many deep learning models. Those functions, such as focal loss function [10] and tversky loss function [14], have achieved significant progress compared with the softmax loss function.

The first approach was used in our method.The detail of the resampling is shown in the algorithm 1.

C. EVALUATION METRICS

The final classification results are divided into four states: TP(true positive), FP(false positive), TN (true negative), FN (false negative), they also are four basic metrics of the confusion matrix. TP is the number of samples that are classified in the normal class. FP is the number of attack samples that are incorrectly classified in the normal class. TN is the number of attack samples that are classified correctly. FN is the number of normal class samples that are classified in attack class.

To evaluate the performance of the proposed method. Four states, the accuracy, precision, detection rate,false alarm rate and F-measure are defined as shown in Eq.(7).

$$\begin{cases} Accuracy = \dfrac{TP + TN}{TP + TN + FN + FP} \\ Precision = \dfrac{TP}{TP + FP} \\ False\ Positive\ Rate(FPR) = \dfrac{FP}{FP + TN} \\ False\ Negative\ Rate(FNR) = \dfrac{FN}{FN + TP} \\ False\ Alarm\ Rate(FAR) = \dfrac{FNR + FPR}{2} \\ Detection\ Rate\ (DR) = \dfrac{TP}{TP + FN} \\ F - measure = \dfrac{2 * (Precision * DR)}{Precision + DR} \end{cases} \tag{7}$$

TABLE 3. Detail of training sets.

Dataset	Type	No. per class	Sample times	Total	Proportion in training set
NSL-KDD	binary	100	5	1000	0.79%
NSL-KDD	multi-class	50	10	2500	Less than 1.98%
USNW-NB15	binary	100	10	2000	1.14%

D. EXPERIMENTS

In our experiment, four crucial factors are influencing our experimental results: the training dataset, the support set, the embedding function model, and the loss function, respectively. We will describe in detail the settings of these four factors in our experiments later. In order to evaluate our method, experiments were carried out on NSL-KDD and UNSW-NB15 datasets.

In the NSL-KDD dataset, the number of *U2R* and *R2L* samples are scarce. Furthermore, the proportion of the sample size of these two categories in the testing set is much more scarce than that in the training set. Although the UNSW-NB15 dataset is a new dataset, it only has a small number of worms samples in multi-class classification, and the proportion of attack samples and training samples is 1:3. The multi class distribution of test dataset can not reflect the detection effect of the class of the small sample size. Therefore, binary classification experiments are carried out on the NSL-KDD and UNSW-NB15 datasets and multi-class classification on the NSL-KDD dataset.

1) THE RESAMPLED TRAINING DATASET

As mentioned previously, FSL does not need a large number of samples for training. So in the experiment, the original training set will not be used as the experimental training set. In order to eliminate the influence of imbalanced dataset as far as possible, the samples training set was balanced by resampling.

For the binary classification experiment, there are only two kinds of categories: normal and abnormal.In NSL-KDD dataset, for normal and attack classes, 100 samples were randomly sampled five times. So there are only 1000 samples ($2 * 100 * 5$) in the training set. In the UNSW-NB15 dataset, for each class, 100 samples were chosen and were randomly sampled ten times.

For the multi-class classification experiment, we selected 50 samples and sampled them ten times for each class. The sampled 2500 ($5 * 50 * 10$) samples were used as the new training dataset. Because the number of samples of the *U2R* class is only 52, there must be a large number of duplicate samples in 10 times.

The experimental results show that the new training set after resampling will not reduce the final experimental effect. Table 3 shows the details of the sampling of the training set in the experiment.

TABLE 4. Detail of support sets.

Dataset	Type	No. per class	Sample times	Total	Proportion in testing set
NSL-KDD	binary	25	1	50	0.22%
NSL-KDD	multi-class	10	1	50	0.22%
USNW-NB15	binary	25	1	50	0.06%

2) THE CHOICE OF SUPPORT SET

The selection of support datasets is so important that it directly influences the performance of the final model. As described in the algorithm 1, instead of selecting samples from the training set as our support set, we randomly selected some samples from each class as the support set from the testing set. The reason for not selecting samples from the training set is that the samples selected from the test set are more consistent with the distribution of test cases.

The table 4 shows the details of the support set in the experiment. For example, in the NSL-KDD multi-class classification experiment, we selected 50 samples(10 samples per class) from the test set as the support set. The support set only accounted for 0.22% of the testing set, but the accuracy has been improved by 4%.

3) THE DIFFERENT EMBEDDING AND DISTANCE FUNCTIONS

Different embedding functions have distinct effects on the model. Two embedding functions were chosen for comparison. The first one is the DNN with three layers, and the second is the CNN.

In the experiment with NSL-KDD, the DNN embedding function input layer has 121 units and 84 nodes at the output layer, and 108 nodes and 96 nodes at hidden layers. After the Embedding function, the model ended up with an embedding of 84 dimensions. Also, the represented embedding is computed as the mean of embedded support data for each class. During the training phase, the model calculates the network parameters by minimizing the distance between the train and represented embedding. In the test phase, the model uses the distance function to classify the test data. Euclidean distance and cosine distance are used to classify and evaluate the model. The comparison results are shown in the following table 6. The model uses the Adam optimizer algorithm to optimize parameters, and the learning rate is chosen to be 0.001.

When using CNN as the embedding function, we treat the sequence data as a 2-dimensional image with one channel. The UNSW-NB15 data size was 196 dimensions; for instance, it was converted into (1, 14, 14) size data. Where 1 is the number of data channels, and 14 is equivalent to the length and height of the image.

After transformation, the dimension of data is small, which may suffer degradation with a multi-layer convolutional neural network. The residual learning [6] can address the degradation problem of multi-layer networks. The residual learning block was applied to the CNN embedding network.

TABLE 5. Parameters of CNN.

Layer Name	Output Shape	Filter Size	Padding	Stride
Input	$11 \times 11 \times 1$	N/A	N	N/A
Convolutional 1	$11 \times 11 \times 4$	$3 \times 3 \times 4$	Y	(1,1)
Max Pooling 1	$5 \times 5 \times 4$	2×2	N	(2,2)
Convolutional 2	$5 \times 5 \times 16$	$3 \times 3 \times 16$	Y	(1,1)
Max Pooling 2	$2 \times 2 \times 16$	2×2	N	(2,2)
Convolutional 3	$2 \times 2 \times 64$	$3 \times 3 \times 64$	Y	(1,1)
Max Pooling 3	$1 \times 1 \times 64$	2×2	N	(2,2)
Convolutional 4	$1 \times 1 \times 121$	$3 \times 3 \times 121$	Y	(1,1)
Max Pooling 4	$1 \times 1 \times 121$	2×2	N	(2,2)
Flatten	121	N/A	N/A	N/A
Shortcut	Flatten + Input			
Output	121	N/A	N/A	N/A

TABLE 6. Performance of different functions on two embedding functions.

Embedding function	Similary function	KDDtest+	KDDtest-21
DNN	Euclidean	86.24%	74.08%
	Cosine	88.14%	77.51%
	Cosine & CenterLoss	**89.38%**	**80.16%**
CNN	Euclidean	88.67%	79.24%
	Cosine	91.75%	85.26%
	Cosine & CenterLoss	**92.33%**	**86.23%**

TABLE 7. Confusion matrix for the binary classification experiments on KDDTest+.

Actual Class \ Predicted Class	Normal	Attack
Normal	9246	465
Attack	1261	11572

TABLE 8. Confusion matrix for the binary classification experiments on KDDTest-21.

Actual Class \ Predicted Class	Normal	Attack
Normal	1725	427
Attack	1262	8436

There are three convolutional layers in our model. The filter layer size is 3*3, which is commonly used in current convolution networks. The CNN model parameters of the NSL-KDD experiment are shown in the table 5 below.

As shown in the table 6, using CNN as an embedding function has a better classification effect on the model. Compared with DNN, the accuracy of CNN has a 3% improvement on KDDTest+ and a more significant improvement on KDDTest−21. Similarly, cosine distance is proved to be more powerful on the model than Euclidean distance.

4) THE LOSS FUNCTION

The CenterLoss function is used to improve the accuracy of the model. As mentioned earlier in this paper, CenterLoss function tries to find a class center with smaller intra-class distance. It also uses Euclidean distances to measure the distance between the embedding and the center of the category.

In the distance function, Euclidean distance is also used to measure the distance between the embedding and the support representative embedding. The class centers of the embedding and representational embedding hardly coincide, so there is a conflict between the two functions.

TABLE 9. Results of the evaluation metrics for the binary classification on NSL-KDD.

Dataset	Class	TP	TN	FP	FN	Precision	DR	F1-score	FPR	Accuracy
KDDTest+	Normal	9246	11572	1261	465	88.00%	95.21%	91.46%	9.83%	92.34%
	Attack	11572	9246	465	1261	96.14%	90.17%	93.06%	4.79%	
KDDtest-21	Normal	1725	8436	1262	427	57.75%	80.16%	67.13%	13.01%	85.75%
	Attack	8436	1725	427	1262	95.18%	86.99%	90.90%	19.84%	

TABLE 10. Binary classification accuracy comparison on NSL-KDD.

Methodology	Training dataset	KDDTest+ Accuracy (%)	KDDTest-21 Accuracy (%)
J48 [20]	20% KDDTrain+	81.05	63.97
Naive Bayes [20]	20% KDDTrain+	76.56	55.77
NB Tree [20]	20% KDDTrain+	82.02	66.16
Random Forest [20]	20% KDDTrain+	80.67	63.25
Random Tree [20]	20% KDDTrain+	81.59	58.51
Multi-layer perceptron [20]	20% KDDTrain+	77.41	57.34
SVM [20]	20% KDDTrain+	69.52	42.29
RNN [26]	Full KDDTrain+	83.28	68.55
Fuzzy based semi-supervised learning [2]	Full KDDTrain+(10% labled data)	84.12	68.82
CBR-CNN [4]	80% KDDTrain+	89.41	80.36
Our model	**Less than 1% KDDTrain+**	**92.34**	**85.75**

Therefore, the model with Euclidean distance will not apply to the CenterLoss.

The CenterLoss function is applied to the method with cosine distance as the distance function. In comparative experiments, the hyperparameters of our method are kept the same as those of other methods. As shown in the table 6,the CenterLoss function improves the performance of classification.

IV. RESULTS AND COMPARISON

A. NSL-KDD DATASET

Other state-of-the-art deep learning methods and the classical machine learning methods were used to compare them with our method.

1) BINARY CLASSIFICATION

KDDTest+ and KDDTest-21 which are two testing sets were used to evaluate the model perforation. In this paper, less than 1% of training data is used for training, while 100% of testing data is used for testing. The confusion matrices on the testing data are shown in table 7 and table 8.

The detailed evaluation metric is calculated according to the confusion matrices, as listed in table 9. Compared with the KDDTest+, the related indicators for the normal class of KDDTest-21 declined significantly. The main reason is that KDDtest-21 has reduced about 7500 samples of normal class from KDDTest+,which is more than twice as many as that of abnormal class.

The traditional machine learning methods, including J48, Naive Bayes, NB Tree, Random Forest, Random Tree, Multi-layer perceptron and SVM, were evaluated by Tavallaee *et al.* [20] with NSL-KDD dataset. These algorithms were evaluated on NSL-KDD set by using 20% training data.They used 20% of datasets for training. The result was listed in table 10. The traditional machine learning methods have a better performance in KDDTest+ than in KDDTest-21. Fuzzy based semi-supervised learning is a method of semi-supervised learning. It divided the training set into two parts and utilized 10% of labeled data and 90% of unlabeled data for training the model. The method has obtained excellent results, and the final effect of the model is better than the RNN model [26].RNN model was a deep learning method which used a full training set to classify the testing dataset. Although the model is better than traditional machine learning methods, it needs more data to train the model. Channel boosted and residual learning based deep convolutional neural network (CBR-CNN) uses the lightweight CNN architecture for classification. It used 80% of the training set and the remaining 20% as the testing set. The performance of CBR-CNN is inferior to that of our method.

In the table 10,it is easy to see that our method exceeds other methods in both two testing sets. Not only that, our method used the least amount of data for training; it only used less than 1% of the data for training and got the best results.

2) MULTI-CLASS CLASSIFICATION

In the case of multi-class classification, the accuracy of different methods on KDDTest+ and KDDTest−21 are compared first. The comparison results are shown in figure 4 and table 11 below.

In [26], traditional machine learning classifiers were compared with RNN model, including J48, Naive Bayes, Random Forest and so on. RNN had surpassed the traditional algorithms in all indicators. It can be seen from the table 11 that the performance of the traditional machine learning classifiers remains poor in both KDDTest+ and KDDTest-21 datasets. The CNN method is better than the traditional machine learning methods, but not as good as RNN.

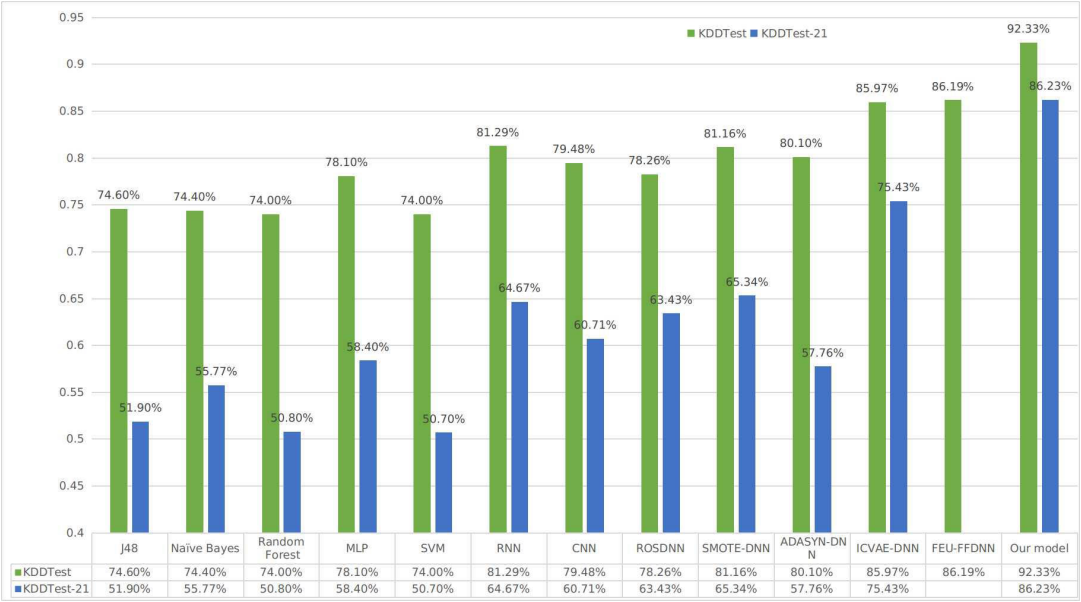


FIGURE 4. Multi-class classification accuracy comparison on NSL-KDD.

TABLE 11. Comparison of multi-class classification on NSL-KDD.

Methodology	Training dataset	KDDTest+ Accuracy (%)	KDDTest-21 Accuracy (%)
J48 [26]	Full KDDTrain+	74.60	51.90
Naive Bayes [26]	Full KDDTrain+	74.40	55.77
NB Tree [26]	Full KDDTrain+	75.40	55.40
Random Forest [26]	Full KDDTrain+	74.00	50.80
Random Tree [26]	Full KDDTrain+	72.80	49.70
Multi-layer perceptron [26]	Full KDDTrain+	78.10	58.40
SVM [26]	Full KDDTrain+	74.00	50.70
RNN [26]	Full KDDTrain+	81.29	64.67
CNN [24]	Full KDDTrain+	79.48	60.71
ROS-DNN [25]	67,245 samples	78.26	63.43
SMOTE-DNN [25]	67,245 samples	81.16	65.34
ADASYN-DNN [25]	67,245 samples	80.10	57.76
ICVAE-DNN [25]	67,245 samples	85.97	75.43
FEU-FFDNN [7]	75% KDDTrain+	86.19	-
Our model	**Less than 2% KDDTrain+**	**92.33**	**86.23**

ROS-DNN, SMOTE-DNN, ADASYN-DNN and ICVAE-DNN are four deep learning models based on the resampling algorithms [25]. They all synthesized minority abnormal samples, including random over sampler (ROS), SMOTE, and ADASYN, based on the over-sampling method.These resampling algorithms were used to generate training sets and then DNN is used to train models. After that, the author used improved conditional variational AutoEncoder (ICVAE) to generate datasets for experimental comparison, namely ICVAE-DNN. The authors used 20% of the training data to generate 67,245 samples, 13,449 for each of the five categories.

From the experimental results, we can see that the ROS-DNN has the lowest accuracy among the four algorithms on the KDDTest+, but its accuracy on KDDTest-21 is higher than that of ADASYN-DNN and slightly worse than that of RNN. The accuracy of the SMOTE-DNN model has surpassed that of RNN in KDDTest+. Like the SMOTE-DNN model, ADASYN-DNN is better than CNN and traditional algorithm on KDDTest+, but its effect on KDDTest-21 is inferior, even lower than some traditional algorithms.

TABLE 12. Confusion matrix for the multi-class classification experiment on KDDTest+.

Actual Class \ Predicted Class	Normal	Dos	R2L	U2R	Probe
Normal	9301	90	94	25	201
Dos	329	7016	24	2	87
R2L	623	3	2091	23	8
U2R	22	1	12	163	2
Probe	18	84	66	10	2243

ICVAE-DNN model is the best algorithm among these resampled algorithms, but it still has some shortcomings compared with our algorithm. First, the algorithm needs more data than our method. Although it only generates data from 20% of the original data, it is still 10 times more samples than our

TABLE 13. Results of the evaluation metrics for the multi-class classification.

Class	TP	TN	FP	FN	Precision	DR	F-measure	FPR
Normal	9301	11841	992	410	90.36%	95.78%	92.99%	7.73%
Dos	7016	14908	178	442	97.53%	94.07%	95.77%	1.18%
R2L	2091	19594	196	663	91.43%	75.93%	82.96%	0.99%
U2R	163	22278	66	37	71.18%	81.50%	75.99%	0.30%
Probe	2243	19825	298	178	88.27%	92.65%	90.41%	1.48%

TABLE 14. Comparison of evaluation metrics for the multi-class classification on NSL-KDD.

Attack Type	Our model		RNN		CNN		ROS-DNN	SMOTE-DNN	ADASYN-DNN	ICVAE-DNN
	DR	FPR	DR	FPR	DR	FPR	DR	DR	DR	DR
Dos	**94.07%**	**1.18%**	83.49%	2.06%	83.21%	2.35%	80.32%	82.19%	83.28%	85.65%
R2L	**75.93%**	0.99%	24.69%	0.80%	21.68%	**0.69%**	12.75%	10.93%	9.84%	44.41%
U2R	**81.50%**	0.30%	11.55%	0.07%	13.00%	**0.06%**	6.00%	11.00%	8.00%	11.00%
Probe	**92.65%**	**1.48%**	83.40%	2.16%	81.87%	2.09%	56.26%	56.75%	59.81%	74.97%

TABLE 15. Binary classification accuracy comparison on UNSW-NB15.

Methodology	Training dataset	Accuracy (%)	FAR (%)
DT [12]	UNSW-NB15 training set	85.56	15.78
LR [12]	UNSW-NB15 training set	83.15	18.48
NB [12]	UNSW-NB15 training set	82.07	18.56
ANN [12]	UNSW-NB15 training set	81.34	21.36
EM clustering [12]	UNSW-NB15 training set	78.47	23.79
GALR-DT [8]	UNSW-NB15 training set	81.42	**6.39**
Our model	**less than 2% UNSW-NB15 training set**	**92.00**	8.01

method. Second, the accuracy of the algorithm is lower than that of our method. Our method is 6% and 10% higher than the ICVAE-DNN model on KDDTest+ and KDDTest-21 datasets, respectively.

FEU-FFDNN used a filter-based feature selection algorithm and DNN architecture to conduct experiments, and the final model effect was higher than other algorithms except ours. Its training dataset used 75% of the original training data, while the remaining 25% was used as validation data. Our method increases accuracy to 92.33% on KDDTest+, which is more than 6% higher than other methods. It also performs well on KDDTest-21 datasets, nearly 35% higher than the traditional machine learning methods and 10% higher than the current popular deep learning methods.

Table 12 shows the confusion matrix of the model using CNN as the embedding function and CenterLoss as the loss function. The parameters in table 13 are calculated based on table 12. We can see that the number of samples have no significant relationship with result.

We compare the indicators of our method with others in table 14. The DR of our method is much better than that of others. In the training and testing set, the number of U2R and R2L samples are scarce, which leads to the low DR of other methods in these two categories. Our method is 31% higher than other methods in u2l and 68% higher in U2R. In the remaining two categories with sufficient samples, the DR of our method is above 90%, while other algorithms are generally below 85%. In terms of the FPR metric, the FPRs of DoS and Probe are better for our method, but the FPRs of U2L and R2L are inferior to the RNN and CNN methods. The F-measure considers both the precision and the detection rate. We can also see that our algorithm performs fairly well. Our proposed method has been evaluated on NSL-KDD testing sets, and the final results have demonstrated that our method is better than the others.

B. UNSW-NB15 DATASET

As table 3 shown, the experiment of binary classification is carried out on the UNSW-NB15 dataset with less than 2% training data. The results are shown in the table 15.

Paper [12] conducted a series of experiments with UNSW-NB15, including Decision Tree (DT), Naive Bayes (NB), Artificial Neural Network (ANN), Logistic Regression (LR) and Expectation-Maximization (EM) Clustering.

Paper [8] proposed a new method called GALR-DT (a Genetic Algorithm and Logistic Regression wrapper of Decision Tree). GALR-DT achieved relative high accuracy (81.42%) and very low FAR (6.39%) on UNSW-NB15. Our method obtained higher accuracy (92%) and similar FAR (8.01%) using less than 2% of dataset as training data. The detection rate of our method is 92.11% on UNSW-NB15.

V. DISCUSSION

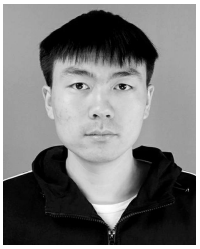
We introduced FSL into intrusion detection and carried out experiments on NSL-KDD and UNSW-NB15 datasets. The results show that our method can achieve remarkable results without large size of sample data. The embedding function, distance function, and loss function are the main aspects which influence classification results using FSL. The samples of the training set were balanced by resampling for both binary classification and multi-class classification. The result of our method shows that though the balanced dataset reduces the number of samples for training, the performance of classification is improved.

VI. CONCLUSION

In this work, we designed a new IDS based on FSL. The abnormal samples and normal samples were balanced by resampling. Embedding of support set and test data were generated with trained embedding function, and similarity measurement was used to evaluate the distance between the test data embedding and each embedding in the support embeddings.The CenterLoss function with cosine distance was applied to improve the accuracy of our method. Compared with traditional machine learning and deep learning methods, our method has high accuracy and detection rate in both binary and multi-class classifications. More importantly, our method only used less than 2% of data for training and achieved leading performance. In future research, we will study the classification performance of initialization based FSL.

REFERENCES

[1] A. Z. A. Casari, *Feature Engineering for Machine Learning*. Sebastopol, CA, USA: O'Reilly Media, 2018.
[2] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Inf. Sci.*, vol. 378, pp. 484–497, Feb. 2017.
[3] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," 2019, *arXiv:1904.04232*. [Online]. Available: http://arxiv.org/abs/1904.04232
[4] N. Chouhan, A. Khan, and H.-U.-R. Khan, "Network anomaly detection using channel boosted and residual learning based deep convolutional neural network," *Appl. Soft Comput.*, vol. 83, Oct. 2019, Art. no. 105612.
[5] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Comput. Secur.*, vol. 28, nos. 1–2, pp. 18–28, Feb. 2009.
[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[7] S. M. Kasongo and Y. Sun, "A deep learning method with filter based feature engineering for wireless intrusion detection system," *IEEE Access*, vol. 7, pp. 38597–38607, 2019.
[8] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Comput. Secur.*, vol. 70, pp. 255–277, Sep. 2017.
[9] 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.
[10] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," Tech. Rep., Aug. 2017.
[11] 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)*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Dec. 2015, pp. 1–6.
[12] 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.
[13] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.
[14] S. S. M. Salehi, D. Erdogmus, and A. Gholipour, "Tversky loss function for image segmentation using 3D fully convolutional deep networks," in *Proc. Int. Workshop Mach. Learn. Med. Imag.* Cham, Switzerland: Springer, 2017, pp. 379–387.
[15] F. Salo, A. B. Nassif, and A. Essex, "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection," *Comput. Netw.*, vol. 148, pp. 164–175, Jan. 2019.
[16] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 815–823.
[17] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.
[18] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.
[19] Y. Tang and S. Chen, "An automated signature-based approach against polymorphic Internet worms," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 7, pp. 879–892, Jul. 2007.
[20] 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.
[21] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.
[22] Y. Wang, Q. Yao, J. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," 2019, *arXiv:1904.05046*. [Online]. Available: https://arxiv.org/abs/1904.05046
[23] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Computer Vision*. Cham, Switzerland: Springer, 2016, pp. 499–515.
[24] K. Wu, Z. Chen, and W. Li, "A novel intrusion detection model for a massive network using convolutional neural networks," *IEEE Access*, vol. 6, pp. 50850–50859, 2018.
[25] Y. Yang, K. Zheng, C. Wu, and Y. Yang, "Improving the classification effectiveness of intrusion detection by using improved conditional variational AutoEncoder and deep neural network," *Sensors*, vol. 19, no. 11, p. 2528, 2019.
[26] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.



YINGWEI YU received the B.S. degree in civil engineering from the Hunan University of Science and Engineering, in 2015. He is currently pursuing the master's degree with the College of Computer Science and Electronic Engineering, Hunan University. His research areas are intrusion detection and deep learning.

NAIZHENG BIAN received the master's degree from the Department of Computer Science, Virginia Polytechnic Institute and State University, in 2002. He is currently working with the College of Computer Science and Electronic Engineering, Hunan University. His research interests are broad, including deep learning-based image recognition, genetic data processing and analysis, and networks security.