Hindawi
Security and Communication Networks
Volume 2022, Article ID 3317048, 13 pages
https://doi.org/10.1155/2022/3317048

Research Article

Few-Shot Learning-Based Network Intrusion Detection through an Enhanced Parallelized Triplet Network

Ji-Yu Tian,[1,2] **Zu-Min Wang**,[1] **Hui Fang**,[3] **Li-Ming Chen**,[4] **Jing Qin**,[5] **Jie Chen**,[1] **and Zhi-He Wang**[6]

[1]*College of Information Engineering, Dalian University, Dalian 116622, China*
[2]*School of Software Technology, Dalian University of Technology, Dalian 116620, China*
[3]*Department of Computer Science, Loughborough University, Loughborough LE113TU, UK*
[4]*School of Computing, Ulster University, Belfast NIC100166, UK*
[5]*School of Software Engineering, Dalian University, Dalian 116622, China*
[6]*Cloud Services Department, SPIC Digital Technology Co. LTD, Jinan 250014, China*

Correspondence should be addressed to Jing Qin; qinjing@dlu.edu.cn

Received 10 October 2021; Revised 21 August 2022; Accepted 5 December 2022; Published 25 December 2022

Academic Editor: Arijit Karati

Network intrusion detection is one of the critical techniques to enhance cybersecurity. Several few-shot learning-based methods have recently been proposed to alleviate the dependence on large training samples in many supervised learning methods. However, it is still a challenge to achieve real-time higher-accuracy intrusion detection which is an essential requirement for high-speed network security. In this study, we propose a novel few-shot learning-based network intrusion detection method to address this challenge. Specifically, we improve the detection accuracy and real-time processing speed simultaneously in the metric procedure via two mechanisms: (i) we utilize a hard sample selection scheme as a refining stage of our triplet network model training to increase the detection accuracy; and (ii) we design a lightweight embedding network and parallelize the metric feature extraction process to achieve real-time analysis speed. To evaluate the proposed method, we construct few-shot learning-based datasets by using two real and heterogeneous network traffic intrusion detection data sources. Extensive results demonstrate that our method outperforms the state-of-the-art methods in terms of real-time performance and high detection accuracy of malicious samples.

1. Introduction

Network intrusion detection plays a key role in establishing secured and reliable networks [1]. Network Intrusion Detection System (NIDS), usually a binary classification model, can effectively distinguish between abnormal attacks and normal traffic, thus ensuring the stable operation of the network. Recently, to tackle the problem of insufficient training network attack data, several few-shot learning-based detection methods, including Siamese networks [2], prototypical networks [3], and the ensemble model [4], have been proposed to deal with limited abnormal samples during training. Compared to traditional methods, such as random forest [5], deep autoencoder [6], and ResNet [7], few-shot learning-based network intrusion detection methods do not rely on a large number of samples for training and have shown better performance in solving complex and frequent types of attacks [8, 9].

Some new approaches have been recently proposed to improve the classification performance of existing few-shot learning-based models. For example, He et al. [10] used an attention mechanism to enhance feature representations extracted from a metric learning network. Xu et al. [9] introduced 3D temporal convolutional networks to exploit temporal information to improve the accuracy of NIDS. However, these methods still suffer from the reliability issue of existing metric learning models when representative data samples are not effectively selected at the training stage. In

addition, these advanced few-shot learning methods have paid little attention to detection efficiency even though real-time processing performance is essential to ensuring high-speed network security. Compared to classical deep neural network models, metric learning-based models have higher computational complexity as similarities between the testing and training samples in a supporting set are computed to extract the features for the final classification [11].

In our work, we propose a novel parallelized triplet network to improve the real-time detection performance. Specifically, the proposed method uses an initialized triplet model to select hard samples which cannot be identified in a standard triplet network to enhance the generalization ability of the model. To achieve real-time processing performance, we develop a lightweight triplet network by introducing a depthwise separable convolution algorithm and a global average pooling (GAP) mechanism to reduce the number of parameters and computational complexity. This new network architecture allows feature extraction from the triplet network to be undertaken in parallel. As such, different triplet sample pairs created from the same test sample during the detection process can be simultaneously input to the model for feature extraction so that the real-time performance can be improved. The contributions of this study are summarized as follows:

(i) We conceive a multistage few-shot training framework that utilizes an initialized triplet model to select and establish a set of representative training pairs for more effective triplet loss training

(ii) We design a lightweight deep triplet network. The efficiency of the network is improved by using depthwise separable convolution and GAP to achieve fast feature extraction and high concurrency of multiple models by reducing computational complexity

(iii) Furthermore, we develop a mechanism that parallelizes the feature extraction of the triplet model when calculating the similarities between the testing sample and samples in the supporting set so that the real-time processing performance is achieved.

The remainder of this study is organized as follows: Section 2 discusses the related work of the proposed method, and Section 3 describes the overall detection framework, including the underlying lightweight triplet network and the parallelized triplet metric model. Section 4 presents the datasets, evaluation metrics, and experiment results. Finally, the work is concluded in Section 5.

2. Related Work

2.1. Methods of Network Intrusion Detection. Many existing research works have been developed to address the reliability of NIDS. These efforts are usually made on exploiting machine learning and deep learning algorithms. A fuzzy rule-based automatic intrusion detection system [12] was proposed as a solution to deal with precise measurement and uncertainty in the judgment of each criterion. Furthermore, fuzzy TOPSIS (technique for order of preference by similarity to ideal solution) was used for response prioritization in multicriteria decision-making. Iannucci and Abdelwahed [13] proposed a probabilistic model-based intrusion detection system built on a multiagent discrete-time Markov decision process (MA-MDP), which effectively captures the dynamics of both the defended system and the attacker. Wu et al. [14] proposed an intrusion detection method by using a convolutional neural network. This method converted the vector format of the original data into an image format and CNNs were applied to extract image features to detect intrusions.

A reliable intrusion detection system needs to consider both detection accuracy and efficiency. Ambusaidi et al. [15] proposed a mutual information-based algorithm that analytically selects the optimal feature for classification. This mutual information-based feature selection algorithm can handle linearly and nonlinearly dependent data features and irrelevant features from the original data. The work proposed by Selvakumar and Muneeswaran [16] deployed a filter and wrapper-based approach where the firefly algorithm was used in the wrapper to search for the best subset of features. The SwiftIDS approach proposed by Jin et al. [17] achieved the abovementioned objectives in two ways. One way was to simplify data preprocessing by using LightGBM support for classification features. The other way was to analyze the traffic data arriving in different time windows through a parallel intrusion detection mechanism. In these ways, the delay caused by the later-arriving data waiting for the end of the intrusion detection cycle of the first-arriving data can be avoided. Due to the devices of WSNs that do not have powerful processing performance due to power limitations, Zhao et al. [18] proposed a lightweight dynamic autoencoder network method for NID, which realizes efficient feature extraction through lightweight structure design. Furthermore, they proposed a novel NID method [19] for IoT based on a lightweight deep neural network. To avoid high-dimensional raw traffic features leading to high model complexity, the method used the PCA algorithm to achieve feature dimensionality reduction. Besides, the classifier used the expansion and compression structure, the inverse residual structure, and the channel shuffle operation to achieve effective feature extraction with low computational cost.

While significant progress has been made, these existing state-of-the-art methods still face challenges. For example, simplifying preprocessing methods for data or dimensionality reduction of data can reduce unnecessary computations, but the computational complexity of the algorithms has not been explored much. Second, these methods do not pay much attention to the scarcity of attack samples in real networks.

2.2. Few-Shot Intrusion Detection. Few-shot learning models have been proposed to address the tasks with a limited number of training samples [8]. For example, the prototypical networks [20] feature the same embedding function for the support and query sets, turning the classification problem into the nearest neighbour problem in the

embedding space. The relational networks [21] are constructed by constructing neural networks to calculate the distance between two samples and thus analyze the degree of matching. The matching networks [22] are characterized by two different embedding functions for the support and query sets, with the output of the classifier being a weighted sum of the predicted values between the support set samples and the query set samples. The Siamese networks [23] are trained by constructing pairs of samples as input to the twin structure by random combination and calculating the distance between the pairs to measure similarity. Yu and Bian [3] exploited a deep convolutional neural network algorithm that was integrated into the metric learning network to calculate the Euclidean distances of different samples to further distinguish between normal traffic samples and attack traffic samples. Ouyang et al. [24] used orchestrating one-hot encoding and principal component analysis for data preprocessing and built a complete FS-IDS model by further training of the preliminary IDS model. He et al. [25] proposed a few-shot detection method based on CNN and autoencoder. The method used some anomalous samples to build a structure for extracting deep features and selected the features of normal samples as training data. The method also introduced an attention mechanism to improve detection accuracy. Similarly, Zhou et al. [2] constructed a Siamese CNN coding network to measure the distance of input samples based on their optimized feature representations and proposed a robust cost function including three specific losses to improve the training efficiency. As an advanced detection method, Xu et al. [9] further processed traffic data from spatial and temporal features. This method combined temporally neighbouring samples in the same connection into a spatial three-channel image and constructed Siamese networks to detect image-based intrusion events using Conv3D convolution operation.

Obviously, the deep learning algorithm still occupies a vital part of the few-shot learning method. Unfortunately, these few-shot intrusion detection methods pay little attention to the computational burden that the metric procedure brings to the final decision. Complex feature extraction networks and similarity metric mechanisms still restrict the real-time processing capability of these methods. In addition, as an important barrier to the network security operation, the abovementioned methods are still inadequate in terms of accuracy and need to be further improved.

2.3. Triple Network. The triplet network, which is derived from Siamese networks, consists of three embedded networks with shared parameters [26]. It has been widely used in metric learning tasks [27–29]. Nguyen et al. [30] automatically learned motion patterns from small image blocks by training a triplet network for change detection based on motion features. Abdullah et al. [31] proposed a bidirectional triplet network to match text with remote-sensing images. The network consisted of a long short-term memory network (LSTM) and CNNs (based on EfficientNet-B2, ResNet-50, Inception-v3, and VGG16) and used averaging fusion strategy to fuse features associated with five image sentences to achieve a more robust embedding. Ji et al. [32] proposed a dual triplet network for image zero-shot learning. The method projected semantic information into visual space by using a mapping network and then learned visual semantic mappings via two triplet networks. In this method, one triplet network focused on negative attribute features and the other triplet network paid attention on negative visual features to ensure that the data information is fully utilized. Gao et al. [33] proposed a new heterogeneous information network embedding algorithm. In the data sampling phase, a metaschema-based random walk was performed to extract semihard quadruplets based on the node type and its degree. In the representation learning phase, a relational triplet loss is designed to optimize the distance of triplet embedding on diverse heterogeneous relationships.

With the widespread use of triplet networks, some improved approaches focused on the unique data format required for algorithm training as the random selection of triplet sample pairs could lead to uneven distribution of the data and unstable performance in the model training process. Schroff et al. [34] proposed a method to select triplet sample pairs by defining three types of samples including easy triplets, semihard triplets, and hard triplets, and he pointed out that the generalization ability of the model will be limited if only easy triplets were used, while semihard and hard samples were more effective to train the network. Hermans et al. [35] further proposed an improved version of triplet sample pair generation with online-batch hard sample mining.

Compared with the abovementioned networks, the triplet network shows greater advantages but still lacks lightness and stability. The parallelized detection structure and lightweight embedding network can cope with large-scale traffic in more real time. Equally, more stable prediction results and a more effective intrusion detection system can be obtained by further exploiting the potential features of the training data.

3. Proposed Method

Despite the significant advantages of the triplet network compared to other metric learning networks, it still suffers from two weaknesses: (1) similar to other metric learning methods, it has a high computational burden at the feature extraction stage. Consequently, it is not suitable for those applications which have real-time processing requirements and (2) the convergence of the network is sensitive to training pair samples used in the triplet loss calculation. In this study, we propose a parallelized lightweight triplet network to achieve real-time intrusion detection performance. Further, we introduce a paired sample selection stage to extract more representative pairs for the network training. As a result, our method achieves high detection accuracy with a real-time processing speed.

3.1. The Architecture of the Few-Shot Learning-Based Intrusion Detection. Figure 1 depicts the architecture of building our few-shot learning intrusion detection by using the designed

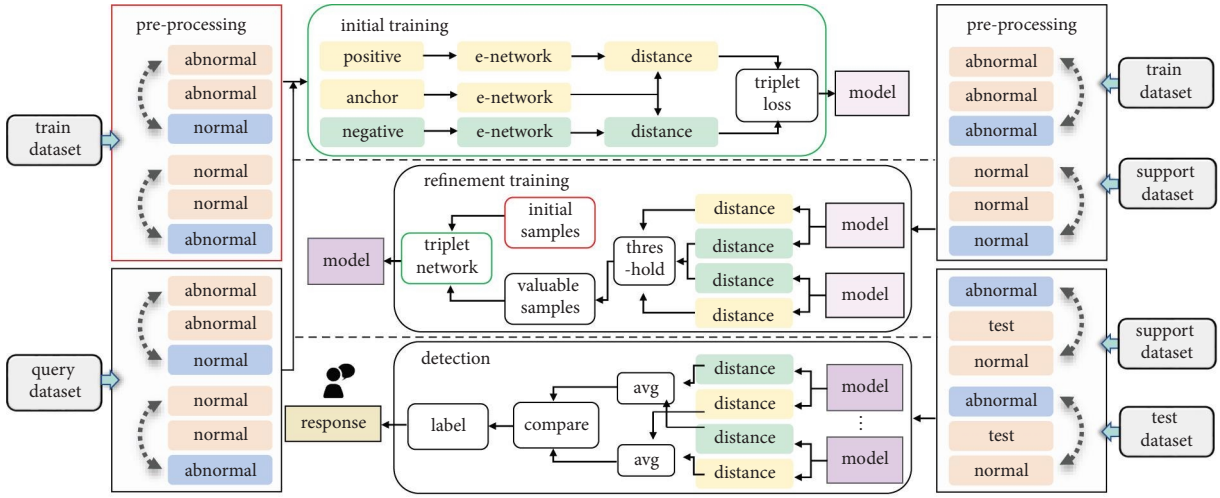

FIGURE 1: The architecture of building few-shot intrusion detection using the triplet network. The model with the same background colour in the diagram is functionally identical. For example, the refinement training module inherits the model trained by the initial training module and transfers that model to the detection module. The ellipsis between models in the detection module indicates where the model can be deployed in parallel.

lightweight parallelized triplet network. The architecture mainly includes four modules, that is, preprocessing stage, initial training stage, refinement training, and detection.

The preprocessing module is to process the few-shot training set, the query set, the support set, and the test set and to create the corresponding input sample pairs according to the position relationships of anchor, positive, and negative, which are described in detail in Section 3.2. The training set is used to train the triplet model; the query set is used to verify the performance of the model in the training phase; the support set is used as a metric for refinement training and detection; and the test set is used to test the effect of the model.

The initial training module is to train a preliminary detection model on the few-shot training set by means of a triplet network, which contains the embedding network and triplet loss. The embedding network takes on the task of feature vector extraction, and it performs feature vector extraction through a deep CNN network with shared weights to achieve a lightweight detection model while avoiding overfitting. The triplet loss mainly undertakes the task of loss calculation in the training process, which metrics the feature vectors output from the embedded network, calculates the distance between the feature vectors, and then updates the network weights.

The refinement training module is to further improve the generalization ability of the model by selecting more representative and hard pairs for training. The selected samples are captured via the initial model to metric the samples in the training set where the hard samples are chosen as valuable samples according to a certain ratio. These samples are used as training samples to obtain the refined detection model.

The detection module is to obtain the labels of the testing samples by using the similarity measurements of the testing sample with samples in the supporting set. A parallel process structure is designed to measure the similarities for real-time performance.

In summary, the method obtains a more accurate model through double training in the training phase and achieves the detection of test samples in the detection phase through the parallel deployment of multiple models. In terms of deployment, the method is a concept of multiple triplet networks combined to provide a parallel network structure to enable feature extraction and real-time detection from different traffic samples. First, in the training phase, the network relies on a small number of samples to obtain an initial detection model and then refines the training to select more representative samples to improve the model's ability to detect malicious traffic. Second, in the detection phase, improved and refined detection models are distributed to edge computing devices for parallelized deployment. This deployment strategy enables real-time detection of online traffic through a lightweight detection model and responds to anomalous traffic for future model refinement.

3.2. The Lightweight Enhanced Triplet Network for Reducing Computational Complexity. The proposed triple network consists of three embedding networks with shared weights, which takes three inputs with an anchor sample, a positive sample, and a negative sample. The parameters of the embedding network are updated by using the backpropagation algorithm. Specifically, during the training process, the triple network randomly selects a sample from the training set as the anchor. Then, a sample of the same type as anchor is randomly selected for positive and a sample of a different type for negative, forming a triplet sample pair (positive, anchor, and negative) as the input to the embedding network. Finally, the Euclidean distances of positive sample pairs (anchor and positive) and negative sample pairs (anchor and negative) are calculated separately based on the feature vectors output from the embedded network to further calculate the loss. As shown in Figure 2, triplet samples are input to the embedding network. The

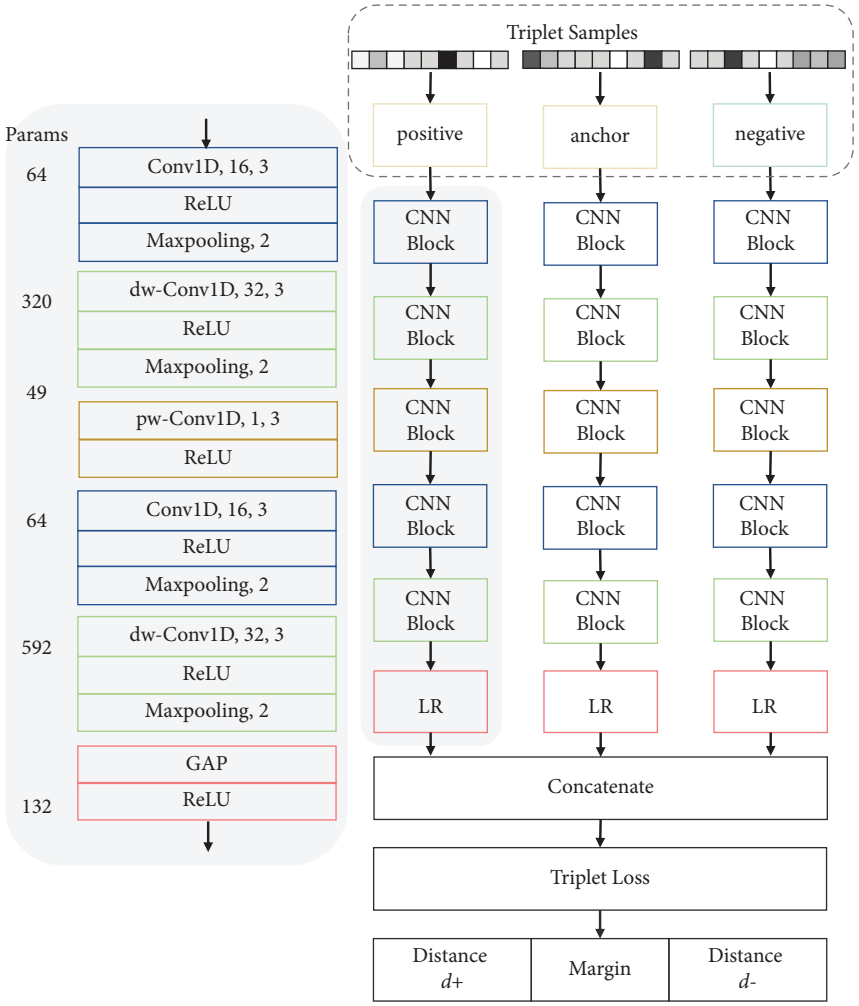

Figure 2: The structure of lightweight triplet network.

embedding obtains the sample features by stacking multiple layers of convolutional pooling operations. The purpose of training is to make the distance between the anchor and the positive as small as possible and the distance between the anchor and the negative as large as possible, i.e., by updating the network weights to make the anchor closer to the positive and away from the negative. Therefore, triplet loss is calculated as shown in the following formula:

$$\text{Loss}(d_+, d_-) = \left\|(d_+, d_- - 1)\right\|_2^2, \tag{1}$$

where d_+ and d_- are the distances of the (anchor and positive) feature vectors and the (anchor and negative) feature vectors, respectively. The specific calculation is shown in the following equations:

$$d_+ = \frac{e^{\|\text{Net}(a)-\text{Net}(p)\|_2}}{e^{\|\text{Net}(a)-\text{Net}(p)\|_2} + e^{\|\text{Net}(a)-\text{Net}(n)\|_2}}, \tag{2}$$

$$d_- = \frac{e^{\|\text{Net}(a)-\text{Net}(n)\|_2}}{e^{\|\text{Net}(a)-\text{Net}(p)\|_2} + e^{\|\text{Net}(a)-\text{Net}(n)\|_2}}. \tag{3}$$

The ability to address large-scale traffic is a very important metric for intrusion detection methods, which is closely related to the computational and parametric quantities of the detection model. Therefore, we used depthwise separable convolution [36] and GAP [37] to implement a lightweight deep embedding network.

Typically, CNNs need to define multiple filters to enrich the underlying features, which allow one-dimensional traffic samples to evolve into multichannel feature maps after convolution. Different from the conventional convolution operation, depthwise separable convolution contains two parts: depthwise convolution (dw-conv) and pointwise convolution (pw-conv). Among them, each filter of dw-conv is responsible for convolving one channel. Pw-conv operation is equivalent to a weighted fusion of multiple feature maps from the previous layer. As shown in Figure 3, dw-convc initializes three convolutional kernels for extracting three different feature maps, respectively. Pw-conv fuses three 1D feature mappings into a new feature mapping through a $1 * 3$ filter, although this fusion inevitably losses some features. However, as shown in Figure 2, we add this operation before the underlying features enter the higher-level convolution, achieving a reduction in computational complexity while barely affecting the effectiveness of feature extraction.

In addition, the embedding network inputs the one-dimensional vectorization (Flatten) of the feature map output from the last convolutional block to the fully connected layer. GAP was proposed to replace fully connected layers to average the output feature maps of the last convolutional layer for compact and efficient feature representations. As shown in Figure 2, the output of the last convolutional layer of each embedding is made as the input of the GAP, and the output feature vectors are aggregated through a concatenate layer after being activated. With the abovementioned two improvements, the number of parameters and computation of the embedded network are greatly reduced. The architecture avoids overfitting issues to improve the reliability of our model.

3.3. Parallelized Triplet Model

3.3.1. Parallelized Triplet Model for Refinement Training. The effectiveness of the model is closely related to the quality of the training samples. Compared to easy samples, hard sample pairs provide more valuable distinctive power in the training of the network. By using these valuable samples, the generalization ability of the model can be improved.

However, the valuable samples are difficult to select due to a large number of combinations. As shown in Figure 4, during the training process, not only the similarity between samples can be obtained by the (anchor, positive) pairs but also the dissimilarity between samples can be obtained through the (anchor, negative) pairs. As described in Section 3.2, the resulting model is trained to maximize distances between different types of input sample pairs and minimize distances between the same types of pairs. In our work, we select a certain proportion of positive and negative samples as sample pairs according to the distance. The threshold for valuable sample selection is shown in the following formula:.

$$\text{thresold} = \{\max(\text{distance}) - \min(\text{distance})\} * (1 - \sigma). \quad (4)$$

When the distance of a sample pair is higher than a defined threshold on distance, we select this pair into the training set for the refinement stage. The σ in formula (4) as a hyperparameter represents the percentage of the valuable samples to be selected in terms of distance, which we usually set to $\sigma = 25\%$. After selecting the hard samples, we use them to construct (anchor, positive, and negative) sample pairs by random selection to add to the initial training set to obtain the final detection model.

3.3.2. Parallelized Triplet Model for Detection. Because the model obtained by the triplet network is more suitable for unsupervised classification tasks and cannot directly give the labels of the tested samples, we build a few-shot intrusion detection framework that can perform metric learning through a parallel structure. The training process of the model contains a measure of the (anchor and positive) part, which can effectively capture the intraclass similarity. In the detection phase, different models with the same weights are able to simultaneously obtain the distance between the test samples and the positive and negative samples as a similarity. The average similarities are then compared to determine the label of the test sample.

In addition, an extremely important constraint of few-shot learning in intrusion detection is the ability to handle large-scale traffic data. Since the determination of the label in the detection process relies on the similarities between the test sample and samples in the support set, it is a computationally intensive process to calculate the similarities across all pairs. Therefore, we build a parallel intrusion detection mechanism using the metric model obtained from the training to reduce the computational complexity. As shown in Figure 5, the parallel triplet network model can simultaneously process multiple input sample pairs established by the test and the support samples to obtain the similarity values simultaneously as a feature representation for the detection. Specifically, a single detection model can extract features on two input samples at a time to obtain the Euclidean distances between the test sample and different types of support samples. Thus, parallelized deployment of multiple models allows simultaneous comparison of a given test sample with different support samples, obtaining the distance of the test sample from the positive sample set as well as the negative sample set and then, by comparing these distances, obtaining the labels of the test samples. The lightweight model can achieve the high concurrency required for parallelism, meets the limitations of storage and computational units on parallel processing, and enables fast detection of large-scale traffic.

4. Experiments, Evaluation, and Discussion

In this section, we conduct experiments on two benchmark datasets to verify the advantages of the proposed method on few-shot intrusion detection. We first introduce the datasets and the experimental environment, then we show the advantages of the method by comparing other algorithms, and finally we choose the state-of-the-art few-shot intrusion detection method to compare with the proposed method under the same experimental conditions. Through the experiments, we can obtain answers to the following questions:

(i) Whether our proposed method has advantages in few-shot scenarios compared to existing general intrusion detection methods?

(ii) Whether our proposed method has an improvement in detection accuracy compared to advanced few-shot intrusion detection methods?

(iii) How does our proposed method perform in terms of detection efficiency compared to advanced few-shot intrusion detection methods?

4.1. Experiment Setup

4.1.1. Description of the Benchmark Datasets. To demonstrate the effectiveness of the proposed approach, we conduct experiments on two publicly available and heterogeneous intrusion detection datasets.

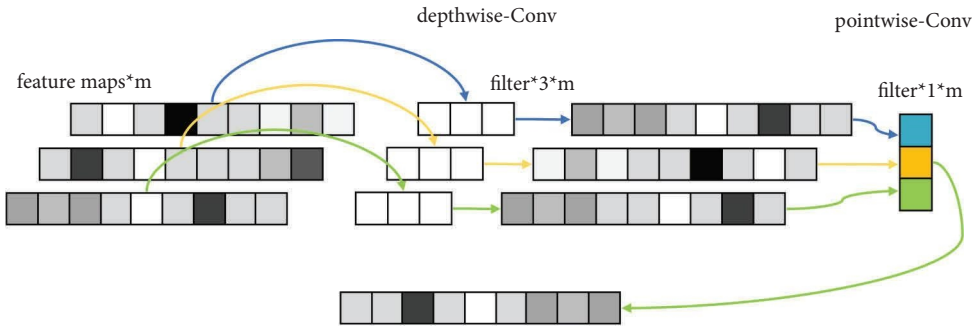

Figure 3: Depthwise separable convolution for feature extraction of 1D samples.

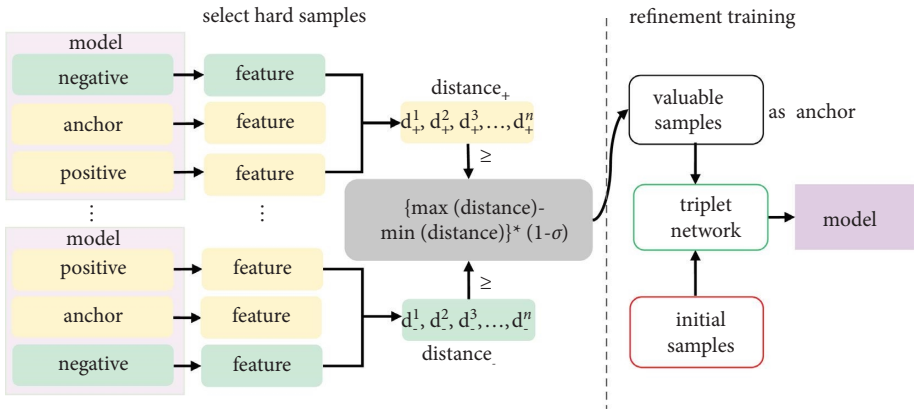

Figure 4: Flowchart of refinement training.

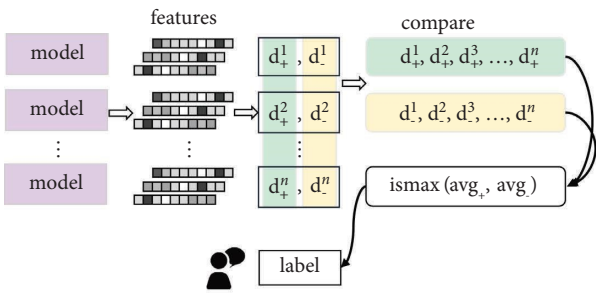

Figure 5: Mechanism of parallel triplet model for detection.

CICIDS-2017 dataset [38]: the dataset was released by the Canadian Institute for Cyber Security (CIC) in 2017 and contains normal samples and 14 of the latest common attacks such as distributed denial of service, SQL injection, and port scanning. Specifically, the dataset contains 2,830,743 samples on continuous time from Monday to Friday. Each sample included 78 different features. We selected 220 of them as the few-shot training set for training and validated the detection effect on 20,710 test sets. The label distribution of the few-shot dataset is shown in Table 1.

In the training set, the relationship between the number of normal samples and abnormal samples is 1 : 1, and the number of samples available for training is 10 for each abnormality type, which satisfies the requirement of few-shot learning. The test set contains 10,000 normal samples and 10,710 abnormal samples, which can fully examine the detection accuracy of the method. Among them, the number of Heartbleed, Infiltration and Web Attack-Sql Injection samples is relatively few. As most advanced NIDS classifiers based on deep learning are less sensitive to unknown attacks, and traffic samples for emerging forms of attacks such as "zero-day" attacks are difficult to obtain, it is important to identify unknown attacks during the detection process [8]. Therefore, we simulate these three types of samples as unknown attacks and include them in the test set only.

UNSW_NB15 dataset [39]: this dataset is the latest intrusion detection dataset created by the Australian Centre for Cyber Security (ACCS). It was generated from approximately 2,540,047 samples of network traffic through the IXIA PerfectStorm. Because the original dataset is too large, the agency also released a dataset containing 82,332 samples for evaluation, and our experiments were launched on this dataset. The dataset covers 9 different attack types such as Backdoor, Denial of Service, Reconnaissance, Shellcode, and Worms. Each of these samples contains 49 features. Since the variety of attacks became less, we used 140 of these samples as a few-shot training set and 16,662 of them as a test set.

Similarly, the ratio of normal to abnormal samples in the few-shot training set created on this dataset is 1 : 1, and the number of samples available for training in each abnormality type is 10. As shown in Table 2, 10,000 normal and 6,662 abnormal samples are included in the test set, and unknown attacks are simulated using Worms and Shellcode types which are not included in the training set.

Table 1: Label distribution of few-shot datasets based on the CICIDS-2017 dataset.

Describe	Type	Train	Test
Normal traffic	Benign	110	10000
Known attack traffic	FTP-Patator	10	1000
	SSH-Patator	10	1000
	DoS Hulk	10	1000
	DoS GoldenEye	10	1000
	DoS Slowloris	10	1000
	DoS Slowhttptest	10	1000
	Web Attack-Brute Force	10	1000
	Web Attack-XSS	10	642
	Bot	10	1000
	DDoS	10	1000
	Port Scan	10	1000
Unknown attack traffic	Heartbleed	0	11
	Infiltration	0	36
	Web Attack-Sql Injection	0	21
—	Total	220	20710

Table 2: Label distribution of few-shot datasets based on the UNSW_NB15.

Describe	Type	Train	Test
Normal traffic	Normal	70	10000
Known attack traffic	Reconnaissance	10	1000
	Backdoor	10	573
	DoS	10	1000
	Exploits	10	1000
	Analysis	10	667
	Fuzzers	10	1000
	Generic	10	1000
Unknown attack traffic	Worms	0	44
	Shellcode	0	378
—	Total	140	16,662

4.1.2. Implementation and Metrics. The few-shot intrusion detection in the study is a process of binary classification of traffic samples, and the detection results are classified into four types as follows:

(i) TP: attack samples are correctly detected as attack samples

(ii) FN: attack samples are incorrectly classified as normal samples

(iii) TN: normal samples are correctly detected as normal samples

(iv) FP: normal samples are incorrectly classified as attack samples

We use precision, recall, and F-measure to validate the detection performance of the method in the study. The precision is the proportion of real positive samples in the samples that are judged to be positive and can be expressed by the formula, precision = TP/TP + FP. The recall refers to the proportion of samples that are judged to be positive in all samples that are truly positive and can be expressed by the formula, recall = TP/TP + FN. F-measure is the summed average of precision and recall, and the formula is $F-\text{measure} = 2 * \text{precision} * \text{recall}/\text{precision} + \text{recall}$.

In addition, we used the number of parameters, floating points of operations (FLOPs), and detection time to measure the performance of the method in terms of operational performance. The number of parameters in the embedded network is related to the size of the kernels and the number of input and output channels and can measure the usage of computational resources such as memory during model training and detection. FLOPs are the number of multiplication and addition operations in the model and are used to measure the computational complexity of the model. Since there are some hard-to-measure factors in the detection process, they can be judged by the actual time consumption. Therefore, the intrusion detection time for large-scale samples is also a very important evaluation metric.

To complete the testing of the method, the training experiments were performed on a CPU Intel Xeon E5-2620, GPU NVIDA GTX1080ti, 64 GB of RAM, 11 GB of video memory, CuDNN 7.6.5, CUDA 11.0, Tensorflow 1.13.1, and Keras 2.2.4.

4.2. Refinement Training Results. To compare the improvement of the detection effect by refinement training, we conducted experiments on a few-shot training set. After obtaining the initial model, we set the distance proportion of the valuable samples to 25%. The distribution of the valuable samples obtained after refinement training is shown in Figure 6.

Although valuable samples occupy 25% of the distribution distance, they only account for a very minor proportion of the number of samples. In order to better fit these samples, we added the valuable samples to the training set in the post-training phase and performed refinement training. Compared with the initial model, the improvement in detection accuracy of the refinement training model is shown in Table 3.

From the two different datasets, the models obtained after refinement training showed a significant improvement in detection accuracy. On the CICIDS-2017 dataset, the model obtained after refinement training improves nearly 3 percentage points in precision rate and maintains the lead in the overall evaluation metric F-measure despite the slight decrease in recall rate. On the UNSW_NB15 dataset, the refinement trained model improved significantly in all three evaluation criteria, including precision, recall, and F-measure. While the number of misclassified normal samples decreased from 719 to 329, the number of correctly detected abnormal samples improved by 131. Therefore, the refinement training under the parallel model structure has a significant improvement on the few-shot intrusion detection method.

4.3. Comparison and Analysis with Other Classifiers in Few-Shot Scenarios

4.3.1. Overall Performance. The merit of an intrusion detection system is evaluated based on its ability to correctly classify traffic samples. To evaluate the classification performance of our proposed method on a few-shot training set, we use three widely used classifiers, including the deep learning algorithms CNN [40], ResNet [41], and XGBoost

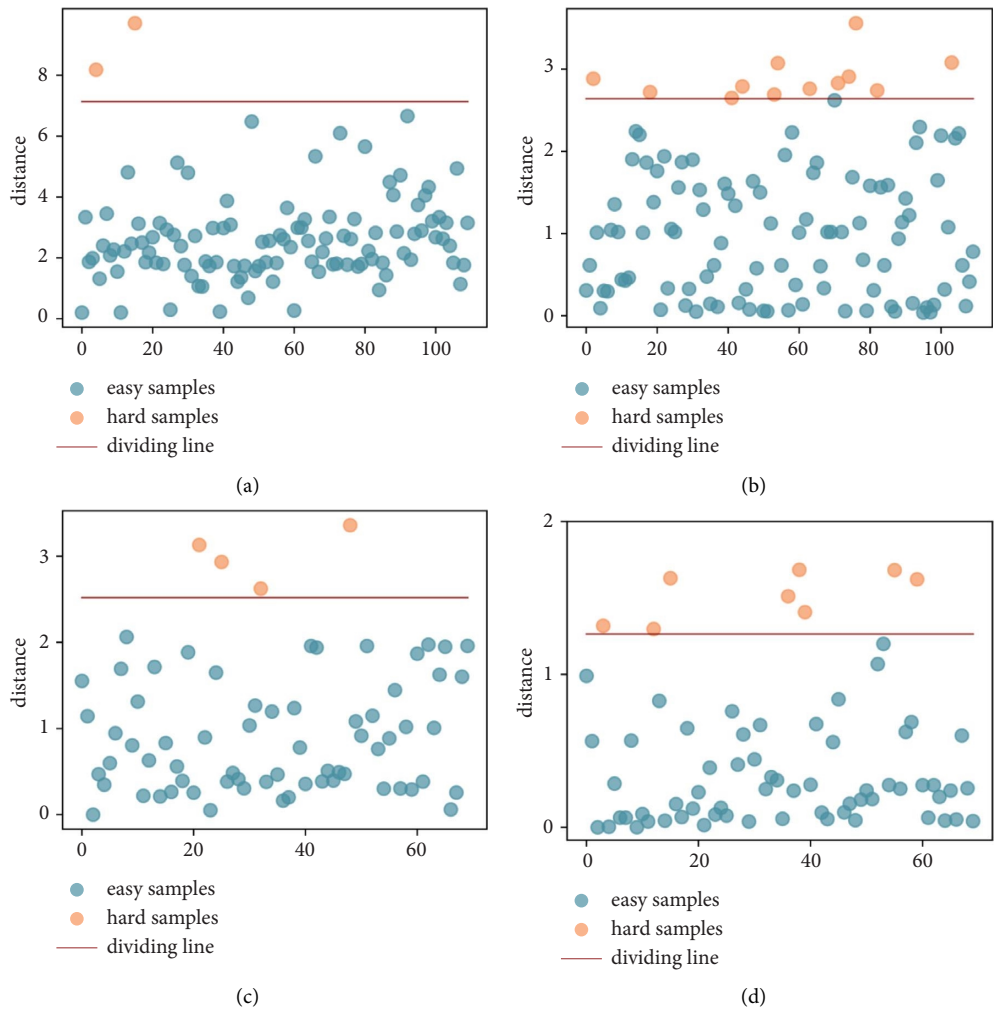

Figure 6: Visualization of valuable sample distribution. (a) Normal samples on the CICIDS-2017. (b) Abnormal samples on the CICIDS-2017. (c) Normal samples on the UNSW-NB15. (d) Abnormal samples on the UNSW-NB15.

Table 3: Refinement training for detection performance improvement.

Dataset	Method	TP	FP	TN	FN	Precision	Recall	F-measure
CICIDS-2017	Initial training	10,055	655	**9,466**	**534**	93.88	**94.96**	94.42
	Refinement training	**10,301**	**409**	9,409	591	**96.18**	94.57	**95.37**
UNSW_NB15	Initial training	6,444	218	9,081	719	96.73	89.96	93.22
	Refinement training	**6,575**	**187**	**9,671**	**329**	**97.23**	**95.23**	**96.22**

The bold values represent the optimal values for the different evaluation criteria in the comparison experiment. For example, Refinement training achieved the best value under the evaluation criterion TP.

[7]. With sufficient training data, all three algorithms can achieve high classification accuracy. However, as shown in Table 4, in scenarios where the number of samples is scarce, both the single deep learning algorithm and the machine learning algorithm produce many misclassifications for both normal and abnormal samples. Obviously, in scenarios with very few sample sizes, machine learning algorithms and deep learning algorithms are prone to overfitting due to the lack of training samples. The few-shot learning method is not limited by the number of samples, and the final result is determined by metric with the support set of samples after feature extraction, with fewer false positives compared to other algorithms for normal samples. In contrast, the tripletnet-based classifier achieved the best performance in terms of all the evaluation metrics.

Figure 7 shows the detection performance of the four classifiers in terms of precision, recall, and comprehensive evaluation metric F-measure. The proposed method accomplishes the successful detection of more than 10,000 samples using only a minority of samples for learning. Among them, on the CICIDS-2017 dataset, the detection rate of the algorithm for attack traffic is 96.18%, which is higher than CNN by 1 percentage point. Among them, the detection rate of the method on the CICIDS-2017 dataset is

Table 4: The performance results based on different datasets.

Dataset	Algorithm	TP	FP	TN	FN
CICIDS-2017	CNN	10,189	521	9,085	915
	ResNet	9,893	817	8,817	1,183
	XGBoost	10,132	578	9,252	748
	Proposed	**10,301**	**409**	**9,409**	**591**
UNSW_NB15	CNN	5,823	839	9,391	709
	ResNet	5,605	1,057	9,281	719
	XGBoost	6,154	508	8,243	1757
	Proposed	**6,475**	**187**	**9,671**	**329**

The bold values represent the optimal values for the different evaluation criteria in the comparison experiment. For example, proposed method achieved the best value under the evaluation criterion TP on CICIDS-2017 dataset.

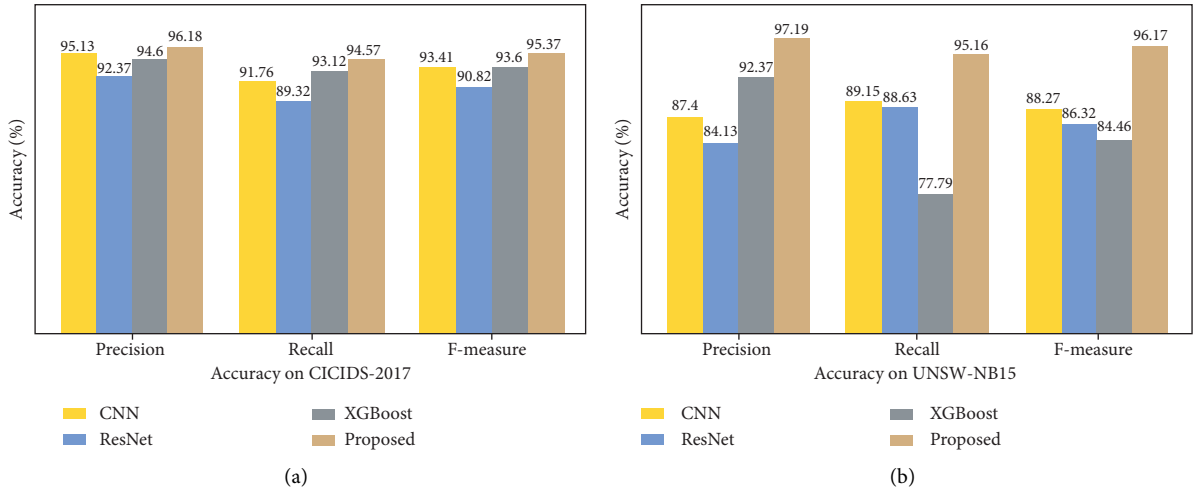

Figure 7: Comparison performance with others classifiers based on the different datasets.

96.18% for attack traffic, which is 1 percentage point higher than that of CNN. Although this advantage does not pull away from the CNN algorithm, the proposed method is better overall when combined with the recall rate and F-Measure. CNN shows more false positives in the detection of normal samples. In addition, the proposed method has an all-round advantage over the other three classification models on the UNSW_NB15 dataset. The structure of the triplet metric model makes it possible to achieve a detection rate of 97.19% for abnormal samples, which is more than 10 percentage points ahead compared to deep learning algorithms such as CNN and ResNet. The above can fully illustrate that the method in the study is capable of the detection task in the case of extreme scarcity of samples.

4.3.2. Comparison of Detection Results under Different Attack Types. Table 5 shows the detection results of different classifiers for 14 attack types on the CICIDS-2017 dataset. Among them, the first 11 attack types appear in the training set. The last 3 attack types are not trained and are used to simulate unknown attacks. It can be clearly seen that the XGBoost algorithm is better at detecting attack samples compared to the deep learning algorithm. This indicates that the general deep learning algorithm is not suitable for the training task in the sample scarcity scenario. Due to the scarcity of samples, deep learning algorithms are highly prone to overfitting, which affects the classification accuracy. However, the detection rate of the XGBoost algorithm for samples is not stable, for example, it is extremely poor for known attacks such as FTP-Patator, Port Scan, Unknown Attacks, Heartbleed, and Infiltration. The proposed method outperforms the other three algorithms in the detection of multiple attack types. Although the advantage is not very significant, there is a stable detection for all known attack types. In addition, for the detection of unknown attacks, our proposed method is still able to achieve better detection accuracy using the capability of metric learning.

In contrast, the experimental results on the UNSW_NB15 dataset fully demonstrate the advantages of the method in the study. The proposed method leads on seven out of the nine attack types shown in Table 6. The detection results of XGBoost outperform the other two deep algorithms. However, combined with the recall shown in Figure 7, the advantage of XGBoost is based on a large number of false positives for normal samples.

4.4. Comparison and Analysis with the State-of-the-Art Methods

4.4.1. Comparison of Detection Accuracy. To demonstrate the detection performance and efficiency of the proposed

Table 5: Detection results for different attack types on the CICIDS-2017 dataset.

Type	CNN	ResNet	XGBoost	Proposed
FTP-Patator	99.80	88.17	81.50	99.50
SSH-Patator	97.90	98.63	98.30	**98.70**
DoS Hulk	95.10	96.87	**99.70**	95.50
DoS GoldenEye	94.90	79.30	**98.90**	95.20
DoS Slowloris	95.80	97.30	**99.00**	96.10
DoS Slowhttptest	93.90	95.10	96.80	**97.90**
Web Attack-Brute Force	99.60	96.90	97.40	**99.70**
Web Attack-XSS	94.23	93.30	**99.07**	97.35
Bot	88.10	**97.66**	92.30	**97.70**
DDoS	93.40	92.20	**99.60**	91.30
Port Scan	97.60	82.40	83.60	91.20
Heartbleed	18.18	**100.00**	27.27	**100.00**
Infiltration	11.11	8.00	8.33	**47.22**
Web Attack-Sql Injection	80.95	57.14	85.71	**90.48**

The values in bold represent the optimal values for the different methods in the comparison experiments.

Table 6: The detection results for different attack types on the UNSW_NB15 dataset.

Type	CNN	ResNet	XGBoost	Proposed
Reconnaissance	77.70	50.7	98.5	95.1
Backdoor	98.08	55.1	97.9	99.65
DoS	93.30	91.1	96.8	97.3
Exploits	88.50	90.6	91.9	96.3
Analysis	97.90	98.80	93.55	99.40
Fuzzers	73.30	86.4	71.2	97.10
Generic	99.3	99.6	99.5	99.7
Worms	70.45	72.72	70.45	100
Shellcode	67.72	47.35	94.9	90.48

Table 7: Comparison performance with few-shot method based on the two datasets.

Dataset	Method	TP	FP	TN	FN	Precision	Recall	F-measure
CICIDS-2017	FC-net	10,022	688	**9,622**	**378**	93.58	**96.37**	94.95
	Proposed	**10,301**	**409**	9,409	591	**96.18**	94.57	**95.37**
UNSW_NB15	FC-net	6,192	486	9,528	422	92.97	93.62	93.30
	Proposed	**6,575**	**187**	**9,671**	**329**	**97.23**	**95.23**	**96.22**

method, we use the existing state-of-the-art method as a comparison. To ensure reliable experimental results, the algorithm model in the FC-Net method is built as described in Section 4 and paragraph B of the original study [9], and the consistency of the dataset is maintained.

As shown in Table 7, compared to FC-Net, the proposed method maintains the leading position on different datasets. From the perspective of detection of attack samples, the few-shot learning method established using the triplet network can reach 96.18% and 97.23% on the two datasets, respectively, which can effectively detect the network traffic of the attack, exceeding the few-shot learning method established on the Siamese network structure. From the perspective of the detection of normal samples, FC-Net has a lower false alarm than our method on the CICIDS-2017 dataset for normal samples, but still has a higher false alarm on another dataset. It cannot be considered as an advantage of the method. Therefore, on the whole, our algorithm surpasses the advanced method FC-Net.

Table 8: Comparison performance with few-shot method based on the two datasets.

Method	Input-dim	FLOPs	Parameter	Detection time (s)
FC-Net	(3,9,9.3)	1,620,656	806,273	191
Proposed	78	**2,281**	**1,221**	**1.8**
FC-Net	(3,7,7,3)	1,608,368	800,129	188
Proposed	43	**2,281**	**1,221**	**1.5**

The values in bold represent the optimal values for the different methods in the comparison experiments.

4.4.2. Comparison of Operational Performance. Intrusion detection requires real-time performance which means it could effectively cope with large-scale network traffic. Constrained by the embedded network and the procedure, FC-Net performs poorly in detecting real-time process. As shown in Table 8, the number of parameters of the 3D-ConV model built by FC-Net reached 806,273, which means that the detection model consumes great memory resources

when running and it is difficult to address large-scale data simultaneously. In terms of FLOPs, the computational complexity of FC-Net's j is enormous. In contrast, our method is only 0.1% of the FC-Net method in terms of computation and number of parameters. Moreover, in terms of the detection time of the two methods for different test sets of samples, the proposed method accomplishes the classification task for 20,710 and 16,662 test samples in 1.8 s and 1.5 s, respectively, on a dual detection framework with higher scalability. Thus, our approach has the capability to address large-scale network traffic in real time.

5. Conclusions

In this study, we proposed a few-shot intrusion detection framework based on triplet network. The method implements staged learning and parallelized detection through the proposed lightweight model with high concurrency. Experimental results demonstrate the significant advantages of the method in terms of detection performance and efficiency in the case of a limited number of samples. High availability is a constant topic in the field of intrusion detection. In future research, we will further explore the application of a meta-learning framework in real-time intrusion detection.

Data Availability

The datasets used in the article are the two public datasets, CICIDS-2017 and UNSW_NB15. They can be downloaded from the following website links: https://www.unb.ca/cic/datasets/ids-2017.html and https://research.unsw.edu.au/projects/unsw-nb15-dataset.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Youth Fund Project of the National Nature Fund of China under grant no. 62002038.

References

[1] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019.

[2] X. Zhou, W. Liang, S. Shimizu, J. Ma, and Q. Jin, "Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5790–5798, 2021.

[3] Y. Yu and N. Bian, "An intrusion detection method using few-shot learning," *IEEE Access*, vol. 8, Article ID 49730, 2020.

[4] M. M. U. Chowdhury, F. Hammond, G. Konowicz, C. Xin, H. Wu, and J. Li, "A few-shot deep learning approach for improved intrusion detection," in *Proceedings of the 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, IEEE, New York, NY, USA, October 2017.

[5] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems," *ACM Computing Surveys*, vol. 51, no. 3, pp. 1–36, 2018.

[6] XuK. Li, W. Chen, Q. Zhang, and L. Wu, "Building auto-encoder intrusion detection system based on random forest feature selection," *Computers & Security*, vol. 95, Article ID 101851, 2020.

[7] J. Song, B. Li, Y. Wu, Y. Shi, and A. Li, "ReAL: a new ResNet-ALSTM based intrusion detection system for the internet of energy," in *Proceedings of the 2020 IEEE 45th Conference on Local Computer Networks (LCN)*, IEEE, Sydney, Australia, November 2020.

[8] Zu-M. Wang, Ji-Yu Tian, J. Qin, H. Fang, and L. M. Chen, "A few-shot learning-based siamese capsule network for intrusion detection with imbalanced training data," *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 7126913, 17 pages, 2021.

[9] C. Xu, J. Shen, and X. Du, "A method of few-shot network intrusion detection based on meta-learning framework," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3540–3552, 2020.

[10] A. He, C. Luo, X. Tian, and W. Zeng, "A twofold siamese network for real-time object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.

[11] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu, "Structured siamese network for real-time visual tracking," in *Proceedings of the European conference on computer vision*, (ECCV), Munich, Germany, September 2018.

[12] D. K. Singh and P. Kaushik, "Framework for fuzzy rule based automatic intrusion response selection system (frairss) using fuzzy analytic hierarchy process and fuzzy topsis," *Journal of Intelligent and Fuzzy Systems*, vol. 35, no. 2, pp. 2559–2571, 2018.

[13] S. Iannucci and S. Abdelwahed, "Model-based response planning strategies for autonomic intrusion protection," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 13, no. 1, pp. 1–23, 2018.

[14] K. Wu, Z. Chen, and W. Li, "A novel intrusion detection model for a massive network using convolutional neural networks," *IEEE Access*, vol. 6, Article ID 50850, 2018.

[15] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986–2998, 2016.

[16] B. Selvakumar and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection," *Computers & Security*, vol. 81, pp. 148–155, 2019.

[17] D. Jin, Y. Lu, J. Qin, Z. Cheng, and Z. Mao, "SwiftIDS: real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism," *Computers & Security*, vol. 97, Article ID 101984, 2020.

[18] R. Zhao, J. Yin, Z. Xue et al., "An efficient intrusion detection method based on dynamic autoencoder," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1707–1711, 2021.

[19] R. Zhao, G. Gui, Z. Xue et al., "A novel intrusion detection method based on lightweight neural network for internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 9960–9972, 2022.

[20] J. Snell, S. Kevin, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[21] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, Salt Lake City, UT, USA, June 2018.

[22] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[23] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," *ICML deep learning workshop*, vol. 2, 2015.

[24] Y. Ouyang, B. Li, Q. Kong, H. Song, and T. Li, "FS-IDS: a novel few-shot learning based intrusion detection system for scada networks," in *Proceedings of the ICC 2021-IEEE International Conference on Communications*, IEEE, Montreal, Canada, June 2021.

[25] M. He, X. Wang, J. Zhou, Y. Xi, L. Jin, and X. Wang, "Deep-feature-based autoencoder network for few-shot malicious traffic detection," *Security and Communication Networks*, vol. 2021, Article ID 6659022, 13 pages, 2021.

[26] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *Proceedings of the International workshop on similarity-based pattern recognition*, Copenhagen, Denmark, October 2015.

[27] W. Min, S. Mei, Z. Li, and S. Jiang, "A two-stage triplet network training framework for image retrieval," *IEEE Transactions on Multimedia*, vol. 22, no. 12, pp. 3128–3138, 2020.

[28] G. He, F. Li, Q. Wang, Z. Bai, and Y. Xu, "A hierarchical sampling based triplet network for fine-grained image classification," *Pattern Recognition*, vol. 115, Article ID 107889, 2021.

[29] G. Andresini, A. Appice, and D. Malerba, "Autoencoder-based deep metric learning for network intrusion detection," *Information Sciences*, vol. 569, no. 2021, pp. 706–727, 2021.

[30] T. P. Nguyen, C. C. Pham, S. V. U. Ha, and J. W. Jeon, "Change detection by training a triplet network for motion feature extraction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 433–446, 2019.

[31] T. Abdullah, Y. Bazi, M. M. Al Rahhal, M. L. Mekhalfi, L. Rangarajan, and M. Zuair, "TextRS: deep bidirectional triplet network for matching text to remote sensing images," *Remote Sensing*, vol. 12, no. 3, p. 405, 2020.

[32] Z. Ji, H. Wang, Y. Pang, and L. Shao, "Dual triplet network for image zero-shot learning," *Neurocomputing*, vol. 373, pp. 90–97, 2020.

[33] X. Gao, J. Chen, Z. Zhan, and S. Yang, "Learning heterogeneous information network embeddings via relational triplet network," *Neurocomputing*, vol. 412, pp. 31–41, 2020.

[34] F. Schroff, D. Kalenichenko, and P. James, "Facenet: a unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Boston, MA, USA, June 2015.

[35] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," 2017, https://arxiv.org/abs/1703.07737.

[36] A. G. Howard, M. Zhu, B. Chen et al., "Mobilenets: efficient convolutional neural networks for mobile vision applications," 2017, https://arxiv.org/abs/1704.04861.

[37] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, https://arxiv.org/abs/1312.4400.

[38] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSp*, vol. 1, pp. 108–116, 2018.

[39] N. Moustafa and Jill Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proceedings of the Military Communications and Information Systems Conference (MilCIS)*, IEEE, Canberra, Australia, November 2015.

[40] H. Zhang, L. Huang, C. Q. Wu, and Z. Li, "An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset," *Computer Networks*, vol. 177, Article ID 107315, 2020.

[41] A. Gouveia and M. Correia, "Network intrusion detection with XGBoost," *Recent Advances in Security, Privacy, and Trust for Internet of Things (IoT) and Cyber-Physical Systems (CPS)*, pp. 137–166, Chapman and Hall/CRC, London, UK, 2020.