RFG-HELAD: A Robust Fine-Grained Network Traffic Anomaly Detection Model Based on Heterogeneous Ensemble Learning

Ying Zhong, Zhiliang Wang, Xingang Shi, *Member, IEEE*, Jiahai Yang, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*— Fine-grained attack detection is an important network security task. A large number of machine learning/deep learning(ML/DL) based algorithms have been proposed. However, attacks not present in the training set pose a challenge to the model (open-set problem). Further, ML/DL based models face the problem of adversarial attacks. Despite the large amount of work attempting to address these problems, there are still some challenges as follows. First, the open-set problem in fine-grained attack detection is difficult to solve because there is no effective representation of the distribution of unknown attacks. Second, in the open set environment, how the fine-grained attack detection model resists the adversarial attack is a more difficult problem. For example, the presence of unknown attacks poses a challenge for adversarial defense. For these reasons, we propose the RFG-HELAD model, which consists of a K classification model based on deep neural network (DNN) with contrastive learning (CL), and a $K+1$ classification model combining a generative adversarial networks (GAN) with two discriminators and deep k-nearest neighbors (Deep kNN). Among them, Deep kNN uses latent features from GAN and contrastive learning as input, which is essentially a distance-based out-of-distribution detection algorithm used to determine unknown attacks. The large category of unknown attacks has been added to the K classification, so it is a $K+1$ classification. To further improve the robustness of the RFG-HELAD model, we perform Fourier transform as well as feature fusion on the features, and also conduct adversarial training on the K classification model. Generative adversarial training of our GAN model can implicitly defend against adversarial attack. Experiments show that our model is superior to other state-of-the-art (SOTA) models in the presence of unknown attacks as well as under adversarial attacks. Especially, our model improves the accuracy by at least 18.7% over the corresponding SOTA model with adversarial defense. Further, we discuss the grounded deployment of the model and demonstrate its feasibility.**

***Index Terms*— Network anomaly detection, adversarial attack, unknown attack detection, ensemble learning, fine-grained attack detection.**

Manuscript received 1 September 2023; revised 30 March 2024; accepted 22 April 2024. Date of publication 17 May 2024; date of current version 31 May 2024. This work was supported by the National Key Research and Development Program of China under Grant 2022YFB3102902. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. George Loukas. *(Corresponding author: Zhiliang Wang.)*

Ying Zhong, Zhiliang Wang, Xingang Shi, and Jiahai Yang are with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China (e-mail: zhongy18@mails.tsinghua.edu.cn; wzl@cernet.edu.cn).

Keqin Li is with the Department of Computer Science, The State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TIFS.2024.3402439, provided by the authors.

Digital Object Identifier 10.1109/TIFS.2024.3402439

I. Introduction

THE continuous upgrading of network technologies and the increasing size of networks are bringing new challenges to cybersecurity. Network traffic anomalies can lead to reduced network performance or complete unavailability. The more common network anomalies are mainly generated by network packets or flows with malicious attacks. For example, scanning attacks, botnets, Denial of Service (DoS) attacks, Distributed Denial of Service (DDoS) attacks etc. Also, network misconfigurations and sudden outages can cause network anomalies. Another aspect that we need to focus on is unknown attacks. According to CrowdStrike's cybersecurity analysis,[1] the scale of network attacks is expanding significantly, and the types of zero-day attacks are increasing. A zero-day attack is an attack in which an undiscovered or undisclosed vulnerability is exploited to gain illegal access to the target system or network. Effective network anomaly detection becomes more important, which means that anomaly detection algorithms can quickly distinguish not only known attacks, but also unknown attacks, thus providing a strong guarantee for network security and network availability.

The first intrusion detection system (IDS) was proposed for detecting host attacks [1]. It was able to detect intrusions, infiltrations, and other forms of computer misuse. Subsequently, considerable progress was made in intrusion detection models, and rule matching based intrusion detection algorithms were proposed. Representative works are Snort [2] and Suricata [3], which cannot detect attacks outside the scope of application of the rules. To keep up with the rapidly evolving technology, machine learning/deep learning based anomaly detection models are flourishing [4], [54]. However, because the problem of network traffic anomaly detection itself is very complex, there are many assumptions that are not close to reality [5]. For example, fine-grained anomaly detection model ACID (adaptive clustering based intrusion detection) does not consider how to deal with unknown attacks [19]. Fine-grained means that the attacks are classified at a more precise level, which corresponds to multiclassification. Most algorithms [8], [9], [10], [11] perform binary classification, which cannot efficiently complete attack classification and determination. In contrast, fine-grained attack classification can provide enough diagnostic information to effectively guide security experts in addressing threats. In real-world network attack environments, fine-grained attack detection is an open set problem and new attacks continue to emerge. Open-set problem means that the identification in the real world is open

[1] https://www.crowdstrike.com/global-threat-report/, 2023.

set, i.e. the identification system should reject unknown/unseen categories when testing [25]. This means that the attacks seen in the tests may not have been present in the training. Existing fine-grained attack detection algorithms that can detect unknown attacks are constantly being proposed [21], [22], [29], but the following challenges still exist.

• **The open-set problem in fine-grained attack detection is difficult to solve because there is no effective representation of the distribution of unknown attacks.** Effective representation means that the distributions of the training and test sets do not mismatch [56]. Known attack distributions are relatively easy to learn through the training set. Therefore, existing algorithms for solving this problem are unsatisfactory due to the inability to capture the distribution of unknown attacks more accurately.

(1) Unknown class detection models based on restricted scenarios. For example, AutoIoT [29] and DeviceMien [57] use Kolmogorov-Smirnov test and probabilistic framework, respectively, to detect unknown IoT devices. However, these customized methods are not applicable to unknown attack detection.

(2) Distance-based unknown attack detection model. For example, KCC [30] uses negative samples to assist in determining the boundaries of known categories and discriminates unknown attacks by the distance between the test sample and the center of the known category. The introduced negative samples do not represent the distribution of unknown attacks and are not effective for detection. Moreover, it has side effects in some network environments. The fundamental reason is that this type of work does not distance the unknown attacks from the known ones.

(3) Unknown attack detection models based on extreme value theory. For example, CVAE-EVT [21] and OpenIDS [31] use extreme value theory to determine unknown attacks. When the real unknown attack distribution only partially conforms to the extreme value theory, the improvement in the detection of unknown attacks will be insignificant (compared to the model without unknown attack processing module).

• **In the open set environment, how the fine-grained attack detection model resists the adversarial attack is a more difficult problem.** At present, there is no corresponding research work to solve this problem.

(1) The presence of unknown attacks poses a challenge for adversarial defense. In an open-set environment, unknown attacks exist only in the detection phase, which is not present in the training phase. This leads to drawbacks of directly using traditional adversarial defense methods like adversarial training. For example, the detection model after adversarial training may not be able to recognize unknown attacks with perturbations. Further, due to the diversity of perturbations, there exist some special perturbations that can misclassify known attacks to unknown attacks [15].

(2) Adversarial defense strategies impact the detection effectiveness of fine-grained attack detection models, which poses design challenges. First, known attacks face the dilemma of compromising between defense effectiveness and detection accuracy [59]. Then, the implementation of fine-grained detection models for adversarial defense in open-set environments needs to be considered holistically. This should consider not only the effectiveness of the defense, but also the impact of the defense strategy on the effectiveness of unknown attack detection.

Based on the above challenges, we propose the RFG-HELAD model (A **R**obust **F**ine-**G**rained Network Traffic **A**nomaly **D**etection Model Based on **H**eterogeneous **E**nsemble **L**earning). To cope with the open-set problem under fine-grained attacks, we propose the RFG-HELAD* model, which covers both K classification model detection and $K+1$ classification model detection. To ensure the detection effect, we customize a fine-grained K classification detection framework. This framework consists of optimal features for different scenarios, normalization, DNN for matching with traffic, and contrastive learning. The corresponding techniques inside this framework can all be replaced to improve the detection results. We propose a $K+1$ classification model for unknown attack detection. The model contains two components, GAN with two discriminators and DNN with contrastive learning [34]. Designed based on DCGAN [6], GAN with two discriminators possesses the ability to **r**ecognize **p**otential unknown attacks, which we call RPGAN. The hidden features of the two components are fused and then the results are fed into Deep kNN for decision making on unknown attacks. In the modeling, Deep kNN is innovatively used as a heterogeneous ensemble learning approach. This effectively pulls apart the distribution of known and unknown attacks, as well as dynamically learns the distribution of potential unknown attacks, which in turn ensures accurate localization of unknown attacks.

Specifically, for known attacks, contrastive learning increases the distance between attack categories and decreases the distance within attack categories. That is, each attack category forms its own cluster that is constantly close to each other. Unknown attacks are not essentially in the same category as all other known attacks. The proximity movement within the known attack categories indirectly increases the distance between the distributions of known and unknown attacks. And, RPGAN (GAN with two discriminators) can reduce the KL divergence and inverse KL divergence between the real data distribution and the data distribution generated by the generator, thus effectively enhancing the generalization performance of the GAN model. RPGAN treats the known categories (known attacks + benign traffic) in the network traffic as one macroscopic category, and the distribution of unknown attacks learned with generative adversarial training as another macroscopic category. With this generative adversarial training, the distance between the distributions of the two macroscopic categories is further increased in order to better discriminate the unknown attacks. The determination of the unknown attack distribution requires the participation of known attacks in the training, which also shows that our algorithm has the ability of dynamic learning. The potential unknown attack distribution can be determined based on the corresponding known attack network environment.

To address the problem of fine-grained attack detection models against adversarial attacks in an open-set environment, we propose a comprehensive hierarchical progressive adversarial defense strategy (CHPDD). In the known attack defense part, we organically combine several defense strategies in the order of the data processing flow. Specifically, Fourier transform enhances the difference after perturbation at the feature level. Hidden space compression (a manifestation of the efficacy of contrastive learning in adversarial defense) reduces the space that can be perturbed at the distribution level after the feature form is determined [14]. Adversarial training is the perturbation of the determined features to train

TABLE I
DETAILED COMPARISON OF EXISTING NETWORK ANOMALY DETECTION ALGORITHMS

K/K+1	Model	Fine-grained	Adversarial attack	Open-set	Generality	Code
K	ACID [19]	yes	no	no	some	yes
	SCADA [13]	no(*)	no	no	some	no(classic)
	FARE [20]	yes	no	no	yes	yes
	DNN-kNN [12]	no(*)	no	no	yes	yes
	1DCNN [18]	no	yes	no	no	no
KS test based ($K+1$)	AutoIoT [29]	yes	no	yes	no	yes
EVT based ($K+1$)	openmax(DNN)+centerloss [23]	yes	no	yes	yes	yes
	CVAE-EVT [21]	yes	no	yes	some	yes
	OpenIDS [31]	yes	no	yes	some	no
Distance based ($K+1$)	scalable-NIDS [22]	yes	no	yes	some	yes
	CADE [24]	no(*)	no	yes	some	yes
	KCC [30]	yes	no	yes	some	no
	Our model	yes	yes	yes	yes	yes

yes means the corresponding attribute is considered; no means the corresponding attribute is not considered; EVT means extreme value theory some means the corresponding attribute is partially considered; no(*) means that the corresponding attribute can be extended by modification. K indicates that no unknown attacks are considered, while $K+1$ considers unknown attacks; KS test means Kolmogorov-Smirnov test.

the model and enhance the robustness of the model. These are combined in a logical sequence from raw data, data distribution, and model training. The intrinsic connection is that the processing of the hidden space compression takes into account the changes in the distribution due to the Fourier transform. Adversarial training also perturbs the features of the Fourier transform. These are essentially to reduce the likelihood that the perturbations will cause the classifier to misclassify samples. Regarding the unknown attack defense, in order to incorporate all the defense properties of the defense part of the known attack, we ensemble the hidden layer output of the K classification model in the final judgment of the unknown attack. Further, the generative adversarial training of our RPGAN model has been able to implicitly defend against adversarial attacks while addressing the problem of recognizing unknown attacks with perturbations. This is because the ideal generator naturally spoofs the discriminator, which makes the discriminator robust to this spoofing (covering the effect of perturbation). Finally, the design of adversarial defense strategies can be complemented with the RFG-HELAD* model. That is, RFG-HELAD* and CHPDD together form the RFG-HELAD model.

The contribution of our paper is summarized as follows.

- **Novel fine-grained network traffic attack detection with adversarial defense for open set environments.** We propose the RFG-HELAD model, which is formed by unknown attack detection module RFG-HELAD* and adversarial defense strategy CHPDD. To the best of our knowledge, this is the first attempt to fuse multiple machine learning/deep learning models (DNN, GAN with two discriminators, Deep kNN, contrastive learning) into a single framework in a heterogeneous ensemble learning manner to address the problem of co-existing unknown and adversarial attacks in fine-grained attack detection scenarios.
- **Multi-aspects high detection performance.** We test our model on several data sets (UKM [43], NSLKDD [44], Kitsune [8]). Experiments show that our model performs better than SOTA K classification models (FARE [20], CADE [24], ACID [19], SCADA [13], DNN-kNN [12]) and $K+1$ classification models (scalable-NIDS [22], CVAE-EVT [21]), and also has good adversarial robustness. In fine-grained attack detection scenarios with both unknown and adversarial attacks, our model improves the accuracy by at least 18.7% over the corresponding SOTA model with adversarial defense. Meanwhile, our RFG-HELAD model has good generalization.
- **Complete fine-grained attack detection system and open source.**[2] We implement an open source fine-grained attack detection system and discuss the grounded deployment of the model and demonstrate its feasibility.

The rest of the paper is organized as follows. We summarize the related work in Section II and provide background and problem scope in Section III. Section IV introduces our RFG-HELAD model. Experimental evaluations and analyses are shown in Section V. In Section VI, we first discuss model-related issues, followed by limitations and future work. Section VII concludes our work.

II. RELATED WORK

With the development of network technology, network traffic is increasing. The packets reached per second are gigantic [7]. To process this data quickly, network packet header-based or flow-based anomaly detection models are constantly evolving. Among them, machine learning and deep learning based models are dominant. These intrusion detection models can be classified as binary [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18] or multi-classified (fine-grained anomaly detection) [19], [20], [21], [22], [23] according to the class of classification. The main idea of binary classification attack detection models (e.g., HELAD [9]) is to construct a model of normal traffic and to treat any traffic that deviates from the normal model as an attack during the detection phase. Multi-classification attack detection models (e.g., our RFG-HELAD) are to learn the distribution of each attack category that can be recognized in the detection phase. These two types of work address different issues. We present a systematic summary of recent work on ML/DL-based network anomaly detection. TABLE I shows detailed comparison of existing network anomaly detection algorithms. We compare anomaly detection models in terms of several properties. For example, whether this is a fine-grained attack detection algorithm or whether this algorithm handles the open set problem. It can be seen that for the SCADA algorithm [13], the fine-grained property is no(*). This indicates that this algorithm is not a fine-grained algorithm in the original paper,

[2]https://github.com/RFG-HELAD/RFG-HELAD

but can be modified with the final output layer to achieve fine-grained attack detection. Specifically, we present related work on unknown attack detection and adversarial attack and defense in intrusion detection.

A. Unknown Attack Detection

The importance of unknown attack detection is being continuously demonstrated. Therefore, a large amount of work has emerged. We highlight the following algorithms [21], [22], [23], [24], [29], [30], [31], [52], [53]. Openmax [23] overcomes the limitations of softmax and enables the neural network to detect unknown attacks. Reference [22] proposed Open Set Classification Network (OCN) to detect unknown attacks, OCN is based on convolutional neural network with nearest class average (NCM) classifier. It is designed with two new losses to jointly optimize it, including Fisher loss and maximum mean difference (MMD). Reference [21] combined the fine-grained known/unknown intrusion detection problem as a two-stage minimization problem. The first phase is to seek a scoring criterion that minimizes the empirical risk of misclassification of known attacks. The second phase is to find another scoring criterion to minimize the identification risk of inferring an unknown attack. In the second stage extreme value theory is further used to build model the distribution of reconstruction errors to distinguish unknown attacks. Reference [29] proposed a novel IoT device identification model named AutoIoT, which updates itself automatically when new types of devices are plugged in. Their unknown attack detection algorithm is mainly based on KS testing. This model is customized for iot traffic data. The disadvantage of these methods is that they cannot accurately capture the distribution of unknown attacks.

B. Adversarial Attack and Defense in Intrusion Detection

Similar to us, these works [58], [59], [60] also study anomaly detection as well as adversarial related theories (e.g. GAN, game theory) and address specific problems in each subfield separately. The difference between our paper and these works is that we focus on how to effectively defend against adversarial attacks in network traffic anomaly detection. The problem of adversarial attacks in the IDS domain has received much attention in recent years [32]. We present several more representative works. Reference [33] systematically evaluated ML-based NIDS for adversarial robustness, which can better provide analytical guidance for our real defense model. Reference [16] presented Whisper, an ML-based real-time malicious traffic detection system that achieves high accuracy and high throughput by exploiting frequency domain features. In particular, attackers cannot easily interfere with the frequency domain features, and thus Whisper is robust to various evasive attacks. Reference [18] investigated an attack detection method based on simple and lightweight neural networks and using frequency domain features to enhance robustness. All of these adversarial defence methods are done based on binary classification algorithms and fine-grained adversarial defense needs to be considered. Further, fine-grained adversarial defense work in open-set environments is not yet available.

Based on the above analysis, we would like to propose a new robust fine-grained anomaly detection model based on ensemble learning [46]. This model can detect unknown attacks, as well as defend against possible adversarial attacks.

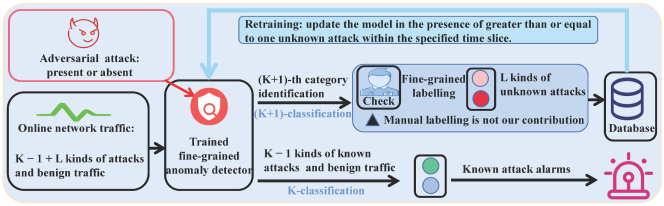


Fig. 1. Flow chart of our research scope on intrusion detection models.

III. Background and Problem Scope

In this section, we first define the problem. Then we describe our motivation for using ensemble learning.

A. Problem Definition

We know that cyber attacks are complex, and new attacks are constantly emerging. Therefore, our model should be constantly iteratively updated. It is not known how many unknown attacks there are in the actual scenario. Since the classifier cannot complete the classification of uncertain categories, we temporarily treat all unknown attack categories reached in the specified time period in our model as one new attack major category. This new attack major category is then given to the network security administrator to analyse and give fine-grained labels. These new fine-grained labelled data sets are used to update model. There may be multiple unknown attacks arriving in a time period. Once it is discovered that our model detects an unknown attack, a fine-grained labeling process is required by the network security administrator. This one or more new attacks are then added to the retraining. This process is repeated over time. Meanwhile, our fine-grained attack detection models are developed based on deep learning and will face the problem of adversarial attacks. Based on the above analysis, **our research problem** is defined as shown below. our goal is to achieve accurate classification of known attacks, detection of one new attack major category, and defence against adversarial attacks within specified time period. The fine-grained labelling of unknown attacks is outside the scope of our study.

To facilitate the evaluation of the model's performance, network traffic from the specified time periods were selected for the experiments. We divide the network traffic collected during this time period into two data sets in chronological order. The former is treated as training set after experimental setup, and the latter is treated as test set directly. Specifically, we assume that there are $(K-1)$ classes of known attacks and benign traffic in the training set and $K-1+L$ kinds of attacks and benign traffic in the test set ($K \geq 1$ and $L \geq 0$). Our goal is to accurately identify K known classes including attacks and benign traffic, as well as to identify all L species as unknown attacks denoting as $(K+1)th$ class, and to be able to defend against adversarial attacks. Fig. 1 shows the flow chart of our research scope on intrusion detection models.

B. Motivation

Our core idea is to ensemble DNN with contrastive learning and GAN model with two discriminators using Deep kNN as a heterogeneous ensemble method. We state the motivation of this idea in the perspective of solving challenges. Fig. 2 shows correspondence diagram between motivations and system components. We show the linkages at the macro level through

the four modules (feature processing module, attack distribution learning module, unknown attack localization module, adversarial defense module). It can be seen that the modules are interconnected and support each other. To better understand the motivation part, we introduce expressions like Part 1. For example, Part 1 belongs to the known attack distribution learning in the attack distribution learning module, which needs to be realized by the DNN+CL components in our model. DNN+CL is used for known attack classification which is composed of DNN and contrastive learning. The robustness of the DNN+CL component needs to be supported by the hidden space compression (another function of contrastive learning) and adversarial training in Part 4. The relationship between other modules can be analyzed similarly. The detailed linkages can be seen in the following specific analysis.

(1) The motivation to address challenge 1 is analyzed as follows. 1) Closed set detection is the basis of open set detection. The basic detection process for closed sets is as follows. The first step is feature extraction followed by normalization. Then a classifier needs to be selected. For classification tasks, representation learning can optimize the performance of the classifier. We select the contrastive learning in representation learning. In order to ensure the detection effect, we need to customize a fine-grained K classification detection framework, which consists of optimal features for different scenarios, normalization, DNN for matching with network traffic, and contrastive learning. The optimal features can be selected according to the scenario. We illustrate the advantages of our framework using basic DNN models suitable for network traffic. Obviously, other better classifiers can be chosen. Contrastive learning can distance different classes and further reduce misclassification. This combination improves the generalization of the model at three levels: raw data representation, distribution shift, and the degree of matching between the model and the data distribution. Also, the three parts of the combination are convenient to extend to adversarial defense. This part learns the optimal K classification (known attack detection) and the distribution of known attacks. (Part 1)

2) The open-set problem in fine-grained attack detection is difficult to solve because there is no effective representation of the distribution of unknown attacks. Inspired by openGAN [37], which uses primitive GAN networks to augment unknown categories, we propose RPGAN (or GAN with two discriminators) to dynamically learn the distribution of potential unknown attacks. Unlike openGAN, we do not need to have a small number of unknown categories in training because of the inaccessibility of unknown attacks in real scenarios. Motivated by D2GAN [36], RPGAN adds a discriminator that uses the inverse KL divergence to compute the loss on top of the original discriminator, which can rationally utilize the complementary statistical properties of the two discriminators. The use of two discriminators increases the diversity of the learned distributions and in turn improves the generalization performance. The generalization performance refers to the fact that the model obtained from the training samples can also be well adapted to the testing samples. Enhancing the generalization performance of the GAN model implies that the GAN-based latent layer features have generalization, which means that Deep kNN using latent layer features also enhances the generalization when deciding unknown attacks, thus improving the detection of unknown attacks. Further, as we mentioned earlier, the RPGAN model has the ability to learn dynamic unknown attack distributions based on known attacks. This part learns the distribution of potentially unknown attacks in RPGAN during the generative adversarial training process. The learned distribution is based on known attacks, so it is characterized as an existential possibility and has limited representational capabilities. (Part 2)

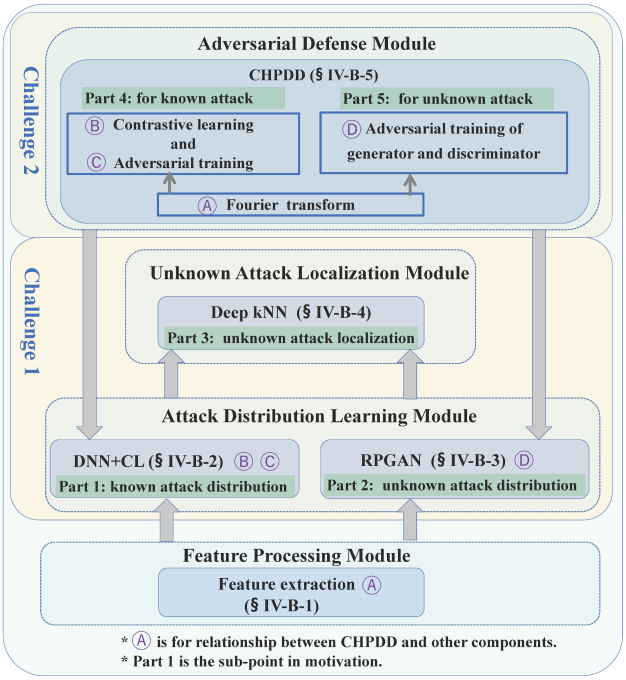


Fig. 2. Correspondence diagram between motivations and system components.

3) After learning the distributions of known and unknown attacks, a rational decision of unknown attacks is needed. The distributions of known and unknown attacks from the Part 1 and Part 2 are combined into a new feature, and then the Deep kNN distance algorithm is used to determine the unknown attacks. We can view this combination of features from two perspectives. First, with features representing the distribution of known attacks, the model is less likely to misclassify known attacks as unknown attacks. Second, with features that represent the distribution of unknown attacks, it is less likely to allow unknown attacks to be misclassified as known attacks. This part can accurately localize unknown attacks (unknown attack detection) based on ensuring the effectiveness of known attack detection. (Part 3)

(2) The motivation to address challenge 2 is analyzed as follows. 1) To address the impact of defense strategies on attack detection, we adopt a complementary philosophy. First, we try to choose components that both improve detection performance and enhance robustness. For example, contrastive learning can both improve the detection effect and reduce the perturbed space. RPGAN can both learn the distribution of potential unknown attacks and defend against unknown attack perturbations. Second, the defense is strengthened according to each part of the optimal detection process, which makes our defense more comprehensive. Specifically, Fourier transform, hidden space compression (contrastive learning), adversarial training are all expanded from the three aspects of the attack detection process for Part 1. (Part 4)

2) To address the challenges posed by the emergence of unknown attacks to the adversarial defense, we have to fully incorporate all the defense properties of the known attack part as well as the advantages of coping with perturbations in the

training of the GAN network itself (adversarial training of generator and discriminator). (Part 5)

(3) To summarize, Part 1 and Part 3 together are fine-grained attack detection for open-set environments. Our ensemble learning model can better detect unknown attacks on the basis of good detection of known attacks. Meanwhile, the components inside the ensemble learning model can be extended to adversarial defense more easily, which meets the demand of fine-grained attack detection model against adversarial attacks in open-set environments. Therefore, it is sensible to ensemble DNN with contrastive learning and GAN model with two discriminators using Deep kNN as a heterogeneous ensemble method to design robust fine-grained attack detection models.

IV. RFG-HELAD Model

In this section, we first explain the systematic structure of our model. Then, we specify the details of the model.

A. The Structure Diagram of RFG-HELAD Model

We propose the RFG-HELAD model, which is formed by unknown attack detection module RFG-HELAD* and adversarial defense strategy CHPDD. This is shown in Fig. 3. The input to the model can be either flow data or raw packet data. After the feature extraction, normalization [55] is then used to obtain the original feature form. This is done by centering the data by the minimum and then scaling by the extreme difference (maximum - minimum). The data is shifted by the minimum of one unit and will be converged to between [1, 0]. The original features and their Fourier transformed forms are combined into the final features. Then the K model (DNN with contrastive learning) and RPGAN models are trained. Our RPGAN network adds a discriminator to the DCGAN infrastructure, and this discriminator uses inverse divergence to compute the loss. The specific RPGAN structure is shown in Fig. 4. Next, K model is adversarial trained using the final features. And the hidden layer features generated in the training phase are saved and used as input to Deep kNN. The distance matrix D_{TV} of the original training set is thus constructed, which is used to calculate threshold and will be described subsequently.

In the test phase, the extracted features are fed directly into the discriminator D_1 and K model to generate the hidden layer features of the current sample. This hidden layer feature is then compared to the training set to calculate the distance vector. A threshold is used to determine if the attack is unknown. If it is an unknown attack, then it needs to be checked by a network security administrator. If not, perform a known attack classification. Considering that some network environments will not have adversarial attacks, we can avoid the tradeoff [47] between accuracy and adversarial defence by not enabling the defence part CHPDD. That is, we can use the RFG-HELAD* model alone.

B. Details of the Model

1) Feature Engineering (Optimal Feature+Normalization): Feature engineering has a significant impact on detection performance. For data in pcap form, we use Damped Incremental Statistics as feature extraction method [8] because it is an excellent method [8], [9]. The extracted features are further processed using the normalization to obtain the final features. For data in the form of flow, we use normalization to obtain the final features directly because it can represent flow data more efficiently. In summary, if there is a better feature extraction method based on flow and pcap, it can also be incorporated into our feature extraction framework: excellent feature extraction method + normalization. The feature extraction method can be left empty. After processing in our feature extraction framework, it can fit well with our DNN+CL model.

2) DNN+CL (Evolving DNN+Class Distribution Optimization Techniques): In this part, we introduce combination evolving DNN+CLass distribution optimisation techniques. To demonstrate the superiority of this combination, we choose a simple DNN as K classification model, and contrastive learning as class distribution optimisation techniques. The simplest DNN is chosen because it is a better match to the traffic (Details can be found in the Part 8 of Appendix.). CL method is recognized as a method that can improve the classification effect, which could increase the distance between classes and decrease the distance within classes. We describe the combination DNN+CL in detail.

Our DNN model has a total of 4 layers. The output of the third layer we denote as $DNN_3(x_i)$. x_i is the i-th input sample (final feature in Fig. 3). The projection space of the third layer of the DNN is denoted as $\boldsymbol{h} \in \mathbb{R}^d$. In our model we have $\boldsymbol{h}_i = DNN_3(x_i)$. First, we present the cross-entropy loss for multiclassification based on DNN, where M is the number of test samples and N is the number of labels.

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{\mathrm{M}} \sum_{j=1}^{\mathrm{M}} \sum_{i=1}^{N} y_{ji} \log \widehat{y_{ji}} \tag{1}$$

In multiclassification, y_{ji} is used to indicate the true label of a sample. $y_{ji} = 1$ means that the label of the j-th sample is i. $y_{ji} = 0$ means that the label of the j-th sample is not i. $\widehat{y_{ji}}$ is used to indicate the predicted label of the sample.

Next we describe the loss of contrastive learning [34], [35].

$$\mathcal{L}_{\mathrm{CL}} = \sum_{j \in I} \frac{-1}{|P(j)|} \sum_{p \in P(j)} \log \frac{\exp\left(\boldsymbol{h}_j \cdot \boldsymbol{h}_p / \tau_{CL}\right)}{\sum_{a \in A(j)} \exp\left(\boldsymbol{h}_j \cdot \boldsymbol{h}_a / \tau_{CL}\right)} \tag{2}$$

Here, τ_{CL} is a temperature parameter and I is the set of sample numbers. $A(j) = I \backslash \{j\}$ stands for the set of numbers except for j. $P(j) = \{p \in A(j) : y_p = y_j\}$ and the meaning of $y_p = y_j$ is that the labels numbered p and j are the same. $|P(j)|$ is its cardinality. $\boldsymbol{h}_j$, $\boldsymbol{h}_a$, and $\boldsymbol{h}_p$ are the projections of the corresponding numbers onto the third layer of the DNN's hidden space, respectively.

The loss of the K classification model can be obtained below.

$$\mathcal{L}_{\mathrm{K-model}} = \mathcal{L}_{\mathrm{CE}} + \mathcal{L}_{\mathrm{CL}} \tag{3}$$

At the end of the training of the DNN+CL model, we can obtain the hidden layer features $Hid_1 \leftarrow (DNN + CL)_h(x_i)$. In the testing phase, this step leads to the K classification result @Res1.

3) RPGAN (Recognize Potential Distribution of Unknown Attacks): Inspired by D2GAN [36] and DCGAN [6], we design RPGAN, which uses two discriminators ($D_1(\mathbf{x})$ and $D_2(\mathbf{x})$) and a generator. RPGAN can learn the potential distribution of unknown attacks well. About $D_1(\mathbf{x})$, a high score is awarded if $\mathbf{x}$ is drawn from the data distribution P_{data}, and a low score is given if it is generated from the

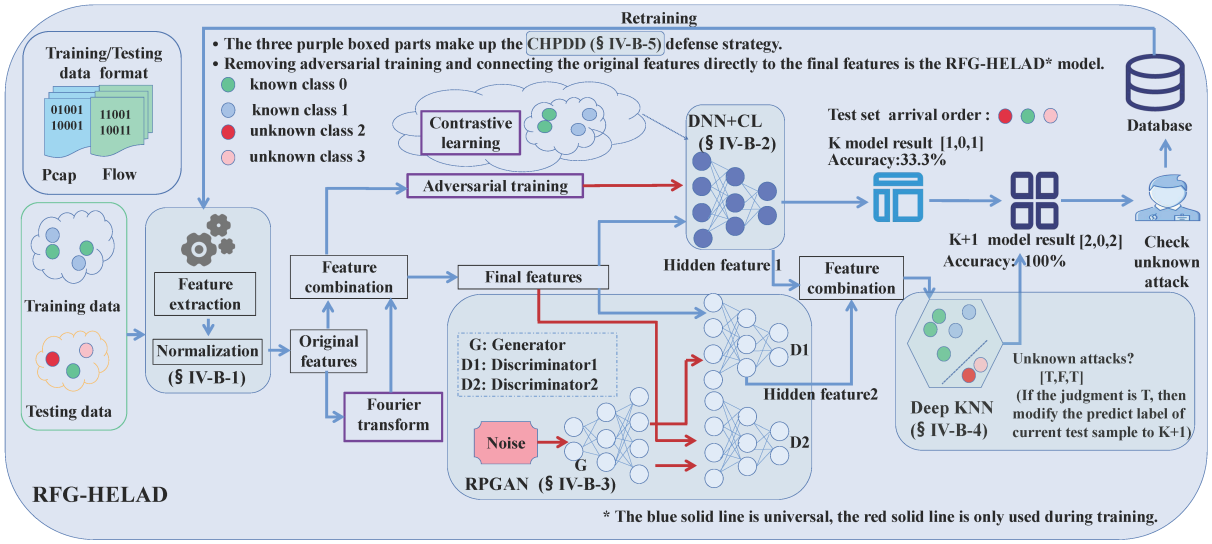


Fig. 3. The structure diagram of RFG-HELAD.

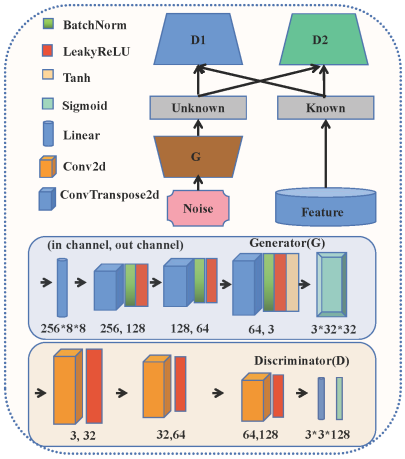


Fig. 4. The framework diagram of our designed RPGAN model.

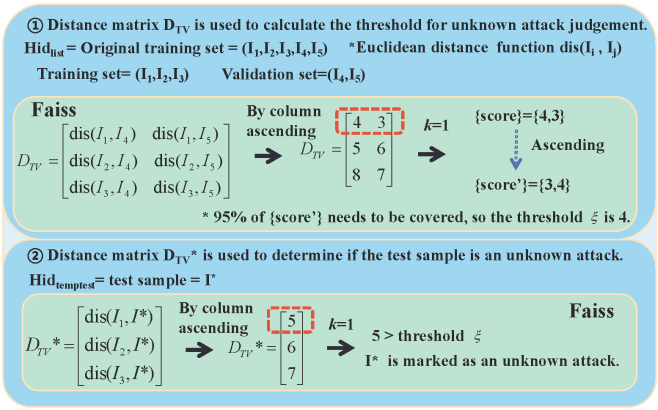


Fig. 5. Calculating distance matrix with decision unknown attacks using Faiss.

generator distribution P_G. Conversely, $D_2(\mathbf{x})$ returns a high score for $\mathbf{x}$ generated from P_G and gives a low score for samples drawn from P_{data}. D_1 and D_2 do not share their parameters. We control the diversity of the generated samples by the value of λ. A more formalised minimax optimization problem is shown below.

$$\begin{aligned}&\min_{G}\max_{D_1,D_2}\mathcal{J}(G,D_1,D_2)\\&=\mathbb{E}_{\mathbf{x}\sim P_{\text{data}}}\left[\log D_1(\mathbf{x})\right]+\mathbb{E}_{\mathbf{x}\sim P_G}\left[-D_1(G(\mathbf{x}))\right]\\&\quad+\mathbb{E}_{\mathbf{x}\sim P_{\text{data}}}\left[-D_2(\mathbf{x})\right]+\lambda\times\mathbb{E}_{\mathbf{x}\sim P_G}\left[\log D_2(G(\mathbf{x}))\right]\end{aligned}\tag{4}$$

Regarding the solution of this optimization problem, we first fix G and then maximise $\mathcal{J}(G,D_1,D_2)$. The optimal discriminators $D_1^\star$ and $D_2^\star$ can be obtained. Next, we fix D_1=$D_1^\star$,D_2=$D_2^\star$, and substitute them into equation (4). The following expression can be obtained.

$$\begin{aligned}\mathcal{J}\left(G,D_1^\star,D_2^\star\right)&=-1+\lambda(\log\lambda-1)\\&\quad+D_{\text{KL}}\left(P_{\text{data}}\|P_G\right)+\lambda D_{\text{KL}}\left(P_G\|P_{\text{data}}\right)\end{aligned}\tag{5}$$

$D_{\text{KL}}\left(P_{\text{data}}\|P_G\right)$ and $D_{\text{KL}}\left(P_G\|P_{\text{data}}\right)$ are the KL divergence and inverse KL divergence between the data and the generator distribution, respectively. These divergences are always non-negative and are only zero if $P_G = P_{data}$. This achieves an optimal solution to the optimization problem.

At the end of the training of the RPGAN, we can obtain the hidden layer features Hid_2 via discriminator D_1 and $Hid_2 \leftarrow (GAN - D_1)_h(x_i)$.

4) Deep kNN(Ensemble Learning to Decide Unknown Attacks): Deep k-nearest neighbors (Deep kNN) have shown great advantages in anomaly detection [38], [39]. We use it as an ensemble learning approach. Specifically, we define the input data for model deep k-nearest neighbors as $\mathbb{L}_n = (\mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_n)$. The elements in $\mathbb{L}_n$ are derived from the fusion of features from Hid_1 and Hid_2.

In the testing phase, the test sample after feature extraction is $\mathbf{l}^*$. We calculate the Euclidean distance $\|\mathbf{l}_i - \mathbf{l}^*\|_2$, where $\mathbf{l}_i \in \mathbb{L}_n$. We arrange the obtained Euclidean distances in increasing order to obtain $\mathbb{L}'_n = (\mathbf{l}'_1, \mathbf{l}'_2, \ldots, \mathbf{l}'_n)$, and the decision function of the unknown attack is as follows.

$$M\left(\mathbf{l}^*;k\right)=\mathbf{1}\left\{-r_k\left(\mathbf{l}^*\right)\geq\delta\right\}\tag{6}$$

where $r_k(\mathbf{l}^*) = \left\|\mathbf{l}^* - \mathbf{l}'_k\right\|_2$ is the distance to the k-th nearest neighbor and $\mathbf{1}\{\cdot\}$ is the indicator function which means

that when the input is True, the output is 1, and when the input is False, the output is 0. This is used to determine unknown attacks. The threshold δ is usually chosen so that a high proportion of known attack data (e.g., 95%) is correctly classified. This can be selected in the validation set. The threshold is independent of the unknown attacks. We use a library named Faiss[3] for efficient nearest neighbour search. Specifically, faiss.IndexFlatL2 is used as an indexing method for Euclidean distances. Fig. 5 illustrates the process of calculating the distance matrix with decision unknown attacks using Faiss. The distance matrix D_{TV} is calculated using Faiss, which is calculated in RAM and is faster. In the testing phase, this step leads to the $K+1$ classification result @Res2.

5) CHPDD(Explicit Methods of Adversarial Defence: Fourier Transform of Features and Adversarial Training of Models): Hidden space compression (contrastive learning) of CHPDD is the implicit adversarial defence method and has already been described in the previous part. Next, we present the other two explicit components of adversarial defence. The expression for the **Fourier transform** [40], as shown below.

$$\hat{g}(k) = \int_{-\infty}^{+\infty} g(t) e^{-2\pi i t k} dt \tag{7}$$

g is some specified function that depends on time t and $\hat{g}$ is the Fourier transform. Here, i represents the imaginary unit and k is the frequency. For periodic data sampling, the following discrete Fourier transform can be used.

$$DFT_k = \sum_{n=0}^{N-1} g_n e^{-2\pi i n k / N}, \tag{8}$$

g_n means g for the n-th sample. The Fourier transform provides the frequency spectrum for the entire measurement time period. We use the Short Time Fourier Transform (STFT) and implement it using a third-party library, librosa.[4]

The **adversarial training** [41] is to add a perturbation Δx to the original input sample x to obtain the adversarial sample, and then train it, and the problem can be abstracted into such a model:

$$\max_{\theta} P(y \mid x + \Delta x; \theta) \tag{9}$$

where y is the ground truth and θ is the model parameter. It means that the parameter maximizes the probability of predicting y even under the perturbation.

V. Experimental Evaluation and Analysis

To explore the effectiveness of our model, we introduce a series of experiments. We first introduce the data set and experimental configuration. Then we validate our model and analyze the experimental results in the order of the main research questions and uncover the insights behind the experimental results. Our code is available for download. The main questions we study are as follows.

- How does the performance of our algorithm on the fine-grained classification submodel (K classification), compared with other state-of-the-art algorithms?
- How does our algorithm's K classification submodel perform in the cases of adversarial attack?
- How does our model perform in an environment with unknown attacks and without adversarial attack, compared with other state-of-the-art algorithms ($K+1$ classification)?
- How does our algorithm perform in the presence of unknown attacks and adversarial attack, comparing multiple metrics with other top algorithms?
- Are the individual components of our model effective (ablation experiments)?

A. Datasets

We use three data sets, NSLKDD2009 [44], Kitsune2018 [8], and UKM2020 [43]. These three data sets represent the most classical, relatively new pcap-based, and relatively new flow-based network attack detection dataset, respectively. The data set selected in this way has better heterogeneity and representativeness.

NSLKDD2009 is a classic data set. This data set includes four broad categories of attacks, Probe, R2L, U2R, and Dos. Each broad category contains multiple attacks. For example, Probe includes attacks such as ipsweep, nmap, etc., and Dos includes attacks such as pod, smurf, etc. Here 41 feature fields are directly provided and we use normalization directly. NSLKDD2009 can also be noted as NSL or NSLKDD.

Kitsune2018 includes four major categories of attacks: reconnaissance, Man in the Middle, Denial of Service, and botnets. Of these, the more fine-grained categories of attacks within the major categories will be presented in the specific data set used. The pre-feature extraction is done using Kitsune's own feature extraction method. This data set is the raw network traffic. The dimensionality after feature extraction is 100. Kitsune2018 can also be noted as Kitsune.

UKM2020 is a relatively new data set, which encompasses four types of attacks, namely DoS, ARP poisoning, exploits, and scanning. This data set represents the attacks in the new network environment. Here 46 feature fields are provided directly and we use normalization directly. UKM2020 can also be noted as UKM.

B. Evaluation Metrics

Accuracy of multiclass classification is used as the main evaluation metric. We define the accuracy of K classification as ACC-K, and the accuracy of $K+1$ classification as ACC-(K + 1). To illustrate that the use of accuracy is representative, we also evaluate other metric [55]: Area Under Curve (AUC), $avg\_Precision$ which means weighted average Precision as **P**, avg_Recall as **R**, and avg_F_1 as **F1**. Weighted average means that calculate metrics for each label, and find their average, weighted by support (the number of instances for each label). AUC is essentially a binary classification evaluation metric. We treat unknown attacks as positive examples, and known attacks and normal traffic as negative examples. To better assess generalizability, some of the experiments calculate the average of individual result, denoted as AVG.

Then, we introduce two representative metrics for binary class classification for experimental evaluation [55]. These include: true positive rate (TPR), false positive rate (FPR). In order to make these metrics usable on multiclassification experiments, we need to modify the metrics. Specifically, TPRs are calculated for each category in the multiclassification, and then the average of these TPRs is calculated, which is denoted as TPR_a. FPR_a is analyzed similarly. On the other hand, the known and unknown attacks are counted as positive

[3]Faiss: https://github.com/facebookresearch/faiss
[4]librosa: http://librosa.org/

TABLE II
DEFINE THE CORRESPONDENCE BETWEEN THE EXPERIMENTAL ENVIRONMENT AND OUR MODEL

	K classification (known classification)	$K+1$ classification (known classification + unknown detection)
adversarial attack (with adversarial defence)	RFG-HELAD-K	RFG-HELAD-(K+1)
no adversarial attack (no adversarial defence)	RFG-HELAD*-K	RFG-HELAD*-(K+1)

samples and the benign traffic as negative samples. The FPR obtained in this case is called FPR_2. For better analysis, we introduce false negative rate (FNR) [55] in this scenario, denoted as FNR_2. **Lower FNR and FPR are better. For other metrics, the higher the value, the better**.

C. Experimental Settings

1) Specific Experimental Environment: We use python to implement our model and comparison algorithm. We select the fine-grained categories of attacks in the dataset for our experiments. Regarding the core experiment, we have two types of data sets. The first category is the K classification experimental data sets (without unknown attack): UKM (training set: 8923, test set: 2226, 5 categories), NSL (training set: 123059, test set: 16179, 8 categories), Kitsune (training set: 810000, test set: 90000, 10 categories). The second category is the $K+1$ classification experimental data sets (with unknown attack): UKM (training set: 8923, test set: 2578, training set 5 categories, test set 8 categories), NSL (training set: 6000, test set: 3000, training set 3 categories, test set 6 categories), Kitsune (training set: 30000, test set: 10000, training set 6 categories, test set 10 categories). Each instance is a labeled record of the dataset. Since the comparison algorithms vary in complexity, we reduce the amount of data in second category for a fair comparison. Also, in order to minimize the effect of imbalance on the individual algorithms (focusing on the core problem), we extract a relatively balanced data set. See Part 4 of Appendix for specific details and analysis. We use the hold-out method for time series data. The hold-out approach directly divides the data set D into two mutually exclusive sets. One of the sets is used as the training set U, and the other is used as the test set V, i.e. $D = U \cup V$, $U \cap V = \phi$. In the temporal order, all samples of the training set are in front of the test set. All results of our experiments are averaged over 5 times. To make it easier to understand, we define the correspondence between the experimental environment and our model, which is shown in TABLE II.

2) Hyperparameter Configuration of Our Model: Our parameters are divided into three main parts, corresponding to DNN with contrastive learning, GAN, and deep k-nearest neighbours. The specific meanings of the parameters and their assignments are as follows. ① DNN+CL: contrastive learning parameters suploss = 0.7, learning rate of DNN lr1 = 1e-2, fourier parameters n_fft = 52, hop_len = 64; ② GAN: noise vector dimension of GAN z_dim = 100, weights of the two generators lamda = 0.1, learning rate of GAN lr2 = 0.0001; ③ Deep kNN: hidden layer vector dimension (consistent for DNN and GAN) hid_dim = 64, nearest neighbor parameter k = 1. The thresholds represent distances and do not have a great meaning in themselves, so they are not shown here.

D. Baseline Model

1) K Classification: Regarding the K classification, two classes of models are selected for comparison. The first class is the classical machine learning/deep learning models. This includes one-dimensional CNNs (onedCNN), MLPs, LSTMs, DNNs, and CNNs [4]. The second category is excellent fine-grained attack detection methods, which includes ACID [19], SCADA [13], FARE [20], and DNN-kNN [12]. ACID consists of multiple kernel networks. SCADA in the original paper is a binary classification algorithm, which is a combined model of classical LSTM and DNN. Thus, this can be easily changed to a multi-classification model. FARE uses the labels determined by multiple unsupervised models for auxiliary classification, which are originally unsupervised and semi-supervised models. For a fair comparison, we fed all the training set labels into FARE to make it a fully supervised model. We use six clustering algorithms [55] kmeans, AgglomerativeClustering, SpectralClustering, MBkmeans, FeatureAgglomeration, and AffinityPropagation to form 30 clustering results. To maintain heterogeneity, each clustering algorithm is reused with a different parameter configuration. DNN-kNN in the original paper is a binary classification algorithm, which is a combined model of classical KNN and DNN. This also can be changed to a multi-classification model.

2) K + 1 Classification: We evaluate the effectiveness of $K+1$ classification, which means that all unknown attacks are detected as one large class. In this case, we choose three types of comparison algorithms. The first category is

TABLE III
K CLASSIFICATION EXPERIMENTAL RESULTS (ACC-K)

model	NSL	Kitsune	UKM	AVG
onedCNN	**0.970**	0.812	**1.000**	0.927
MLP	0.968	0.808	0.997	0.924
LSTM	0.966	0.858	0.996	0.940
DNN	0.968	0.928	0.996	0.964
CNN	0.960	0.766	**1.000**	0.908
ACID	0.910	0.676	0.949	0.845
SCADA	0.969	0.929	0.997	0.965
FARE	0.962	0.928	0.998	0.963
DNN-kNN	0.969	0.917	0.998	0.961
ours (RFG-HELAD*-K) •	**0.970**	**0.944**	**1.000**	**0.971**

TABLE IV
ABLATION EXPERIMENTS OF OUR DEFENSE MODEL IN THE CASE OF FGSM ADVERSARIAL ATTACK AND UKM DATASET (EPSILONS=12/255,ACC-K). ADVT STANDS FOR ADVERSARIAL TRAINING. CLEAN_TEST STANDS FOR ACCURACY IN RFG-HELAD WITHOUT ADVERSARIAL ATTACKS. ADV_TEST REPRESENTS THE ACCURACY OF RFG-HELAD IN THE PRESENCE OF ADVERSARIAL ATTACKS

model	clean_test	adv_test	AVG
DNN	0.996	0.719	0.857
DNN+advT	0.996	0.719	0.857
DNN+STFT	0.988	0.893	0.940
DNN+advT+STFT	0.988	0.893	0.940
DNN+CL	1.000	0.772	0.886
DNN+CL+advT	0.991	0.789	0.890
DNN+CL+STFT	0.975	0.911	0.943
DNN+CL+advT+STFT(ours) •	0.969	0.940	**0.954**

• *DNN+CL+advT+STFT(ours) means RFG-HELAD-K;*

TABLE V

DETECTION EFFECTIVENESS OF UNKNOWN ATTACK PROCESSING ALGORITHMS IN KNOWN ATTACKS (ACC-K). NSL(3:3) REPRESENTS A TRAINING SET WITH 3 KNOWN CATEGORIES AND AN ACTUAL TEST ENVIRONMENT WITH 3 KNOWN AND 3 UNKNOWN CATEGORIES. HERE IS THE DETECTION OF KNOWN ATTACKS, SO THE UNKNOWN ATTACKS IN THE TEST SET ARE REMOVED. SO, THE TRAINING SET AND TEST SET OF NSL ARE 3 CATEGORIES. OPENMAX(DNN)-C REFERS TO OPENMAX(DNN)+CENTERLOSS [23]

model	NSL (3:3)	Kitsune (6:4)	UKM (5:3)
DNN(softmax)	**1.000**	0.970	0.996
CNN(softmax)	**1.000**	0.970	**1.000**
openmax(DNN) [23]	**1.000**	0.970	**1.000**
scalable-NIDS [22]	0.970	0.778	0.829
CADE [24]	0.993	0.947	0.972
CVAE-EVT [21]	0.998	0.982	0.999
openmax(DNN)-C [23]	**1.000**	0.983	**1.000**
capnet [26]	0.997	0.666	0.994
gcm [28]	0.998	0.979	0.998
cgdl [27]	0.998	0.980	0.976
RFG-HELAD*-K	**1.000**	**0.984**	**1.000**

TABLE VI

$K+1$ CLASSIFICATION (WITH UNKNOWN ATTACK) RESULTS (ACC-(K + 1)). NSL(3:3) REPRESENTS A TRAINING SET WITH 3 KNOWN CATEGORIES AND AN ACTUAL TEST ENVIRONMENT WITH 3 KNOWN AND 3 UNKNOWN CATEGORIES. △ IMPLIES A TOP3 ALGORITHM ON THE UKM DATA SET. THE ORIGINAL OPENMAX(CNN) COULD NOT BE TRAINED IN OUR DATA SET CONFIGURATION

model	NSL (3:3)	Kitsune (6:4)	UKM (5:3)
DNN+CL(softmax)	0.499	0.591	0.859
DNN(softmax)	0.499	0.580	0.859
CNN(softmax)	0.499	0.580	0.859
openmax(DNN) [23]	0.661	0.754	0.878△
scalable-NIDS [22]	0.603	0.820	0.727
CADE [24]	0.847	0.589	0.837
CVAE-EVT [21]	0.559	0.590	0.864△
openmax(DNN)-C [23]	0.748	0.767	0.836
capnet [26]	0.819	0.554	0.861
gcm [28]	0.499	0.573	0.862
cgdl [27]	0.583	0.844	0.846
RFG-HELAD*-(K+1)	**0.928**	**0.918**	**0.948**△

the model without unknown attack detection. This type of model can only perform K classification. These algorithms are standard baselines, which include DNN, CNN, DNN+CL. The second category is the state-of-the-art models for unknown attacks in the field of network anomaly detection. These include openmax(DNN), scalable-NIDS, CADE, CVAE-EVT, openmax(DNN) with centerloss. openmax based on CNN is the pioneering work in the field of open set identification. In our experiments, CNN-based openmax does not perform as well as DNN-based openmax. Thus, we choose DNN-based openmax method for comparison. The third class of comparison algorithms are excellent open-set problem recognition algorithms in the field of computer vision, which include capnet [26], cgdl [27], and gcm [28].

E. Answers to Key Questions and Analysis of Experimental Results

We conduct a series of experiments in this section to verify our model.

1) How Does the Performance of Our Algorithm on the Fine-Grained Classification Submodel (K Classification), Compared With Other State-of-the-Art Algorithms: In this section, we compare our model with traditional machine learning/deep learning algorithms and SOTA fine-grained attack detection algorithms. As you can see from the TABLE III, the original DNN has a better detection performance for such type of time-series data as traffic. This is because DNNs have superiority in network traffic classification (Part 8 of Appendix). CNN, as an excellent classification algorithm, achieves good results with an accuracy of 1.000 in the UKM data set. However, with the complexity of the data set, the convolutional pooling layer loses some important information for non-image data. Furthermore, the CNN model is more sensitive to imbalanced data. In contrast, DNN is less sensitive to imbalance, which is also an advantage. DNN-kNN and SCADA are ensemble learning methods that generally outperform the use of a single classifier. However, there is still room for performance growth due to the lack of distribution optimization. FARE is an unsupervised/semi-supervised algorithm. For a fair comparison, we use all the labels in the training set. It can be found that it can infinitely approximate the detection effect of DNN. To evaluate the generality of the model for different data sets, we calculate the average of the results of the three data sets in the last column of the table. NSL is classical and there is a large amount of work based on this data set. We directly normalize the original data set to get features and observe the relative performance between classifiers. There are many ways to perform even better on this data set, e.g., more processing can be done at the feature level. Experimental results show that our K classification model (DNN+CL) achieves the best detection results. Experimental results show that different complex scenarios, our customized K model can effectively guarantee the performance and play a fundamental role.

2) How Does Our Algorithm's K Classification Submodel Perform in the Cases of Adversarial Attack: To verify the effectiveness of adding defence components, we design ablation experiments to observe the contribution of each component to the CHPDD defence strategy. Specifically, we conduct experiments on both fast gradient sign method (FGSM) [41] and projected gradient descent (PGD) [42] adversarial attack models. Our model has relatively good robustness in both attack environments. As shown in TABLE IV, when the test set is all perturbed data (i.e. adv_test), the accuracy of DNN alone is 0.719, while the accuracy of DNN+CL is 0.772. This shows that contrastive learning can improve robustness by compressing the low-dimensional space. The accuracy of our DNN+CL model improves to 0.911 after adding STFT, which is a significant improvement and shows that the STFT plays a key role. Finally, adding adversarial training to the DNN+CL+STFT model improves the accuracy to 0.940, which indicates that adversarial training can also improve robustness, but not significantly. That is, adversarial training can improve the robustness of the model to some

TABLE VII

COMPREHENSIVE METRICS COMPARISON BETWEEN OUR RFG-HELAD* MODEL AND THE TOP3 MODEL IN THE THREE DATA SETS(TABLE VI)

Data set	model	P	R	F1	AUC	TPR_a	FPR_a	FPR_2	FNR_2	ACC-(K+1)
UKM	CVAE-EVT	0.830	0.860	0.810	0.473	0.825	0.059	**0.001**	0.316	0.864
	openmax(DNN)	0.900	0.880	0.880	0.827	0.829	0.036	0.075	0.125	0.878
	RFG-HELAD*-(K+1)	**0.960**	**0.950**	**0.950**	**0.953**	**0.950**	**0.011**	0.053	**0.014**	**0.948**
Kitsune	CVAE-EVT	0.412	0.589	0.468	0.474	0.841	0.065	**0.012**	0.023	0.590
	openmax(DNN)	0.632	0.754	0.679	0.770	0.675	0.056	1.000	0.000	0.754
	RFG-HELAD*-(K+1)	**0.922**	**0.918**	**0.919**	**0.938**	**0.928**	**0.015**	0.058	**0.010**	**0.918**
NSL	CVAE-EVT	0.471	0.559	0.489	0.556	0.612	0.165	1.000	**0.000**	0.559
	openmax(DNN)	0.774	0.661	0.648	0.786	0.799	0.107	0.038	0.250	0.661
	RFG-HELAD*-(K+1)	**0.929**	**0.928**	**0.928**	**0.940**	**0.938**	**0.026**	**0.058**	0.032	**0.928**

TABLE VIII

COMPREHENSIVE METRICS COMPARISON BETWEEN OUR RFG-HELAD MODEL AND TOP3 MODEL UNDER UKM DATASET (TABLE VI) AND THE PRESENCE OF FGSM ADVERSARIAL ATTACK. $\odot$ STANDS FOR PRESENCE ADVERSARIAL TRAINING

model	P	R	F1	AUC	TPR_a	FPR_a	FPR_2	FNR_2	ACC-(K+1)
CVAE-EVT$\odot$	0.610	0.700	0.650	0.521	0.471	0.121	0.098	0.628	0.704
CVAE-EVT	0.580	0.540	0.540	0.544	0.489	0.124	0.363	0.398	0.539
openmax(DNN)$\odot$	0.470	0.690	0.560	0.500	0.166	0.166	**0.000**	1.000	0.685
openmax(DNN)	0.840	0.700	0.730	0.785	0.588	0.067	0.286	**0.083**	0.696
RFG-HELAD-(K+1)	**0.900**	**0.890**	**0.890**	**0.809**	**0.847**	**0.030**	0.070	0.100	**0.891**

TABLE IX

ABLATION EXPERIMENTS(ACC-(K + 1)). THE LAST LINE REPRESENTS OUR RFG-HELAD*-(K + 1) MODEL. THE COMPONENTS OF ADVERSARIAL DEFENCE HAVE BEEN DISCUSSED IN THE PREVIOUS PART OF EXPERIMENTS AND WILL NOT BE DISCUSSED HERE. GAN-FEA AND CL-FEA REPRESENT THE HIDDEN LAYER FEATURES GENERATED BY GAN AND DNN+CL, RESPECTIVELY

	Model component						Data set		
Combination method	DNN+CL	DCGAN	RPGAN	GAN-fea	CL-fea	Deep kNN	UKM	NSL	Kitsune
1)	√	×	×	×	×	×	0.859	0.499	0.591
2)	√	√	×	×	×	×	0.911	0.850	0.852
3)	√	×	√	×	×	×	0.920	0.858	0.865
4)	√	×	√	√	×	√	0.932	**0.952**	0.840
5)	√	×	√	×	√	√	0.946	0.941	0.820
6)	√	×	√	√	√	√	**0.948**	0.928	**0.918**

extent. However, the Fourier feature transformation is more effective in enhancing the robustness of our model. Experiments show the correctness of following the detection process sequentially to enhance robustness. Looking at the DNN+CL and DNN+CL+advT+STFT models of clean_test, it can be seen that the accuracy of RFG-HELAD-K (with adversarial defence) decreases a little in the absence of adversarial attacks. This is because using adversarial defence in the absence of an adversarial attack will degrade detection performance [47]. This suggests that there are also scenarios where RFG-HELAD* is used alone. Similar analysis can be done for the PGD experiments in Part 5 of Appendix.

3) How Does Our Model Perform in an Environment With Unknown Attacks and Without Adversarial Attack, Compared With Other State-of-the-Art Algorithms ($K+1$ Classification): We first test the unknown attack detection algorithm in a K classification scenario. As shown in TABLE V, almost most of the models achieved good classification results, which indicates the effectiveness of feature extraction for each data set. Next, to verify that our unknown attack detection model is valid ($K+1$ classification), we first compare it with K-class classifiers that cannot handle unknown attack. Second, we compare with the SOTA unknown attack detection algorithm. All models used for comparison are fed with the same features as our model. As you can see from the TABLE VI, CVAE-EVT and DNN+opemax achieve good detection results on the UKM data set. But on the Kitsune data set, scalable-NIDS and cgdl algorithms perform better. Our model performs the best on all data sets and the minimum ACC-(K + 1) of our algorithm can reach 0.918. This indicates that our method is effective and has good generality. CADE detects unknown attacks by contrastive learning and distance algorithms. The results become poor when the distribution of unknown attacks is complex. The shortcoming of scalable-NIDS, which is also distance-based, is that it does not distance the unknown attack from the known ones. Regarding CVAE-EVT, we drop the clustering part of CVAE-EVT in order to be consistent with other methods. When the real unknown attack distribution only partially conforms to the extreme value theory, the improvement in the detection of unknown attacks will be insignificant. Algorithms in this category including CVAE-EVT,openmax(DNN) + centerloss, openmax(DNN). cgdl, gcm. capnet can detect unknown classes very well in the domain of images. However, the model is too customized and not very suitable for unknown attack detection in the field of network anomaly detection.

Further, to verify whether ACC-(K + 1) can respond well to the detection performance of unknown attacks, our model and top 3 algorithms are compared on three data set for more metrics. In an environment with unknown attacks and without adversarial attack, we use the RFG-HELAD*-(K+1) model. As you can see from TABLE VII, our detection model performs best on almost all metrics. In particular, regarding the evaluation of false positives in fine-grained attacks, FPR_a

is more reasonable. Our model has the lowest FPR_a. Furthermore, the FPR_2 metrics for binary classification do not fully reflect the multi-classification attack detection, which can only be used as a reference value. For example, CVAE-EVT has a low FPR_2, but a very high FNR_2. Therefore, it is reasonable to use accuracy as the main assessment indicator.

4) How Does Our Algorithm Perform in the Presence of Unknown Attacks and Adversarial Attack, Comparing Multiple Metrics With Other Top Algorithms: We evaluate the overall RFG-HELAD model in the presence of unknown attacks and adversarial attack. Specifically, we choose the top 3 algorithms to experiment on UKM data set. As you can see from the TABLE VIII, regarding ACC-(K + 1), we are at least 18.7% higher than the case of two SOTA algorithms with adversarial training, which indicates our RFG-HELAD-(K + 1) model can effectively cope with these problems. Similarly, comparing to other models, our model has the lowest FPR_a.

5) Are the Individual Components of Our Model Effective (Ablation Experiments): We have previously verified the validity of the components of the adversarial defence, and next we verify the validity of the components of the RFG-HELAD*-(K + 1) model which does not take into account the adversarial defence. We conduct model ablation experiments on three data sets. TABLE IX shows ablation experiments of our RFG-HELAD*-(K + 1) model. The first row is DNN+CL, which is a baseline model for K class classification. The second row is a separate addition of a DCGAN and output of the discriminator is used directly as a judgement for unknown attacks. RPGAN (GAN with two discriminators) are used in the third row, which obviously increases the generalization performance of the model. Fourth row indicates the hidden layer features are generated using only GAN (input to Deep kNN). The fifth row represents the hidden layer features generated using only contrastive learning (input to Deep kNN). The last row denotes the complete our RFG-HELAD*-(K + 1) model. For the first three rows, when Deep kNN is not selected we discuss it in two cases. The first row is the classification result using DNN+CL directly. The second and third rows are cases where DNN+CL and GAN models are combined. In this case we first use the discriminator of GAN to directly discriminate whether it is an unknown attack or not, and then use DNN+CL to classify the known attacks.

As can be seen from TABLE IX, the detection performance of the last row is better than that of the first three rows, which illustrates the effectiveness of ensemble learning. In particular, on the NSL data set, the accuracy using the hidden layer features of GAN alone can reach 0.952. There are three specific reasons for this. Firstly, the NSL data set is simpler and it is easier to learn an effective feature representation. Secondly, NSL data set has a 3:3 ratio of known attacks to unknown attacks, which is the case where the number of unknown attacks is high. GAN performs better for the case where there are more unknown attacks. For the more complex Kitsune data set, the performance gains from ensemble learning are clear. Overall, ensemble learning is necessary in the model, as are the other components.

F. Analysis of Issues Related to Model Runtime and Deployment

1) How Efficient is Our Model in the Existing Hardware Environment: We use a server with Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz and 128G of RAM as the device. The type of GPU we use is GTX 1080Ti. We perform the tests using UKM's 8,923 training set and 2,578 testing set and record the maximum values of CPU and memory usage. Our model is divided into three parts and executed sequentially. Therefore, we perform the tests separately. As can be seen from TABLE X and TABLE XI, our model requires a relatively low percentage of memory and a slightly higher CPU, which is acceptable.

TABLE X

EVALUATION OF OUR MODEL HARDWARE REQUIREMENTS ON UKM'S 2578 TESTING SET

Component	Memory	Memory (%)	CPU (%)	Test time
DNN+CL	513.2MB	0.4%	9.2%	0.643s
GAN	1,155.3MB	0.9%	13.9%	1.070s
Deep kNN	385.1MB	0.3%	3.9%	0.272s

TABLE XI

EVALUATION OF OUR MODEL HARDWARE REQUIREMENTS ON UKM'S 8923 TRAINING SET. TOE DENOTES THE TIME TO TRAIN AN EPOCH. WT HERE DENOTES THE TIME WHEN THE MODEL IS WELL TRAINED

Component	Memory	Memory (%)	CPU (%)	TOE	WT(epochs)
DNN+CL	2,052.8MB	1.6%	28.1%	2.577s	36.564s(15)
GAN	9,750.8MB	7.6%	40.7%	9.235s	31.133s(3)
Deep kNN	385.1MB	0.3%	4.7%	0.456s	0.456s(-)

2) Regarding Runtime, How Does Our Algorithm Compare to Various Top 3 Algorithm: In real applications, the training of the model can be done offline, and we are concerned about the speed of detection. So, let's first analyze the time complexity of the detection part. Suppose the training set has m packets and the test set contains n packets. In the detection phase, the time for a single packet to run on DNN+CL (storing the hidden features) is T_1, and the time for a single packet to run on GAN (storing the hidden features) is T_2. The time to compute the pairwise packet distances in the Deep kNN module is T_{31}, while the time for the threshold judgment of each packet is T_{32}. Also with respect to Deep kNN, the number of elements in the selected subset of the training set used for distance calculation is m_s $(1 <= m_s <= m)$. The overall time complexity of the detection part is $\mathcal{O}(T_1 * n + T_2 * n + m_s * n * T_{31} + n * T_{32})$. For all the above experiments, our m_s is equal to m. When the size of the training set is large, a representative m_s can be selected for retrieval. For example, for samples in the training set that are very close together, only one is selected. The m_s is chosen only in the third stage of our model (Deep kNN), so it does not affect the representation ability of known and unknown attacks. The m_s are chosen to be representative, so the distances calculated are also differentiated. Overall, the choice of m_s does not affect the performance. Further, in the real world, the training sets are manually handpicked data that is very limited. The actual network traffic for testing is always continuous and massive. Thus, $m \ll n$ is satisfied. In our future work, we may consider controlling the size of m_s to a multiple of constant related to the number of attack categories with full consideration of intra-class concept drift.

Detection accuracy and speed are both important metrics. To evaluate the detection speed of our algorithm, we compare it with top3 algorithms in each of the three scenarios. As can be seen from TABLE XII, TABLE XIII, and TABLE XIV, our detection speeds are all superior. In particular, in TABLE XIV, the test set is used twice. The test set here is a splice of the original test set and the test set after the perturbation, which is

TABLE XII

TEST TIME COMPARISON BETWEEN OUR RFG-HELAD*-K MODEL AND THE TOP3 MODEL IN THE UKM DATA SET (CONSISTENT WITH TABLE III'S CONFIGURATION, UKM'S 2226 TESTING SET)

model	DNN	Scada	RFG-HELAD*-K
Test time	0.300s	1.073s	0.324s

TABLE XIII

TEST TIME COMPARISON BETWEEN OUR RFG-HELAD*-(K + 1) MODEL AND THE TOP3 MODEL IN THE UKM DATA SET (CONSISTENT WITH TABLE VII'S CONFIGURATION, UKM'S 2578 TESTING SET)

model	CVAE-EVT	openmax(DNN)	ours
Test time	2.129s	2.566s	1.985s

• *Here, ours means RFG-HELAD*-(K+1);*

TABLE XIV

TEST TIME COMPARISON BETWEEN OUR RFG-HELAD-(K + 1) MODEL AND TOP3 MODEL UNDER UKM DATA SET. N/Y STANDS FOR WHETHER OR NOT ADVERSARIAL TRAINING WAS USED (CONSISTENT WITH TABLE VIII'S CONFIGURATION, USING UKM'S 2578 TESTING SET TWICE)

model	CVAE-EVT(N/Y)	openmax(DNN)(N/Y)	ours
Test time	4.305s/4.317s	5.139s/5.187s	3.821s

• *Here, ours means RFG-HELAD-(K+1);*

equivalent to the original test set being used twice. The reason for this setup is to be closer to the real situation (even if there is a adversarial attack, the traffic without adversarial attack will still be captured in the test traffic). All of the above are the test times for corresponding test set. We use TCP Flood in ukm to test the DDoS detection time. There are 121 records in TCP Flood in the test set. Test time of our RFG-HELAD*-K is 0.017s, and the test time of our RFG-HELAD*-(K + 1) is 0.301s. According to the longer test time of our RFG-HELAD*-(K + 1) model, the test time of a single flow is 0.002s, which indicates that our model has a fast detection speed.

3) How are Our Models Deployed: Our attack detection is currently offline, similar to [33], and we can use traffic replay to verify the landable deployment of our algorithm. Our traffic is not altered in any way and can obviously be replayed online. The specific online deployment detection scheme is shown in Fig. 6. We can use virtual machines and Dockers to simulate the experiments referring to [8] and [33], which simulates the experimental platform for the Kitsune data set. Then, we use Tcpreplay and Tcplivereplay to replay on the testbed.

As you can see, every part of our process is landable. This proves that our model is fully deployable. Regarding the packet throughput, we can refer to Kitsune [8] for the analysis of feature extraction efficiency. The speed in its original paper is relatively substantial. Regarding the case where the data source is flow, we just change the corresponding feature extraction part. The detection speed and advantages of our model have been demonstrated in Section-V-F-I and Section-V-F-II. Because the adversarial defense is costly, the cost is to affect the detection accuracy in the environment without adversarial attacks. Therefore, our adversarial defense module is optional.

4) Does Our Model Require Retraining: The trained model has limitations to some extent. This is because network attacks are constantly evolving and changing. For this reason, we should update the model when unknown attacks are

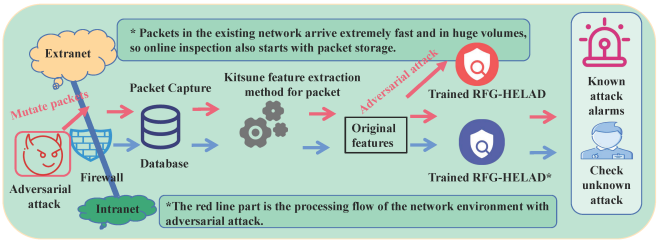


Fig. 6. Deployment diagram of our model.

detected and analyzed manually. We perform experimental validation on the UKM data set. Our accuracy is 0.948 (ACC-(K + 1)) in the case of 5 train 8 test, where 5 train 8 test means that there are 5 categories in the training set and 8 categories in the test set and that all categories in the training set appear in the test set. After that, we retrain and our accuracy is 0.998 (ACC-(K + L)) in the case of 8 train 8 test. This illustrates the need for retraining.

VI. DISCUSSIONS

In this section, we first discuss issues related to the model. Then, we discuss the limitations and future work.

A. Discussion of Model-Related Issues

Q1: Is the distribution of unknown and known attacks well distinguishable in our model detection?

In order to explore the intuition behind the detection mechanism, we quantitatively analyze the distributional features for unknown and known attacks. We set up two sets of experiments to show that the distribution of unknown attacks can be relatively well distinguished from that of known attacks (using Euclidean distance calculation of features). For the first set of experiments, we set up the following: 1) Compute multiple distances between known attacks in the test set and known attacks in the training set, 2) Calculate multiple distances between unknown attacks in the test set and known attacks in the training set. Multiple distances include closest, farthest and average. We introduce the distance calculation method by taking an unknown attack as an example. An unknown attack sample in the test set finds the closest sample in the training set and distance dis_1 is obtained. All the unknown attack samples in the test set are calculated in this way to obtain an array of distances. We sort this array in ascending order and then denote it as $array_f$. The smallest distance in $array_f$ is the closest and the largest is the farthest. The average value of the array is the average distance. Next, we compute the cumulative distribution function based on $array_f$ in both cases(unknown/known). The features used here are our combined CL-fea+GAN-fea features. The experimental results are shown in TABLE XV and Fig. 7. It is clear that the distance from unknown attacks in the test set to known attacks in the training set is much larger. This makes it easier to distinguish between unknown and known attacks.

For the second set of experiments, we compute the average distance between unknown and known attacks in the test set. Unlike the previous method of distance calculation, we calculate the distance between each sample of unknown attacks and each sample of known attacks in the test set. These distances are averaged and the result is denoted as direct average distances. We set up different features for comparison. Specifically, we observe the following three cases: GAN-fea,

TABLE XV

CALCULATE THE DISTANCE BETWEEN KNOWN/UNKNOWN ATTACKS IN THE TEST SET AND KNOWN ATTACKS IN THE TRAINING SET

Data set	Category	Closest	Farthest	Average
UKM	Known	3.576e-06	0.280	0.001
	Unknown	3.840e-04	0.558	0.085
Kitsune	Known	0.001	0.308	0.021
	Unknown	0.004	0.812	0.233
NSL	Known	0.000	0.076	2.245e-4
	Unknown	1.070e-04	0.586	0.026

TABLE XVI

COMPUTE THE DIRECT AVERAGE DISTANCE BETWEEN UNKNOWN AND KNOWN ATTACKS IN THE TEST SET IN THE CASE OF DIFFERENT FEATURES

Case	UKM	Kitsune	NSL
CL-fea	0.633	1.146	0.963
GAN-fea	0.198	0.162	0.094
CL-fea+GAN-fea (ours)	0.713	1.166	0.985

TABLE XVII

PERFORMANCE OF OUR MODEL UNDER DIFFERENT UNKNOWN ATTACK TYPE RATIOS (ACC-(K + 1), UKM DATA SET). 5:3 MEANS FIVE KNOWN ATTACK TYPES AND THREE UNKNOWN ATTACK TYPES. KNOWN/UNKNOWN REPRESENTS THE NUMBER OF KNOWN AND UNKNOWN ATTACKS IN THE TEST SET

Proportions	Known/unknown	RFG-HELAD*-K	our*-(K+1)
8:0	2578/0	0.998	0.998
7:1	2470/108	0.959	0.980
6:2	2349/229	0.908	0.955
5:3	2226/352	0.859	0.948
4:4	2110/468	0.818	0.952

• *Here, our*-(K+1) means RFG-HELAD*-(K+1);*

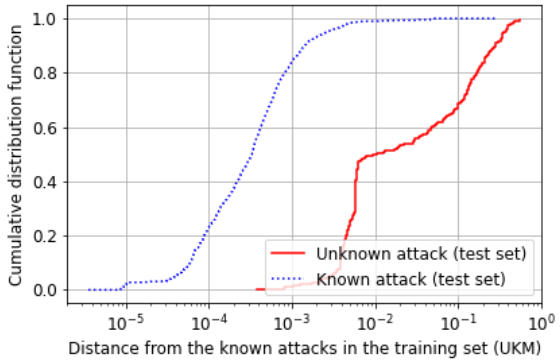
CL-fea, and CL-fea+GAN-fea(ours). The experimental results are shown in TABLE XVI. It is obvious that the distance increases after the combination, which is more beneficial to distinguish the known attacks from the unknown ones. These two sets of experiments further illustrate the generalization and effectiveness of our model in open-set scenarios. Our model can separate unknown and known attacks well and achieve better detection results. The essence of this is that the components of our model, DNN+CL and RPGAN, can learn well the distributions used to distinguish between known and unknown attacks.

Q2: How does our model perform with varying proportions of known and unknown attack types?

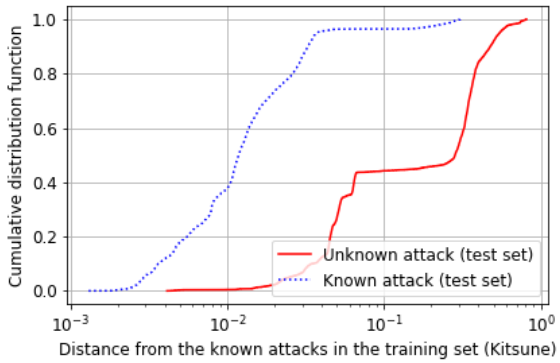
In the previous experiments, we have well demonstrated our superiority over other SOTA algorithms. Here, we use the UKM data set to test the effectiveness of our model in detecting unknown attacks at different ratios of unknown attack types. If the unknown attack types are larger than the known attack types and most of the attacks have to be manually relabeled, then it makes less sense to detect the unknown attacks. Therefore, we set the maximum unknown attack type ratio to 0.5 (in the case of 4:4). As shown in TABLE XVII, our model has good detection results under various unknown attack type ratios.

Q3: How does our model perform in other data sets?

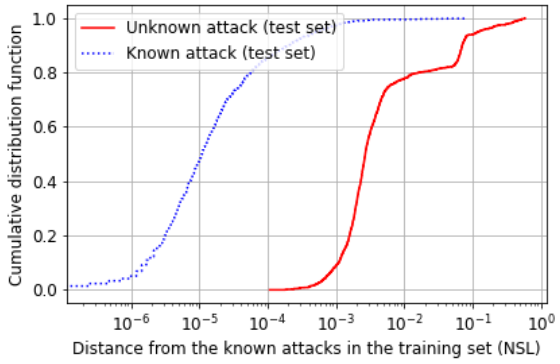
In the preceding experiments, we have sufficiently demonstrated the effectiveness of our model's adversarial defense. To understand how well our model detects on other data sets, we add K and $K+1$ experiments and compare our model to



(a) UKM



(b) Kitsune



(c) NSL

Fig. 7. Distance between known/unknown attacks in the test set and known attacks in the training set.

the top3 comparison algorithms. We add CICIDS2018 [45] and TON2019 [61] data sets for experiments. The training set of CICIDS2018 is [Benign,SSH-Bruteforce, DoS-Hulk] [12000, 4000, 10000] and the test set is [Benign, SSH-Bruteforce, DoS-Hulk, Infiltration] [3249, 346, 1697, 4238]. What follows is the number of attacks. The training set of TON2019 is [backdoor, ddos, dos, injection, normal] [800 800 800 800 5600] and the test set is [backdoor, ddos, dos, injection, normal, password, scanning, xss] [200 200 200 200 1400 200 200 200]. As shown in TABLE XVIII and TABLE XIX, our model has excellent performance, which suggests that our model has good detection performance in other data sets as well.

Since the CICIDS2018 dataset is more specific in terms of the collection time of each type of attack, we further evaluate the detection speed of our model. We test it using the full attack data of SSH-Bruteforce, DoS-Hulk, and Infiltration in the CICIDS2018 data set. As shown in TABLE XX, the

TABLE XVIII
K CLASSIFICATION EXPERIMENTS ON OTHER DATA SETS (ACC-K, DROPPING UNKNOWN ATTACKS)

model	CICIDS2018(3:1)	TON2019(5:3)
DNN	1.000	1.000
SCADA	1.000	1.000
ours (RFG-HELAD*-K)	1.000	1.000

TABLE XIX
$K+1$ CLASSIFICATION EXPERIMENTS ON OTHER DATA SETS (ACC-(K + 1))

model	CICIDS2018(3:1)	TON2019(5:3)
RFG-HELAD*-K	0.555	0.787
CVAE-EVT	0.555	0.783
openmax(DNN)	0.555	0.880
ours(RFG-HELAD*-K+1)	0.994	0.924

TABLE XX
COMPARISON OF DATA COLLECTION TIME WITH THE DETECTION TIME OF OUR MODEL IN CICIDS2018 DATA SET

Attacks	Number	Collection time	Detection time
SSH-Bruteforce	187,589	90 min	76s
DoS-Hulk	461,912	34 min	189s
Infiltration	68,871	155 min	32s

TABLE XXI
VALIDATING THE PERFORMANCE OF OUR MODEL ON LARGE KITSUNE DATA SET

model	P	R	F1	AUC	ACC-(K+1)
RFG-HELAD*-K	0.800	0. 860	0.820	0.449	0.856
ours•	0.940	0.930	0.930	0.995	0.934
Setup: Train: 733582 Test-known:82114 Test-unknown: 7886					

• *Here, ours means RFG-HELAD*-(K+1);*

detection time of our model is less than the data collection time, which further indicates the superiority of our model's detection speed.

Q4: How do our models perform in the case of unknown attacks and large data sets?

We use the large Kitsune data set of 900,000 for K + 1 experiments. As shown in TABLE XXI, our RFG-HELAD* model detection can achieve an accuracy of 0.934, which is better than the small data set (0.918 in TABLE VI). The experiment illustrates that a large amount of data can slightly improve the detection effect.

Q5: Our model uses FGSM and PGD as adversarial attack, does this affect the evaluation of defense performance?

Both feature-level adversarial attacks and traffic-space adversarial attacks are essentially perturbations to bypass the classifier's detection. The actual adversarial attacks generated at the traffic level have more constraints [33]. Moreover, feature-level attacks are generally more effective because they are not constrained at the physical level. Therefore, the choice of feature level for the adversarial attack does not affect our evaluation of the adversarial robustness.

B. Limitation and Future Work

For space reasons, some of our model-related issues were not included in the design framework. This leaves our framework extensible.

The first one is attack class imbalance, which is a classic problem for network traffic anomaly detection [48]. Similar to [49], our future work can use data augmentation methods to generate categories with small amount of data.

The second one is the label acquisition problem. Many works use semi-supervised methods [10], [17], [29], and our model can be combined with more general semi-supervised methods, such as Mean teacher [50] and Mixmatch [51].

VII. CONCLUSION

Network attack detection is an important network management task. The continuous changes in the network attack environment make anomaly detection more and more difficult. Among them, fine-grained attack detection is an important tool for attack detection and analysis. As technology upgrades, new attacks are constantly emerging, which places higher requirements on detection models. Current mainstream detection models are based on ML/DL, which can potentially have the problem of adversarial attacks. For this reason, we propose the RFG-HELAD model, which integrates a currently effective deep learning model (RPGAN) with a technique (adversarial defense using STFT in IDS) that can well solve the above problem. Experiments show that our model outperforms other SOTA models and can be well used in practical grounded scenarios. Specifically, in the presence of unknown and adversarial attacks, the accuracy of our model improves by at least 18.7% compared to the corresponding SOTA model with adversarial defense. In the future, our work will be positioned on the problem of data augmentation.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the anonymous reviewers whose constructive comments have greatly helped to improve this manuscript.

REFERENCES

[1] D. E. Denning, "An intrusion-detection model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 222–232, Feb. 1987.
[2] SNORT. (2019). *Snort 2.9.7.6*. [Online]. Available: https://www.snort.org/
[3] (2018). *Suricata 4.0.4*. [Online]. Available: https://suricataids.org/about/
[4] A. Aldweesh, A. Derhab, and A. Z. Emam, "Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues," *Knowl.-Based Syst.*, vol. 189, Feb. 2020, Art. no. 105124.
[5] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 305–316.
[6] A. Radford et al., "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. ICLR*, 2016.
[7] T. Yang et al., "Elastic sketch: Adaptive and fast network-wide measurements," in *Proc. Conf. ACM Special Interest Group Data Commun.*, Aug. 2018, pp. 561–575.
[8] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018.
[9] Y. Zhong et al., "HELAD: A novel network anomaly detection model based on heterogeneous ensemble learning," *Comput. Netw.*, vol. 169, Mar. 2020, Art. no. 107049.
[10] J. Camacho et al., "Semi-supervised multivariate statistical network monitoring for learning security threats," in *Proc. TIFS*, 2019, pp. 2179–2189.
[11] P.-F. Marteau, "Random partitioning forest for point-wise and collective anomaly detection—Application to network intrusion detection," in *Proc. TIFS*, 2021, pp. 2157–2172.
[12] C. A. de Souza, C. B. Westphall, R. B. Machado, J. B. M. Sobral, and G. D. S. Vieira, "Hybrid approach to intrusion detection in fog-based IoT environments," *Comput. Netw.*, vol. 180, Oct. 2020, Art. no. 107417.

[13] J. Gao et al., "Omni SCADA intrusion detection using deep learning algorithms," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 951–961, Jan. 2021.

[14] A. Mustafa, S. Khan, M. Hayat, R. Goecke, J. Shen, and L. Shao, "Adversarial defense by restricting the hidden space of deep neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3384–3393.

[15] A. Meinke, J. Bitterwolf, and M. Hein, "Provably adversarially robust detection of out-of-distribution data (almost) for free," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. Curran Associates, Jan. 2022, pp. 30167–30180. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/c2c62117283dda155db754e54dbe8d71-Paper-Conference.pdf

[16] C. Fu, Q. Li, M. Shen, and K. Xu, "Realtime robust malicious traffic detection via frequency domain analysis," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2021, pp. 3431–3446.

[17] Y. Li, X. Yuan, and W. Li, "An extreme semi-supervised framework based on transformer for network intrusion detection," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2022, pp. 4204–4208.

[18] M. Kravchik and A. Shabtai, "Efficient cyber attack detection in industrial control systems using lightweight neural networks and PCA," *IEEE Trans. Depend. Secure Comput.*, vol. 19, no. 4, pp. 2179–2197, Jul. 2022.

[19] A. F. Diallo and P. Patras, "Adaptive clustering-based malicious traffic classification at the network edge," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2021, pp. 1–10.

[20] J. Liang, W. Guo, T. Luo, V. G. Honavar, G. Wang, and X. Xing, "FARE: Enabling fine-grained attack categorization under low-quality labeled data," in *Proc. NDSS*, 2021.

[21] J. Yang, X. Chen, S. Chen, X. Jiang, and X. Tan, "Conditional variational auto-encoder and extreme value theory aided two-stage learning approach for intelligent fine-grained known/unknown intrusion detection," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3538–3553, 2021.

[22] Z. Zhang, Y. Zhang, D. Guo, and M. Song, "A scalable network intrusion detection system towards detecting, discovering, and learning unknown attacks," *Int. J. Mach. Learn. Cybern.*, vol. 12, no. 6, pp. 1649–1665, Jun. 2021.

[23] C. Wang, "Intrusion detection for industrial control systems based on open set artificial neural network," *Proc. Secur. Commun. Netw.*, vol. 2021, pp. 1–14, Aug. 2021.

[24] L. Yang et al., "CADE: Detecting and explaining concept drift samples for security applications," in *Proc. 30th USENIX Secur. Symp. (USENIX Security)*, 2021, pp. 2327–2344.

[25] A. Bendale and T. E. Boult, "Towards open set deep networks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, Aug. 2016, pp. 1563–1572.

[26] Y. Guo, G. Camporese, W. Yang, A. Sperduti, and L. Ballan, "Conditional variational capsule network for open set recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 103–111.

[27] X. Sun, Z. Yang, C. Zhang, K.-V. Ling, and G. Peng, "Conditional Gaussian distribution learning for open set recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, Jun. 2020, pp. 13477–13486.

[28] Z. Yue, T. Wang, Q. Sun, X.-S. Hua, and H. Zhang, "Counterfactual zero-shot and open-set visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15404–15414.

[29] L. Fan et al., "AutoIoT: Automatically updated IoT device identification with semi-supervised learning," *IEEE Trans. Mobile Comput.*, vol. 22, no. 10, pp. 5769–5786, Oct. 2023.

[30] S. Xu, L. Li, H. Yang, and J. Tang, "KCC method: Unknown intrusion detection based on open set recognition," in *Proc. IEEE 33rd Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2021, pp. 1343–1347.

[31] G. Ping and X. Ye, "Open-set intrusion detection with MinMax autoencoder and pseudo extreme value machine," in *Proc. Int. Jt. Conf. Neural Netw. (IJCNN)*, 2022, pp. 1–8.

[32] I. Rosenberg, A. Shabtai, Y. Elovici, and L. Rokach, "Adversarial machine learning attacks and defense methods in the cyber security domain," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–36, Jun. 2022.

[33] D. Han et al., "Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2632–2647, Aug. 2021.

[34] P. Khosla et al., "Supervised contrastive learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 18661–18673.

[35] J. Winkens et al., "Contrastive training for improved out-of-distribution detection," 2007, *arXiv:2007.05566*.

[36] T. D. Nguyen, T. Le, H. Vu, and D. Q. Phung, "Dual discriminator generative adversarial nets," in *Proc. NIPS*, 2017, pp. 2670–2680.

[37] S. Kong and D. Ramanan, "OpenGAN: Open-set recognition via open data generation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 793–802.

[38] L. Bergman, N. Cohen, and Y. Hoshen, "Deep nearest neighbor anomaly detection," 2020, *arXiv:2002.10445*.

[39] Y. Sun, Y. Ming, X. Zhu, and Y. Li, "Out-of-distribution detection with deep nearest neighbors," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 20827–20840.

[40] N. G. de Dick Bruijn, "Uncertainty principles in Fourier analysis," *Inequalities*, vol. 2, pp. 57–71, Aug. 1967.

[41] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. ICLR*, 2015.

[42] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. ICLR*, 2018.

[43] M. S. Al-Daweri, S. Abdullah, and K. A. Z. Ariffin, "An adaptive method and a new dataset, UKM-IDS20, for the network intrusion detection system," *Comput. Commun.*, vol. 180, pp. 57–76, Dec. 2021.

[44] NSLKDD. (2009). *NSL-KDD Data Set for Network-Based Intrusion Detection Systems*. [Online]. Available: http://nsl.cs.unb.ca/NSL-KDD/

[45] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*, 2018, pp. 108–116.

[46] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: CRC Press, 2012.

[47] X. Wang et al., "Protecting neural networks with hierarchical random switching: Towards better robustness-accuracy trade-off for stochastic defenses," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 6013–6019.

[48] S. Al and M. Dener, "STL-HDL: A new hybrid network intrusion detection system for imbalanced dataset on big data environment," *Comput. Secur.*, vol. 110, Nov. 2021, Art. no. 102435.

[49] S. T. Jan et al., "Throwing darts in the dark? Detecting bots with limited data using neural data augmentation," in *Proc. 41st IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 1190–1206.

[50] A. Tarvainen et al., "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. NIPS*, 2017.

[51] D. Berthelot et al., "MixMatch: A holistic approach to semi-supervised learning," in *Proc. NeurIPS*, 2019.

[52] Z. Zhang, Y. Zhang, J. Niu, and D. Guo, "Unknown network attack detection based on open-set recognition and active learning in drone network," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 10, p. e4212, 2022.

[53] M. Souza, C. Pontes, J. Gondim, L. P. F. Garcia, L. DaSilva, and M. A. Marotta, "A novel open set energy-based flow classifier for network intrusion detection," 2021, *arXiv:2109.11224*.

[54] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019.

[55] R. C. Qiu, Z. Hu, H. Li, and M. C. Wicks, "Machine learning," in *Cognitive Radio Communication and Networking: Principles and Practice*. Cham, Switzerland: Springer, 2013, pp. 283–321.

[56] M. Salehi, H. Mirzaei, D. Hendrycks, Y. Li, M. H. Rohban, and M. Sabokrou, "A unified survey on anomaly, novelty, open-set, and out of-distribution detection: Solutions and future challenges," *Trans. Mach. Learn. Res.*, 2022. [Online]. Available: https://openreview.net/forum?id=aRtjVZvbpK

[57] J. Ortiz et al., "DeviceMien: Network device behavior modeling for identifying unknown IoT devices," in *Proc. IoTDI*, 2019, pp. 106–117.

[58] X. Zhang, J. Chen, R. Zhang, C. Wang, and L. Liu, "Attacking recommender systems with plausible profile," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4788–4800, 2021.

[59] P. Addesso, M. Barni, M. Di Mauro, and V. Matta, "Adversarial Kendall's model towards containment of distributed cyber-threats," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3604–3619, 2021.

[60] X. Liu, T. J. Lim, and J. Huang, "Optimal Byzantine attacker identification based on game theory in network coding enabled wireless ad hoc networks," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2570–2583, 2020.

[61] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, Sep. 2021, Art. no. 102994.