MASiNet: Network Intrusion Detection for IoT Security Based on Meta-Learning Framework

Yiming Wu, Gaoyun Lin, Lisong Liu, Zhen Hong, Yangyang Wang, Xing Yang*, Zoe L. Jiang, Shouling Ji, Zhenyu Wen

***Abstract*—The rapid proliferation of Internet of Things (IoT) devices has led to an increased need for robust and efficient intrusion detection systems capable of identifying and mitigating novel threats. Traditional methods often struggle with the scarcity of labeled anomaly data, which is highly consequential, particularly in the context of IoT. In this study, we propose a novel few-shot learning approach by leveraging a Multi-Stage Attention Siamese Network (MASiNet) for network traffic intrusion detection based on meta-learning framework. Unlike traditional methods, the proposed MASiNet model is capable of detecting intrusions with minimal labeled samples, addressing the challenge of scarce anomaly data. The model is trained using various attack samples and evaluates unknown samples by comparing similarities with a small set of known attack types. A well-structured cost function design, incorporating two specific losses, is introduced to optimize the effectiveness of the training process. Tested on the NSL_KDD and UNSW-NB15 datasets in a simulated few-shot learning environment, the MASiNet model demonstrates superior performance in terms of accuracy, precision, False Alarm Rate (FAR), outperforming existing methods. Furthermore, we have validated our approach through real-world evaluations. The proposed method provides an effective solution for intrusion detection in the context of few-shot learning, offering a proficient solution that aligns with the dynamic nature of IoT networks.**

***Index Terms*— IoT security, Intrusion detection, Meta-learning framework, Few-shot learning.**

I. Introduction

With the growing ubiquity of the Internet in our daily lives, there is a significant rise in the number of interconnected devices that can communicate through networks. This increased connectivity, while bringing numerous benefits, also introduces concerns about cyberspace security [1]. One of the most critical dimensions of this security challenge revolves around the identification and swift detection of unauthorized or malicious activities within computer networks and systems, commonly known as an intrusion detection system (IDS).

The IoT (Internet of Things) environment, which involves a wide variety of devices, communication methods, and protocols, presents new security risks and challenges for intrusion detection that traditional cybersecurity systems struggle to handle. In recent years, Machine Learning (ML) has emerged as a promising approach for IDS in such dynamic environments. From the perspective of ML, IDS can be viewed as a classification task, where the system aims to accurately classify network traffic into normal or malicious categories [2]. ML algorithms demonstrate potential to effectively address the challenges posed by the IoT environment by leveraging various features of network traffic [3]. These algorithms, leveraging techniques like feature extraction, enhance the accuracy and efficiency of intrusion detection.

Despite their promise, ML-based IDS methods do have their limitations, particularly when it comes to adapting to novel and sophisticated cyber threats, where only a limited amount of network traffic data is available. These methods heavily rely on labeled training data and require a substantial number of representative examples to effectively train the models. Additionally, ML-based models need to be regularly updated and retrained to adapt to new attack types and evolving threat landscapes. This imposes significant challenges in maintaining up-to-date and comprehensive training datasets, as well as in continuously monitoring and updating the ML models to ensure their effectiveness.

Addressing these challenges, few-shot learning (FSL)-based IDS is introduced, this innovative approach trains the system to recognize and respond to new types of attacks, with minimal training data. Existing FSL methods face challenges in effectively capturing the complexity of varied network intrusions due to limited training sample sizes and struggle to adapt to evolving threats with techniques centered on feature extraction and fine-tuning from known data [4]. Other approaches focus on learning the differences between samples, but also lack nuanced understanding of network traffic patterns [5]. This paves the way for our novel intrusion detection model, the Multi-Stage Attention Siamese Network (MASiNet), which

* Xing Yang is the corresponding author.

Yiming Wu and Zhenyu Wen is with the Institute of Cyberspace Security, and the College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China, and also with the University of Science and Technology of China, Hefei, China. (e-mail: wyiming@zjut.edu.cn; zhenyuwen@zjut.edu.cn)

Gaoyun Lin is with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China. (e-mail: lingaoyun_zj@163.com)

Lisong Liu, Zhen Hong are with the Institute of Cyberspace Security, and the College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China. (e-mail: lls_zstu@163.com; zhong1983@zjut.edu.cn)

Zoe L. Jiang is with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Shenzhen, 518055, China, and also with the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Guangdong, 510632, China. (e-mail: zoeljiang@hit.edu.cn)

Shouling Ji is with the NESA Lab, College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China. (e-mail: sji@zju.edu.cn)

Xing Yang and Yangyang Wang are with the State Key Laboratory of Pulsed Power Laser Technology, and the Electronic Countermeasure Institute, National University of Defense Technology, Hefei 230037, China. (e-mail: yangxing17@nudt.edu.cn; wyyedu@163.com)

is specifically tailored to excel in such data-constrained environments. The key idea behind our approach lies in an enhanced comparative learning framework, designed to enable our model to effectively differentiate and compare network traffic samples. Central to this framework is the integration of a Siamese convolution network, which aids in learning similarity metrics between sample pairs. Complementing this, we subtly incorporate attention mechanisms for focused feature analysis and residual learning to boost learning efficiency and accuracy. By training the network to learn similarity metrics between pairs of samples, we can effectively capture the intrinsic characteristics of normal and malicious network traffic. This composite approach is beneficial in few-shot intrusion detection scenarios, as it allows for a comprehensive understanding of both normal and malicious network traffic, ensuring robust generalization even with limited labeled training data.

We conduct extensive experiments on benchmark datasets to evaluate the performance of our approach. The results demonstrate that our proposed method achieves superior performance compared to existing FSL-based IDS. At the same time, we also verify the practical feasibility of the model in the real experimental environment of the IoT. The ability to effectively leverage limited labeled training data makes our approach a promising solution for real-world intrusion detection tasks where obtaining a large amount of labeled data is often challenging or time-consuming.

The primary contributions of our work are outlined as follows:

1) We propose the MASiNet framework, a novel few-shot intrusion detection method, which prominently features two attention modules within the Siamese convolutional neural network. This framework provides effective capabilities in extracting network traffic features and is further enhanced by the integration of residual learning.
2) We introduce a composite loss function to improve the training performance of the MASiNet. First, we introduce a feature coding loss function for efficient and effective feature extraction in network traffic analysis. Additionally, we employ a dot product-based contrast loss function to enhance the network's comparative accuracy and robustness in analyzing traffic sample pairs.
3) To evaluate the practicality and effectiveness of the MASiNet approach in few-shot network intrusion scenarios, we conduct analysis in a customized real environment. The results of our experiments demonstrate the superiority of our method in terms of both practical applicability and performance.

II. Prior Knowledge and Related Work

A. Problem Contextualization: K-way-N-shot Meta-Learning

In our study, we consider a specific binary classification problem: distinguishing between normal sample and attack sample. As previously highlighted, conventional ML techniques grapple with limitations when confront with restricted sample sizes or a paucity of novel attack instances. To surmount this quandary, we introduce a K-way-N-shot meta-learning strategy [6].

The K-way-N-shot term refers to the setup of each learning task [7], [8]. Here, K represents the count of distinct classes or categories involved in each task, while N signifies the number of instances per class that the model encounters during the learning process. The K-way-N-shot meta-learning approach involves training the model on both "meta-training tasks" and culminates in the pursuit of the "meta-testing tasks". Within each task, the model is trained using a "support set" and subsequently evaluated using a "query set". The support set is composed of $K \times N$ examples, with N instances randomly drawn from each of the K classes. While the query set incorporates multiple samples from the K classes, typically distinct from those featured in the support set.

B. Related Work

Recent advancements in IDS for IoT have been marked by significant research contributions focusing on deep learning and network complexities. Hu et al. [9] introduced a one-class intrusion detection scheme for software-defined industrial networks, emphasizing dimension reduction for enhanced accuracy. Djenouri et al. [10] proposed a deep learning framework for anomaly detection in the Internet of Everything, combining advanced decomposition with neural networks and evolutionary computation. Duan et al. [11] utilized a dynamic line graph neural network with semi-supervised learning for intrusion detection, targeting the contextual dynamics in network communications. Lastly, Chen et al. [12] explored cross-domain industrial intrusion detection using a model trained on imbalanced data, applying information-enhanced adversarial domain adaptation. These studies collectively highlight the evolution of IDS, focusing on deep learning solutions for complex network environments. However, the majority of these methods demand extensive amounts of data for training, posing challenges for real network traffic scenarios where only a small percentage of data is abnormal, yet potentially the most damaging.

FSL has emerged as a critical deep learning technology, especially valuable in scenarios with limited labeled data, highlighting its significance in network intrusion detection where abnormal data is scarce [13–16]. Some FSL methods necessitated training with limited sample data during the training phase to enable the model to identify crucial sample features for testing [13]. Another methodology focused on learning feature extraction capabilities from known data, followed by fine-tuning to adapt to new types of attacks with only a handful of examples [14]. Expanding the horizon of FSL strategies, an alternative method involved training a model on an extensive dataset to discern "similarities and differences." This training empowered the model to utilize patterns learned from a smaller dataset, based on unseen data, for classifying samples during testing [17, 18]. This pioneering concept was introduced by [15], proposing FC-Net in the context of few-shot detection in network traffic. Building upon this foundation, [16] introduced the use of Siamese networks (FSL-SCNN) to enhance the learning of sample features.

Our work deeply investigates the effectiveness of attention mechanisms in the realm of meta-learning and proposes a

network intrusion traffic detection system based on attention Siamese networks. This system comprises a pair of convolutional neural networks with attention and residual mechanism, including a core network and a clone network sharing weights, each expressed through a two-level architecture consisting of an encoding stage and a decoding stage. This methodology strengthens the model's ability to learn sample features, resulting in enhanced performance on few-shot detection tasks.

III. Methodology

A. Overview

The architecture is outlined in Figure 1. The system encompasses three primary components: data preprocessing module, feature extraction module and similarity metric module.

1) *Data preprocessing module:* Within this module, the original one-dimensional network traffic data, initially perceived as a time series signal, is reshaped into a two-dimensional format. Subsequently, the data is partitioned into meta-training and meta-testing sets, each inclusive of support and query subsets.
2) *Feature extraction module:* This module leverages MASiNet to derive feature embeddings from the input samples. The MASiNet comprises a pair of convolutional neural networks with attention and residual mechanism, encompassing a core network and a clone network with shared weights, each is expressed through a two-stage architecture consisting of a encoding stage followed by a decoding stage.
3) *Similarity metric module:* This module orchestrates a comparison between the two sets of extracted feature embeddings. It quantifies the similarity between feature embeddings through dot product.

We take the advantage of meta-learning framework. In the meta-training phase, the feature extraction module is responsible for deriving feature embeddings for each sample through coding and encoding phase. Subsequently, the similarity metric module calculates the dot product between the feature embeddings of the support set and the query set samples, the obtained dot product is compared with the pre-set threshold to obtain the prediction label of the pair of samples. Throughout this process, both the coding loss and contrastive loss functions (as elaborated in Section III-D) are utilized to enhance the model's capabilities in feature extraction and classification. During the meta-testing phase, the feature extraction module employs pre-trained model to extract feature embeddings for both samples in support set and samples that need to be classified. This is followed by a matching process, where each sample that needs to be classified is compared with every support sample. The similarity between these samples is determined using the dot product, and the pair with the highest dot product, indicating the greatest similarity, is selected for classification.

An Example. We take an example to illustrate the overall process. In this scenario, we consider six types of network traffic samples: normal samples (O) and five distinct types of malicious samples (A, B, C, D, E). Among these, A, B, C, and D represent four known malicious attack types with ample labeled samples (e.g., DoS, DDoS, and port scanning attacks). In contrast, E signifies a novel attack type with only a limited number of available instances. In this context, the model's training involves discerning the differences between A, B, C, D, and O, and subsequently classifying an unknown sample by comparing its similarity with E and O, thereby determining whether it belongs to the class E attack or normal traffic O. Building upon the foundation laid in our earlier discussion of the K-way-N-shot meta-learning framework, K signifies the count of distinct classes or categories in each learning task, thus, in our study, we assign K as 2, signifying the normal and attack sample categories.

Specifically, our model undergoes a series of iterative training sessions during the meta-training phase, spanning a diverse array of tasks. This iterative process continues until the loss value attains a state of stabilization. Each distinct task within this phase consists of two discernible data sets: the support set and the query set. Each of these support and query sets encompasses two subgroups: the attack sample set and the normal sample set. During the formulation of these subsets, a specific type is selected from the available four types of attack samples (A, B, C, D). Within the support set, a total of N samples are randomly drawn from the chosen attack sample type to form the attack sample set. While remaining samples of the same attack sample type are then allocated to the attack sample set within the query set. Furthermore, both the support and query sets symmetrically augment their corresponding normal sample sets with an equal number of attack samples. After iterative training, the model improves its ability to learn on unseen categories by learning how to adapt and generalize to new categories.

To gauge the model's adaptability and learning efficacy towards new categories, we conduct meta-testing stage. Within the meta-testing stage, we only have a limited number of samples of the support set (N attack samples E and N normal samples O) to classify a mass of samples (E or O). For each sample to be classified, that sample is compared to all samples in the support set. The feature embedding dot product between the two samples in each comparison is computed. Specifically, the sample with classification is fed into the model to obtain the feature embedding $F(x)'$. The dot product of $F(x)'$ and all the feature embeddings $F(x_1), F(x_2), \ldots, F(x_{2N})$ in the support set is obtained to yield $\mathcal{P}(1)$ to $\mathcal{P}(2N)$:

$$\mathcal{P}(i) = F(x)' \cdot F(x_i)^T \tag{1}$$

The dot product $\mathcal{P}_i$ quantifies the similarity between two feature embeddings, with a larger dot product indicating greater similarity in the embedding space. Based on this principle, the classification of a sample x is determined by identifying the sample in the support set that yields the maximum dot product similarity:

$$\mathcal{C}(x) = \underset{i \in \{1,2,\ldots,2N\}}{\arg\max} \ \mathcal{P}_i \tag{2}$$

Here, $\mathcal{C}(x)$ denotes the assigned class of the sample x, and $\mathcal{P}_i$ represents the dot product similarity score of sample x with the i-th sample in the support set.

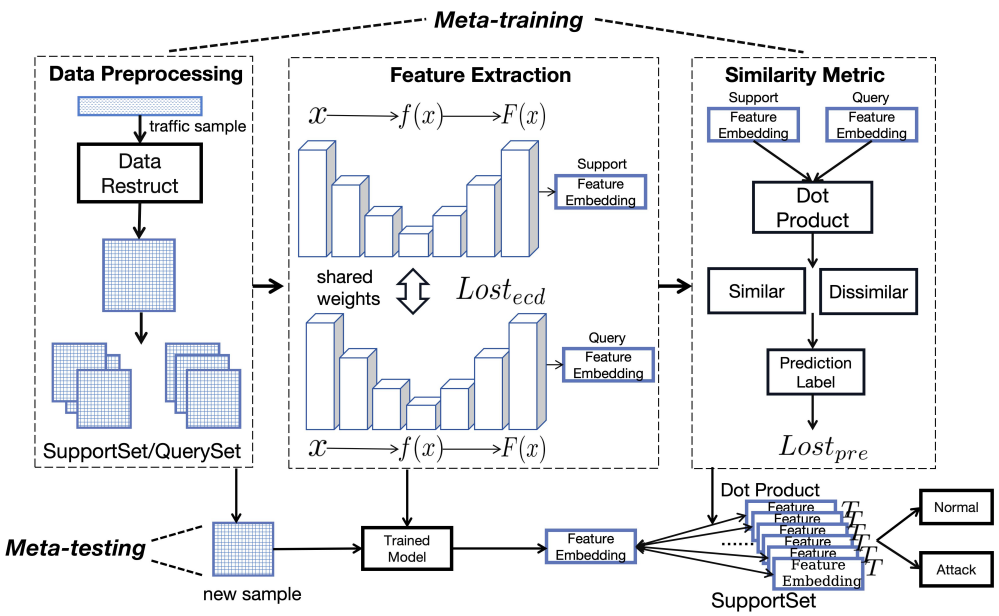


Fig. 1. A framework for network traffic intrusion detection based on meta-learning framework

B. Data Preprocessing Module

It is evident that network traffic data is highly intricate [19]. Therefore, the standardization and formatting of traffic data are required.

1) *Converting non-numeric attributes:* Network traffic contains non-numeric attributes, such as "Protocol_type", "Service", and "Flag". We convert these attributes into multi-dimensional vectors by utilizing one-hot encoding. As an example, the "Protocol_type" attribute, which consists of properties like tcp, udp, and icmp, is transformed into three-dimensional vectors: $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$ respectively.
2) *Handling constant attributes:* Static attributes, like "num_outbound_cmds" in the NSL_KDD dataset [20], contribute nothing to classification. The removal of these irrelevant attributes is crucial to ensure both the dataset's pertinence and efficiency in classification endeavors.
3) *Normalization for attribute value disparities:* To counteract the skewed influence of attributes with varying scales in machine learning models, a two-step normalization process is implemented:
   a) *logarithmic transformation:* Attributes undergo logarithmic scaling $x'_i = \log(1 + x_i)$ to harmonize their distribution, making them more uniformly comparable.
   b) *Min-Max normalization:* Post-logarithmic transformation, attributes are scaled within [0, 1] using min-max normalization $x''_i = \frac{x'_i - \min(x')}{\max(x') - \min(x')}$, ensuring equal contribution to the learning process.
4) *Data reshaping:* Furthermore, to facilitates a unified data structure, enabling more effective feature extraction and analysis, particularly for models like convolutional neural networks that require fixed-width inputs. we reshape one-dimensional data into a $h \times w$ two-dimensional format by padding with "0"s, where h and w represent the equal dimensions.

C. Feature Extraction

Once the network data has been converted into two-dimensional format, it is sent into MASiNet to extract features.

MASiNet's architecture, illustrated in Figure 2, comprises two phases: encoding and decoding.

In the encoding phase, MASiNet employs Residual Blocks, pivotal for feature processing. These blocks consist of convolutional layers, crucial for initial feature extraction. Enhanced by ReLU activation [21], which introduces vital non-linearities, capturing complex data patterns effectively. Residual Blocks integrate residual connections [22], addressing the vanishing gradient problem in deep networks. By facilitating gradient flow, these connections enable deeper architectures, enhancing learning and preventing performance degradation.

Within each Residual Block, Channel Attention layer [23], as implemented through the Squeeze-and-Excitation operation plays a pivotal role. It performs a unique function of recalibrating the network's feature maps adaptively. The operation dynamically emphasizes important features while diminishing the less relevant ones, thereby refining the feature maps. Complementing the Channel Attention layer is the Spatial Attention Layer [24]. This layer concentrates on identifying and accentuating salient spatial regions within the input data. By doing so, it enables the network to better understand and utilize spatial dependencies and contextual information, significantly enhancing the overall feature extraction process. This spatial focus is particularly advantageous in tasks where the contextual relationship between different regions of the input data is crucial.

Moreover, the decoding phase employs Sequential blocks consisting of ConvTranspose2d layers. These layers play a crucial role in upscaling the feature maps, effectively reconstructing the higher-dimensional spatial information from the

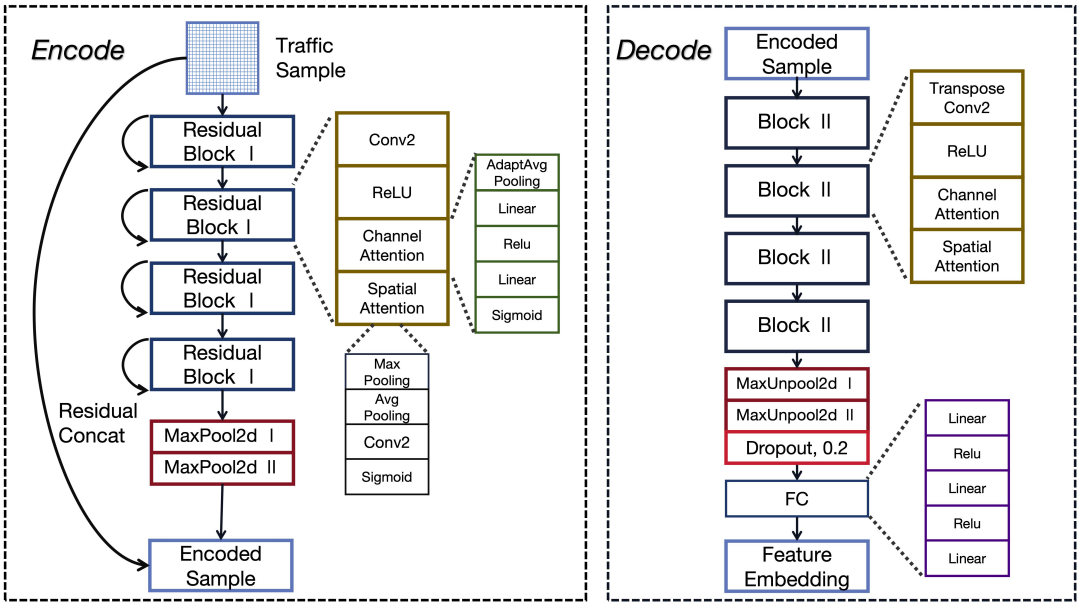


Fig. 2. The feature extraction kernel of MASiNet

compressed features obtained in the encoding phase. Mirroring the encoding stage, Channel Attention layers and Spatial Attention Layers are also incorporated within these blocks. Their presence ensures that the process of feature reconstruction remains focused and retains critical information, thereby facilitating a more nuanced and detailed feature representation.

Complementing the core structure are MaxPool2d and MaxUnpool2d layers [25], strategically placed for efficient spatial dimensionality manipulation through downsampling and upsampling, respectively. A Dropout layer [26] with a rate of 0.2 is included to impart robustness to the model, aiding in generalization by mitigating overfitting tendencies. The architecture culminates with a Sequential block, consisting of Linear layers and ReLU activations. This block serves to refine the final feature representations, preparing them for the output layer.

D. Cost Function Design for Meta-Training

In the process of training the MASiNet for few-shot detection, ensuring the accuracy of anomaly detection predictions is of utmost importance, especially when dealing with a limited number of labeled samples. The scarcity of new attack samples often poses challenges in feature extraction and can lead to overfitting issues within the model. Therefore, the selection of an appropriate loss function becomes crucial in this context. In our approach, we consider two distinct losses to address these challenges. Starting from the coding and decoding stages of the training phase, we introduce two loss function used in two different stages.

Encode and Decode. Consider the input sample x_i into MASiNet, the feature extraction module encodes the samples, which we represent as follows:

$$x_i = \{\ x_i \mid i \in N, x_i \in R^{h \times w}\} \tag{3}$$

$$f(x_i) = Encode(x_i) \tag{4}$$

h and w represent the length and width of the two-dimensional data x_i respectively.

Next the feature embedding goes to the decoding stage as follows:

$$F(x_i) = Decode(f(x_i)) \tag{5}$$

Coding Loss. Within the MASiNet framework, the introduction of coding loss is fundamental, particularly during the feature extraction phase. This is attributed to its role in optimizing feature representation while ensuring that crucial information within network traffic data is meticulously preserved. The adoption of coding loss is pivotal in ascertaining that the model effectively retains essential details from the original data, thereby augmenting both the accuracy and robustness of the system.

The challenge in quantifying information loss during the encoding process arises from its inherent non-observability. To navigate this complexity, the research delineated in [27] proposes the use of relative cross entropy as a viable means to quantify information loss. This is accomplished by contrasting the actual distribution of data with its theoretical counterpart. Notably, relative cross entropy is also synonymous with Kullback-Leibler (KL) divergence. Consequently, in our study, we employ KL divergence as the basis for coding loss. This divergence serves as a widely recognized metric for gauging the discrepancy between two probability distributions. The calculation of coding loss, predicated on KL divergence, is articulated as follows:

$$L_{ecd} = \sum_i p(x_i) \log \left(\frac{p(x_i)}{q(F(x_i))} \right) \tag{6}$$

In Equation 6, $p(x_i)$ symbolizes the distribution of the original dataset, and $q(F(x_i))$ represents the distribution of post-reconstruction by the model. The KL divergence is effectively minimized when the disparity between these two distributions, $p(x_i)$ and $q(F(x_i))$, is at its least, signifying a high fidelity of the reconstructed data to the original.

The coding loss in feature extraction module plays a crucial role in retaining critical information, enabling the model to learn the feature embedding of network traffic data more proficiently. This, in turn, helps alleviate the overfitting problem caused by the scarcity of training data.

Dot Product-based Contrast Loss. In addition to encoding loss, our model incorporates a second loss function—dot product-based contrast loss. This loss function is pivotal in our similarity metric module, facilitating the alignment of samples within the feature space. The adoption of dot product-based contrast loss, is motivated by its effectiveness in capturing the directional relationship and magnitude correlation between feature embeddings. This has been increasingly recognized in machine learning, particularly in tasks involving high-dimensional data, where the angle between vectors can be more informative than their absolute spatial difference [28].

The core of this function lies in the computation of cosine similarity between feature embeddings, $F(x_i)$ and $F(x_j)$. This is mathematically expressed as:

$$\mathcal{P}(F(x_i), F(x_j)) = F(x_i) \cdot F(x_j)^T \tag{7}$$

$$\cos(\theta) = \frac{\mathcal{P}(F(x_i), F(x_j))}{\|F(x_i)\|\|F(x_j)\|} \tag{8}$$

Subsequently, prediction label is determined using the equation:

$$y = \begin{cases} 1 & , \cos(\theta) > \alpha \\ 0 & , \cos(\theta) < \alpha \end{cases} \tag{9}$$

Here, y represents binary labels $\{0, 1\}$, where 1 signifies similar sample types and 0 denotes dissimilar types. The threshold α is empirically set to $\frac{1}{2}$ during training, which is a common choice for balancing the sensitivity towards similarity and dissimilarity. The contrast loss is then calculated as follows:

$$\begin{aligned} L_{\text{pre}} = \sum_{k=1}^{n} & y \times (1 - \cos(\theta))^2 \\ & + (1 - y) \times \max(0, m - (1 - \cos(\theta)))^2 \end{aligned} \tag{10}$$

In this equation, y denotes the prediction label. The term m denotes a predetermined margin that modulates the impact of dissimilar class pairs. This mechanism enables the model to discern and differentiate the output features, effectively capturing the nuances in sample pairs and optimizing the cosine similarity to increase it for similar pairs and decrease for dissimilar pairs.

The contrast loss function, combined with the feature encoding loss, contributes to the overall training performance of the MASiNet model. The total loss is computed as the sum of these two losses:

$$L_{MASiNet} = \lambda L_{ecd} + L_{pre} \tag{11}$$

Here, λ is the temperature coefficient, which acts as a balancing factor to control the impact of the feature encoding loss L_{ecd} during the training process.

Meta-Training. By training on multiple tasks, the network can learn more features and patterns, improving its generalization performance. Algorithm 1 shows the implementation of the few-shot detection task based on the original dataset.

Algorithm 1 Generating a Few-Shot Detection Task

input : Label set $L = \{0, 1, \ldots, C\}$,
Dataset generated by random sampling: $D = \{(x_l, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, with $y_i \in L$.
The number of samples per class for the support set is N, the number of samples per class for the query set is M.
output: Few-shot detection task $T = \{S, Q\}$

Label $\leftarrow$ RandomSample($\{1, \ldots, C\}$, 1)
// Generate support set
$S(0) \leftarrow$ RandomSample($D(0)$, N)
S(Label) $\leftarrow$ RandomSample(D(Label), N)
$S \leftarrow S(0) \cup S$(Label)
// Generate query set
$Q(0) \leftarrow$ RandomSample($D(0)$, M)
Q(Label) $\leftarrow$ RandomSample(D(Label), M)
$Q \leftarrow Q(0) \cup Q$(Label)
// Change labels
for (x_i, y_i) *in* $\{S, Q\}$ **do**
  if $y_i \neq 0$ **then**
    $y_i \leftarrow 1$
  end
end
// Generate detection tasks
$T \leftarrow \{S, Q\}$

Algorithm 2 Few-Shot Model Training

input : tasks $T = \{S, Q\}$, Max_training_iteration E,
Support set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{2N}, y_{2N})\}$,
Query set $Q = \{(x_1^q, y_1^q), (x_2^q, y_2^q), \ldots, (x_{2M}^q, y_{2M}^q)\}$,
$y_i \in \{0, 1\}$
output: optimal M_{MASiNet}

$L_{\text{MASiNet}} \leftarrow 0$
for K^{th} *epoch in* E **do**
  for x_i *in Support set* **do**
    Eq.4, Eq.5 $\rightarrow F(x_i)$
    Eq.6 $\rightarrow L_{\text{ecd}}$
    for x_j *in Query set* **do**
      Eq.4, Eq.5 $\rightarrow F(x_j)$
      Eq.7, Eq.8 $\rightarrow \mathcal{P}(F(x_i), F(x_j))$, $\cos(\theta)$
      Eq.9, Eq.10 $\rightarrow L_{\text{pre}}$
      Eq.11 $\rightarrow L_{\text{MASiNet}}$
      minimize L_{MASiNet}, update M_{MASiNet}
    end
  end
end
return M_{MASiNet}

The model is trained using the provided loss functions to enable effective differentiation between similar and dissimilar samples in the feature space. The training process is outlined in Algorithm 2, which iteratively refines the model until convergence. The Adam optimizer algorithm [29] is employed to optimize the parameters, with a learning rate of 0.001 and a weight decay [30] coefficient of $1e-5$ to prevent

overfitting. The training process involves 400 iterations to examine the model's performance. In each iteration, $Batch$ number of meta-training tasks are executed. The network hyperparameters used in the experiments, including the weight decay, are summarized in Table I.

TABLE I
NETWORK STRUCTURE SETTINGS OF MASINET

Hyperparameter	Value
Training cycle	400
Learning rate	0.001
β_1	0.9
β_2	0.999
M	15
N	5
$Batch$	100
Weight decay	1e-5

IV. EVALUATION

A. Dataset

The performance evaluation of few-shot intrusion detection methods is conducted using the NSL_KDD [31] dataset and UNSW-NB15 [32] dataset, which are well-known benchmark in the field of network systems.

The NSL_KDD dataset is divided into three subsets: KDDTrain+, KDDTest+, and KDDTest-21, each subset serving different purposes based on its proportion and prediction difficulty. Specifically, the dataset contains 125,973 records in the KDDTrain+ subset, 22,544 records in the KDDTest+ subset, and 11,850 records in the KDDTest-21 subset. Each record consists of 43 attributes, with 41 attributes related to the traffic input and the remaining two attributes indicating the label (normal or abnormal) and the potential harm of the traffic input.

The attacks in the dataset are classified into four primary categories, as presented in Table II.

TABLE II
THE DISTRIBUTION OF DIFFERENT CATEGORIES IN NSL_KDD DATASET

Type	KDDTrain+	KDDTest+
Normal	67345	9711
Dos	45926	7458
Probe	11655	2421
R2L	995	2754
U2R	52	200
Total	125973	22544

The UNSW-NB15 dataset, addressing limitations of previous datasets like KDD[33] and NSL-KDD, is highly relevant for IoT security research. It includes a blend of real and synthetic data, reflecting the unique threat landscape of IoT networks. This dataset aids in developing advanced intrusion detection systems suitable for IoT, containing 175,341 training and 82,332 testing entries with IoT-specific traffic and attack scenarios. The class distribution is detailed in Table III.

TABLE III
THE DISTRIBUTION OF DIFFERENT CATEGORIES IN UNSW-NB15 DATASET

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

B. Experimental Settings

The experiments were conducted on an Intel(R) Xeon(R) Gold 5218R CPU @2.10 GHz, 128 GB RAM, NVIDIA A100-PCIE-80GB, Aurora Gridview cluster, utilizing CUDA 11.7, cuDNN 8.5, and Pytorch 1.12.1.

In our study, we employed the KDDTrain+ and UNSW-NB15 training sets to create a simulated experimental environment for few-shot learning. Specifically, the KDDTrain+ dataset, which comprised five distinct classes as outlined in Table II, served as a basis for constructing various data combinations for our experiment. We designed four types of data combinations (detailed in Table IV), each simulating a new attack category with only a few samples, while utilizing the remaining attack samples for model training. Particularly, in our first data combination, we integrated three attack types: Prob, R2L, and U2R - along with normal samples to form the meta-training tasks. For the meta-testing task, we built scenarios using a limited number of labeled DoS samples and normal class samples as the support set, which were randomly selected from the testing set. We used these limited samples to detect a large number of samples; specifically, we detected all the remaining DoS attacks in the testing set, and an equal number of normal samples as the attack samples.

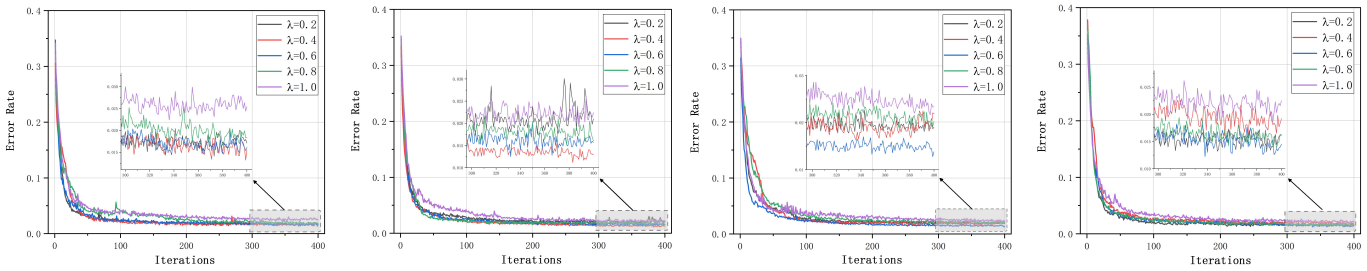
The same methodology was applied to the UNSW-NB15 dataset, ensuring a consistent experimental approach across both datasets. This strategy allowed us to effectively simulate few-shot learning scenarios, focusing on the capability of the model to adapt to new attack categories.

TABLE IV
COMBINATIONS OF TRAINING AND TESTING CLASSES

No.	Meta-training tasks	Meta-testing tasks
1	(normal, Probe), (normal, R2L), (normal, U2R)	(normal, DoS)
2	(normal, DoS), (normal, R2L), (normal, U2R)	(normal, Probe)
3	(normal, DoS), (normal, Probe), (normal, U2R)	(normal, R2L)
4	(normal, DoS), (normal, Probe), (normal, R2L)	(normal, U2R)

C. Experimental Result and Analysis

Due to the nascent stage of few-shot network intrusion detection, there was a lack of dedicated benchmark datasets. Consequently, we leveraged three recently-researched few-shot



(a)DoS (b)Probe (c)R2L (d)U2R

Fig. 3. Evaluation on training efficiency: encoding loss curve with different temperature coefficients for different meta-testing class.

learning methods for comparative analysis: FSL-Intrusion [13], FC-Net [15], and FSL-SCNN [16].

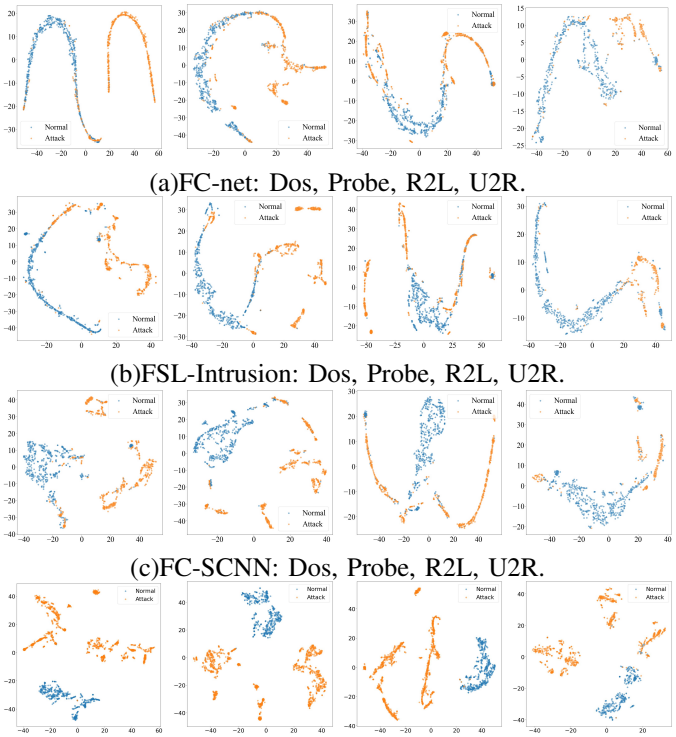
Temperature Coefficient. In this study, first, we used the four combinations formed in the KDDTrain+ dataset shown in Table IV to conduct experiments to select a suitable temperature coefficient. The results for coding loss are shown in Figure 3. We observed that the overall performance of the loss decreased rapidly before stabilizing around 100 iterations. After considering the effects of the encoding loss, we selected the temperature coefficient $\lambda = 0.6$ as the hyperparameter for subsequent experiments.

Feature Embedding. To assess the impact of feature embedding on our model's accuracy, we employed principal component analysis (PCA) as our baseline. While our training was based on the KDDTrain+ dataset, PCA was done using the meta-testing class of four combinations on KDDTest+. Notably, our model demonstrated distinguishing prowess, evident from the clear differentiation between the two feature embedding classes, as illustrated in Figure 4. Besides, our model surpassed the other three methods in terms of clustering capability.

Performance Evaluation. The effectiveness of our few-shot detection approach was assessed using the KDDTest+ dataset and the UNSW-NB15 testing set. Key performance metrics included Accuracy, Precision, and False Alarm Rate (FAR)[34]. We conducted a series of experiments for each meta-testing task combination, averaging the results to derive the final detection performance. As indicated in TableV for the KDDTest+ dataset and Table VI for the UNSW-NB15 testing set, our model demonstrated robust performance in terms of all metrics.

V. Real-World Evaluation

In this section, we conducted an in-depth evaluation of the MASiNet for few-shot intrusion detection in a real-world scenario. Our evaluation environment was a bespoke smart home platform equipped with wireless network infrastructure and a Raspberry Pi serving as the primary computing device. It should be explained that the real-world experiment was primarily aimed at supporting the theoretical method we proposed and demonstrating the feasibility of our approach in practical scenarios. Therefore, in our experiments, we only collected and analyzed the Deauthentication Attack and



(a)FC-net: Dos, Probe, R2L, U2R.

(b)FSL-Intrusion: Dos, Probe, R2L, U2R.

(c)FC-SCNN: Dos, Probe, R2L, U2R.

(d)MASiNet: Dos, Probe, R2L, U2R.

Fig. 4. Feature embedding evaluation based on PCA in KDDTest+.

Beacon Flooding Attack dataset to verify that our method also yielded practical results in real-life scenarios.

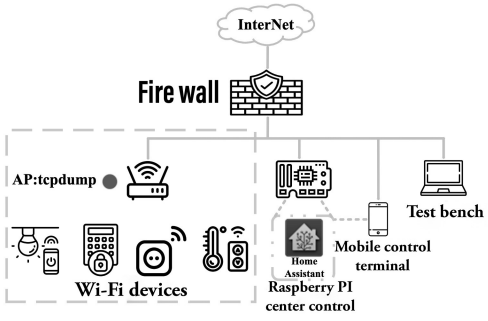


Fig. 5. Overall architecture of smart home system platform based on Raspberry Pi and Home Assistant

TABLE V
PERFORMANCE COMPARISON ON KDDTEST+

	FC-Net			FSL-Intrusion			FSL-SCNN			MASiNet		
Class	Acc	Prec	FAR	Acc	Prec	FAR	Acc	Prec	FAR	Acc	Prec	FAR
DoS	87.03%	81.51%	21.71%	85.61%	82.29%	19.52%	91.97%	87.15%	14.51%	95.46%	92.43%	8.10%
Probe	87.50%	86.12%	14.40%	86.73%	84.24%	16.89%	89.93%	86.74%	10.07%	96.04%	93.97%	6.52%
R2L	86.73%	81.69%	21.21%	80.45%	71.17%	41.45%	85.16%	78.05%	27.50%	90.27%	83.82%	19.25%
U2R	90.78%	89.74%	10.52%	89.21%	87.81%	12.63%	88.94%	85.23%	16.31%	96.05%	94.87%	5.26%

TABLE VI
PERFORMANCE COMPARISON ON UNSW-NB15 TESTING SET

	FC-Net			FSL-Intrusion			FSL-SCNN			MASiNet		
Class	Acc	Prec	FAR	Acc	Prec	FAR	Acc	Prec	FAR	Acc	Prec	FAR
Generic	91.72%	90.74%	9.48%	90.14%	89.06%	11.23%	90.51%	89.45%	10.83%	97.79%	97.36%	2.65%
DoS	87.25%	86.75%	13.42%	84.53%	83.73%	16.65%	86.47%	85.96%	14.23%	91.04%	90.39%	9.75%
Fuzzers	86.25%	85.47%	14.84%	85.71%	84.95%	15.37%	86.89%	86.28%	13.93%	90.38%	89.72%	10.44%
Reconnaissance	86.97%	86.05%	14.30%	85.08%	84.00%	16.50%	87.77%	86.95%	13.32%	91.08%	90.31%	9.86%
Shellcode	78.96%	75.64%	27.51%	76.85%	74.10%	28.83%	80.82%	76.78%	26.71%	83.86%	79.62%	23.28%
Worms	76.13%	73.46%	29.54%	62.50%	61.22%	43.18%	70.45%	68.75%	34.09%	77.27%	74.00%	29.54%
Backdoor	87.04%	86.48%	13.72%	85.50%	84.84%	15.43%	86.36%	85.57%	14.75%	91.68%	90.23%	10.12%
Exploits	92.26%	91.96%	8.09%	89.66%	88.61%	11.68%	92.37%	92.07%	7.97%	94.46%	94.06%	5.98%

Experimental Environment Settings. The Smart Home platform was designed based on Raspberry Pi and the Home Assistant control management platform as shown in Figure 5. It used WiFi technology to control and monitor various smart home devices, including smart lighting, smart sockets, and smart access control. The platform was divided into three parts: control end, communication end, and execution end. The control end included a Raspberry Pi computer Model 4B system equipped with Home Assistant control and mobile network devices for remote control. It used Raspberry Pi as the core controller, communicating with terminal devices through wireless WiFi. Users could remotely control smart home devices, through mobile APPs, voice control, or automation control. The communication end primarily used wireless routing devices and WiFi technology to interconnect the control and execution ends. The execution end mainly included smart IoT devices with wireless networking capabilities.

The platform employed the tcpdump utility to capture and analyze network traffic, further refining the wireless traffic data through the use of Tshark, Pandas, and Numpy. For experimental purposes, a laptop running the Kali operating system served as the test bench, which was used for penetration testing and network security assessment.

This comprehensive Smart Home platform, with its diverse smart devices and robust control and communication systems, provided a realistic and challenging environment for testing our intrusion detection method.

Data Collection. Considering the security vulnerabilities of existing wireless Local Area Network(LAN), DoS attacks can disrupt smart home devices or wireless LAN functionality, causing significant disruptions to IoT systems [35]. Common DoS attacks on wireless LANs include Ping Flood, SYN Flood, Deauthentication, and Beacon Flooding attacks. We identified two common, easily executable, and observable attacks: Deauthentication Attack and Beacon Flooding Attack. During the operation of our smart home system, we simulated Deauthentication and Beacon Flooding attacks, capturing all transmitted traffic packets in the channel where the target wireless LAN was located. The specific implementation process was as follows: First, the captured wireless LAN packet was encapsulated into a unified frame. Then, by analyzing and deleting the fixed and redundant complex fields, the captured packet attributes which could be used for intrusion detection were extracted. These attributes covered the clear code information of all frames in the wireless LAN packet, forming a specific attribute vector, which not only retained the data frame in the wireless LAN but also included most of the useful information of the management frame and control frame. Combined with the design of this experiment, some of the relevant attributes extracted by this method were retained, as shown in Table VII.

We manually marked normal and abnormal traffic based on the record of the captured packets. Next, we randomly sampled the real wireless LAN dataset collected as described above, as shown in the table VIII. In addition, we preprocessed the collected samples using the method as aforementioned.

Performance. We subjected our MASiNet to evaluation using this real-world dataset, with the results demonstrating its practical utility (Table IX). These findings underscored the effectiveness and feasibility of our proposed few-shot network intrusion detection method for smart home wireless LANs in real-world applications.

VI. FUTURE WORK

In this study, we utilized MASiNet-based few-shot learning approach to detect normal and attack network traffic, which showed promising results. However, several areas warrant further exploration:

1) *Adaptive learning mechanisms:* Further research should focus on enhancing the model's adaptability to evolv-

TABLE VII
ATTRIBUTES OF WIRELESS LAN INTRUSION DETECTION BASED ON UNIFIED FRAME TECHNOLOGY

No.	Attribute	Description	Representation
1	Frame.len	Frame length	Numeric
2	Frame.time_delta	Frame time increment	Numeric
3	Wlan.fc.type	Frame type	Categorical
4	Wlan.fc.subtype	Frame subtype	Categorical
5	Wlan.fc.ds	Ds field	Categorical
6	Wlan.fc.retry	Retry field	Categorical
7	Wlan.fc.moredata	More data field	Categorical
8	Wlan.fc.protected	Protected frame field	Categorical
9	Wlan.duration	Duration field	Numeric
10	Wlan.seq	Sequence number	Numeric
11	Wlan.fixed.timestamp	Fixed timestamp field	Numeric
12	Wlan.fixed.reason_code	Fixed reason code field	Numeric
13	Data.len	Data length	Numeric

TABLE VIII
DATA DISTRIBUTION OF WIRELESS NETWORK ATTACK DATASET

Data type	Number of data samples
Normal	10000
Deauthentication Attack	2000
Beacon Flooding Attack	2000

TABLE IX
EXPERIMENTAL RESULTS

Model	Data category	Accuracy	Prec
MASiNet	Deauthentication Attack	99.08%	98.76%
	Beacon Flooding Attack	98.67%	98.55%

ing threat landscapes. This includes developing adaptive learning algorithms that can update the model in real-time, catering to the dynamic nature of IoT network behaviors and emerging threats such as zero-day attack.

2) *Targeted data augmentation techniques:* Investigate data augmentation methods specifically designed for imbalanced network traffic data. This might include synthetic data generation that can mimic rare but critical attack vectors, thus providing a more robust training dataset for the model.
3) *Experimental platform improvement:* Our Raspberry PI-based smart home experimental platform employs relatively simple implementation logic, an idealized attack process, and a limited variety of collected wireless network attack data. This simplification may limit the detection results. In future work, we will enhance the smart home system, simulate a wider range of wireless network attacks, and collect a more diverse set of wireless data packets. This will allow us to verify the effectiveness and practicability of our proposed method from a more comprehensive perspective.
4) *Integration with decentralized security systems:* Explore the potential for integrating MASiNet within decentralized security frameworks, like blockchain-based IoT security systems, to enhance data integrity and resilience against distributed network attacks.

VII. CONCLUSION

In this study, we introduced MASiNet, a pioneering model in network intrusion detection for IoT security, harnessing the potential of few-shot learning with a meta-learning framework. Our approach effectively leveraged minimal labeled data to address the challenge of anomaly detection in IoT networks. Through extensive experimentation, MASiNet demonstrated its superiority in feature extraction, owing to the integration of attention mechanisms and a crafted cost function. Evaluated against the NSL_KDD and UNSW-NB15 datasets, our model achieved remarkable improvements over existing methods, particularly in terms of accuracy, precision, and reducing the FAR. It offered a robust solution to the pressing issue of detecting new and evolving threats in dynamic network environments.

Subsequently, our model was applied to real-world scenarios through a custom-built smart home system platform. We developed an authentic wireless LAN attack dataset and evaluated our proposed intrusion detection method on it, achieving remarkably high accuracy in identifying samples. These results served as strong evidence, confirming the effectiveness and feasibility of our novel approach to intrusion detection.

VIII. ACKNOWLEDGEMENT

This work was partly supported by the National Natural Science Foundation of China (NSFC) under Grant No. 62302454, the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies under Grant No. 2022B1212010005, the Zhejiang Provincial Natural Science Foundation of China through a Major Program (Youth Original Project) under Grant No. LDQ24F020001, and the China Postdoctoral Science Foundation under Grant No. 2023M743403.

REFERENCES

[1] A. Alwarafy, K. A. Al-Thelaya, M. Abdallah, J. Schneider, and M. Hamdi, "A survey on security and privacy issues in edge-computing-assisted internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4004–4022, 2020.

[2] T. Zebin, S. Rezvy, and Y. Luo, "An explainable ai-based intrusion detection system for dns over https (doh) attacks," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2339–2349, 2022.

[3] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "Corrauc: A malicious bot-iot traffic detection method in iot network using machine-learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242–3254, 2020.

[4] A. S. Iliyasu, U. A. Abdurrahman, and L. Zheng, "Few-shot network intrusion detection using discriminative representation learning with supervised autoencoder," *Applied Sciences*, vol. 12, no. 5, p. 2351, 2022.

[5] Z. Zhang, Q. Liu, S. Qiu, S. Zhou, and C. Zhang, "Unknown attack detection based on zero-shot learning," *IEEE Access*, vol. 8, pp. 193 981–193 991, 2020.

[6] S. Thrun and L. Pratt, *Learning to learn*. Springer Science & Business Media, 2012.

[7] P. Tian, Z. Wu, L. Qi, L. Wang, Y. Shi, and Y. Gao, "Differentiable meta-learning model for few-shot semantic segmentation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 087–12 094.

[8] L. Liu, T. Zhou, G. Long, J. Jiang, L. Yao, and C. Zhang, "Prototype propagation networks (ppn) for weakly-supervised few-shot learning on category graph," in *IJCAI International Joint Conference on Artificial Intelligence*, 2019.

[9] B. Hu, Y. Bi, M. Zhi, K. Zhang, F. Yan, Q. Zhang, and Z. Liu, "A deep one-class intrusion detection scheme in software-defined industrial networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4286–4296, 2021.

[10] Y. Djenouri, D. Djenouri, A. Belhadi, G. Srivastava, and J. C.-W. Lin, "Emergent deep learning for anomaly detection in internet of everything," *IEEE Internet of Things Journal*, 2021.

[11] G. Duan, H. Lv, H. Wang, and G. Feng, "Application of a dynamic line graph neural network for intrusion detection with semisupervised learning," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 699–714, 2022.

[12] Y. Chen, S. Su, D. Yu, H. He, X. Wang, Y. Ma, and H. Guo, "Cross-domain industrial intrusion detection deep model trained with imbalanced data," *IEEE Internet of Things Journal*, vol. 10, no. 1, pp. 584–596, 2022.

[13] Y. Yu and N. Bian, "An intrusion detection method using few-shot learning," *IEEE Access*, vol. 8, pp. 49 730–49 740, 2020.

[14] T. Li, Z. Hong, L. Liu, Z. Wen, and L. Yu, "Meta-wf: Meta-learning-based few-shot wireless impersonation detection for wi-fi networks," *IEEE Communications Letters*, vol. 25, no. 11, pp. 3585–3589, 2021.

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

[16] X. Zhou, W. Liang, S. Shimizu, J. Ma, and Q. Jin, "Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5790–5798, 2020.

[17] C. Cheng, L. Song, R. Xue, H. Wang, H. Sun, Y. Ge, and Y. Shan, "Meta-adapter: An online few-shot learner for vision-language model," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[18] S. Khodadadeh, L. Boloni, and M. Shah, "Unsupervised meta-learning for few-shot image classification," *Advances in neural information processing systems*, vol. 32, 2019.

[19] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep learning for network traffic monitoring and analysis (ntma): A survey," *Computer Communications*, vol. 170, pp. 19–41, 2021.

[20] L. Dhanabal and S. Shantharajah, "A study on nsl-kdd dataset for intrusion detection system based on classification algorithms," *International journal of advanced research in computer and communication engineering*, vol. 4, no. 6, pp. 446–452, 2015.

[21] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[24] X. Zhu, D. Cheng, Z. Zhang, S. Lin, and J. Dai, "An empirical study of spatial attention mechanisms in deep networks," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6688–6697.

[25] N. Murray and F. Perronnin, "Generalized max pooling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2473–2480.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[27] G. Flamich, M. Havasi, and J. M. Hernández-Lobato, "Compressing images by encoding their latent representations with relative entropy coding," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 131–16 141, 2020.

[28] S. Sra, "Directional statistics in machine learning: a brief review," *Applied Directional Statistics: modern methods and case studies*, vol. 225, no. 6, 2018.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[31] R. D. Ravipati and M. Abualkibash, "A survey on different machine learning algorithms and weak classifiers based on kdd and nsl-kdd datasets," *International Journal of Artificial Intelligence and Applications (IJAIA)*, vol. 10, no. 3, 2019.

[32] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[33] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee, 2009, pp. 1–6.

[34] L. Li, Y. Yu, S. Bai, J. Cheng, and X. Chen, "Towards effective network intrusion detection: A hybrid model integrating gini index and gbdt with pso," *Journal of Sensors*, vol. 2018, pp. 1–9, 2018.

[35] A. E. Abdallah, M. Hamdan, M. S. Gismalla, A. O. Ibrahim, N. S. Aljurayban, W. Nagmeldin, and M. H. Khairi, "Detection of management-frames-based denial-of-service attack in wireless lan network using artificial neural network," *Sensors*, vol. 23, no. 5, p. 2663, 2023.

Yiming Wu is an Assistant Professor with the Institute of Cyberspace Security and the College of Information Engineering at *Zhejiang University of Technology*. She got her Ph.D. in Cyberspace Security from *Zhejiang University*. Her research interests include Data-driven Security, Software and System Security, and AI Security.

Gaoyun Lin was born in Zhengjiang, China. He is currently pursuing the bachelor's degree with the College of Computer Science, *Zhejiang University of Technology*, Hangzhou, China. His research interests include edge computing and deep learning.

Lisong Liu received the master's degree in electronic information from the School of Information Engineering, *Zhejiang University of Technology*, in 2023. Currently, he is working in a Power Technology Company, mainly engaged in data analysis, software research, and development.

Zhen Hong is a professor with the Institute of Cyberspace Security and College of Information Engineering, *Zhejiang University of Technology*, China. His research interests include cyber-physical systems, the Internet of Things, wireless sensor networks, cybersecurity, and data analytics.

Xing Yang is a researcher at the State Key Laboratory of Pulsed Power Laser Technology, *National University of Defense Technology*. He received his BS, MS, and Ph.D. degrees from Hefei Electronic Engineering Institute in 2006, 2009, and 2012, respectively. Currently, his research interests mainly focus on Optoelectronic Engineering, Artificial Intelligence, and Cyberspace Security.

Yangyang Wang (1986-) comes from Xuchang, Henan, China. He received the B.S. degree from *University of Science and Technology of China* in 2009, and the M.S. degree from *Ordnance Engineering College* in 2011. Currently, he is an associate professor at the State Key Laboratory of Pulsed Power Laser Technology, *National University of Defense Technology*. His main research interests include Artificial Intelligence and Cyberspace Security.

Zoe L. Jiang *Zoe L. Jiang* received the Ph.D. degree from *The University of Hong Kong*, Hong Kong, in 2010. She is currently a professor with School of Computer Science and Technology, *Harbin Institute of Technology*, Shenzhen, China. Her research interests include information security and applied cryptography.

Shouling Ji (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from *The Georgia Institute of Technology* and the Ph.D. degree in computer science from *Georgia State University*. He is currently a *ZJU 100-Young Professor* with the College of Computer Science and Technology, *Zhejiang University*, and a Research Faculty Member of the School of Electrical and Computer Engineering, *The Georgia Institute of Technology*. His current research interests include AI security, data-driven security, privacy, and data analytics. He is a member of ACM and was the Membership Chair of the IEEE Student Branch with Georgia State University (2012–2013).

Zhenyu Wen (Senior Member, IEEE) is currently a tenure-tracked professor with the Institute of Cyberspace Security and College of Information Engineering, *Zhejiang University of Technology*. His research interests include IoT, crowdsourcing, AI systems, and cloud computing.