Semisupervised Federated-Learning-Based Intrusion Detection Method for Internet of Things

Ruijie Zhao, *Graduate Student Member, IEEE*, Yijun Wang, Zhi Xue, Tomoaki Ohtsuki, *Senior Member, IEEE*, Bamidele Adebisi, *Senior Member, IEEE*, and Guan Gui, *Senior Member, IEEE*

***Abstract*—Federated learning (FL) has become an increasingly popular solution for intrusion detection to avoid data privacy leakage in Internet of Things (IoT) edge devices. Existing FL-based intrusion detection methods, however, suffer from three limitations: 1) model parameters transmitted in each round may be used to recover private data, which leads to security risks; 2) not independent and identically distributed (non-IID) private data seriously adversely affect the training of FL (especially distillation-based FL); and 3) high communication overhead caused by the large model size greatly hinders the actual deployment of the solution. To address these problems, this article develops an intrusion detection method based on a semisupervised FL scheme via knowledge distillation. First, our proposed method leverages unlabeled data via distillation method to enhance the classifier performance. Second, we build a model based on convolutional neural networks (CNNs) for extracting deep features of the traffic packets, and take this model as both the classifier network and discriminator network. Third, the discriminator is designed to improve the quality of each client's predicted labels, and to avoid the failure of distillation training caused by a large number of incorrect predictions under private non-IID data. Moreover, the combination of the hard-label strategy and voting mechanism further reduces communication overhead. The experiments on the real-world traffic data set with three non-IID scenarios show that our proposed method can achieve better detection performance as well as lower communication overhead than state-of-the-art methods.**

***Index Terms*—Federated learning (FL), intrusion detection, knowledge distillation, semisupervised learning.**

Manuscript received 8 February 2022; revised 12 April 2022; accepted 13 May 2022. Date of publication 18 May 2022; date of current version 9 May 2023. This work was supported in part by the Cyber Security from the National Key Research and Development Program of Shanghai Jiao Tong University under Grant 2019QY0703; in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant JP19H02142; in part by the Summit of the Six Top Talents Program of Jiangsu under Grant XYDXX-010; and in part by the Project of the Key Laboratory of Universal Wireless Communications (BUPT) of Ministry of Education of China under Grant KFKT-2020106. *(Corresponding authors: Yijun Wang; Guan Gui.)*

Ruijie Zhao, Yijun Wang, and Zhi Xue are with the School of Electronic Information and Electrical Engineering, Shanghai Jiao tong University, Shanghai 200240, China (e-mail: ruijiezhao@sjtu.edu.cn; ericwyj@sjtu.edu.cn; zxue@sjtu.edu.cn).

Tomoaki Ohtsuki is with the Department of Information and Computer Science, Keio University, Yokohama 108-8345, Japan (e-mail: ohtsuki@keio.jp).

Bamidele Adebisi is with the Department of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M1 5GD, U.K. (e-mail: b.adebisi@mmu.ac.uk).

Guan Gui is with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: guiguan@njupt.edu.cn).

Digital Object Identifier 10.1109/JIOT.2022.3175918

I. Introduction

INTERNET of Things (IoT) has developed rapidly in recent years and provides useful services in multiple applications, such as health and transportation systems. In the meantime, a great number of IoT devices lack appropriate security defenses and therefore, intruders can easily target many of them [1]–[6]. Thus, it is of vital importance to adopt intrusion detection systems (IDS) for IoT devices. Benefiting from the development of deep learning (DL) technology, DL-based methods have achieved great success in the field of traffic analysis and intrusion detection [7]–[14]. Typically, centralized DL (CDL) techniques collect a large amount of traffic data from various IoT devices and upload these data to a central server for training a deep neural network (DNN) model, which is used for detection. Although these CDL methods can achieve high accuracy in intrusion detection task, there exists a problem of information leakage, i.e., private-sensitive raw data are uploaded to the central server and possibly abused by the server. To address the problem of private data leakage of CDL, federated learning (FL) is applied in IDS recently [15]–[20]. In the FL framework, devices collaboratively train their local DL models through the periodical exchange and aggregation of DL model parameters or gradients at the central server without uploading their raw data. However, since traditional FL requires clients to frequently upload the parameters of the DL model, its high communication overhead seriously hinders actual deployment [21]–[27]. Besides, IoT devices are usually large scale, which means that a large number of devices (clients) need to upload information (e.g., gradients and parameters) when applying FL, resulting in more expensive communication overhead [28]. What is worse, there have been many attacks that can restore the raw data of client from the uploaded model parameters, which means that this scheme still poses the security risk of information leakage [29].

Recently, the federated distillation (FD) scheme is proposed to reduce the communication overhead and security risk caused by exchanging model parameters [30]–[32]. Clients in FD can upload outputs of DL model rather than model parameters to the server. Then, the server aggregates outputs, also called local logits. The aggregated local logits, i.e., global logits are downloaded and used by clients in further local training process. Although FD reduces the communication overhead and achieves similar accuracy to that of FL based on independent and identically distributed (IID) data, the performance of FD

is poor when the distribution of data is not IID (non-IID). As data distribution is non-IID, the local data of device does not represent the population distribution and the global logits retain similar information to the local labels already attached to each device. Training based on global logits can be almost identical to training based on a local data set. Hence, several practical applications of FL-based intrusion detection method should achieve these aims.

1) *Security:* FL is proposed to protect users' data privacy. Hence, the FL-based schemes should ensure private data cannot be obtained or restored.
2) *Accuracy:* Clients often have non-IID private data in real-world IoT network, so FL training needs to avoid the adverse effects of non-IID data.
3) *Efficiency:* Successful deployment of the scheme in a real-life environment requires the FL method with low communication overhead. This is to ensure timely delivery of the final model.

To address these challenges mentioned above, we develop a semisupervised FL scheme via knowledge distillation for intrusion detection, denoted as semisupervised FL method (SSFL), which consists of multiple clients and a central server. Specifically, the client first downloads the unlabeled open data from the central server, and this step is performed only once for each client. The easy-to-obtain unlabeled traffic data does not require manual labeling, which saves labor and time costs. Then, the local non-IID labeled data are used to train the first-stage classifier network. Since the first-stage classifier cannot learn some traffic categories that are not in the local labeled data set, it may make incorrect predictions for unfamiliar unlabeled traffic data. Thus, we introduce a discriminator to discriminate whether the unlabeled traffic packets are "familiar," in which the unfamiliar traffic packets will be marked, so as to improve the quality of the predicted soft labels. Note that both the classifier network and the discriminator network use the model based on convolutional neural networks (CNNs), which can effectively extract the deep features of the traffic packet. Next, in order to further reduce communication cost, the client converts soft labels into hard labels and uploads them to the central server. After the central server has collected the predictions of all clients, the global labels of the unlabeled open data are determined through a voting mechanism, rather than directly aggregating. Finally, the aggregated labels are sent to each client for training the second-stage classifier. The above steps are repeated to achieve an excellent performance model for intrusion detection via SSFL. Hence, the main contributions of this article can be summarized as follows.

1) We propose an SSFL for intrusion detection, which uses unlabeled data and knowledge distillation to achieve a privacy-preserving and communication-efficient FL-based intrusion detection method for IoT.
2) A CNN-based model is developed for extracting the deep features of the traffic packet, and take this model as both the classifier network and the discriminator network.
3) To make our scheme more effective, we design a discriminator to make a distinction as to whether the traffic packets are familiar or not, which improves the quality of each client's predicted labels. In addition, the combination of the hard-label strategy and voting mechanism further reduces communication overhead.
4) We construct three challenging non-IID scenarios and evaluate SSFL on the real-world IoT traffic data set. The experimental results show that our scheme can achieve better detection performance as well as lower communication overhead than state-of-the-art methods. Thus, it is suitable for federated training and detecting intrusions in the IoT network.

The remainder of this article is organized as follows. Section II reviews the state-of-art methods used in IDS. In Section III, we introduce the baseline schemes for federated training, including the FL scheme, the FD scheme, and the distillation-based semisupervised FL (DS-FL) scheme. The details of our method are described in Section IV. In Section V, we experimentally evaluate the performances of SSFL. Finally, we conclude this article in Section VI.

II. Related Works

A. DL-Based IDS

Researchers have proposed many intrusion detection methods to defend against cyber attacks. In general, IDSs are divided into three categories: 1) rule-based IDS; 2) ML-based IDS; and 3) DL-based IDS. The rule-based IDS determines whether the traffic matches the corresponding rule, which is based on the collected abnormal samples. The main drawbacks of rule-based IDS are that some key features (e.g., port number) prone to failure in a complex network environment, and it is difficult to detect unknown intrusions. In recent years, to better analyze the complex network traffic, some ML algorithms have been applied in the field of intrusion detection [33]–[37]. Although the ML-based IDSs can detect unknown intrusions based on the payload or statistical features of the traffic effectively, they usually rely heavily on feature engineering and can only extract shallow features. Some works have proposed methods based on DL algorithms for intrusion detection and demonstrated better performance than ML algorithms. For instance, Mirsky *et al.* [7] designed an ensemble of neural networks to collectively differentiate between normal and abnormal traffic, which can realize low-complexity online processing. Lin *et al.* [13] proposed a time-related intrusion detection method based on the stacked sparse autoencoder (SSAE) and recurrent neural network (RNN), which effectively extracts features with the greedy layerwise strategy for better detection performance. Andresini *et al.* [14] proposed to use a CNN architecture that is trained on the imagery representation of the network flows to detect the attacking flow behavior. Obviously, DL-based methods have achieved success in the field of intrusion detection, but they rely on a large number of traffic samples to train a model with excellent detection performance. Since these data have user's private information, most users are unwilling to provide it. On the other hand, some laws (e.g., GDPR)[1] on data privacy have strict regulations on the use of user data.

[1] https://gdpr.eu/data-privacy

B. FL-Based IDS

FL is proposed for privacy-sensitive data, which is in contrast to CDL that requires to collect private data of the user. Recently, due to privacy advantages, the FL method has been widely used in IDS and IoT. Kang *et al.* [16] first introduced reputation as the metric to measure the reliability and trustworthiness of the mobile devices for reliable FL. Popoola *et al.* [17] proposed an FL method for detecting the zero-day botnet attack to avoid data privacy leakage in IoT edge devices. Al-Marri *et al.* [18] proposed to use mimic learning in FL framework to overcome the problem of reverse engineering in FL to minimize the risk of jeopardizing users' privacy in IDS. Hei *et al.* [19] proposed the FL framework for IDS and this framework stored the model training process information and behavior on the blockchain to meet the storage needs of a large number of alert training data in real scenarios. Zhang *et al.* [20] proposed a novel semisupervised FL method that leverages unlabeled data to train an unsupervised model based on consistency regularization, and then they aggregate the unsupervised model, supervised model, and global model into a new global model.

However, the DL-based model has a huge amount of parameters, which makes the client suffer from the communication bottleneck. To reduce the communication overhead in FL-based schemes, some researchers have applied various methods. Liu *et al.* [28] proposed an efficient communication FL scheme, which uses the gradient compression mechanism based on top-k selection in FL-based IDS to reduce the communication overhead. Qin *et al.* [38] used binarized neural networks in FL-based IDS to keep the communication overheads of training small. Although the above methods reduce the communication overhead, it will scale up according to the model size. Inspired by some works of the distillation-based FL method [30], [31], we develop a semisupervised FL scheme for intrusion detection, which transmits the predictions of unlabeled data instead of model parameters in each round of training. Moreover, we propose a discriminator for improving the quality of each client's predicted labels. The results show that our method outperforms state of the art on the real-world traffic data set, as detailed in the later experiment section.

III. Preliminaries

In this section, we summarize the baseline schemes for federated training, including the FL scheme, the FD scheme, and the DS-FL scheme, detailed as follows.

A. Baseline 1: FL Scheme

The process of the FL scheme consists of four steps. First, IoT devices or mobile phones participating FL, called clients as per terminology, train their DL model with private data sets. Second, each client uploads the parameters of model to a central server. The server adds up the parameters from each client and calculates the average value of the parameters to obtain new parameters. Finally, the new parameters are broadcast to each client and then each client continues the loop until training converges.

More specifically, we consider there are K clients participating FL and each client $k = 1, 2, 3, \ldots, K$ has a private labeled data set $D^k = \{(\boldsymbol{x}_i^k, \boldsymbol{y}_i^k) | i = 1, 2, \ldots, n\}$, where $\boldsymbol{x}_i^k$, $\boldsymbol{y}_i^k$, and n, respectively, represent the vectorized feature of a sample, the one-hot label of the sample, and the number of samples in the data set D^k. For an L-class classification task, $\boldsymbol{y}_i^k$ is a one-hot vector, i.e., $\boldsymbol{y}_i^k = [y_{i,0}^k, y_{i,1}^k, \ldots, y_{i,L-1}^k]^T$. For simplicity, we, respectively, concatenate $\{\boldsymbol{x}_i^k | i = 1, 2, \ldots, n\}$ and $\{\boldsymbol{y}_i^k | i = 1, 2, \ldots, n\}$ as matrices named $\boldsymbol{X^k}$ and $\boldsymbol{Y^k}$.

Before the training of the first communication round, the DL model of client k is initialized with the DL model of the server, i.e., $\boldsymbol{w}^k \leftarrow \boldsymbol{w}^s$, where $\boldsymbol{w}^s$ and $\boldsymbol{w}^k$, respectively, represent the initial parameters of the DL model of the server and initialized parameters of the DL model of client k.

In the training step of first communication round, client k trains its DL model with its private data set. Then, each client $k = 1, 2, 3, \ldots, K$ uploads the model parameters $\boldsymbol{w}^k$ to the server. The server aggregates the parameters and generates new parameters $\boldsymbol{w}^s$ as follows:

$$\boldsymbol{w}^s = \sum_{k=1}^{K} \frac{N^k}{N} \boldsymbol{w}^k \tag{1}$$

where N is the total number of private samples i.e., $N = \sum_{k=1}^{K} N^k$. In the second communication round, new parameters of DL model of server are broadcast to each client, i.e., $\boldsymbol{w}^k \leftarrow \boldsymbol{w}^s$ and the above processes will loop for several communication rounds.

B. Baseline 2: FD Scheme

Clients participating in the FD scheme do not need to upload parameters, instead, all clients need to upload the averaged predictions of each label, which reduces communication overhead throughout the training process [30]. The private data set D^k owned by client k can be divided into L subsets according to the labels, i.e., $D^k = D^{k,0} \cup D^{k,1} \cup \cdots \cup D^{k,L-1}$. For a subset $D^{k,l} = \{(\boldsymbol{x}_i^{k,l}, \boldsymbol{y}_i^{k,l}) | i = 1, 2, 3, \ldots, n\}$, each sample's label is a one-hot vector represent the label is l, i.e., $\boldsymbol{y}_i^{k,l} = [y_{i,0}^{k,l}, y_{i,1}^{k,l}, \ldots, y_{i,l}^{k,l}, \ldots, y_{i,L-1}^{k,l}]^T$ where $y_{i,l}^{k,l} = 1$ and others are 0. There are six steps in the FD method.

First, each client k trains its local model $\boldsymbol{w}^k$ with the data set D^k, which is same as FL. Then each client computes the average value of each label also called a local-average logit vector. The detail is as follows:

$$\hat{\boldsymbol{y}}_i^{k,l} = F\left(\boldsymbol{x}_i^{k,l} | \boldsymbol{w}^k\right) \tag{2}$$

$$\bar{\boldsymbol{y}}^{k,l} = \frac{1}{N^{k,l}} \sum_{i=1}^{N^{k,l}} \hat{\boldsymbol{y}}_i^{k,l}. \tag{3}$$

Note that client k may not have the samples of which label is l, i.e., $D^{k,l} = \emptyset$. In such situation, the local-average logit vector of label l is $\mathbf{0}$, i.e., $\bar{\boldsymbol{y}}^{k,l} = \mathbf{0}$.

After that, each client uploads the local-average logit vectors $\{\bar{\boldsymbol{y}}^{k,0}, \bar{\boldsymbol{y}}^{k,1}, \ldots, \bar{\boldsymbol{y}}^{k,L-1}\}$ to the server. Then, the global-average logit vectors are broadcast to each client. Each client computes new local-average logit vectors based on them as follows:

$$\bar{\boldsymbol{y}}^{k,l} \leftarrow \frac{1}{N^l - 1}\left(N^l \times \bar{\boldsymbol{y}}^{s,l} - \bar{\boldsymbol{y}}^{k,l}\right). \tag{4}$$

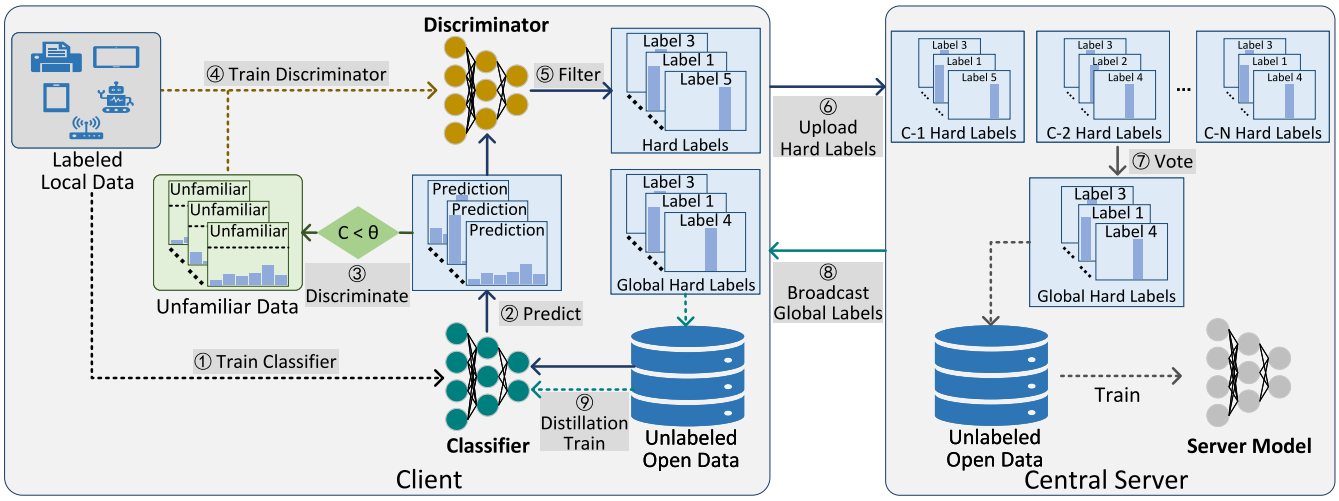


Fig. 1. Overview of proposed semisupervised FL scheme for intrusion detection. Our CNN-based classifier is trained on the labeled local data with supervised training and on unlabeled open data with distillation training. Furthermore, we introduce multiple mechanisms to jointly improve the quality of global labels.

Note that the local-average logit vector $\bar{\boldsymbol{y}}_i^{k,l}$ of each sample $\boldsymbol{x}_i^k$ whose ground truth label is l is equal to $\bar{\boldsymbol{y}}^{k,l}$. In the next, client k trains its model with the new local-average logit vectors and the ground-truth label. Although FD greatly reduces the communication overhead, the performance of FD is poor when the distribution is non-IID.

C. Baseline 3: DS-FL Scheme

The DS-FL scheme [31] consists six steps and each client not only holds a labeled private data set D^k but also an unlabeled data set $D^o = \{\boldsymbol{x}_j^o | j = 1, 2, \ldots, N^o\}$, where $\boldsymbol{x}_j^o$ and N^o, respectively, represent the vectorized feature of a sample and the number of samples in the open data set. For simplicity, we use a matrix $\boldsymbol{X}^o$ to denote the concatenated vectorized samples of D^o.

In the first step of the first communication round, each client trains its DL model with private labeled data set, which is same as FL, as shown in (1). The next step is called "Prediction" step: each client k makes predictions, i.e., local logits for the unlabeled samples as follows:

$$\hat{\boldsymbol{p}}_j^k = F\left(\boldsymbol{x}_j^o | \boldsymbol{w}^k\right) \tag{5}$$

and we use the matrix $\hat{\boldsymbol{P}}^k$ to represent the concatenated $\{\hat{\boldsymbol{p}}_j^k | j = 1, 2, \ldots, N^o\}$ of the DL model. Then, each client uploads the local logits to the central server. Then, the server aggregates the local logits and reduces the entropy of global logits $\hat{\boldsymbol{p}}_j^s$ as follows:

$$\hat{\boldsymbol{p}}_j^s = \frac{1}{K} \sum_{k=1}^{K} \hat{\boldsymbol{p}}_j^k \tag{6}$$

$$\hat{\boldsymbol{p}}_j^s \leftarrow S\left(\hat{\boldsymbol{p}}_j^s | T\right) \tag{7}$$

where $S(\cdot|T)$ is the softmax function with the T temperature, i.e.,

$$S\left(\hat{\boldsymbol{p}}_j^s | T\right) = \frac{exp\left(\frac{\hat{\boldsymbol{p}}_j^s}{T}\right)}{\sum_{l=0}^{L-1} \exp\left(\frac{\hat{p}_{j,l}^s}{T}\right)}. \tag{8}$$

When T is less than 1, the distribution of $\boldsymbol{p}_j$ will be sharper, which means that the information entropy is reduced. The motivation for reducing the entropy of global logits is to accelerate and stabilize DS-FL, particularly in non-IID data distributions [31].

Here, $\hat{\boldsymbol{P}}^s$, the matrix denoting the concatenated $\{\hat{\boldsymbol{p}}_j^s | i = 1, 2, \ldots, N^o\}$, is then broadcast to each client. Finally, each client trains its DL model with the global logits. More specifically, the process, called distillation, is as follows:

$$\boldsymbol{w}^k \leftarrow \boldsymbol{w}^k - \gamma \nabla \psi \left(\hat{\boldsymbol{P}}^k, \hat{\boldsymbol{P}}^s\right). \tag{9}$$

In addition, the server has a global model $\boldsymbol{w}^s$ and the server trains the global model with the shared unlabeled data set and the global logits, i.e.,

$$\boldsymbol{w}^s \leftarrow \boldsymbol{w}^s - \gamma \nabla \psi \left(F\left(\boldsymbol{X}^o | \boldsymbol{w}^s\right), \hat{\boldsymbol{P}}^s\right). \tag{10}$$

The above procedures are looped for a number of communication rounds. Each client in the DS-FL method needs to train its local model with private data set; however, the trained model makes predictions on open data set. If the distribution of private data set is non-IID, the distribution of samples in open data set can be extremely different from that of the private data set so this can result in wrong predictions on the part of the trained model and subsequently lower the quantity of local logits.

IV. Proposed SSFL Scheme

In this section, we first introduce the CNN-based detection model. Then, the proposed SSFL method is described in detail. The overview of our approach is shown in Fig. 1.

A. CNN-Based Baseline Model

The CNN model has a strong feature extraction ability and it can promote the information interaction between adjacent time windows and features, so we apply CNN in our model for extracting the deep features of the traffic packet. The details of the CNN-based baseline model are shown in Table I.

TABLE I
DETAILS OF CNN-BASED BASELINE MODEL

Layer	Unit	Output Size
Input Layer	Input	(80, 23, 5)
Convolution Layer 1-4	Conv1D (64,3,1)	(80, 64, 5)
Convolution Layer 5-6	Conv1D (128,3,1)	(80, 128, 5)
Convolution Layer 7	Conv1D (128,3,2)	(80, 128, 3)
Convolution Layer 8	Conv1D (128,3,2)	(80, 128, 2)
Fully-Connected Layer	Linear	(80, 128)
Output Layer	Output	(80, 11 or 2)

[1] The filter, kernel size, and stride of Conv1D settings are expressed as (filter, kernel size, stride).
[2] The batch size is set to 80 in our experiment.

Except for the output layer, the structure of the classifier network and the discriminator network is the same. Each model has eight convolutional layers to extract the feature maps of the traffic packet features effectively and two fully connected layers to output the prediction results of the model. The input channels of the first convolutional layer are 23, which is equal to the number of rows of the input. Each layer of the first four convolutional layers of the model has 64 convolution kernels and the kernel size is 3. Each layer of the last four convolutional layers has 128 kernels and the kernel size is same as the first four layer. Finally, the output of the last convolutional layer is flattened and a multilayer perceptron (MLP) with three fully connected layers is used for classification. The final layer of the classifier network and the discriminator network has 11 and 2 neurons, respectively.

Since our work focuses more on the design of an efficient federated training framework, we only briefly describe the structure of our CNN-based model. Note that our federated training scheme does not aggregate model parameters, which means that it can work even if clients adopt different model structures.

B. Proposed SSFL Scheme

FD leverages the knowledge of the clients by aggregating the logit of each class; however, it can only enhance the training effect of the corresponding class in each client, which means that efficient classification for all classes cannot be achieved when clients lack samples of some classes. Thus, we introduce an unlabeled open data set to solve this problem. Our method can use global logit to identify what class each sample in the unlabeled data set pertains. Furthermore, since it is difficult for the client model (i.e., trained with non-IID data) to generate high-quality logits on the unlabeled open data set, we introduce multiple mechanisms to jointly improve the quality of logits.

We consider there are K clients and each client $k \in \{1, 2, \ldots, K\}$ has two data sets: 1) a private labeled data set $D^{k,c} = \{(\boldsymbol{x}_i^{k,c}, \boldsymbol{y}_i^{k,c}) | i = 1, 2, \ldots, n\}$ and 2) an unlabeled open data set $D^o = \{\boldsymbol{x}_j^o | j = 1, 2, \ldots, N^o\}$ shared all over the clients. For the L-class classification task, $\boldsymbol{y}_i^{k,c}$ is a one-hot vector. In addition, each client has a classifier model $\boldsymbol{w}^{k,c}$, which is trained with the private data set. Moreover, each client has a discriminator $\boldsymbol{w}^{k,d}$. We describe the details as follows.

1) Train the Classifier: First, each client k trains its classifier $\boldsymbol{w}^{k,c}$ with the labeled private data set. This detail is expressed as

$$\boldsymbol{w}^{k,c} \leftarrow \boldsymbol{w}^{k,c} - \gamma \nabla \psi \left(\hat{\boldsymbol{Y}}^{k,c}, \boldsymbol{Y}^{k,c} \right) \tag{11}$$

where $\psi(\cdot, \cdot)$ denotes the loss function, which is minimized in this step, $\hat{\boldsymbol{Y}}^{k,c}$ represents the output of the classifier model function F, i.e., $\hat{\boldsymbol{Y}}^{k,c} = F(\boldsymbol{X}^{k,c} | \boldsymbol{w}^{k,c})$, and γ denotes the learning rate. For multiclassification tasks, the loss function can be a cross-entropy function. Then, each sample in D^o is passed through the classifier, which makes prediction on it and the confidence score of the sample is computed as

$$c_j^{k,o} = \max \left(\hat{\boldsymbol{p}}_j^k \right) = \max \left(F \left(\boldsymbol{x}_j^o | \boldsymbol{w}^{k,c} \right) \right). \tag{12}$$

2) Train the Discriminator: We first create an empty set $D^{k,d} = \emptyset$. Then, a sample $\boldsymbol{x}_j^o$ will be added in this set if the confidence score is less than a boundary number θ and the sample will be discriminated as "unfamiliar" to client k. The process is described as

$$D^{k,d} = D^{k,d} \cup \left\{ \left(\boldsymbol{x}_j^o, [0, 1]^T \right) | c_j^{k,o} < \theta \right\} \tag{13}$$

where we use one-hot label $[0, 1]^T$ to denote the unfamiliar label. What is more, we consider each sample in the private data set is "familiar" to the client k, so all the samples in private data set are added into $D^{k,d}$ and the labels are "familiar." The process is expressed as

$$D^{k,d} = D^{k,d} \cup \left\{ \left(\boldsymbol{x}_i^{k,c}, [1, 0]^T \right) | i = 1, 2, \ldots, N^{k,c} \right\} \tag{14}$$

where we use one-hot label $[1, 0]^T$ to denote the familiar label. Hence, $D^{k,d}$ data set contains two categories data after this step has finished. Then, the discriminator $\boldsymbol{w}^{k,d}$ is trained with the $D^{k,d}$ data set.

3) Filter and Upload: The predictions of unlabeled open samples in step 2 are filtered by the trained discriminator. We first compute the discriminating result $d_j^{k,o}$ as

$$d_j^{k,o} = \arg\max \left(F \left(\boldsymbol{x}_j^o | \boldsymbol{w}^{k,d} \right) \right). \tag{15}$$

The output of $F(\cdot | \boldsymbol{w}^{k,d})$ is a 2-D vector. We regard $\arg\max(\cdot)$ as the index of the max value of a vector. As discussed in the previous step, if $d_j^{k,o} = 0$, $\boldsymbol{x}_j^o$ is familiar to client k or the

sample is unfamiliar. For each sample $\boldsymbol{x}_j^o$ in D^o, the prediction $\hat{p}_j^k$ is filtered as follows:

$$\hat{p}_j^k = \begin{cases} \arg\max\left(F\left(\boldsymbol{x}_j^o|\boldsymbol{w}^{k,c}\right)\right), & \text{if } d_j^{k,o} = 0 \\ -1, & \text{if } d_j^{k,o} = 1 \end{cases} \tag{16}$$

where we use "-1" to denote the sample is unfamiliar. Then, each client uploads the hard labels of unlabeled open data (i.e., $\{\hat{p}_j^k | j = 1, 2, \ldots, N^o\}$) to the central server. Note that the unlabeled open data is the same for each client; however, the "familiar" samples in the unlabeled open data discriminated by the discriminator of each client are different.

4) Vote and Broadcast: For a sample $\boldsymbol{x}_j^o$, we consider that there are L voting sets, i.e., $\{V^{j,0}, V^{j,1}, \ldots, V^{j,L-1}\}$. Class -1 is used to identify unfamiliar samples, so the class -1 is not included in the L voting sets. For the predicted label $\hat{p}_j^k$ of the sample, if $\hat{p}_j^k \neq -1$, this predicted label is added into the corresponding voting set.

After all clients input the predicted label of the sample (excluding unfamiliar samples) into the corresponding voting set, the central server determines the global hard label of this sample according to the number of votes in each voting set. The global hard label of the sample is calculated as follows:

$$\hat{p}_j^s = \arg\max\left(\left[\left|V^{j,0}\right|, \left|V^{j,1}\right|, \ldots, \left|V^{j,L-1}\right|\right]\right). \tag{17}$$

For simplicity, we use matrix $\hat{\boldsymbol{P}}^s$ to represent the concatenated $\{\hat{p}_j^s | j = 1, 2, \ldots, N^o\}$. Finally, global hard labels (i.e., $\hat{\boldsymbol{P}}^s$) are broadcast to each client.

5) Distillation: Each client treats itself as a student and uploads the predicted labels of unlabeled open data to the central server, and then the aggregated global hard labels of unlabeled open data from the central server act as the teacher, where each local client model uses unlabeled open data with global hard labels for distillation training. The process is as follows:

$$\boldsymbol{w}^{k,c} \leftarrow \boldsymbol{w}^{k,c} - \gamma \nabla \psi\left(\hat{\boldsymbol{P}}^{k,c}, \hat{\boldsymbol{P}}^s\right) \tag{18}$$

where $\hat{\boldsymbol{P}}^{k,c} = F(\boldsymbol{X}^o|\boldsymbol{w}^{k,c})$. The classifier model of central server is also trained with $\hat{\boldsymbol{P}}^s$ and it will be used for evaluation.

The overall training procedure of SSFL is illustrated in Algorithm 1.

Algorithm 1: Pipeline of the Proposed Method on K Clients for T Rounds

```
for each client k in parallel do
    1.Train the classifier:
    train its classifier w^{k,c} via (11)
    for each x_j^o in D^o do
        compute c_j^{k,o} via (12)
    end
    2.Train the discriminator:
    create D^{k,d} via (13) and (14)
    train w^{k,d} with D^{k,d}
    3.Filter and upload:
    filter predictions via (16)
    upload predictions to server
end
4.Vote and broadcast (on the central server):
for j ← 1 to N^o do
    vote and compute p̂_j^s via (17)
end
broadcast the vote result to each client
5.Distillation:
for each client k in parallel do
    train classifier via (18)
end
The 5 steps are iterated for T rounds
```

V. Experimental Results

In this section, we present and discuss our experimental results. In particular, the experiments will answer the three research questions.

1) *RQ1:* Can SSFL achieve higher detection accuracy than the state-of-the-art methods? (Section V-D).
2) *RQ2:* Can SSFL achieve faster training process and lower communication overhead? (Section V-E).
3) *RQ3:* How effective is each component of SSFL in improving performance? (Section V-F).

A. Description of Data Set

The *N-BaIoT* data set is a public data set, which has nine subdata sets collected from nine IoT devices (e.g., doorbell and camera) [39]. Among them, seven IoT devices have 11 categories of traffic: one benign traffic and ten attack traffic. The other two devices have six categories of traffic: one benign traffic and five attack traffic. Each traffic packet has a total of 115 features extracted from most recent five different time windows: 100 ms, 500 ms, 1.5 s, 10 s, and 1 min. These features can be computed quickly to meet the requirements of real-time detection of malicious traffic packets. The *N-BaIoT* data set has comprehensive traffic categories and a large number of traffic records, so it is widely used in the field of intrusion detection as a baseline data set.

B. Data Preprocessing

We preprocess the data in three steps, namely, data set partition, normalization, and two-dimensionalization.

1) Data Partition: In the real world, there are usually a large number of clients with non-IID data participating in FL training. Thus, we divide the raw data set to meet this deployment situation.

Using the raw data set for training and analysis requires expensive resources and infrastructure, so we first devised a new data set *mini-N-BaIoT* consisting of 11 categories of traffic from nine IoT devices. Specifically, we consider the subdata sets of nine IoT devices in the raw data set as $D_{d_1}, D_{d_2}, \ldots, D_{d_9}$. D_{d_i} is divided into L_{d_i} subsets according to the traffic category, which means that each subset $D_{d_i,l}$ only contains traffic data whose category is l. We select 1000 traffic data records in each subset $D_{d_i,l}$ as $D_{d_i,l}^{\text{mini}}$ for the *mini-N-BaIoT* data set.

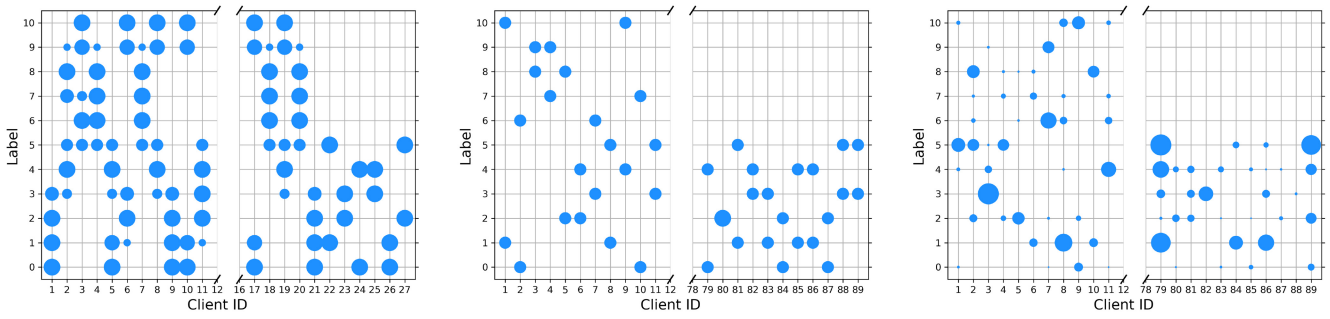


Fig. 2. Schematic illustration of samples per class allocated to each client in different scenarios, where the x-axis indicates client IDs, the y-axis indicates class labels, and the size of scattered points indicates the number of labeled samples. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

Then, our new data set is divided into private data set D^p, open data set D^o, and test data set D^{test} according to the proportion of 70%, 10%, and 20%. Note that each set is disjoint with the other two sets, and D^o does not contain the label.

Note that all data records of D^p are part of D_{d_i} and D^p is distributed to K_{d_i} clients in the following three scenarios for experiments.

1) *Scenario 1:* D^p is sorted by its classification label and is divided into $2 \times K_{d_i}$ shards of size $|D^p|/(2K_{d_i})$ among which two shards are assigned to each client. In this scenario, K_{d_i} is set to 3. This strategy follows the pioneer study [31].
2) *Scenario 2:* The distribution strategy in Scenario 2 is similar to Scenario 1 but K_{d_i} is equal to L_{d_i}. So if D_{d_i} contains 11 categories traffic data, K_{d_i} is 11. If D_{d_i} contains six categories traffic data, K_{d_i} is 6.
3) *Scenario 3:* In this scenario, we use a dirichlet distribution $\boldsymbol{Dir}(\alpha)$, in which a smaller α indicates higher data heterogeneity. We set $\alpha = 0.1$ in experiments and K_{d_i} is equal to L_{d_i}.

After the partition, the distributions of these data sets satisfy the following conditions.

1) The private data set of client k only comes from one device and this is consistent with the real world.
2) The amount of data and classes of data owned by different clients are quite different. The distribution is non-IID.
3) The intersection of D^o, D^p, and D^{test} with each other is the empty set.

Fig. 2 shows the examples of distribution results of three scenarios, where the x-axis indicates client IDs, y-axis indicates labels, and the size of scattered points indicates the number of training samples.

2) Normalization: Since the traffic data have features of different dimensions, the level of each dimension of the data set varies greatly. To train our model more effectively, the values of features are scaled to numbers between 0 and 1 using min–max normalization.

3) Dimensionalization: Each sample X_i has 115 features, which can be divided into five parts according to the time window. So we divide the vectorized feature of a sample into five parts and transfer the vector to a matrix, which has five columns and 23 rows to input the sample to the CNN model proposed in the next. The detail is as follows:

$$\boldsymbol{x}_i = \begin{bmatrix} x_{i,0} & x_{i,23} & x_{i,46} & x_{i,69} & x_{i,92} \\ x_{i,1} & x_{i,24} & x_{i,47} & x_{i,70} & x_{i,93} \\ x_{i,2} & x_{i,25} & x_{i,48} & x_{i,71} & x_{i,94} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i,22} & x_{i,45} & x_{i,68} & x_{i,91} & x_{i,114} \end{bmatrix}. \tag{19}$$

C. Experiment Setup

To comprehensively evaluate the performance and verify the scalability of the method, we conduct evaluations in three different scenarios. The data distributions in the three scenarios are all non-IID, but the data splitting strategies are different (see Fig. 2). Scenario 2 and Scenario 3 both contain 89 clients, which means that the traffic data of each client in Scenario 2 and Scenario 3 are less than Scenario 1 (27 clients). In Scenario 3, the dirichlet data splitting strategy ($\alpha = 0.1$) is used to distributing the private data set. To make the experimental evaluation clearer, we introduce the experimental environment, implementation details, and evaluation metrics as follows.

1) Experimental Environment: All the evaluations are conducted in Python 3.7 with the PyTorch framework of version 1.9.0 and running on the PC with Intel Core i9-11900K@3.50 GHz, 64-GB RAM, and an NVIDIA GeForce RTX3090 GPU.

2) Implementation Details: In the training phase, we train the model with an Adam optimizer. The learning rate, the batch size, and the local train epochs in each communication round is set to 0.0001, 100, and 5, respectively. The threshold θ for judging unfamiliar samples is not set to a fixed value for all clients. For a specific client, this value was set to the median of the clients' predicted probabilities.

3) Evaluation Metrics: To measure the performance of our method, we calculate the number of true positive (T_p), true negative (T_n), false positive (F_p), and false negative (F_n). Based on the above definition, recall, precision, and F_1 can be obtained

$$\text{Recall} = \frac{T_p}{T_p + F_n} \tag{20}$$

$$\text{Precision} = \frac{T_p}{T_p + F_p} \tag{21}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{22}$$

TABLE II
COMPARISON OF METRICS WITH OTHER METHODS

Method	Scenario 1			Scenario 2			Scenario 3		
	Accuracy	F_1-Score	Precision	Accuracy	F_1-Score	Precision	Accuracy	F_1-Score	Precision
FL	86.11%	85.13%	91.29%	81.38%	81.92%	87.34%	81.13%	81.64%	86.70%
FD	48.54%	35.76%	33.69%	20.12%	8.53%	10.37%	53.06%	43.27%	44.31%
DS-FL	50.49%	40.85%	48.27%	53.53%	43.95%	57.45%	20.01%	7.31%	10.96%
MLP	82.78%	81.29%	87.45%	82.85%	81.31%	87.57%	81.28%	79.61%	87.14%
LSTM	72.24%	66.77%	74.50%	69.69%	65.20%	64.37%	60.80%	58.08%	60.86%
Ours	**87.40%**	**86.50%**	**92.33%**	**86.70%**	**84.95%**	**91.73%**	**84.22%**	**82.47%**	**90.17%**

D. Detection Performance of SSFL for Intrusion Detection

In this experiment, we evaluate the detection performance of different methods by measuring *Accuracy*, *F_1-Score*, and *Precision*. Our SSFL is compared with the following five baseline methods.

The detailed description of the three federated training schemes (i.e., FL, FD, and DS-FL) can be found in Section III. *FL* is the most common scheme in federated learning, which directly aggregates model parameters uploaded by different clients. *FD* is proposed by Jeong *et al.* as a distillation-based FL scheme that clients share per-class logits instead of model parameters. *DS-FL* is proposed by Itahara *et al.* [31] that leverages the exchanged model outputs to label the unlabeled open data. Compared with DS-FL scheme, our method has two key differences: 1) we take the discriminator to improve the quality of uploaded labels and 2) our hard-label strategy and voting mechanism can further reduce the communication overhead. Moreover, *MLP* and *LSTM* are introduced to compare the detection performance with our CNN model, and both are trained using our scheme. To ensure fair comparison, we perform on the same data set in different scenarios, and carefully follow the same evaluation protocol.

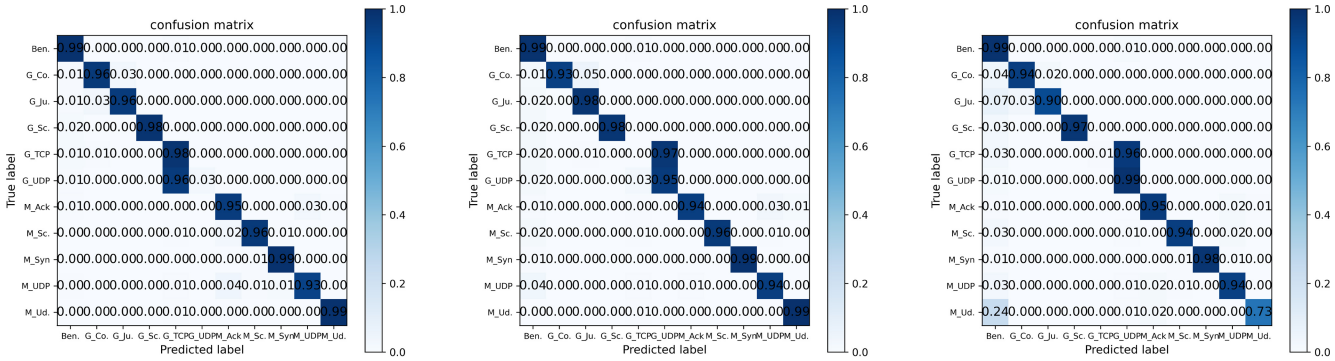
Table II illustrates the results. We can observe that SSFL outperforms other baselines with a considerable margin. In the federated training manner, neither FD nor DS-FL can achieve high detection accuracy. Specifically, the detection accuracy of FD in Scenarios 1 and 3 is about 50%, and the accuracy rate in Scenario 2 is only 22.12%. There are two reasons: 1) the distillation training of FD is difficult to implement when the data distribution is non-IID (i.e., the local data of the client cannot represent the population distribution), so the highest accuracy is obtained by one client trained with its local traffic data and 2) in Scenario 2, each client has fewer categories of traffic, which leads to lower accuracy. Since the best detection performance of FD is obtained by a single client, it can be considered that the FD method is ineffective in the case of non-IID data distribution. The accuracy of DS-FL is higher than that of FD in Scenario 1 and Scenario 2, indicating that the introduction of unlabeled open data improves the usability of the distillation-based FL under non-IID data distribution. However, the more uneven data distribution in Scenario 3 makes DS-FL unable to predict correctly on the open data, resulting in poor detection performance. As a traditional federated training scheme, the FL scheme achieves good detection performance, but the problems of huge communication overhead cannot be ignored, which will be discussed later. The FL scheme also suffers from the risk of gradient leakage, that is, the attacker can recover the raw data through the model parameters uploaded by the client, which means that the FL scheme is not very secure [29]. Our method uploads hard labels from each client instead of gradients or parameters, making attack methods that recover user data from uploaded gradients infeasible. Our scheme achieves more effective federated training for the following reasons: 1) similar to the DS-FL, we also leverage open data to better suit the case of non-IID data distribution; 2) the discriminator of each client can significantly improve the quality of predicted labels; and 3) the voting mechanism of the central server can effectively aggregate the training results from different clients. We can also find that the detection performance of both the LSTM model and the MLP model is lower than our CNN model. Besides, the federated training task of Scenario 3 is the most challenging, which also results in a lower accuracy of SSFL than the other two scenarios.

To further evaluate the classification performance of SSFL, we analyze the confusion matrices of SSFL in the three scenarios on the test data set. As shown in Fig. 3, SSFL can effectively classify different traffic categories, and the detection accuracy of benign traffic in all three scenarios reaches more than 99%. G_TCP and G_UDP are two categories of attack traffic that are easily confused. We conjecture this is largely due to that they are both DoS attacks and have very similar characteristics. It can be concluded that our scheme with the excellent detection performance is suitable for federated training and detecting intrusions in the IoT networks.

E. Communication Efficiency of the Proposed SSFL Method

Tables III and IV answer *RQ2* by showing the results of our model against baselines. In overall, SSFL achieves faster training process and lower communication overhead.

In terms of learning efficiency, we can observe from Table III that SSFL has the fastest learning speed with excellent performance and outperforms other baselines. Specifically, SSFL achieves high accuracy in the first ten epochs, and achieves convergence around 150 epochs. Our multiple mechanisms jointly obtain high-quality global labels for federated training, which are undoubtedly the key to achieve fast convergence and high performance. The Top-1 accuracy of the



(a) (b) (c)

Fig. 3. Confusion matrices of SSFL on the test data set. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

TABLE III
COMPARISON OF TOP-1 TEST ACCURACY WITH OTHER METHODS AT DIFFERENT COMMUNICATION ROUND

Method	Top-1 Test Accuracy (%) @ Communication Round														
	Scenario 1					Scenario 2					Scenario 3				
	10	50	100	150	200	10	50	100	150	200	10	50	100	150	200
FL	39.31	59.71	68.00	73.24	73.79	10.11	28.66	38.81	48.00	57.96	10.22	33.89	45.41	56.27	63.35
FD	44.88	47.21	48.45	48.54	48.54	20.10	20.11	20.11	20.12	20.12	44.07	47.56	52.21	53.06	53.06
DS-FL	50.49	50.49	50.49	50.49	50.49	30.76	53.53	53.53	53.53	53.53	19.85	19.93	20.01	20.01	20.01
MLP	68.86	78.28	81.02	81.87	82.78	26.84	**80.51**	81.53	82.11	82.85	70.97	76.74	79.19	80.13	81.28
LSTM	37.31	62.47	68.77	70.44	72.24	10.11	41.29	48.21	55.76	60.80	10.11	41.29	48.21	55.76	60.80
Ours	**77.90**	**83.81**	**84.90**	**87.19**	**87.40**	**75.26**	80.43	**85.09**	**86.31**	**86.70**	**72.16**	**78.39**	**83.28**	**83.84**	**84.22**

TABLE IV
COMPARISON OF COMMUNICATION OVERHEAD AND TOP-ACC

Method	C@D^o	Communication Cost (MB) @ Test Accuracy (%)											
		Scenario 1				Scenario 2				Scenario 3			
		C@50	C@75	C@Top-Acc	Top-Acc	C@50	C@75	C@Top-Acc	Top-Acc	C@50	C@75	C@Top-Acc	Top-Acc
FL	–	15.81	216.29	1711.43	86.11%	137.04	514.14	1745.06	81.38%	110.69	473.75	1700.12	81.13%
FD	–	–	–	0.13	48.54%	–	–	0.02	20.12%	–	–	0.19	53.06%
DS-FL	0.96	5.04	–	5.04	50.49%	–	–	22.63	53.53%	–	–	46.35	20.01%
Ours	0.96	0.01	0.02	0.55	87.40%	0.01	0.02	0.49	86.70%	0.01	0.04	0.47	84.22%

[1] We highlight the best in and the worst in .
[2] "C@D^o" is the communication overhead of distributing the open dataset. "C@x" represents the cumulative communication overhead required to achieve a test accuracy of x.

FL scheme as 200 epochs is still far from the highest accuracy, which reflects that the speed of federated training by directly averaging parameters is slow. In addition, FD and DS-FL quickly reach the upper limit of detection performance and cannot further improve performance through communication. The convergence speed of MLP and LSTM models is slower than that of CNN models. Since our method is not affected by the size of model parameters, subsequent experiments on communication overhead are not implemented on MLP and LSTM models.

Table IV lists the communication overhead of each method. Note that in FL and FD, there is no need to distribute the open data set to each client. From this table, we notice that our proposed method is communication efficient and the amount of data uploaded by our method is very small. Since the FL method communicates model parameters in each round, the large model size leads to huge communication overhead. The communication overhead of other three distillation-based methods depends only on the output dimensions of the models and does not scale up according to the model size, so these schemes can achieve low communication costs. Furthermore, the client in our proposed method uploads the hard labels of local predictions, so the communication overhead is much lower than DS-FL in which a client needs to upload the vectors of local predictions. Unfortunately, both DS-FL and FD methods have not achieved satisfactory detection performance due to the lack of effective mechanisms to deal with non-IID private traffic data. It can be concluded from Table IV that SSFL has the best overall performance because of its high classification accuracy and communication efficiency.

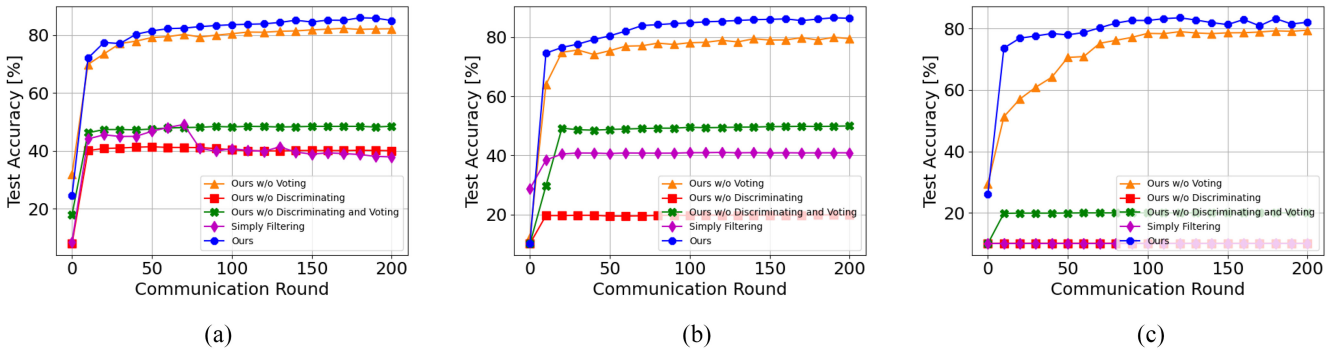


Fig. 4. Test accuracy curves of our SSFL, ours w/o voting, ours w/o discriminating, ours w/o discriminating and voting, and simply filtering. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

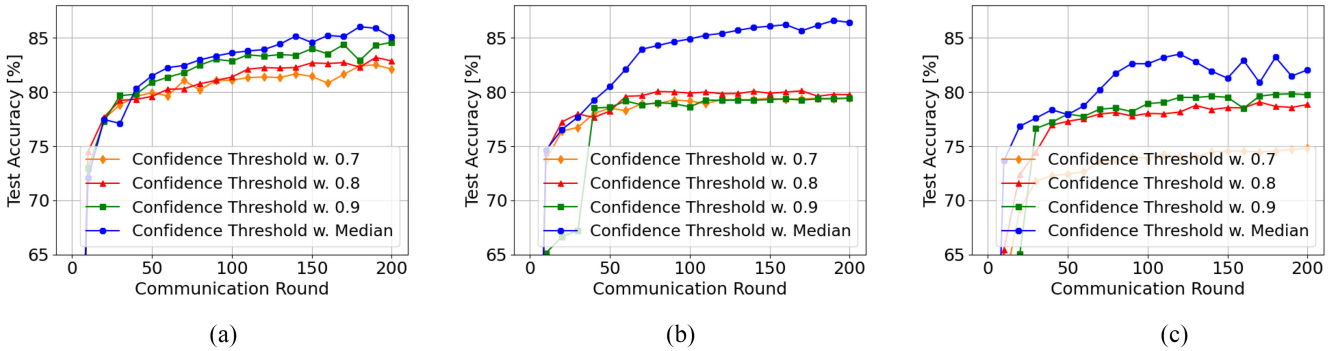


Fig. 5. Impact of confidence threshold θ_c. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

F. Ablation Study

To further examine the level of benefit that each component of SSFL brings to the performance, we conduct a three-stage ablation study on SSFL. In the first stage, we ablate different key components and analyze the detection accuracy. In the second stage, we further evaluate the impact of different confidence thresholds. In the third stage, we analyze the impact of different labeling strategies on detection performance and communication overhead.

As shown in Fig. 4, we first ablate discriminating, voting, and both discriminating and voting from our method. Note that the discriminating corresponds to the third to fifth steps of our method (see the description in Section IV). Without the discriminating, client k makes predictions on all unlabeled samples, including familiar and unfamiliar ones. Besides, we evaluate the detection performance of using simply filtering instead of the discriminator. In simply filtering, whether an open sample $\boldsymbol{x}_j^o$ is familiar or not only depends on whether its confidence score $c_j^{k,o}$ is less than the threshold θ_s (i.e., also without the discriminator), and the threshold θ_s for each client is equal to the median confidence score of its predicted labels. If $c_j^{k,o} < \theta_s$, the prediction $\hat{p}_j^k$ of $\boldsymbol{x}_j^o$ will be set -1. It can be seen that the removal of discriminating in all three scenarios leads to significant performance drops, which illustrates the necessity of discriminator for improving the quality of each client's predicted labels. Obviously, the use of the discriminator has the largest impact on detection performance. The ablation research on the voting mechanism shows that the voting mechanism makes the model performance worse when the client makes too many incorrect predictions. In other words, the voting mechanism is a voting majority, so it needs to be used in conjunction with a discriminator that improves the quality of client predicted labels. Moreover, the filtering method, on its own, cannot effectively deal with the problems caused by non-IID private data. In Scenario 3, the extremely uneven data distribution makes the simply filtering completely ineffective. Thus, it could be concluded that both discriminating and voting contribute to SSFL to a considerable extent.

θ_c is a key hyperparameter of our method. As shown in (12) and (14), if $c_j^{k,o} < \theta_c$, the sample $\boldsymbol{x}_j^o$ will be regraded as "unfamiliar" to client k and will be used to train the discriminator. So the values of θ_c can have a certain influence on the performance of our method. In this section, we set the value of to 0.9, 0.8, 0.7, and the median value. More specifically, we consider the set of confidence scores is $C^k = \{c_j^{k,o} | j = 1, 2, \ldots, N^o\}$ and the median mentioned above is the median of this set. As shown in Fig. 5, when we set a fixed value of θ_c for all clients participating in our method, the performance is worse than that the value of θ_c is median value of C^k. Furthermore, comparing the results of the three scenarios, we can observe that using the median value as the confidence threshold has a greater performance advantage if the client has less local labeled data with fewer traffic categories.

We also investigate the impact of labeling strategies on detection performance and communication overhead. Our

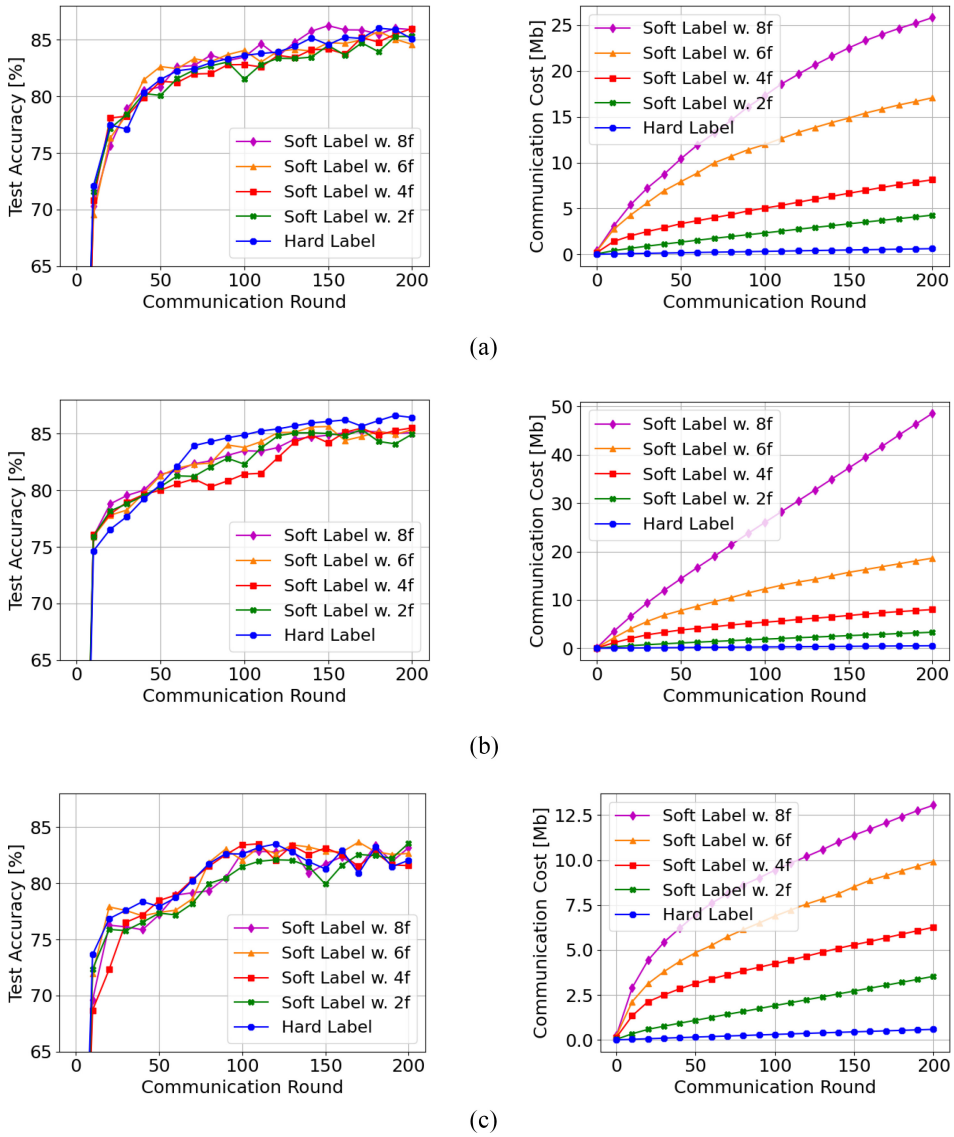


Fig. 6. Test accuracy curve and communication overhead curve of different label strategies. "Soft Label w. xf" means that a client uploads soft labels to the server and each float number in the soft labels is rounded down to "x" decimal places. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

motivation for adopting the hard-label strategy is not to improve the detection performance, but to reduce the communication overhead generated during federated training. The soft label of a sample is a probability vector in which each number is a float number of double type. Before a client uploads this vector, each double number in this vector can be rounded down to a certain decimal places in order to reduce the memory size of soft labels. As shown in Fig. 6, rounding down to 8, 6, 4, or 2 decimal places can achieve almost same accuracy but the communication overhead is quite different. The results show that the fewer decimal places kept, the lower the communication overhead. Therefore, uploading hard labels can greatly reduce the communication overhead at the same time achieving high detection performance.

VI. Conclusion

In this article, we proposed a semisupervised FL scheme via knowledge distillation for intrusion detection, which is the first distillation-based FL method in the field of traffic classification. Our key idea is to leverage the unlabeled open data for enhancing the classifier performance. Then, we build a CNN-based model for extracting deep features of the traffic packets, and take this model as both the classifier network and discriminator network. The discriminator is used to improve the quality of each client's predicted labels, which avoids the failure of distillation training caused by a large number of incorrect predictions under private non-IID data. Additionally, we further reduce communication overhead with hard-label strategy and voting mechanism. To simulate the real deployment environment in the IoT network (i.e., with a large number of devices and uneven traffic data distribution), we set up three challenging scenarios to evaluate our method. The experimental results show that SSFL can achieve better detection performance as well as lower communication overhead than other schemes. Our method meets the security, accuracy, and efficiency required for federated training and detecting intrusions in the IoT network. In future work, we will set a strategy

to score the client's contribution, which can better solve the situation where malicious clients affect federated training.

REFERENCES

[1] H. Yao, P. Gao, P. Zhang, J. Wang, C. Jiang, and L. Lu, "Hybrid intrusion detection system for edge-based IIoT relying on machine-learning-aided detection," *IEEE Netw.*, vol. 33, no. 5, pp. 75–81, Sep./Oct. 2019.

[2] S. Rajendran, Z. Sun, F. Lin, and K. Ren, "Injecting reliable radio frequency fingerprints using metasurface for the Internet of Things," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 1896–1911, Dec. 2020, doi: 10.1109/TIFS.2020.3045318.

[3] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the Internet of Things networks," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4944–4956, Mar. 2021, doi: 10.1109/JIOT.2020.3034156.

[4] S. Bu, F. R. Yu, X. P. Liu, and H. Tang, "Structural results for combined continuous user authentication and intrusion detection in high security mobile ad-hoc networks," *IEEE Trans. Commun.*, vol. 10, no. 9, pp. 3064–3073, Sep. 2011.

[5] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *Proc. GLOBECOM*, Taipei, Taiwan, Dec. 2020, pp. 1–7.

[6] R. Zhao *et al.*., "A novel intrusion detection method based on lightweight neural network for Internet of Things," *IEEE Internet Things J.*, early access, Oct. 11, 2020, doi: 10.1109/JIOT.2021.3119055.

[7] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proc. NDSS*, San Diego, CA, USA, Feb. 2018, pp. 1–15.

[8] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multi-tiered hybrid intrusion detection system for Internet of Vehicles," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 616–632, Jan. 2022.

[9] T. Ye, G. Li, I. Ahmad, C. Zhang, X. Lin, and J. Li, "FLAG: Few-shot latent Dirichlet generative learning for semantic-aware traffic detection," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 1, pp. 73–88, Mar. 2022.

[10] C. Fu, Q. Li, M. Shen, and K. Xu, "Realtime robust malicious traffic detection via frequency domain analysis," in *Proc. CCS*, Virtual, South Korea, Nov. 2021, pp. 3431–3446.

[11] L. Yang, A. Moubayed, I. Hamieh, and A. Shami, "Tree-based intelligent intrusion detection system in Internet of Vehicles," in *Proc. GLOBECOM*, Dec. 2019, pp. 1–6.

[12] G. Gui, M. Liu, F. Tang, N. Kato, and F. Adachi, "6G: Opening new horizons for integration of comfort, security and intelligence," *IEEE Wireless Commun. Mag.*, vol. 27, no. 5, pp. 126–132, Oct. 2020.

[13] Y. Lin, J. Wang, Y. Tu, L. Chen, and Z. Dou, "Time-related network intrusion detection model: A deep learning method," in *Proc. GLOBECOM*, Dec. 2019, pp. 1–6.

[14] G. Andresini, A. Appice, and D. Malerba, "Nearest cluster-based intrusion detection through convolutional neural networks," *Knowl. Based Syst.*, vol. 216, Mar. 2021, Art. no. 106798.

[15] Y. Cheng, J. Lu, D. Niyato, B. Lyu, J. Kang, and S. Zhu, "Federated transfer learning with client selection for intrusion detection in mobile edge computing," *IEEE Commun. Lett.*, vol. 26, no. 3, pp. 552–556, Mar. 2022, doi: 10.1109/LCOMM.2022.3140273.

[16] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.

[17] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in IoT edge devices," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3930–3944, Mar. 2022, doi: 10.1109/JIOT.2021.3100755.

[18] N. A. A.-A. Al-Marri, B. S. Ciftler, and M. Abdallah, "Federated MIMIC learning for privacy preserving intrusion detection," in *Proc. BlackSeaCom*, Odessa, Ukraine, May 2020, pp. 1–6.

[19] X. Hei, X. Yin, Y. Wang, J. Ren, and L. Zhu, "A trusted feature aggregator federated learning for distributed malicious attack detection," *Comput. Security*, vol. 99, Dec. 2020, Art. no. 102033.

[20] Z. Zhang *et al.*, "Robust semi-supervised federated learning for images automatic recognition in Internet of Drones," *IEEE Internet Things J.*, early access, Feb. 16, 2022, doi: 10.1109/JIOT.2022.3151945.

[21] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.

[22] Y. Liu *et al.*, "Boosting privately: Federated extreme gradient boosting for mobile crowdsensing," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Singapore, Nov./Dec. 2020, pp. 1–11.

[23] Y. Wang, G. Gui, H. Gacanin, T. Ohtsuki, O. A. Dobre, and H. V. Poor, "An efficient specific emitter identification method based on complex-valued neural networks and network compression," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2305–2317, Aug. 2021.

[24] N. Zhang and M. Tao, "Gradient statistics aware power control for over-the-air federated learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 8, pp. 5115–5128, Aug. 2021.

[25] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, Cadiz, Spain, May 2016, pp. 1273–1282.

[26] F. Ang, L. Chen, N. Zhao, Y. Chen, W. Wang, and F. R. Yu, "Robust federated learning with noisy communication," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3452–3464, Jun. 2020.

[27] X. Fu *et al.*, "Lightweight automatic modulation classification based on decentralized learning," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 1, pp. 57–70, Mar. 2022, doi: 10.1109/TCCN.2021.3089178.

[28] Y. Liu *et al.*, "Deep anomaly detection for time-series data in Industrial IoT: A communication-efficient on-device federated learning approach," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6348–6358, Apr. 2021.

[29] L. Zhu and S. Han, "Deep leakage from gradients," in *Proc. Feder. Learn.*, 2020, pp. 17–31.

[30] E Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data," in *Proc. NIPS*, Dec. 2018, pp. 1–6.

[31] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-IID private data," *IEEE Trans. Mobile Comput.*, early access, Mar. 31, 2021, doi: 10.1109/TMC.2021.3070013.

[32] D. Sui, Y. Chen, J. Zhao, Y. Jia, Y. Xie, and W. Sun, "FedED: Federated learning via ensemble distillation for medical relation extraction," in *Proc. EMNLP*, Nov. 2020, pp. 2118–2128.

[33] Z. Ma and A. Kaban, "K-nearest-neighbours with a novel similarity measure for intrusion detection," in *Proc. UKCI*, Guildford, U.K., Sep. 2013, pp. 266–271.

[34] R. Kumari, Sheetanshu, M. K. Singh, R. Jha, and N. K. Singh, "Anomaly detection in network traffic using K-mean clustering," in *Proc. RAIT*, Dhanbad, India, Mar. 2016, pp. 387–393, doi: 10.1109/RAIT.2016.7507933.

[35] M. Usha and P. Kavitha, "Anomaly based intrusion detection for 802.11 networks with optimal features using SVM classifier," *Wireless Netw.*, vol. 23, no. 8, pp. 2431–2446, 2017.

[36] Y. Shen, K. Zheng, C. Wu, M. Zhang, X. Niu, and Y. Yang, "An ensemble method based on selection using BAT algorithm for intrusion detection," *Comput. J.*, vol. 61, no. 4, pp. 526–538, 2018.

[37] H. Wang, J. Gu, and S. Wang, "An effective intrusion detection framework based on SVM with feature augmentation," *Knowl. Based Syst.*, vol. 139, pp. 130–139, Nov. 2017.

[38] Q. Qin, K. Poularakis, K. K. Leung, and L. Tassiulas, "Line-speed and scalable intrusion detection at the network edge via federated learning," in *Proc. IFIP Netw. Conf.*, Paris, France, Jun. 2020, pp. 352–360.

[39] Y. Meidan *et al.*, "N-BaIoT: Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul.–Sep. 2018.

Ruijie Zhao (Graduate Student Member, IEEE) received the M.S. degree from Shanghai Jiao Tong University, Shanghai, China, in 2021, where he is currently pursuing the Ph.D. degree with the School of Electronic Information and Electrical Engineering.

His research interests include machine learning, network security, and Internet of Things.

Yijun Wang received the B.S. and M.S. degrees from Shanghai Jiao Tong University, Shanghai, China, in 2001 and 2004, respectively.

He is currently an Engineer with the School of Electronic Information and and Electrical Engineering, Shanghai Jiao Tong University. His research interests include system security, network security, and machine learning.

Zhi Xue received the B.S. and Ph.D. degrees from Shanghai Jiao Tong University, Shanghai, China, in 1992 and 2001, respectively.

He is currently a Professor with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University. His research interests include machine learning, network security, and data science.

Tomoaki Ohtsuki (Senior Member, IEEE) received the B.E., M.E., and Ph.D. degrees in electrical engineering from Keio University, Yokohama, Japan, in 1990, 1992, and 1994, respectively.

From 1994 to 1995, he was a Postdoctoral Fellow and a Visiting Researcher of Electrical Engineering with Keio University. From 1993 to 1995, he was a Special Researcher of Fellowships of the Japan Society for the Promotion of Science for Japanese Junior Scientists. From 1995 to 2005, he was with Science University of Tokyo, Tokyo, Japan. In 2005, he joined Keio University, where he is currently a Professor. From 1998 to 1999, he was with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA, USA. He is engaged in research on wireless communications, optical communications, signal processing, and information theory. He has published more than 205 journal papers and 415 international conference papers.

Dr. Ohtsuki is a recipient of the 1997 Inoue Research Award for Young Scientist, the 1997 Hiroshi Ando Memorial Young Engineering Award, the Ericsson Young Scientist Award 2000, the 2002 Funai Information and Science Award for Young Scientist, the IEEE the 1st Asia–Pacific Young Researcher Award 2001, the 5th International Communication Foundation Research Award, the 2011 IEEE SPCE Outstanding Service Award, the 27th TELECOM System Technology Award, the ETRI Journal's 2012 Best Reviewer Award, and the 9th International Conference on Communications and Networking in China 2014 Best Paper Award. He served as the Chair of IEEE Communications Society, and Signal Processing for Communications and Electronics Technical Committee. He served as a Technical Editor of the *IEEE Wireless Communications Magazine* and an Editor of *Physical Communications* (Elsevier). He is currently serving as an Area Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY and an Editor of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS. He has served as the General Co-Chair, Symposium Co-Chair, and TPC Co-Chair of many conferences, including IEEE GLOBECOM 2008, SPC, IEEE ICC 2011, CTS, IEEE GLOBECOM 2012, SPC, IEEE ICC 2020, SPC, IEEE APWCS, IEEE SPAWC, and IEEE VTC. He gave tutorials and keynote speeches at many international conferences, including IEEE VTC, IEEE PIMRC, and IEEE WCNC. He was the Vice President and the President of the Communications Society of IEICE. He is a Distinguished Lecturer of IEEE, a Fellow of IEICE, and a member of the Engineering Academy of Japan.

Bamidele Adebisi (Senior Member, IEEE) received the bachelor's degree in electrical engineering from Ahmadu Bello University, Zaria, Nigeria, in 1999, and the master's degree in advanced mobile communication engineering and the Ph.D. degree in communication systems from Lancaster University, Lancaster, U.K., in 2003 and 2009, respectively.

He was a Senior Research Associate with the School of Computing and Communication, Lancaster University from 2005 to 2012. He joined the Manchester Metropolitan University, Manchester, U.K., in 2012, where he is currently a Full Professor (Chair) of Intelligent Infrastructure Systems. He has published over 140 peer-review papers and given several talks/panel discussions in the research areas of Internet of Things, smart cities, smart grids, communication systems, and cyber physical systems.

Prof. Adebisi received the 2020 U.K. Best Knowledge Transfer Partnership Project of the Year Awards for one of his projects with an SME. He is currently the Vice Chair of IEEE TC-PLC, and was the General Chair of IEEE ISPLC'18, U.K., and the Co-Chair of the 6th IEEE Int'l Conference on Smart Grid Communications, Miami, U.S., in 2015. He is a Panel Member of the U.K. Engineering and Physical Sciences Research Council Peer Review College, and an EU H2020 Expert Reviewer/Rapporteur. He has been part of multipartner, multicountry, multimillion pounds projects as PI and Co-I. He is a Fellow of IET, and Higher Education Academy, and a Chartered Engineer.

Guan Gui (Senior Member, IEEE) was born in Zongyang county, Anhui, China, in 1982. He received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2012.

From 2009 to 2014, he joined the Tohoku University, Sendai, Japan, as a Research Assistant as well as a Postdoctoral Research Fellow. From 2014 to 2015, he was an Assistant Professor with Akita Prefectural University, Akita, Japan. Since 2015, he has been a Professor with Nanjing University of Posts and Telecommunications, Nanjing, China. He has published more than 200 IEEE journal/conference papers and won several best paper awards, e.g., ICC 2017, ICC 2014, and VTC 2014-Spring. His recent research interests include intelligence sensing and recognition, intelligent signal processing, and physical layer security.

Dr. Gui received the IEEE Communications Society Heinrich Hertz Award in 2021, the Top 2% Scientists of the World by Stanford University in 2021, the Clarivate Analytics Highly Cited Researcher in Cross-Field in 2021, the Highly Cited Chinese Researchers by Elsevier in 2020 and 2021, the Member and Global Activities Contributions Award in 2018, the Top Editor Award of *IEEE Transactions on Vehicular Technology* in 2019, the Outstanding Journal Service Award of *KSII Transactions on Internet and Information System* in 2020, the Exemplary Reviewer Award of IEEE COMMUNICATIONS LETTERS in 2017, the 2012 Japan Society for Promotion of Science (JSPS) Postdoctoral Fellowships for Foreign Researchers, and the 2018 JSPS International Fellowships for Overseas Researchers. He was also selected as the Jiangsu Specially Appointed Professor in 2016, the Jiangsu High-level Innovation and Entrepreneurial Talent in 2016, and the Jiangsu Six Top Talent in 2018. Since 2022, he has been a Distinguished Lecturer of the IEEE Vehicular Technology Society. He is a member of the IEEE Communications Society and the IEEE Vehicular Technology Society. He is serving or served on the editorial boards of several journals, including IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, *IEICE Transactions on Communications*, *Physical Communication*, *Wireless Networks*, IEEE ACCESS, *Journal of Circuits Systems and Computers*, *Security and Communication Networks*, *IEICE Communications Express*, *KSII Transactions on Internet and Information Systems*, and *Journal on Communications*. In addition, he served as the IEEE VTS Ad Hoc Committee Member in AI Wireless, the TPC Chair of PRAI 2022, ICGIP 2022, PHM 2021, and WiMob 2020, the Executive Chair of VTC 2021-Fall, the Vice Chair of WCNC 2021, the Symposium Chair of WCSP 2021, the General Co-Chair of Mobimedia 2020, the Track Chair of EuCNC 2021 and 2022 and VTC 2020 Spring, the Award Chair of PIMRC 2019, and the TPC Member of many IEEE international conferences, including GLOBECOM, ICC, WCNC, PIRMC, VTC, and SPAWC.