Edge Device Identification Based on Federated Learning and Network Traffic Feature Engineering

Zhimin He, *Graduate Student Member, IEEE*, Jie Yin, Yu Wang, *Graduate Student Member, IEEE*, Guan Gui, *Senior Member, IEEE*, Bamidele Adebisi, *Senior Member, IEEE*, Tomoaki Ohtsuki, *Senior Member, IEEE*, Haris Gacanin, *Fellow, IEEE*, and Hikmet Sari, *Life Fellow, IEEE*

***Abstract*—With the ubiquitous deployment and applications of internet of things (IoT), security issues pose a critical challenge to IoT devices. External attackers often utilize vulnerable IoT devices to invade the target's internal network and then further cause a security threat to the whole network. To prevent such attacks, it is necessary to develop a security mechanism to control the access of suspicious IoT devices and manage the internal devices. In recent years, deep learning (DL) algorithm has been widely used in the field of edge device identification (EDI), and has made great achievements. However, these previous methods are essentially centralized learning-based EDI (CentEDI) that trains all data together, which can not guarantee data security and not conducive to deployment on edge devices. To address this problem, we introduce a federated learning-based EDI (FedeEDI) method via network traffic to automatically identify edge devices connected to the whole network. Experimental results show that the training efficiency of our proposed FedeEDI method is much higher than that of the CentEDI method, although its classification accuracy is slightly reduced. In contrast to the CentEDI method, the proposed FedeEDI method has two main advantages: faster training speed and safer training process.**

***Index Terms*—Internet of things, edge device identification, network traffic, centralized learning, federated learning, deep learning.**

Manuscript received February 23, 2021; revised May 13, 2021; accepted July 24, 2021.

This work was supported in part by the JSPS KAKENHI under Grant JP19H02142, Major Project of the Ministry of Industry and Information Technology of China under Grant TC190A3WZ-2, National Natural Science Foundation of China under Grant 61901228, the Summit of the Six Top Talents Program of Jiangsu under Grant XYDXX-010, the Program for High-Level Entrepreneurial and Innovative Team under Grant CZ002SC19001, the project of the Key Laboratory of Universal Wireless Communications (BUPT) of Ministry of Education of China under Grant KFKT-2020106, the Open Project Program of the State Key Lab of CAD&CG under Grant A2102, Zhejiang University. *(Corresponding authors: Jie Yin, Guan Gui).*

Zhimin He, Yu Wang, Guan Gui, Hikmet Sari are with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (Emails: 1019010407@njupt.edu.cn, 1018010407@njupt.edu.cn, guiguan@njupt.edu.cn, hsari@ieee.org).

Jie Yin is with the Department of Computer Information and Cyber Security, Jiangsu Police Institute, Nanjing 210031, China. He is also with the Jiangsu Province Electronic Data Forensics and Analysis Engineering Research Center, Key Laboratory of Digital Forensics of Jiangsu Provincial Public Security Department, amd State Key Lab. for Novel Software Technology, Nanjing University 210023, Nanjing, China. (Email: yinjie@jspi.cn).

Bamidele Adebisi is with the Department of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M15 6BH, United Kingdom (Email: b.adebisi@mmu.ac.uk).

Tomoaki Ohtsuki is with the Department of Information and Computer Science, Keio University, Yokohama 223-8521, Japan (Email: ohtsuki@ics.keio.ac.jp).

Haris Gacanin is with the Faculty of Electrical Engineering and Information Technology, RWTH Aachen University, Aachen 52062, Germany (Email: harisg@ice.rwth-aachen.de).

I. Introduction

Internet of things (IoT) is regarded as the third revolution of the world information industry. It combines various information sensing devices with the Internet to form a network for the purpose of connecting and exchanging data with other devices [1]–[5]. It is forecast that the IoT devices will grow exponentially in line with market demand, and the number will reach 75 billion by 2025 [6]. An example of edge device identification (EDI) in IoT is shown in Fig. 1, the deployment and application of a large number of IoT devices has greatly facilitated our life, but the heterogeneity in services, technologies, devices and protocols (such as wireless, wired, satellite, cellular and bluetooth, etc.) makes the management of IoT devices more complex [7], [8]. Since most of IoT devices have inherent security flaws, they are more vulnerable to external attacks, which lead to security threats to the target network [9]–[11].

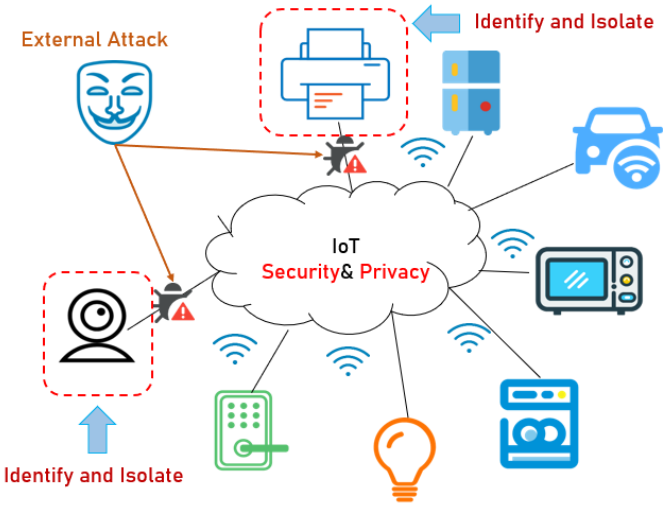


Fig. 1. An example of edge device identification in IoT. We need to identify the suspicious devices connected to networks and isolate for communication to protect the security of the networks

Owing to the openness of the wireless network itself, hackers can gain unauthorized access to internal network by using spoofing identities, thus interfering or destroying the normal communication of legitimate users, which seriously threatens the security of wireless network [12]. There have been many IoT devices security incidents around the world. The One that caught people's attention was in 2006, hackers

hijacked these IoT devices through a malicious software called Mirai, and used these devices to launch large-scale distributed denial of service (DDoS) attacks on domain name system (DNS) servers, which eventually led to network paralysis [13].

For network administrators, identifying the various IoT devices connected to the network is a challenge. According to the survey, internal employees tend to bring a large number of their own IoT devices into the workplace, and 25%∼50% of employees indicated that one of these IoT devices has been connected to the enterprise network [14]. These devices, which are connected to the internal network of the enterprize through employees, become vulnerable to attacks. To address these challenges, we need to identify the suspicious devices connected to networks and isolate for communication to protect the security of the networks.

In the face of these kinds of complex attacks, the internal network of various organizations should reconsider whether to allow these devices to easily connect to the network, and consider how to manage the devices connected to the internal network. To achieve this goal, we first identify the IoT devices connected to the network by feature extraction capability of deep learning. We consider statistical attributes obtained from the network traffic as features and focal loss is applied for class imbalance in dataset. Considering that the centralized learning based EDI (CentEDI) method cannot protect the data privacy, while having the high demand for computing resources. To address these challenges, we propose federated learning (FL) based EDI (FedeEDI) method to detect different IoT. In this paper, our main contributions are summarized in three aspects:

- We design a device identification method based on network traffic features to achieve the purpose of classifying different devices connected to the network with accuracy of 88.2%, which shows comparable performance to the CentEDI method.
- The proposed FedeEDI method is faster in speed and safer in training under the condition of no significant decrease in classification accuracy. In contrast to CentEDI, the training time of FedeEDI is shortened by about 65% without considering communication delay and other conditions.
- Our proposed FedeEDI can combine multiple devices with weak computing power. Compared with CentEDI which generally needs powerful computing devices for training, FedeEDI is suitable for edge devices.

II. Related Work

A. Network Traffic Classification

In recent years, with the rapid development of the Internet, more and more new network applications are emerging, and the network composition is becoming more and more complex. Network traffic can record and reflect network activities and operating conditions. As one of the basic technologies to enhance network controllability, it can not only help network operators to provide better quality of service (QoS), but also can effectively supervise and manage the network to ensure network security. At present, the research on network traffic classification mainly includes four categories: the port-based technique, the payload based technique, the machine learning (ML) based technique, and the DL based technique.

The traditional flow classification method relies on the analysis of the port number in TCP or UDP packets, and uses the non-mandatory port number recommended by the Internet Assigned Number Authority (IANA) to distinguish different network traffic application types. In 2004, S. Sen *et al*. [15] conducted experiments on P2P protocol, the default P2P port number only accounts for about 30% of the total test, this method is simple and fast, and is suitable for real-time flow classification on high-speed networks. The realization of this method is simple and the classification speed is fast, which is suitable for real-time stream classification on high-speed networks. However, it does have some limitations such as that some applications may not register port numbers with IANA or use well-known ports.

To avoid over dependence on the port number, a payload based analysis method is proposed. The payload-based method, also known as deep packet detection, analyzes the signature signature of the packet and tries to match the signature stored in the database. By analyzing whether the payload of the packet contains the special signature of the known application, the method has a high accuracy. W. Li *et al*. [16] devised a classification method that relies on the full packet payload, and their experimental classification accuracy is close to 100%. But with the new application and the continuous emergence of load encryption, the method gradually fails. Although the accuracy of [16] is high, it is more complex, requires higher computing costs, and can not identify encrypted traffic packets. There are many methods [17], [18] to identify encrypted traffic, Okada [19] proposed a classification method EFM based on feature estimation on the basis of analyzing the feature changes caused by traffic encryption. According to the relationship between unencrypted and encrypted traffic, strong correlation features were selected. The experimental results show that the classification accuracy of this method can reach 97.2%.

The classification method based on ML can reduce the amount of computation and identify the encrypted traffic easily. This method uses statistical information of network traffic (arrival time of continuous packets, average packet length, etc.) to classify different traffic. In [20], the authors proposed an ML model to realize online traffic classification using flow level features, they achieve an overall accuracy of 97.92% for classifying eight major applications by means of C4.5 decision tree.

The ML based method often requires manual selection of traffic features that requires expert experience, and whether the feature extraction is reasonable or not will directly affect the final classification effect. The DL based method can use the hidden layer structure of deep neural network to extract features automatically. The authors in [21] applied deep bidirectional long short-term memory to analyse the long-term inter-related changes in the low-dimensional feature set, and achieve good generalization ability in binary and multi-class classification scenarios.

B. Device Identification

Device identification refers to identify IoT devices connected to the network for the purpose of network management and security protection. The traditional device identification method based on MAC address may be invalid because skilled attackers can forge the MAC address of the vulnerable IoT devices [22]. In addition, although the MAC address can be used to identify the manufacturer of a particular device, it cannot identify the brand or type of device. The current works mainly focus on network traffic which has been widely used in a variety of security applications in the past.

In recent years, many machine learning algorithms are widely used in device identification, including support vector machine (SVM), random forest and k-nearest neighbor (KNN), *et al.* [23], [24]. Y. Meidan *et al.* [25] proposed a supervised machine learning based method for detecting unauthorized IoT devices. The features used in the work were extracted from network traffic data collected from 17 distinct IoT devices. With the majority rule and 20-session sequences, the classifier can successfully detect IoT device types that are not on the white list in 90% of test cases, and white listed device types in 99% of cases. Authors in [26] adopt network traffic analytics to identify IoT devices, and their random forest classifier reaches a high accuracy of 95% in classifying 20 IoT devices.

With the rise of deep learning, more and more researchers adopt deep learning to improve identification accuracy. A. Aksoy *et al.* [27] introduced a system for automatically classifying IoT devices based on network traffic. Genetic algorithm (GA) is used to extract features of different protocol header files, and then various machine learning algorithms are deployed to classify different device types. This method allows fully automated classification of devices using TCP/IP packets without expert input and the accuracy rate of identifying device type from a single packet is more than 95%. S. Aneja *et al.* [28] proposed an inter arrival time (IAT) method between two consecutive packets to study device fingerprinting (DFP). Due to the different hardware and software used by the device, IAT is unique to the device. They proposed a new idea of DFP, that is, by drawing the IAT diagram of the data packet, each diagram draws 100 IATs, and then processing the resulting diagram to identify the device. The authors use convolutional neural network (CNN) to identify the device and the experimental accuracy reaches 86.7%. In [29], the authors proposed a novel identification method for edge devices using activation maximization based filter-level pruning technique that omits the less important convolutional filter. In [30], the authors developed an important framework to transform complex-valued signal waveforms into images with statistical significance, which can transfer deep level statistical information from the raw wireless signal waveforms while being represented in an image data format.

C. FL and Its Applications

FL is a machine learning solution for training data decentralization, which aims to get a high-quality centralized machine learning model by training and learning data distributed over a large number of edge devices. FL allows the data to be kept in the local clients and only the model parameters are shared with the central server, which can reduce the communication overhead and protect privacy of each client's raw data [31].

Federal learning is considered as an effective means of privacy protection, which has been widely used in various fields recently. For instance, Qi *et al.* [32] proposed a consortium blockchain-based federated learning framework to update the model from distributed vehicles and then store on the blockchain, the experimental results show that it can effectively prevent data poisoning attacks and improve privacy protection. In [33], the authors proposed a federated-learning traffic classification protocol which can classify the new application instantly when participants join the learning of a new application, and the accuracy of packet classification can reach 88% under non-independent and identically distributed (non-IID) traffic across clients. Yi *et al.* [34] introduced a secure FL framework based on blockchain where the central aggregation identifies malicious and unreliable participants by automatically executing smart contracts to prevent poisoning attacks, thereby improving the security of FL in 5G networks. Y. Liu *et al.* [35] performed a research on applying FL to make accurate traffic predictions while preserving privacy of raw data. In the process the authors propose an FL-based gated recurrent unit (GRU) neural network algorithm that integrates emerging FL with a practical GRU neural network for traffic prediction. In addition, they design a joint-announcement protocol in the aggregation mechanism which uses random subsampling for participating organizations to reduce the communication overhead of the algorithm. In addition, FL was also adopted for both image classification and language modeling task [36]–[38], and achieved good performance.

III. EDI Problem Description and Dataset

A. Problem Description

EDI refers to the identification of IoT devices connected to the network for the purpose of network management and security. Due to the lack of vulnerability update mechanism, IoT devices are often accompanied by huge security risks. These devices are vulnerable to external attacks thus cause information leakage, which will threaten the security of the network they connected. Network managers can employ EDI to identify suspicious IoT devices connected to the network, and then isolate them to avoid communication with the gateway, so as to protect the security of the network.

In this paper, the method we consider to adopt is to first extract the network traffic features of the IoT devices, and then use the obtained features to establish a device classification model. DL model can automatically extract features and classify, which is exactly what we need. The DL model based on the EDI method can be described as

$$\hat{y} = \arg\max_{y \in Y} \mathcal{F}_{DL}(y \mid X) \tag{1}$$

where X is the input and $\hat{y}$ is the predicted label. The input consists of the network traffic features of different devices, Y represents all device types, and y is one of them. $\mathcal{F}_{DL}$ is the mapping function between the input X and the output $\hat{y}$, which can be used to train the DL model to achieve convergence.

B. Network Traffic Features and Dataset

Network traffic can help operators to provide suitable network environment for different applications and provide better service quality for users. At present, the classification of network traffic is widely studied and many methods are used. However, it is mainly based on the following several different granularity or levels.

1) Bit level: It mainly focuses on the data features of network traffic, such as the transmission rate and the changes of throughput.
2) Packet level [39]: It mainly focuses on the features of packet and its arrival process, such as packet size, distribution, arrival time interval, packet loss rate, etc.
3) Flow level: Flow is a relatively loose definition that is divided primarily on the basis of address and application protocol. The definition given by C.Barakat et al. [40] is a five-tuple consisting of the source IP address and port, destination IP address and port, and the application protocol. Studies in this field mainly focus on the arrival process and interval of Flow as well as its local features.
4) Stream level: [41] gives the definition of stream as a triplet composed of source IP address, destination IP address and application protocol. The main purpose is to study the long-term traffic statistics features of backbone network at a coarser granularity.

The dataset used in our research is from [42], and it consists of a large number of traffic data collected by Wireshark from 10 different IoT devices: baby monitor, lights, motion sensor, security camera, smoke detector, socket, thermostat, TV, watch, and water sensor (see Table I) and recorded as *.pcap files. The final *.pcap files are input into the feature extractor together with the publicly available external database such as Alexa Rank [43] and GeoIP [44] to obtain a Comma-Separated Values (CSV) file containing feature vectors such as hypertext transfer protocol (HTTP), secure sockets layer (SSL), etc. The dataset contains 297 features that might model their behavior, such as HTTP protocol request mode, the total number, maximum value and median value of cookie, the size and arrival time of packet, the content, type of SSL protocol and so on. There are approximately 400,000 instances in the training set, 1,000 instances in the validation set and 1,000 instances in the testing set. Table II presents several instances of features extracted from different protocols and network traffic layers [45].

Each feature in the dataset (see Table II) represents a connection session (a TCP connection consisting of source and destination IP addresses and port numbers, from the SYN packet to the FIN packet). For each cumulative feature, we calculate the following statistics as additional features: minimum, first quartile, median, third quartile, maximum, mean, standard deviation, variance, and entropy [45]. At last, the obtained features are divided beforehand into three different datasets: training dataset, validation dataset, and testing dataset. Table I shows their numbers and categories respectively. It is worth mentioning that there are some instances with missing data in the dataset, and the approach we take is to delete the instances with missing data. In addition, we notice that different features have different range of values, which might lead to less accurate results and problems in training. Therefore, we decide to use the MinMaxScaler [46] normalization algorithm of sklearn library in Python to convert the every value in the dataset to the same range $(0, 1)$.

TABLE I
VARIOUS IOT DEVICES ARE INCLUDED IN THE DATASET [42].

Device type	Training dataset	Validation dataset	Testing dataset
Security camera	14,320	100	100
TV	59,144	100	100
Smoke detector	168	100	100
Thermostat	18,914	100	100
Water sensor	1,026	100	100
Watch	4,187	100	100
Baby monitor	51,577	100	100
Motion sensor	50,912	100	100
Lights	100,000	100	100
Socket	100,000	100	100

TABLE II
A FEATURES EXAMPLE OF DIFFERENT PROTOCOL.

Protocol	Features
HTTP	Hostname Cookie User Agent Response Request
SSL	Server name Client name Handshake duration
TCP	Packet size Desination port Response
DNS	Time-to-live Response flags Transaction ID

IV. THE PROPOSED FEDEEDI METHOD

In this paper, we propose a FedeEDI method based on network traffic. Considering the powerful feature extraction capability of deep learning, CNN and LSTM are adopted to classify different IoT devices. In view of that the traditional deep learning method adopts data centralization for training, which cannot guarantee the privacy of data and has a high demand for hardware devices. Our proposed FedeEDI method can effectively avoid this problem and improve communication efficiency, which is easy to deploy on hardware with weak computing power. The system model is illustrated in Fig. 2.

A. Loss Functions for Deep Learning Models

1) Cross Entropy Loss Function: Cross-entropy (CE) is mainly used to measure the actual output and the predictive output, which can solve the problem of slow update of the weight of the quadratic cost function. Given a training set consisting of N_B sample-label pairs $\{(x_i, y_i)\}_{i=1}^{N_B}$, the number

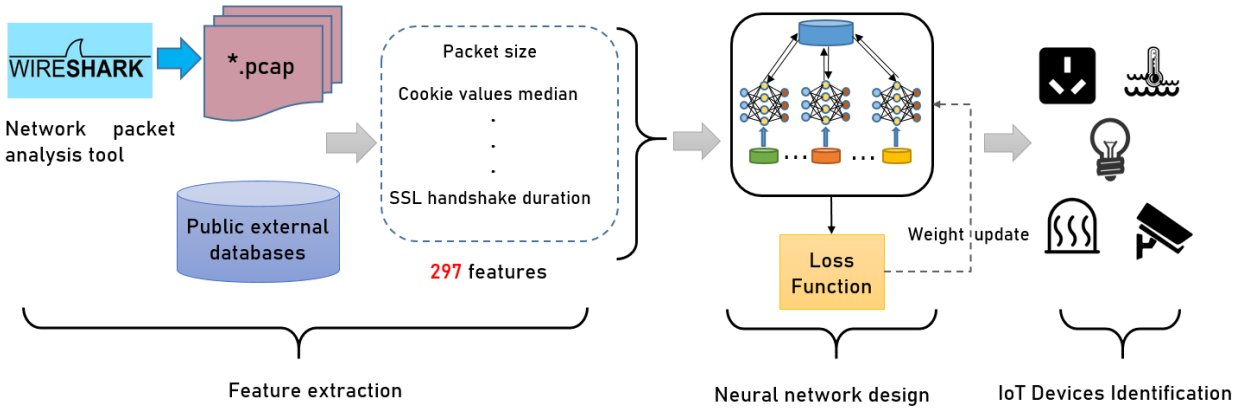


Fig. 2. System model.

of different classes in the training set is M, then the CE loss function can be defined as

$$\begin{aligned} L_{CE} &= -\frac{1}{N_B}\sum_{i=1}^{N_B} y_i \log\left(f\left(\theta; x_i\right)\right) \\ &= -\frac{1}{N_B}\sum_{m\in M}\sum_{i=1}^{N_B} y_i^m \log\left(f\left(\theta; x_i\right)\right). \end{aligned} \tag{2}$$

where θ represents the model weight (or model parameters), f is the mapping function.

2) Focal Loss Function: It is quite common to encounter a dataset of class imbalance which might cause the model to perform poorly. To deal with it, the concept of focal loss [47] is introduced. Focal loss essentially a cross-entropy loss that weights the contribution of each sample to the loss based in the classification error. If a sample is already classified correctly by the model, then its contribution to the loss decreases, which makes the loss implicitly focus on those problematic classes. The definition of focal loss function is given as

$$L_{\text{Focal}} = -\frac{1}{N_B}\sum_{m\in M}\alpha^m \sum_{i=1}^{N_B}\left(y_i^m - f\left(\theta; x_i^m\right)\right)^{\gamma} \times \log\left(f\left(\theta; x_i\right)\right) \tag{3}$$

where α is used to balance between positive and negative samples, γ is a focusing parameter, and $\left(y_i^m - f(\theta; x_i)\right)^{\gamma}$ is modulating factor which reduces the loss contribution from easy examples and extends the range in which an example receives low loss [47].

B. Model Architecture

Compared with the traditional neural network, CNN is not fully connected between layers, thus the number of parameters is greatly reduced through parameter sharing, which can suppress overfitting to a certain extent. In our proposed CNN model, we have two convolutional layers and four fully connected layers. Except for the activation function of the last fully connected layer is softmax, the activation function of the rest layers all use the ReLU function, which can effectively avoid the problem of gradient vanishing. There are 64 convolution kernels (1×8) in the first convolution layer and 128 convolution kernels (1×4) in the second convolution kernel. The number of neurons in the remaining four fully connected layers is 256, 128, 64, and 10, respectively. Since the dropout layer can reduce feature redundancy by randomly removing neurons, which weakens the dependence among the neurons and improves the generalization ability of the network. Therefore, we use the Dropout layer to remove neurons from the network with a probability of 50%, and the specific structure of CNN is shown in Table III.

As a variant of recurrent neural network (RNN), LSTM can solve the problem of RNN's performance decline over time, and it can alleviate the gradient vanishing problem to a certain extent. In this paper, the proposed model consists of two LSTM layers and four fully connected layers. The LSTM layer has 256 and 128 memory units, respectively. The structure and activation function of the fully connected layer are the same as those of the CNN. We also use the dropout layer to prevent overfitting in LSTM model. The structure is shown in Table IV. In our proposed model, we use the Adam optimizer to update the weights, along with categorical cross-entropy as the loss and accuracy as the evaluation metric. We also applied focal loss to improve the classification accuracy of small samples. In all our experiments, we set the learning rate $\eta = 0.001$, the batch size $B = 128$, and the total training epochs $E = 100$.

C. The Proposed FedeEDI Method

Considering that CNN and LSTM train all the data together in one machine, they are not suitable for deployment in the edge equipment with weak computing power. To solve this problem and protect data security, we propose a device identification method based on FL. The main advantages of our proposed method are as follows:

- The data is stored in the local client, and the server cannot obtain the client's data, thus protecting the clients privacy.
- Distributed data architecture reduces the consumption of data centralized storage in the server.
- Training with multiple edge devices at the same time is faster than centralized training.

The FL system is generally composed of clients which store data locally and central server. The amount of local data of

TABLE III
NETWORK STRUCTURE OF CNN.

Layers	Activation function	Output dimensions
Input	/	1×297×1
Convolution (64 filters, 1×8)	ReLU	1×290× 64
Dropout	/	/
Convolution (128 filters, 1×4)	ReLU	1×287×128
Dropout	/	/
Flatten	/	18368
Dense	ReLU	256
Dropout	/	/
Dense	ReLU	128
Dropout	/	
Dense	ReLU	64
Dropout	/	/
Dense	Softmax	10

TABLE IV
NETWORK STRUCTURE OF LSTM.

Layers	Activation function	Output dimensions
Input	/	1×297
LSTM	ReLU	1×256
Dropout	/	/
LSTM	ReLU	1×128
Dense	ReLU	256
Dropout	/	/
Dense	ReLU	128
Dropout	/	
Dense	ReLU	64
Dropout	/	/
Dense	Softmax	10

each client may not be sufficient to support one successful model training, so the data of other client is needed. It collects the gradients from the clients, and returns the new gradient after aggregation operation in the server. In a collaborative process of FL, the clients only train the data locally to protect data privacy. The gradient obtained by iteration is used as interactive information instead of local data and uploaded to the trusted server of the third party. After the server returns the aggregated parameters, the model is updated [48]. The FL process of client-server architecture is demonstrated in Fig. 3.

As shown in **Algorithm 1**, the complete training phase of FL model consists of the following three steps.

1) **Parameter Initialization:** all clients acquire a pre-assigned neural network model in their devices, and the central server sends initialization parameters to each client for local training.
2) **Local Training:** in a particular epoch of communication, each clients first download global model parameters from the central server, then the client updates its model parameters with local data and send these updated parameters to the central server.
3) **Model Average:** The central server aggregates each client's model parameters from different training data through model average.
4) **Model Update:** The central server sends the aggregated model weights, i.e., the global model to each client, and each client applies the global weight to replace the original local model weight for training, and then repeats the steps (2)∼(4) until the loss convergence.

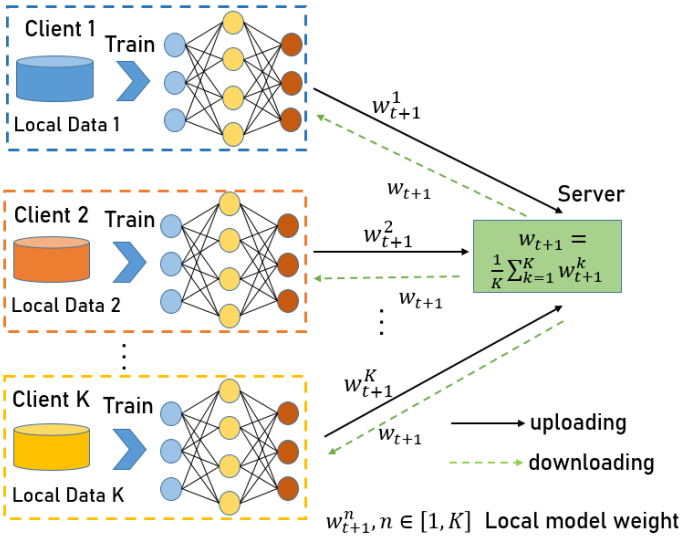


Fig. 3. FedeEDI for client-server architecture.

Algorithm 1: Our proposed FedeEDI method.

Input: Clients $\mathcal{K} = \{K_1, K_2, \ldots, K_N\}$. local mini-batch size: B, number of local epochs: T, learning rate: η,gradient optimization function: $\nabla\ell(\cdot;\cdot)$.
Output: Parameter ω.

Initialize pre-trained parameter ω^0;
for *epoch* $t = 1, 2, \cdots$ **do**
  Server executes: Broadcast initial global model ω^0 to client in $\{K_i\}$;
  foreach *client* $k \in \{K_i\}$ ***in parallel*** **do**
    Initialize $\omega_t^k = \omega^o$;
    $\omega_{t+1}^o \leftarrow \text{ClientUpdate}\left(k, \omega_t^k\right)$
  end
  $w_{t+1} \leftarrow \frac{1}{K}\sum_{k=1}^{K} w_{t+1}^k$;
end
ClientUpdate$\left(k, \omega_t^k\right)$
foreach *local epoch* $t = 1, 2, \cdots, T$ **do**
  if *batch* $b \in B$ **then**
    Compute the current gradient
    $w_{t+1}^k \leftarrow w_t^k - \eta * \nabla\ell_{k,b}\left(w_t^k\right)$;
  end
end

V. Simulation Results And Discussions

To evaluate our model on the testing dataset, we use accuracy (A), precision (P), recall (R), and the F_1-score (F1) as our evaluation criteria which are defined as

$$A = \frac{TP + TN}{TP + TN + FP + FN}, \tag{4}$$

$$P = \frac{TP}{TP + FP}, \tag{5}$$

$$R = \frac{TP}{TP + FN}, \tag{6}$$

$$F_1 = 2 \times \frac{P \times R}{P + R}. \tag{7}$$

where the TP (true positive) is the number of positive samples correctly predicted to positive samples, TN (true negative) is that of negative samples correctly predicted to negative samples, FP (false positive) is that of negative samples incorrectly predicted to positive samples, FN (false negative) is that of positive samples incorrectly predicted to negative samples. In addition, we also analyzed the complexity of the model with parameters to measure space complexity, and floating point operations (FLOPS) to measure time complexity.

A. Experimental Results based on CentEDI

In this part, we discuss two CentEDI models, one based on CNN (CentCNN) and the other based on LSTM (CentLSTM). The optimizer used in our CentEDI method is Adam, and we trained our models for 100 epochs, with batch size of 128 (Table V) and learning rate of 0.001 (Table VI). Due to the existence of class imbalance in the dataset, the focal loss function is adopted to improve the classification accuracy of small classes, and the cross-entropy is used as comparison.

TABLE V
Results of different batch size.

Batch Size	Method	Accuracy	Precision	Recall	F_1-score
64	CNN	0.728	0.711	0.728	0.700
	LSTM	0.724	0.677	0.724	0.691
128	CNN	**0.786**	**0.813**	**0.786**	**0.787**
	LSTM	**0.748**	**0.696**	**0.748**	**0.716**
256	CNN	0.734	0.701	0.734	0.700
	LSTM	0.724	0.680	0.724	0.684

TABLE VI
Results of different learning rate.

Learning Rate	Method	Accuracy	Precision	Recall	F_1-score
0.0005	CNN	0.756	0.703	0.756	0.723
	LSTM	0.656	0.626	0.656	0.613
0.001	CNN	**0.786**	**0.813**	**0.786**	**0.787**
	LSTM	**0.748**	**0.696**	**0.748**	**0.716**
0.005	CNN	0.642	0.568	0.642	0.583
	LSTM	0.521	0.364	0.521	0.419

The experimental results of our proposed model are shown in Table VII. It can be seen that the classification accuracy of the CentCNN and CentLSTM with the cross-entory (CE) function reaches 78.6% and 74.8%, respectively. After using the focal loss (Focal) as the loss function, the accuracy of CentCNN (88.1%) and CentLSTM (88.2%) both further improved by 9.5% and 13.4%, respectively. In addition, Fig. 4 shows that the application of focal loss can effectively accelerate the training both in CNN and LSTM.

TABLE VII
Results based on CentEDI.

Method	Accuracy	Precision	Recall	F_1-score
CNN (CE)	0.786	0.813	0.786	0.787
LSTM (CE)	0.748	0.696	0.748	0.716
CNN (Focal)	0.881	0.833	0.881	0.848
LSTM (Focal)	0.882	0.931	0.882	0.853

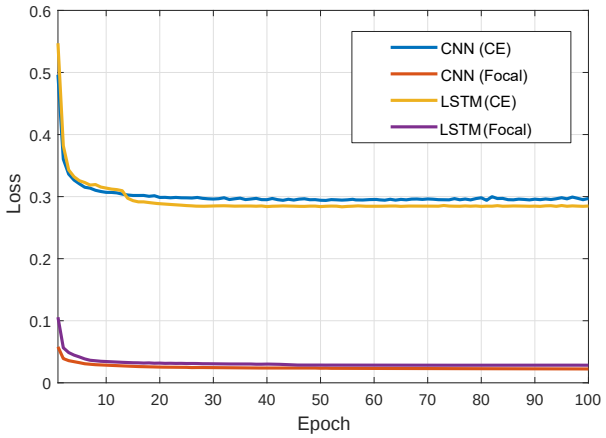
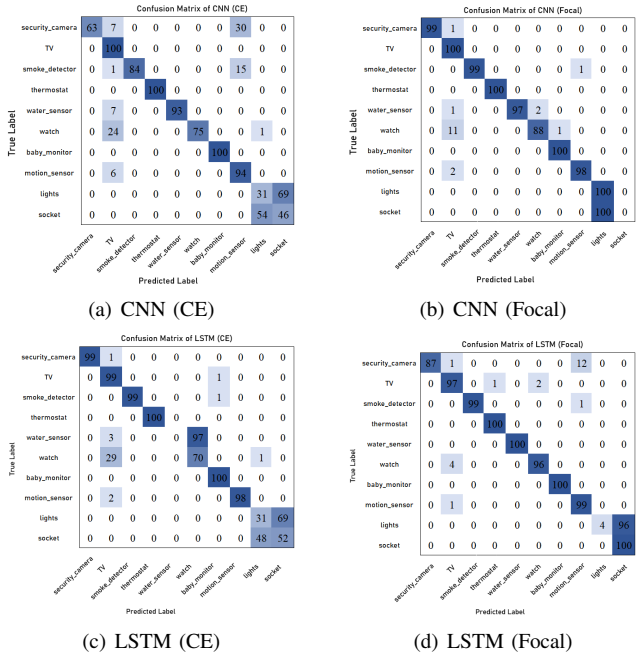


Fig. 4. Training losses of CentEDI.

From the confusion matrix, we can observe that CNN can correctly classify TV, thermostat, water sensor, baby monitor, and motion sensor with over 90% accuracy, however, it cannot identify light and socket accurately, even if their training samples are large. After using the focal loss as the loss function, the classification accuracy of smoke detector is improved by 15% and the lights can also be identified with 100% accuracy. However, it completely classified the socket as light. Similar to CNN, LSTM cannot classify water sensor, watch, lights and socket well. The application of focal loss can significantly improve the classification accuracy of small devices (such as lights and watch), but it still wrongly identifies lights as socket (see in Fig. 5).

B. Experimental Results based on FedeEDI

The previous models put all the data together for training, as a result, it cannot protect the data privacy, and the training speed is rather slow. On the other hand, this centralized training method has high requirements for hardware equipment, which is not suitable for deployment on edge devices with



(a) CNN (CE) (b) CNN (Focal)

(c) LSTM (CE) (d) LSTM (Focal)

Fig. 5. Confusion matrix of CentEDI.

weak computing power. To address the above problems, we proposed an edge device identification method based on FL i.e., FedeEDI. In this part, we discuss the effect of different number of local models (clients) and aggregation epoch on the performance of FedeEDI. To make a comparison with the CentEDI, the neural network structure used by the FedeEDI in client and server is the same as that of the previous model. According to the previous experiments, focal loss can effectively improve the classification accuracy of small samples caused by class imbalance, thus we only consider the case of focal loss as the loss function in this part. Moreover, we consider two FL algorithms, one is based on CNN (FedeCNN), the other is based on LSTM (FedeLSTM).

1) Classification Performance vs. Clients: We divide the training set and test set into several independent subsets as the local data of each client. To test the effect of different number of clients on the experimental results, the test set should be kept unchanged. The performances of FedeCNN and FedeLSTM with different numbers of clients are shown in Table VIII and TableIX, respectively.

TABLE VIII
RESULTS BASED ON FEDECNN MODEL.

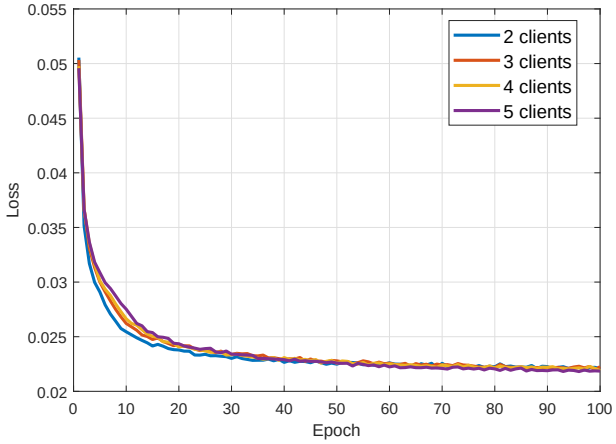
Number of clients	Accuracy	Precision	Recall	F_1-score
2 clients	0.869	0.896	0.869	0.843
3 clients	0.872	0.825	0.872	0.839
4 clients	0.846	0.853	0.846	0.846
5 clients	0.859	0.860	0.859	0.851

It can be concluded from the experimental results that the performance of the FedeCNN model is generally better than that of FedeLSTM model. From the simulation results we can observe that the highest accuracy of FedeCNN (0.872) and FedeLSTM (0.882) are reached when there are 3 clients

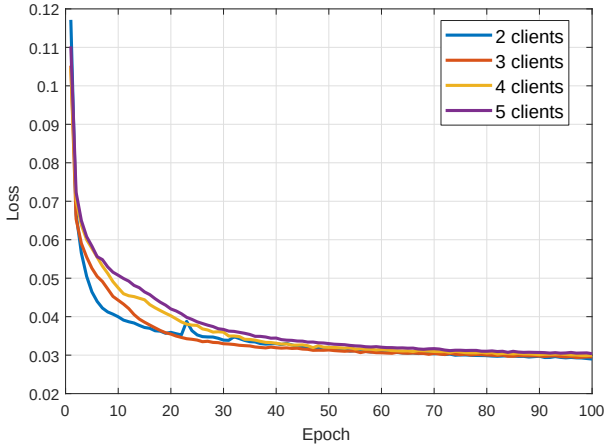
TABLE IX
RESULTS BASED ON FEDELSTM MODEL.

Number of clients	Accuracy	Precision	Recall	F_1-score
2 clients	0.845	0.843	0.845	0.832
3 clients	0.882	0.832	0.882	0.849
4 clients	0.785	0.688	0.785	0.721
5 clients	0.778	0.690	0.778	0.716

for local training. In the FedeCNN model, the number of clients has little effect on the final classification result, and the accuracy reduced by 2.6% in the worst case (4 clients). In contrast, the accuracy of FedeLSTM model is more easily influenced by the number of clients. In the case of using 5 clients for local training, the classification accuracy decreases by 10.4%. The training loss of FedeCNN and FedeLSTM with different number of clients is shown in Fig. 6. It can be seen that although the number of clients is different, they all converge at almost the same epoch, regardless of in FedeCNN or FedeLSTM.



(a) Training loss of FedeCNN with different number of clients.



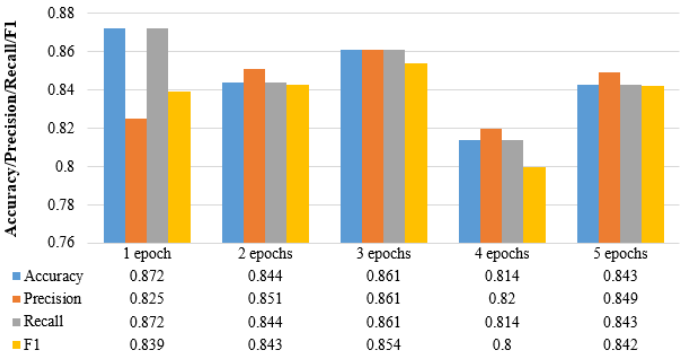
(b) Training loss of FedeLSTM with different number of clients.

Fig. 6. Training loss of FedeEDI with different number of clients.

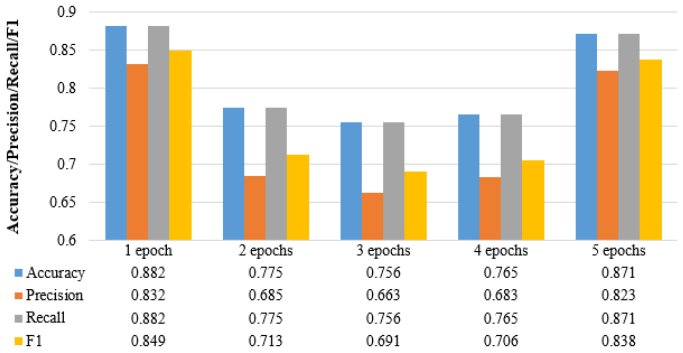
2) Classification Performance vs. Aggregation Epoch: In addition to the number of clients, we also consider the impact

of aggregation epoch on the experimental results. From the experimental results in the previous section, we can see that the scenarios of 3 clients for local training have the best performance both in FedeCNN and FedeLSTM. Therefore, our experiment in this section is based on this scenario of 3 clients.

In this section, we average these local model weights of each client in the aggregation center (server) every α epochs ($\alpha = \{1, 2, 3, 4, 5\}$). It can be concluded from Fig. 7 that FedeCNN had the best performance with classification accuracy of 0.872 under the condition of model average every 1 epoch. The change of aggregation epoch has little influence on the performance of FedeCNN, the classification accuracy also reached 0.814 at the worst performance (aggregation epoch is 4), which decreased by 5.8% compared with the best performance. FedeLSTM achieved the best performance with classification accuracy of 0.882 when aggregation epoch is 1 and the worst performance with classification accuracy of 0.756 when it is 3. It can be observed in the FedeLSTM model that despite the time span of aggregation is relatively large (5 epochs), the performance is quite good with accuracy of 0.871. Combined with the above, it seems that aggregation epoch has no decisive effect on the performance of FedeEDI as far as our experimental results are concerned.



(a) Performance evaluation of FedeCNN v.s. aggregation epoch.



(b) Performance evaluation of FedeLSTM v.s. aggregation epoch.

Fig. 7. Accuracy, precision, recall, F_1-score of FedeEDI with different aggregation epoch.

3) Classification Performance of Our Proposed Model: According to the previous experimental results, in our proposed model, we divided the dataset into three independent subsets for local training of three clients, respectively, and the weight of the model obtained by local training was averaged once every 1 epoch in the client center (server). The experimental results based on FL are shown in the following Table X.

The loss function used of all the experiments in Table X is focal loss. LocalCNN and LocalLSTM respectively represent the experimental results of three local models (clients) in the FL model, and FedeCNN and FedeLSTM are represent the experimental results of global models (server) in the FL model. Since the training data of FL is only a part of the original dataset, the performance of FL model is slightly worse than that of CentEDI. It can be seen from Table X that the accuracy of the FedeCNN method has only decreased by 0.9%, while the accuracy of the FedeLSTM has not changed. This further indicates that the FedeEDI has little impact on the performance of the original model. We also calculated the training time of completing one epoch of the CentEDI and the FedeEDI, and the results are shown in Table XI.

Fig. 8 shows the training loss curves of FedeEDI in three clients scenario and Fig. 9 shows the comparison of the training loss curves of the two models (CentEDI and FedeEDI). The losses of three local models (clients) are similar whether in FedeCNN or FedeLSTM. It is noted that although loss of FedeCNN and FedeLSTM have difference before convergence, they converge almost at the same epoch. Compared to CentEDI (CentCNN and CentLSTM), FedeEDI (FedeCNN and FedeLSTM) can effectively accelerate the training. It can be concluded from Table XI that the training speed of FedeEDI is faster than that of CentEDI without significant performance deterioration, and the training time is shortened by about 65% without considering communication delay and other conditions.

TABLE XI
TIME COST IN EACH TRAINING EPOCH.

	CentEDI	**FedeEDI**
CNN	46.2 s/epoch	16.0 s/epoch
LSTM	66.5 s/epoch	17.7 s/epoch

VI. CONCLUSION

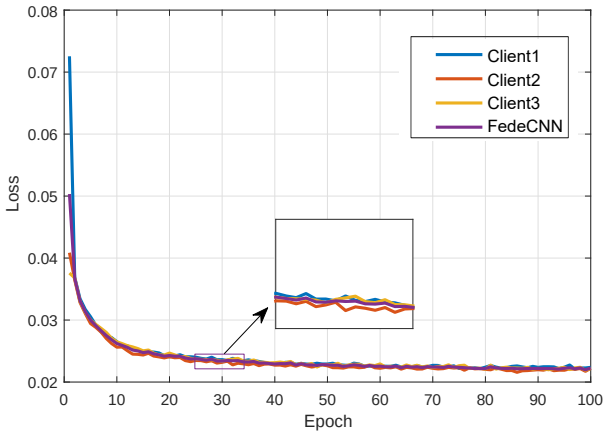
In this paper, we propose a FedeEDI method via network traffic, which has three clients for local data training and adopts model average as the aggregate method for weight update. Compared with the CentEDI method, our proposed FedeEDI method can improve the communication efficiency with limited classification performance loss. The results of our experiments show that the proposed FedeEDI method can accelerate convergence and reduce the training time needed for one epoch. In contrast to CentEDI that generally needs powerful computing devices for training, the FedeEDI can combine multiple devices with weak computing power which is more suitable for edge devices.

REFERENCES

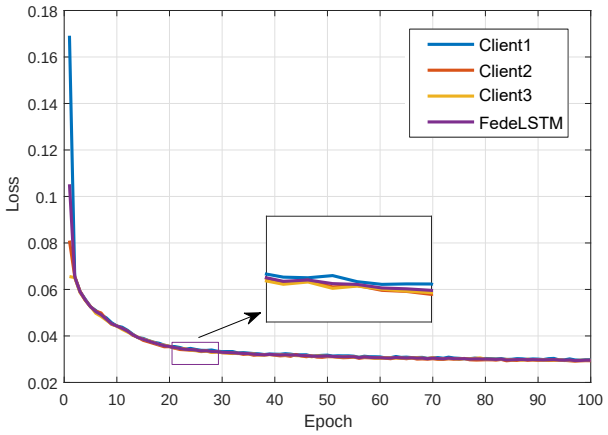
[1] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang and W. Zhao, "A survey on internet of things: architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

TABLE X
RESULTS BASED ON FEDEEDI

Method	Accuracy	Precision	Recall	F_1-score	Parameters	FLOPS
CentCNN	0.881	0.833	0.881	0.848	4,778,255	9,555,213
CentLSTM	0.882	0.931	0.882	0.853	839,242	1,019,157
Clients (CNN)	0.728/0.783/0.780	0.777/0.781/0.817/	0.782/0.783/0.780	0.718/0.721/0.720	4,778,255	9,555,213
Clients (LSTM)	0.870/0.791/0.772	0.821/0.806/0.860	0.870/0.791/0.772	0.837/0.776/0.723	839,242	1,019,157
FedeCNN	**0.872**	**0.825**	**0.872**	**0.839**	**4,778,255**	**9,555,213**
FedeLSTM	**0.882**	**0.832**	**0.882**	**0.849**	**839,242**	**1,019,157**



(a) Training loss curves of FedeCNN.



(b) Training loss curves of FedeLSTM.

Fig. 8. Training loss curves of FedeEDI.

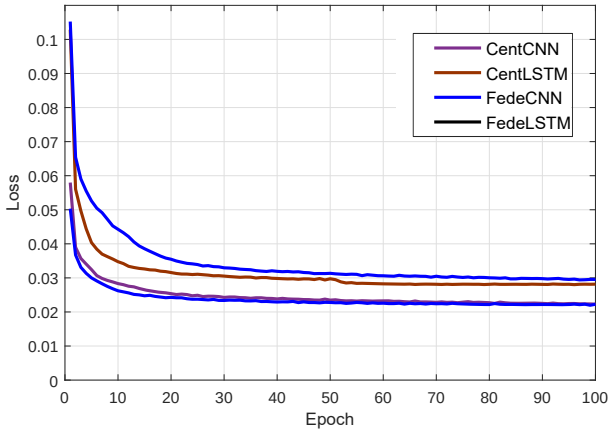


Fig. 9. Comparisons of different training loss curves.

[2] M. J. Farooq and Q. Zhu, "A multi-layer feedback system approach to resilient connectivity of remotely deployed mobile Internet of Things," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 422–432, June 2018,

[3] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Mumtaz, A. Jolfaei, S. H. Ahmed, and A. K. Bashir, "Learning-based context-aware resource allocation for edge computing-empowered industrial IoT," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4260–4277, May 2020.

[4] I. Bisio, C. Garibotto, F. Lavagetto, and A. Sciarrone, "When eHealth meets IoT: A smart wireless system for post-stroke home rehabilitation," *IEEE Wireless Communications*, vol. 26, no. 6, pp. 24–29, Dec. 2019.

[5] K. Zhang, Y. Zhu, S. Maharjan, and Y. Zhang, "Edge intelligence and blockchain empowered 5G beyond for industrial internet of things," *IEEE Network*, vol. 33, no. 5, pp.12–19, Sept./Oct. 2019.

[6] Statista Research Department, "Internet of things (IoT) connected devices installed base worldwide from 2015 to 2025," [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, Nov. 27, 2016.

[7] K. Sood, S. Yu and Y. Xiang, "Software-defined wireless networking opportunities and challenges for internet-of-things: a review," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 453–463, Aug. 2016.

[8] Y. Zhao, Y. Yin, and G. Gui, "Lightweight deep learning based intelligent edge surveillance techniques," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1146–1154, Apr. 2020.

[9] W. Zhang, B. Zhang, Y.Zhou, H. He, and Z. Ding, "An IoT honeynet based on multiport honeypots for capturing IoT attacks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 3991–3999, May 2020.

[10] Y. Jia, F. Zhong, A. Alrawais, B. Gong, and X. Cheng, "FlowGuard: an intelligent edge defense mechanism against IoT DDoS attacks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9552–9562, May 2020.

[11] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use AI to enhance security?" *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 41C-49, Sept. 2020.

[12] I. Andrea, C. Chrysostomou and G. Hadjichristofi, "Internet of things: Security vulnerabilities and challenges," in *IEEE Symposium on Computers and Communication (ISCC)*, pp. 180–187, 2015.

[13] C. Kolias, G. Kambourakis, A. Stavrou and J. Voas, "DDoS in the IoT: mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, Jul. 2017.

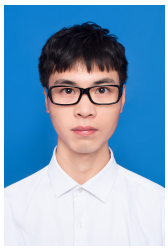
[14] "3 ways the internet of things will impact enterprise security," [Online]. Available: https://www.calero.com/mobility-service-support/3-ways-the-internet-of-things-will-impact-enterprise-security/, March 26, 2015,

[15] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of P2P traffic using application signatures," in *Proceedings of the 13th International Conference on World Wide Web*, pp. 512–521, 2004.

[16] W. Li, M. Canini, A. W. Moore, and R. Bolla, "Efficient application identification and the temporal and spatial stability of classification scheme," *Computer Networks*, vol. 53, no. 6, pp. 790–809, Apr. 2009.

[17] M. Korczyski and A. Duda, "Classifying service flows in the encrypt-

ed skype traffic," *IEEE International Conference on Communications (ICC)*, pp. 1064–1068, 2012.
[18] G. Xiong, W. Huang, Y. Zhao, M. Song, Z. Li, L. Guo, "Real-Time detection of encrypted thunder traffic based on trustworthy behavior association," in *International Conference on Trustworthy Computing and Services (ISCTCS)*, pp. 132–139, 2012.
[19] Y. Okada, S. Ata, N. Nakamura, Y. Nakahira and I. Oka, "Comparisons of machine learning algorithms for application identification of encrypted traffic," in *International Conference on Machine Learning and Applications and Workshops*, pp. 358–361, 2011.
[20] D. Tong, Y. R. Qu and V. K. Prasanna, "Accelerating decision tree based traffic classification on FPGA and multicore platforms," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3046–3059, Nov. 2017.
[21] 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 of Things Journal*, vol. 8, no. 6, pp. 4944–4956, Mar. 2021.
[22] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, " Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (MobiCom)*, pp. 116–127, 2008
[23] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "Audi: Toward autonomous IoT device-type identification using periodic communication," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1402–1412, Jun. 2019.
[24] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Managing ToT cyber-security using programmable telemetry and machine learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 60–74, Mar. 2020.
[25] Y. Meidan, M. Bohadana, A. Shabtai, *et al*., "Detection of unauthorized IoT devices using machine learning techniques," [Online]. Available: https://arxiv.org/abs/1709.04647.
[26] A. Sivanathan, D. Sherratt, H. H. Gharakheili, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Characterizing and classifying iot traffic in smart cities and campuses," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2017, pp. 559–564.
[27] A. Aksoy and M. H. Gunes, "Automated IoT device identification using network traffic," in *IEEE International Conference on Communications (ICC)*, pp. 1–7, 2019.
[28] S. Aneja, N. Aneja, and M. S. Islam,"IoT device fingerprint using deep learning," in *IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*, pp. 174–179, 2018.
[29] Y. Lin, Y. Tu, and Z. Dou, "An improved neural network pruning technology for automatic modulation classification in edge device," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5703–5706, May 2020.
[30] Y. Lin, Y. Tu, Z. Dou, L. Chen, and S. Mao, "Contour stella image and deep learning for signal recognition in the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 1, pp. 34–46, Mar. 2021.
[31] D. Conway-Jones, T. Tuor, S. Wang, and K. K. Leung, "Demonstration of federated learning in a resource-constrained networked environment," *IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 484–486, 2019.
[32] Y. Qi, M. S. Hossain, J. Nie, X. Li, "Privacy-preserving blockchain-based federated learning for traffic flow prediction," *Future Generation Computer Systems*, vol. 117, pp.328–337, Apr. 2021.
[33] H. Mun, Y. Lee, "Internet traffic classification with federated learning," *Electronics*, vol. 10, no. 1, pp.27–27, Dec. 2020.
[34] Y. Liu, J. Peng, J. Kang, A. M. Iliyasu, D. Niyato and A. A. A. El-Latif, "A secure federated learning framework for 5G networks," *IEEE Wireless Communications Magazine*, vol. 27, no. 4, pp. 24–31, Aug. 2020.
[35] Y. Liu, J. J. Q. Yu, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: a federated learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7751–7763, Aug. 2020.
[36] H. B. Mcmahan, E. Moore, D. Ramage, and B. A. Arcas, "Federated learning of deep networks using model averaging," [Online]. Available: https://uk.arxiv.org/abs/1602.05629v1.
[37] T. Chilimbi, Y. Suzue,J. Apacible,K. Kalyanaraman,"Project ADAM: building an efficient and scalable deep learning training system," in *The 11th USENIX Symposium on Operating Systems Design and Implementation*, pp. 571–582, 2014.
[38] X. Zhu, J. Wang, Z. Hong, T. Xia and J. Xiao, "Federated learning of unsegmented chinese text recognition model," in *IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 1341–1345, 2019.
[39] C. Fraleigh, *et al*., "Packet-level traffic measurements from the sprint IP backbone," *IEEE Network*, vol. 17, no. 6, pp. 6–16, Dec. 2003.
[40] C. Barakat, P. Thiran, G. Iannaccone, C. Diot and P. Owezarski, "Modeling internet backbone traffic at the flow level," *IEEE Transactions on Signal Processing,* vol. 51, no. 8, pp. 2111–2124, Aug. 2003.
[41] T. He, H. Zhang, and Z. Li, "A methodology for analyzing backbone network traffic at stream-level," in *International Conference on Communication Technology Proceedings*, Beijing, China, 2003, pp. 98–102, 2003.
[42] I. Mosseri, and D. Eckert, "IoT device identification using machine learning techniques," in *IEEE International Conference On Trust, Security And Privacy In Computing And Communications*, Rotorua, New Zealand, Aug. 5–8, 2019.
[43] "Alexa rank," [Online]. Available: http://www.alexa.com/topsites.
[44] "Geoip lookup service," [Online]. Available: http://geoip.com/.
[45] D. Bekerman, B. Shapira, L. Rokach and A. Bar, "Unknown malware detection using network traffic classification," in *IEEE Conference on Communications and Network Security (CNS)*, pp. 134–142, 2015.
[46] F. Pedregosa, G. Varoquaux, *et al*., "Scikit-learn: machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, Oct. 2011.
[47] T. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, Feb. 2020.
[48] J. Konen, H. B. Mcmahan, D. Ramage, P. Richtarik, "Federated optimization: distributed machine learning for on-device intelligence," [Online]. Available: https://arxiv.org/abs/1610.02527.



Zhimin He (Graduate Student Member, IEEE) received his B.S. degree in Communication Engineering from Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China in 2019. He is currently pursuing the master's degree in communication and information engineering with the Nanjing University of Posts and Telecommunications, Nanjing, China. His research interest includes machine learning for wireless communications.

Jie Yin received his M.S. degree in Software Engeering from Nanjing University of Science and Technology, Nanjing, China in 2008. He is currently a Senior Engineering with the Department of Computer Information and Cyber Security, Jiangsu Police Institute, Nanjing, China. He is also with the Jiangsu Province Electronic Data Forensics and Analysis Engineering Research Center, Key Laboratory of Digital Forensics of Jiangsu Provincial Public Security Department, amd State Key Lab. for Novel Software Technology, Nanjing University, Nanjing, China. He has published more than 10 international journal/conference papers. His recent research interests include machine learning, big data, and network security.

Yu Wang (Graduate Student Member, IEEE) received his B.S. degree in Communication Engineering from Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China in 2018. He is currently working toward the Ph.D. degree in NJUPT. Mr. Wang has published more than 20 IEEE Journal/Conference papers. He received several best paper awards, i.e., ICEICT 2019, CSPS 2019, CSPS 2018. His research interests include deep learning, optimization, and its application in wireless communications.

Guan Gui (Senior Member, IEEE) 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 as a research assistant as well as a postdoctoral research fellow, respectively. From 2014 to 2015, he was an Assistant Professor in the Akita Prefectural University. Since 2015, he has been a professor with Nanjing University of Posts and Telecommunications, Nanjing, China. Dr. Gui 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. He received the IEEE Communications Society Heinrich Hertz Award in 2021, the Elsevier Highly Cited Chinese Researchers in 2020, the Member and Global Activities Contributions Award in 2018, the Top Editor Award of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY in 2019. He served as Executive Chair of IEEE VTC 2021-Fall and Vice Chair of IEEE WCNC 2021.

Bamidele Adebisi (Senior Member, IEEE) received his Bachelor's degree in electrical engineering from Ahmadu Bello University Zaria, Nigeria, in 1999, and his Masters degree in advanced mobile communication engineering and Ph.D. degree in communication systems from Lancaster University, United Kingdom, 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 Manchester Metropolitan University in 2012, where he is currently a Full Professor (Chair) of intelligent infrastructure systems. He is the current Vice Chair of IEEE TC-PLC; was General Chair, IEEE ISPLC'18, UK, Co-Chair, 6th IEEE Int'l Conference on Smart Grid Communications, 2015, Miami, US, etc. He is a Panel Member of the UK Engineering and Physical Sciences Research Council (EPSRC) Peer Review College, and an EU H2020 Expert Reviewer/ rapporteur. He has been part of multi-partner, multi-country, multi-million pounds projects as PI and Co-I. One of his projects with an SME received the 2020 UK Best Knowledge Transfer Partnership Project of the Year Awards. 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. B amidele is a Fellow of IET, a Fellow of Higher Education Academy and a Chartered Engineer.

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 Post Doctoral Fellow and a Visiting Researcher in Electrical Engineering at 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. In 2005 he joined Keio University. He is now a Professor at Keio University. He served as a Chair of IEEE Communications Society, Signal Processing for Communications and Electronics Technical Committee. He served as a technical editor of the IEEE Wireless Communications Magazine and an editor of Elsevier Physical Communications. He is now 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 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, IEEE WCNC, and so on. He was Vice President and President of the Communications Society of the IEICE. He is a senior member and a distinguished lecturer of the IEEE, a fellow of the IEICE, and a member of the Engineering Academy of Japan.

Haris Gacanin (Fellow, IEEE) received his Dipl.-Ing. degree in Electrical engineering from the University of Sarajevo in 2000. In 2005 and 2008, respectively, he received MSc and Ph.D. from Tohoku University in Japan. He was with Tohoku University from 2008 until 2010 first as Japan Society for the Promotion of Science (JSPS) postdoctoral fellow and later, as an Assistant Professor. He joined Alcatel-Lucent Bell (now Nokia Bell) in 2010 as a Physical-layer Expert and later moved to Nokia Bell Labs as Department Head. Since April 2020, he joined RWTH Aachen University. He is a head of the Chair for Distributed Signal Processing and co-director of the Institute for Communication Technologies and Embedded Systems. His professional interests are related to broad areas of digital signal processing and artificial intelligence with applications in wireless communications. He has 200+ scientific publications (journals, conferences and patents) and invited/tutorial talks. He is a fellow of IEEE. He was a Distinguished Lecturer of IEEE Vehicular Technology Society and an Associate Editor of IEEE COMMUNICATIONS MAGAZINE, while he served as the editor of IEICE Transactions on Communications and IET Communications. He acted as a general chair and technical program committee member of various IEEE conferences. He is a recipient of several Nokia innovation awards, IEICE Communications Society Best Paper Award in 2021, IEICE Communication System Study Group Best Paper Award (joint 2014, 2015, 2017), The 2013 Alcatel-Lucent Award of Excellence, the 2012 KDDI Foundation Research Award, the 2009 KDDI Foundation Research Grant Award, the 2008 JSPS Postdoctoral Fellowships for Foreign Researchers, the 2005 Active Research Award in Radio Communications, 2005 Vehicular Technology Conference (VTC 2005-Fall) Student Paper Award from IEEE VTS Japan Chapter and the 2004 Institute of IEICE Society Young Researcher Award.

Hikmet Sari (F'95-LF'20) received the engineering diploma and the Ph.D. degree from ENST, Paris, France, and the Habilitation degree from the University of Paris XI. From 1980 to 2002, he held various research and management positions at Philips Research Laboratories, SAT, Alcatel, Pacific Broadband Communications, and Juniper Networks. From 2003 to 2016, he was a Professor and the Head of the Telecommunications Department, Supelec, and a Chief Scientist at Sequans Communications. He is currently a Professor with the Nanjing University of Posts and Telecommunications (NJUPT). Dr. Sari's distinctions include the Andre Blondel Medal in 1995, the Edwin H. Armstrong Achievement Award in 2003, the Harold Sobol Award in 2012, as well as election to the Academia Europaea (Academy of Europe) and the Science Academy of Turkey in 2012. He was the Chair of the Communication Theory Symposium of ICC 2002, a Technical Program Chair of ICC 2004, a Vice General Chair of ICC 2006, a General Chair of PIMRC 2010, a General Chair of WCNC 2012, an Executive Chair of WCNC 2014, a General Chair of ICUWB 2014, a General Co-Chair of IEEE BlackSeaCom 2015, a Technical Program Chair of EuCNC 2015, an Executive Co-Chair of ICC 2016, a General Co-Chair of ATC 2016, an Executive Chair of ICC 2017, a General Co-Chair of ATC 2018, and a General Co-Chair of PIMRC 2019. He also chaired the Globecom and ICC Technical Content (GITC) Committee from 2010 to 2011, and was the Communications Society (ComSoc) Vice President for Conferences from 2014 to 2015, the Director for Conference Operations of ComSoc from 2018 to 2019, and the Vice President for Conferences of the IEEE France Section from 2017 to 2019. He is currently serving as a General Chair of WCNC 2021 and an Executive Co-Chair of GLOBECOM 2023. He served as an Editor for the IEEE Transactions on Communications from 1987 to 1991, an Associate Editor for the IEEE Communications Letters from 1999 to 2002, and a Guest Editor for several special issues of the IEEE Journal on Selected Areas in Communications, European Transactions on Telecommunications (ETT), and other journals. He served as a Distinguished Lecturer of ComSoc from 2001 to 2006, a member of the IEEE Fellow Evaluation Committee from 2002 to 2007, and a member of the IEEE Awards Committee from 2005 to 2007.