https://doi.org/10.1590/jatm.v17.1376 ORIGINAL PAPER

A Multi-Modal Traffic Classification-Based Device Identification Method

Yuanyuan Ma[1], Yunfan Wang[1,2,*], Zesheng Xi[1,3], Chuan He[1,2]

1. China Electric Power Research Institute – State Grid Laboratory of Power Cyber-Security Protection and Monitoring Technology – Nanjing – China.
2. Southeast University – School of Cyber Science and Engineering – Nanjing – China.
3. Nanjing University of Science and Technology – School of Computer Science and Engineering – Nanjing – China.

***Correspondence author:** lanai_pylon.05@icloud.com

ABSTRACT

Internet of things (IoT) devices are widely used in various fields, with their growing diversity and complexity posing challenges for traditional security measures. Device fingerprint identification can enhance network security and reliability by verifying device features. However, traditional device fingerprint identification methods usually rely on a single mode of traffic characteristics. In the face of changing network environments and diversified device types, it is often difficult to ensure efficient identification performance and robustness. To address the above challenges, this paper proposes a multi-modal traffic classification method for device identification in IoT networks to address the challenges in accuracy and robustness posed by traditional single-modal traffic feature approaches. The method combines various traffic features, such as packet size, transmission interval, flow duration, packet rate, byte rate, and protocol number. It includes four modules: data collection, preprocessing, model training, and fingerprint identification. Network traffic data are collected using deep packet inspection and capture tools, and features are standardized. The *bidirectional encoder representations from transformers* (BERT) model is applied for sensitive text feature extraction, while the convolutional neural network (CNN) model aids in device identification. Experimental results demonstrate high accuracy and robustness across different network environments and device types.

Keywords: Multi-modal traffic classification; Device fingerprinting; Feature Identification; Network security.

INTRODUCTION

Today, due to the rapid expansion of internet of things (IoT) devices, both the number of connected devices and the number of nodes in the network have increased significantly, making device management and data monitoring more difficult. Additionally, cybersecurity threats are increasingly diverse, with attackers using camouflage devices, virtualization technologies, or other sophisticated means to launch attacks, making it difficult to fully cover existing security measures. By collecting and analyzing device characteristics and generating unique identifiers, device fingerprint identification technology can accurately identify and verify devices, effectively improving the network's ability to perceive abnormal behaviors, and providing reliable security for complex network environments. In this context, device fingerprint recognition technology has been widely used.

Existing device fingerprint recognition technology is mainly divided into two categories: hardware fingerprint recognition technology and software fingerprint recognition technology. Hardware fingerprint identification depends on the physical characteristics of the device, such as the unique hard disk serial number, MAC address, and CPU ID. In contrast, software fingerprinting focuses on the characteristics of the software environment of the device, such as the operating system version, browser type, time zone settings, etc. This software-level information can be obtained through network requests or system scanning.

Received: Oct. 30, 2024 | **Accepted:** Mar. 06, 2025
Peer Review History: Single Blind Peer Review.
Section editor: Paulo Renato Silva

However, there are two challenges to this approach:

• Single mode: existing device fingerprint recognition technology relies on a single recognition mode, either through hardware features or software features to identify the device. When the device hardware is replaced or the software is updated, the original fingerprint information may change. As a result, the device cannot be effectively identified. Additionally, attackers can circumvent identification by tampering with software information or using forged hardware features. Therefore, existing device fingerprint recognition technology is difficult to cope with device configuration changes or malicious camouflage.

• Accuracy: existing methods usually rely on hardware, software, or behavioral characteristics, and these characteristics may be affected by environmental changes or differences in user behavior. The user's operating habits and behavior patterns may also change, leading to misjudgments in fingerprint recognition. Furthermore, software fingerprints can be easily modified or camouflaged, and attackers can bypass the identification by changing the operating system or browser configuration. Therefore, faced with the ever-changing network environment and complex attack modes, the existing methods often cannot maintain sufficient accuracy and stability.

In view of the shortcomings of existing fingerprint identification methods, this paper proposes a device fingerprint identification method based on multi-modal network traffic characteristics. The method first collects device feature data from multiple dimensions through multi-modal fusion, including network traffic, hardware information, software configuration, and user behavior. *Bidirectional encoder representations from transformers* (BERT)'s bidirectional coding capabilities are then used to capture contextual dependencies in data streams, extract global timing patterns, and identify device characteristics over long connections or complex interactions. Subsequently, a convolutional neural network (CNN) is used to extract local features from the data and identify subtle change patterns in the traffic through the convolutional layer, such as packet size and transmission rate fluctuations. Finally, by combining the global and local features extracted by BERT and CNN, the fingerprint information of the device is generated through a unified classification model.

This paper's main contributions:

• A multi-modal traffic feature combination is constructed by combining data packet size, flow duration, protocol number, etc., to realize multi-dimensional identification of device fingerprints and enhance the model's adaptability to complex network environments.

• By combining the strengths of BERT and CNN, this approach simultaneously captures global timing information and local device features, yielding more comprehensive and accurate recognition results. BERT's advanced context modeling complements CNN's local feature extraction, enhancing device fingerprinting with improved accuracy and robustness. This method addresses classification challenges arising from similar device behaviors, optimizing both real-time performance and classification precision.

• This paper proposes a device fingerprint recognition method based on multi-modal network traffic characteristics. This method fully utilizes of the complementarity of multi-modal information and can integrate features from different sources. By combining the context modeling capability of BERT and the local feature extraction advantage of CNN, it can accurately capture the complex timing relationships and dependencies between devices in the industrial internet and extract small and detailed changes from network traffic. Thus, high precision and high robustness of device identification can be achieved.

Related works

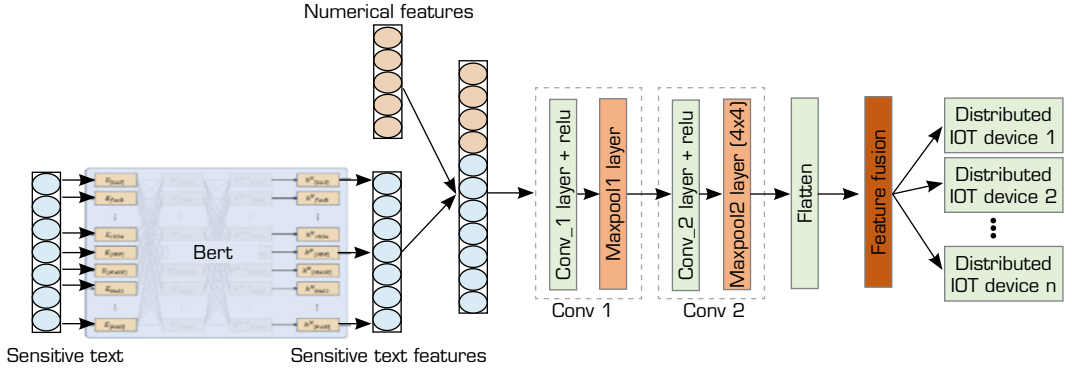
Liang *et al.* (2022) proposed a classification and identification method for encrypted traffic based on a self-attention hybrid pooling CNN (AHPCNN). In this method, by improving the pooling layer of the CNN, the average pooling layer and the maximum pooling layer are combined in parallel to form a two-layer synchronous pooling model. Self-attention module is embedded to enhance the model's dependence on encrypted traffic characteristics, thus capturing both the overall and local characteristics of network encrypted traffic, so as to classify encrypted traffic more accurately. According to the experimental results, the model improves the accuracy of identifying encrypted traffic and the F1 score. Liu *et al.* (2020) proposed a method of fusing byte-level encoding with an improved pre-training task for the application of the pre-training model BERT in encrypted traffic classification. This method improves the model's ability to model the semantic diversity and coherent order of encrypted traffic through the design of a novel vocabulary and a new self-supervised pre-training task. Jian *et al.* (2020) proposed a multidimensional RF fingerprint extraction method based on deep learning. By expanding the observed dimension of signal features and combining multiple difference intervals, the recognition rate of

RF fingerprints is improved, and the accuracy of device recognition is significantly enhanced. Li *et al.* (2024) proposed a flow-based adversarial learning network intrusion detection method, which leverages the advantages of Generative Adversarial Networks (GANs) to capture normal network traffic patterns through adversarial training, thereby enhancing detection capabilities against unknown attacks. Chen *et al.* (2016) explored the traffic identification and separation techniques for network infrastructure devices, studying the threat analysis and detection methods for the traffic of these devices based on the intelligent algorithm models. Li *et al.* (2023) proposed a real-time recognition scheme for IoT devices based on feature vector splitting, which improves recognition accuracy and reduces the time and storage occupation of training models. The above research provides effective methods and theoretical support for encrypted stream classification, device identification, and threat detection in the field of network security.

In comparison to the aforementioned studies, this paper presents several innovative and optimized aspects that distinguish it from the current literature. Firstly, it innovatively combines numerical features with sensitive textual features, leveraging the BERT model to process the latter and fusing multi-modal features into a unified feature vector. This approach ensures a more comprehensive capture of device behavioral characteristics, thereby enhancing the accuracy and robustness of device identification. Secondly, the paper introduces a novel device recognition methodology that comprehensively harnesses multiple traffic features. By pre-training the BERT model to extract sensitive text features and dynamically weighting and fusing them with numerical features, this method significantly boosts the model's capability to discern devices. This dynamic fusion strategy not only leverages the strengths of both numerical and textual modalities but also optimizes their combined representation for improved recognition performance. Furthermore, the utilization of the BERT model for extracting sensitive text features and integrating them with traffic numerical features through weighted fusion results in a more comprehensive feature representation. This enriched representation, when coupled with a CNN model tailored for multi-modal features, leverages the CNN's robust feature extraction capabilities to further elevate the accuracy and robustness of device recognition. The CNN's ability to effectively process and analyze the fused multi-modal features ensures that intricate patterns within the data are captured and utilized for accurate device identification. Lastly, the combination of dynamic weighted fusion, the BERT pre-training model, and the real-time feature extraction and recognition capabilities of the CNN model presents a solution that guarantees both high efficiency and accuracy in real-time device recognition. This approach optimizes the training process and overall performance of the model, making it a suitable candidate for practical applications requiring rapid and reliable device identification. In summary, this paper contributes significantly to the field by introducing a comprehensive, multi-modal, and dynamically adaptive approach to device recognition.

METHODOLOGY

Figure 1 shows the device identification process based on multi-modal traffic classification. Specifically, first, network communication flow information between IoT devices is collected using relevant techniques and tools. Then, the collected initial data are cleaned, and a suitable standardization and splicing method is selected for stream integration of data traffic as well as text



Source: Elaborated by the authors.

Figure 1. Device identification process via multi-modal traffic classification.

stream information. Then, multi-modal feature data are then extracted from the integrated stream, making full use of numerical features and sensitive text features to improve the accuracy of device fingerprinting, The BERT model is used to efficiently learn the semantics and contextual relationships of the text and output them as a feature vector, which is then fused with multi-modal features to form a unified feature vector. After that, a CNN model is designed for the specification of the feature vectors, and supervised training is carried out. The processed and fused data are used for model training to ensure that it meets the requirements of device recognition based on multi-modal traffic classification. Finally, for the trained model, real-time collected network traffic is used as input, and the recognition results are returned to each IoT device terminal for device identification and verification.

Data acquisition and processing

The data collection part uses Wireshark and Tcpdump to collect network communication traffic information of all network traffic data between IoT devices. Specifically, the collected raw data contain the following main features: stream duration, stream packet rate, stream interval, stream byte rate, download/upload ratio, protocol number, User-Agent/JA3 hash, DHCP, and DNS text data.

Data cleansing

For raw data, the number and length of data packets with numerical features and text features differ to some extent. First, the data packets are divided respectively. After partitioning, redundant data and noise data are cleared by comparing the quintuples of data packets (source IP address, destination IP address, source port, destination port, and protocol type). If identical records are found, one of them is kept and delete the rest of the duplicate records are deleted.

If the value of abnormal data, such as stream duration, stream packet rate, and other characteristics, exceeds the reasonable range, the standard deviation method is used to identify abnormal data and eliminate this portion of the data. For missing data records, features whose values are out of the reasonable range are directly deleted. The standard deviation method is applied for anomalous data identification to reject this part of the data. Records with missing data are rejected directly.

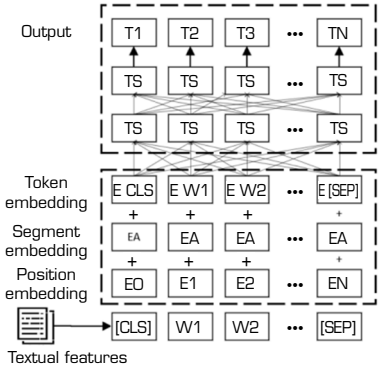
Data format normalization

The packet timestamp format generated by the communication between different devices may differ. This paper chooses ISO 8601 format as the conversion benchmark and converts all timestamps to this format uniformly.

Data from different sources also have certain differences in data types, and in this paper, the floating-point type is chosen as the benchmark for processing all data for conversion. For text data, the first five fields (source IP, destination IP, source port, destination port, protocol type) in each packet are selected for splicing and stream integration.

Text feature extraction based on the BERT model

To effectively capture the semantic and contextual information in the text, this paper adopts the BERT pre-trained language model to encode the sensitive textual data, which is then converted into high-dimensional feature vectors. The specific process is shown in Fig. 2.



Source: Elaborated by the authors.

Figure 2. Text feature extraction process.

Text conversion and pre-processing

The input sensitive textual data is disambiguated and encoded using the BERT disambiguator and converted into the input format required by the model. First, each textual data is disambiguated and converted into a series of lexical identifiers. These identifiers are then encoded to adapt them to the input requirements of the BERT model. To ensure the consistency and validity of the input data, the text data are truncated and padded to a fixed length. For example, for each input text, its length is truncated to 512 vocabulary units and padded where necessary to ensure that all input texts have the same length.

BERT model structure

The transformer architecture consists of an encoder-decoder structure, in which the encoder is primarily composed of a self-concerned layer, and a feedforward neural network, and their respective normalized layers. These components are connected by a residual structure to ensure that the input information is transmitted completely to the next layer. The self-attention layer is the core of the encoder. The input embedding vector is multiplied with a randomly generated matrix of dimension (64, 512) in the self-attention layer to obtain three new vectors: Q-query, K-key, and V-value, which are then used in the attention computation:

$$Attention(Q,K,V) = Soft\max(\frac{QK^T}{\sqrt{d_K}}) \times V \tag{1}$$

where d_K is the dimension of the K-vector and Softmax is the normalized exponential function:

$$ó(Z)_I = \frac{e^{Z_i}}{\acute{O}_{j=1}^{K} e^{Z_i}} \tag{2}$$

To extract information about different spatial semantics, BERT uses a multi-head attention mechanism that combines attention values computed using multiple initialization matrices W multiplied by the input embedding, and then performs another linear change to obtain the final output.

$$MultiHead(Q,K,V) = Concat(head_1, head_2, head_3, ...) \times W \tag{3}$$

where $head_1$ is obtained from Eq. 3 is a randomly generated initialization matrix.

BERT model inputs

The input to the BERT model includes label embedding, segment embedding, and position embedding. Token embedding represents word embedding, where each sensitive text data is preceded by a CLS identifier as the start identifier and ended by a SEP token as the end identifier. Segment embedding represents the segment identifier; when the model performs a relationship prediction task, it merges two text data into the model for training, and this vector differentiates between the two data. Position embedding represents temporal information for each word.

Model training

The three embedding vectors (word embedding, segment embedding, and position embedding) obtained in the previous step are fed into a bidirectional multi-layer transformer encoder for processing. The corresponding feature word vectors are output using the transformer-encoder's multi-head attention mechanism. At this point, the first vector of the output layer is the word vector corresponding to the CLS flag bit, which can be directly input into a multi-layer perceptron for classification, i.e., the sensitive text feature vector to be acquired. To ensure that the two different types of features can be effectively fused, feature merging and weighted averaging are used. The stream feature vector after weighted processing F_{flow} can be expressed as:

$$F_{flow} = 0.2 \times x_{duration} + 0.2 \times x_{package\_rate} + 0.2 \times x_{interval} + 0.2 \times x_{byte\_rate} + 0.2 \times x_{download/upload\_radio} + 0.2 \times x_{protocol\_number} \quad (4)$$

Finally, the word feature vectors are concatenated with the weighted stream feature vectors to form a composite feature vector:

$$F_{combined} = [F_{word}, F_{flow}] \quad (5)$$

In this way, the fused feature vectors contain not only the semantic information in the textual data but also retain the key features in the numerical data, forming a unified and comprehensive feature representation.

CNN model based data training

In this paper, a CNN model containing double convolutional layers, a pooling layer, a dropout layer, and a fully connected layer is designed. This model can effectively extract and fuse text and streaming data features, process and fuse the completed data for model training, and optimize the training process using methods such as cross-validation and grid search to ensure the accuracy and robustness of the model.

For the stream features in the initial data, the temporal patterns of the stream features are first extracted by a one-dimensional (1D) convolutional layer. Sixty-four 3 × 1 convolution kernels are used for the convolution operation, and the activation function is chosen to be the rectified linear unit (ReLU) to enhance the expression ability of the features. as shown in Eq. 6:

$$F' = Flatten\left(\begin{array}{c}\text{MaxPooling1D}(2)\\ \left(\text{Conv1D}(64,3,\text{activation}='relu')\text{MaxPooling1D}(2)\right)\\ (Flow_{Features})\end{array}\right) \quad (6)$$

Next, a pooling window of size 2 is used for down-sampling through a 1D maximum pooling layer to reduce the dimension and computation of features while preserving important features. Finally, the pooled features are flattened into 1D vectors using a flattening layer. The processed fusion features are fused with the stream features to form a new feature matrix that is further processed using a 2D CNN, as shown in Eq. 7:

$$C = \text{CONCAT}(C,F') \quad (7)$$

For the fused feature matrix CCC, a 2D convolutional layer is used to further extract higher-order features. First, 64 3 × 3 convolution kernels are used for the convolution operation, and the activation function is selected as ReLU to capture local spatial relations and enhance feature expression. Then, a 2D maximum pooling layer is used for down-sampling, with a pooling window of size 2 × 2 to reduce feature dimensions and computation while retaining important features. Finally, the pooled features are flattened into 1D vectors using a flattening layer for further feature extraction and classification, as shown in Eq. 8:

$$C' = Flatten\left(\begin{array}{c}\text{MaxPooling1D}(2,2)\\ \left(\text{Conv2D}\left(\begin{array}{c}64,(3,3),\\ \text{activation}='relu'\end{array}\right)(C)\right)\end{array}\right) \quad (8)$$

where Conv2D(64,(3,3)) denotes the use of 64 3 × 3 convolutional kernels with 2 × 2 pooling layers for down-sampling.

After feature extraction is completed, the fully connected layer is used to map the feature vector to a higher feature space and further extract the advanced features.

The output dimension of the fully connected layer is 128, and the activation function is selected as ReLU. To prevent overfitting, the dropout layer randomly drops some neurons with a 50% probability, enhancing the generalization ability of the model, as shown in Eq. 9:

$$\mathrm{H}' = ReLU\left(\begin{array}{c}\mathrm{Dropout}(0.5)\\ \left(\begin{array}{c}\mathrm{Dense}\\ (128,\mathrm{activation}='relu')(C')\end{array}\right)\end{array}\right) \tag{9}$$

where dense (128) denotes a fully connected layer that maps feature vectors to 128 dimensions, and dropout (0.5) denotes randomly dropping some neurons with a 50% probability.

Finally, multi-class classification is performed using the Softmax activation function, which maps the feature vectors to the probability distributions of the device categories. The final output dimension is the number of device classes, with the probability of each class calculated by the Softmax function, as shown in Eq. 10.

$$O = SoftMax\left(\mathrm{Dense}\left(\mathrm{num}_{\mathrm{classes,activation}}\right)(H')\right) \tag{10}$$

METODOLOGY

Experimental environment

In the process of processing the dataset, the Windows 10 operating system is used. The system's operating hardware environment includes 16 GB of random access memory (RAM), an AMD Ryzen 5 6600 H processor with Radeon Graphics, and a 1TB SSD. The deep learning framework used for the experiment is Pytorch, with the main packages being Numpy, Pandas, Pytorch, Keras, Sklearn, and Matplotlib. The specific experiment configuration is shown in Table 1.

Table 1. Experimental configuration.

Form	Name	In detail
Software	CPU	AMD Ryzen 5 6600 H
	GPUs	NVIDIA GeForce RTX 4050 GPUs
	(of a computer) run RAM	16 GB
Hardware	Program multilingualism	Python 3.6
	Count flat-roofed building	Pytorch 1.10.2
	Flux artifact	Scapy and SplitCap 2.1

Source: Elaborated by the authors.

Experimental dataset

The dataset used in the experimental study of the method proposed in this paper is primarily the IoT-Sentinel dataset, which contains the information about all devices, as shown in Table 2. The IoT-Sentinel dataset includes information about 31 representative IoT devices that are commonly used in the market, including smart lighting, door sensors, security cameras, smart sockets, and health monitoring devices. Similarly, the researchers captured traffic through Tcpdump software at the gateway, and repeated the setup process for each device 20 times after the traffic was captured vis filtering rules set based on the device's physical address, and stored it locally in a daily pcap file. The physical address of each device is included in the dataset at the time of capture and storage, so it is easy to distinguish the traffic of each device from the dataset and extract the feature set for experiments respectively.

Table 2. IoT-sentinel dataset.

Serial number	Equipment name	MAC address	Type of communication
1	Aria	20:f8:5e:ca:91:52	WiFi
2	D-Link Cam	b0:c5:54:25:5b:0e	WiFi
3	D-Link Day Cam	b0:c5:54:1c:71:85	WiFi/Ethernet
4	D-Link Door Sensor	1c:5f:2b:aa:fd:4e	Z-Wave
5	D-Link Home Hub	1c:5f:2b:aa:fd:4e	WiFi/Ethernet/Z-Wave
6	D-Link Sensor	90:8d:78:a8:e1:43	WiFi
7	D-Link Siren	90:8d:78:dd:0d:60	WiFi
8	D-Link Switch	90:8d:78:a9:3d:6f	WiFi
9	D-Link Water Sensor	6c:72:20:c5:17:5a	WiFi
10	EdimaxCam1	74:da:38:80:7a:08	WiFi/Ethernet
11	EdimaxCam2	74:da:38:80:79:fc	WiFi/Ethernet
12	EdimaxPlug1101W	74:da:38:4a:76:49	WiFi
13	EdimaxPlug2101W	74:da:38:23:22:7b	WiFi
14	EdnetCam1	3c:49:37:03:17:db	WiFi/Ethernet
15	EdnetCam2	3c:49:37:03:17:f0	WiFi/Ethernet
16	Ednet Gateway	ac:cf:23:62:3c:6e	WiFi/Other
17	Home Matic Plug	00:1a:22:05:c4:2e	Other
18	Hue Bridge	00:17:88:24:76:ff	ZigBee/Ethernet
19	Hue Switch	00:17:88:24:76:ff	ZigBee
20	iKettle2	5c:cf:7f:06:d9:02	WiFi
21	Lightify	84:18:26:7b:5f:6b	WiFi/ZigBee
22	MAX Gateway	00:1a:22:03:cb:be	Ethernet
23	Smarter Coffee	5c:cf:7f:07:ae:fb	WiFi
24	TP-LinkPlugHS100	5c:cf:bf:00:fc:a3	WiFi
25	TP-LinkPlugHS110	50:c7:bf:00:c7:03	WiFi
26	WeMo Insight Switch	94:10:3e:41:c2:05	WiFi
27	WeMo Insight Switch2	94:10:3e:42:80:69	WiFi
28	WeMo Link	94:10:3e:cd:37:65	WiFi/ZigBee
29	WeMo Switch	94:10:3e:35:01:c1	WiFi
30	WeMo Switch2	94:10:3e:34:0c:b5	WiFi
31	Withings	00:24:e4:24:80:2a	WiFi

Source: Elaborated by the authors.

Experimental evaluation indicators

Accuracy, precision, recall and F1 (the mean of precision and recall) were used to evaluate the classification performance of the model. The calculation formulas are shown in Eqs. 11–14:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{11}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{12}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{13}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

where true positive (TP) indicates the number of predictions and the number of facts classified, FP (false positive) represents the number of predictions for the classification and the number of facts for the classification, TN (true negative) indicates the number of predictions and the number of facts of the category, and FN (false negative) represents the number of predictions that do not belong to the class and the number of facts that belong to the class.

Analysis of experimental results

To reduce error, this paper performs a preliminary processing on the IoT-Sentinel dataset by excluding the traffic of IoT devices with fewer than 500 traffic records. Multi-classification experiments were conducted on the IoT-Sentinel dataset for category and model identification of specific IoT devices by using different deep learning models (e.g., CNN/MLP). By training and evaluating the network models, the classification results of different models for IoT device type identification based on the IoT-Sentinel dataset were obtained, as shown in Table 3. The confusion matrix drawn based on the IoT-Sentinel dataset and the *receiver operating characteristic* (ROC) curve graphs demonstrate the performance of this paper's method in targeting the category and model recognition of specific IoT devices. The rows of the confusion matrix represent the real labels for IoT devices, and the columns represent the predictive labels for IoT devices identified by the model, as shown in Figs. 3 and 4.

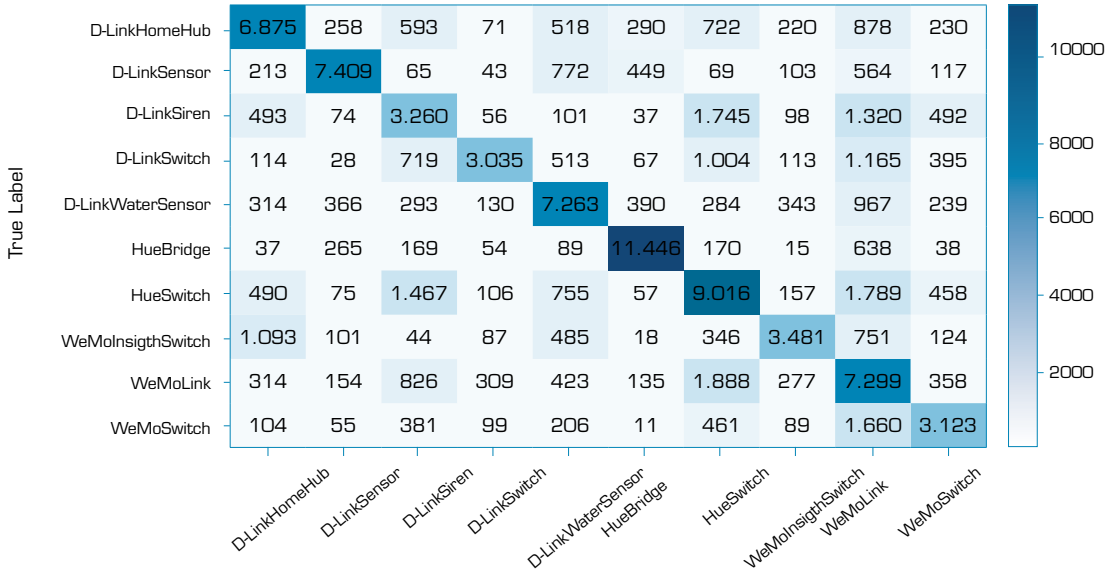
According to Fig.3, it can be seen that the device identification method based on traffic classification proposed in this paper suffers from different degrees of classification overlapping problems in the identification of a few specific device types. There are two main reasons for the proposed method's misidentification: one is that some of these devices have similar functions, and therefore their device communication behaviors are similar. For example, 14% of the D-Link Switches in the IoT-Sentinel dataset are incorrectly identified as Hue Switches because they are all of the same device type, as Switch, and are incorrectly identified due to their similar device communication behaviors. The second reason is that most of these devices are from the same manufacturer or different models of the same device, so the devices will have similar characteristics because they share the same firmware. For example, the D-LinkSensor and D-LinkWaterSensor in the IoT-Sentinel dataset. The IoT devices in these groups have the same hardware and firmware, so in some cases, there is a degree of confusion in identifying them.

From the analysis in Fig. 4, the following detailed conclusions can be drawn: the overall classification performance of the ResNet model is good, with the area under the micro-average ROC curve (AUC) of the classifier being 0.89, which indicates that the classifier performs strongly overall. This means that the classifier is able to distinguish between positive and negative categories better in the overall classification task for different categories. Moreover, the performance difference between categories is evident, with a significant difference in classification performance across categories. The HueBridge category has the highest AUC of

Table 3. Classification results of different models.

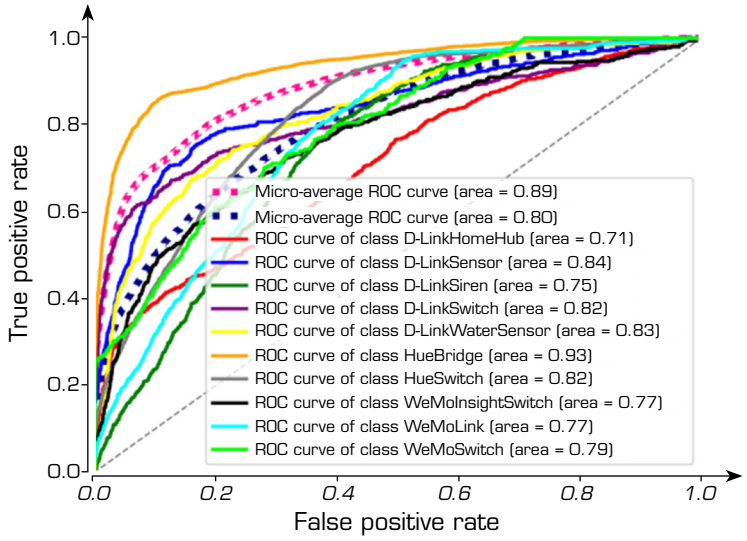
Relevant papers	Model	Accuracy	Precision	Recall	F1
This text	Multi-Modal TC-Device Identification	0.94	0.94	0.94	0.93
Chen et al. (2024) IDS	RF, DT, SVM ,kNN, ANN, Gaussian NB	0.91	0.95	0.87	0.91
Li et al. (2024) Flowgananomaly	GAN	0.87	0.80	0.81	0.88
Wang et al. (2024) NTC-based on federated semi-supervised learning	Naive-Bayes Random-Forest	0.88	0.87	0.81	0.89
Li et al. (2023) NTC	CNN	0.89	0.85	0.95	0.84

Source: Elaborated by the authors.



Source: Elaborated by the authors.

Figure 3. Multi-classification confusion matrix.



Source: Elaborated by the authors.

Figure 4. Multi-class ROC.

0.93, which indicates that the classifier performs very well in distinguishing this category. The D-LinkHomeHub category has the lowest AUC of 0.71, indicating that the classifier has more room for improvement in this category. Most of the other categories have AUC values between 0.75 and 0.84, indicating a more stable and better classification performance for these categories. The AUC of the micro-averaged ROC curve (0.89) is higher than that of the macro-averaged ROC curve (0.80), which suggests that the classifier performs better on categories with higher frequencies. Micro-averaging takes into account the number of samples

in each category, so categories with higher frequencies have a greater impact on the micro-averaging results. The macro-average ROC curve assigns the same weight to each category, so its AUC value reflects the overall performance of the classifier across all categories independent of the number of samples in the category.

Comparison of methods

Table 3 summarizes the advanced classification and recognition technologies and performance of IoT devices in recent years from the perspective of algorithms and experimental indicators. Twenty-three features in the first 12 packets of each device are selected as device fingerprints, including source and destination IP addresses, packet attributes, and protocol information. However, the IP address count refers to the number of devices with which each device communicates, which is affected by the environment. A random forest classification algorithm was used to detect 17 out of 27 device types with an accuracy of more than 95%. The method adopted partial features of the IoT-Sentinel dataset, as well as three payload-related features, and used GB, DT, and k-NN classification algorithms to achieve an 81% recall rate per device and an average accuracy of 88%. However, it used a much smaller number of devices and samples.

Recall rates of up to 95% were achieved on the complete IoT-Sentinel dataset. In contrast, the proposed model selects part of its published dataset, and under the same amount of training data, the recall rate is only 1% lower, the average accuracy is higher, and the training cost is lower, demonstrating its high efficiency and advanced nature. Additionally, the lack of fine-grained (same type, same manufacturer, different models) comparison of IoT devices in the dataset was optimized in this paper. Furthermore, some attributes of its feature set are not sufficient to characterize IoT devices stably, such as stream size, average stream rate, etc. These features vary due to differences in device states, so the dataset and features selected in this paper are more comprehensive and representative. The BiLSTM model and CNN model were used to classify and identify data packets, but the prediction accuracy was low for devices with sparse traffic in idle states, such as sockets, door locks, and Tmall assistants.

To sum up, the classification and recognition model proposed in this paper has a higher average accuracy than existing similar technologies, increasing by 3 to 7%, which can classify and identify IoT devices more efficiently and in a more fine-grained manner, thus coping with the complex and dynamic IoT environment.

CONCLUSIONS

Facing the new challenges of network security brought by the surge in the number of IoT devices, the device identification method based on multi-modal traffic classification proposed in this paper performs well. It integrates multiple traffic features, consists of four modules, collects and processes data with the help of advanced tools, and combines BERT and CNN models for identification. The experimental results prove that it can achieve high accuracy and robustness in diverse network environments and device types.

Future work

Future research can focus on integrating more diverse data sources and using dynamic feature selection to adapt to different network environments and device types. As the number of IoT devices grows, real-time performance and efficiency become critical, suggesting the use of edge or distributed computing to swiftly process traffic data without sacrificing accuracy. Model interpretability is also essential; improving transparency and developing explainable methods will help network administrators and security experts trust and effectively manage system decisions. Additionally, cross-domain adaptation and transfer learning can ensure the identification system remains accurate and robust in new network contexts. Finally, defending against increasingly sophisticated attacks – such as disguised traffic – will be key to enhancing the security of device identification systems.

CONFLICT OF INTEREST

Nothing to declare.

AUTHORS' CONTRIBUTION

Conceptualization: Ma Y; **Methodology:** Wang Y; **Software:** Ma Y; **Validation:** Wang Y; **Investigation:** Xi Z; **Formal analysis:** Xi Z; **Resources:** He C; **Final approval:** Wang Y.

DATA AVAILABILITY STATEMENT

The data will be available upon request.

FUNDING

National Key Research and Development Program of China
Grant No: 2022YFB3104300

REFERENCES

Chen X, Wang P, Yang Y, Liu M (2024) Resource-constraint deep forest based intrusion detection method in internet of things for consumer electronic. IEEE Trans Consumer Electron 70(2):4976-4987. https://doi.org/10.1109/TCE.2024.3373126

Chen Z, Xu G, Mahalingam V, Ge L, Nguyen J, Yu W, Lu C (2016) A cloud computing based network monitoring and threat detection system for critical infrastructures. Big Data Research 3:10-23. https://doi.org/10.1016/j.bdr.2015.11.002

Jian T, Tong J, Bruno CR, Emmanuel O, Nasim S, Zifeng W, Kunal S, Andrey G, Jennifer D, Kaushik C, Stratis I (2020) Deep learning for RF fingerprinting: a massive experimental study. IEEE Internet Things J 3(1):50-57. https://doi.org/10.1109/IOTM.0001.1900065

Li Z, Wang P, Wang Z, Li Z, Wang P, Wang Z (2024) Flowgananomaly: flow-based anomaly network intrusion detection with adversarial learning. Chin J Electron 33(1):58-71. https://doi.org/10.23919/cje.2022.00.173

Li Z, Zhang Z, Fu M, Li Z, Zhang Z, Fu M, & Wang P (2023) A novel network flow feature scaling method based on cloud-edge collaboration. Paper presented 2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications. IEEE; Exeter, United Kingdom. https://doi.org/10.1109/TrustCom60117.2023.00265

Liang Z, Tao M, Xie J, Liang Z, Tao M, Xie J, Yang X, Wang L (2022) A Radio signal recognition approach based on complex-valued CNN and self-attention mechanism. IEEE Trans Cogn Commun Netw 8(3):1358-1373. https://doi.org/10.1109/TCCN.2022.3179450

Wang ZX, Li ZY, Fu MY, Wang Z, Li Z, Fu M, Ye Y, Wang P (2024) Network traffic classification based on federated semi-supervised learning. J Syst Archit 149:103091. https://doi.org/10.1016/j.sysarc.2024.103091