IoT Devices Classification Base on Network Behavior Analysis

Lingan Chen[1(✉)], Xiaobin Tan[1,2], Chuang Peng[2], Mingye Zhu[3], Zhenghuan Xu[3], and Shuangwu Chen[1,2]

[1] Department of Automation, University of Science and Technology of China, Hefei, China
cla@mail.ustc.edu.cn, {xbtan,chensw}@ustc.edu.cn

[2] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China
pengchuang@mail.ustc.edu.cn

[3] Institute of Advanced Technology, University of Science and Technology of China, Hefei, China
mingyezhu@mail.ustc.edu.cn, zhxu86@ustc.edu.cn

Abstract. IoT devices classification from the network traffic have attract increasing attention due to the manage requirement of the growing IoT applications.

Current statistic feature based methods needs to sample most of the packets and requires much computation during the feature extracting. The features are easily affected by the network environment and user operations.

This paper proposes a network behavior analysis (NBA) based IoT devices classification scheme which takes the sequence of network session features as input and learns the behavior feature with LSTM for IoT devices classification. NBA bases on the network behavior analysis without parsing the packet of network traffic, thus can handle the traffic using encryption protocol. In addition, we built a testbed with IoT devices for evaluation.

The traffic generated in the testbed was captured for evaluation. The result of the experiment indicates that by incorporating destination and interval features, our method can accurately classify IoT and non-IoT devices and achieves the accuracy over 99%.

Keywords: Internet of Thing · Machine learning · Deep learning · Traffic classification

1 Introduction

The growing Internet of Things (IoT) devices in campuses and cities brings the need for knowing what IoT devices are connected and whether they are functioning normally for the administrator. Identifying IoT devices from the network

J. Lin and Q. Tang (Eds.): AC3 2022, LNICST 448, pp. 173–188, 2022.
https://doi.org/10.1007/978-3-031-17081-2_11

traffic is a reality method to address this problem. The traffic of IoT devices features mean rate, average packet size, sleep time, DNS interval etc. Such statistical characteristics can be used for Random Forest algorithm to classify IoT devices [1]. However, by using traffic camouflage technologies, malware may make the statistical features of its traffic look like those of normal IoT devices [2].

Because malware may split data into fragments of any size for transmission, such packet-length-based algorithms are also vulnerable to traffic disguise. IP addresses are difficult to conceal because of its importance in packet routing and forwarding. Web apps can be classified according to the destinations with which they communicate [3]. However network destinations may change over time and IoT devices from the same manufacturer may share the same set of destination.

The traffic generated by IoT devices comes from their network behaviors, such as DHCP, reporting logs, broadcasting device information, providing remote control services for users, etc. Different functions of IoT devices lead to different network behaviors. The traffic generated by different network behaviors of devices corresponds to different sessions, thus the sequence of sessions contains the network behavior information of devices. The network behavior of the device is not only which server to access, but also how to access the target server, in what order and frequency.

An IoT traffic categorization framework based on network behavior analysis was developed in this study. The framework introduces IoT devices' destinations to prevent traffic camouflage and overcomes the issue of changing IP addresses by grouping comparable destinations into IP pools. Our contribution can be summarised as follows:

- We designed an IoT traffic classification framework based on network behavior analysis. The framework analyses the sequence of session features and LSTM with attention mechanism was applied to extract network behavior features for IoT devices classification.
- We researched the network behavior of IoT devices and two effective features in the preliminary experiment. We introduces the behavior features into our framework to classify IoT devices. Our algorithm can apply to encrypted IoT traffic since it is based on network behavior analysis without parsing the content of packets.
- An IoT devices testbed was constructed. The network traffic generated in the testbed was collected for experiment and evaluation. We implemented the proposed IoT devices classification framework and evaluated it with the traffic collected from the testbed. Experiment result shows that our classifier can not only distinguish between IoT and non-IoT devices, but also uniquely classify IoT devices with over 99% accuracy.

2 Related Work

Traffic identification of IoT devices has attracted researchers' attention in recent years [4]. Work in [1] built an experimental testbed for IoT devices and collected the traffic generated by IoT devices. On this basis, they studied the statistical

characteristics of these devices, such as mean rate, average packet size, sleep time, and DNS interval etc. After dividing the statistical characteristics into multiple bins, they applied random forest for device identification. In the next work [5] of the same team, more devices were added to their testbed and the distributions of remote port number, domain name and cipher suites were analysed. Then a two-stage classification algorithm based on random forest and Bayesian classifier was designed to classify IoT devices.

Work in [6] also applied machine learning algorithms to the identification of IoT devices based on statistical features. The difference is that PCA method is used to remove correlated features, and simpler machine learning algorithms are used, such as support vector machine, logistic regression and decision tree. Besides, some works [7,8] attempted to infer user activities from the network traffic of IoT devices.

In the broader field of traffic identificationthe DPI-based (Deep Packet Inspection) algorithms can reliably detect different types of traffic [9]. However, the DPI's application scenario has been limited by the growing adoption of the transport layer security (TLS) protocol [10], which is also applied to encrypt IoT traffic.

Following the development of deep learning technology in the areas of Computer Vision and Natural Language Processing (NLP), researchers began to study with applying it to classify encrypted traffic. Convolutional Neural Network (CNN) [11] was investigated for automatically extracting characteristics from encrypted traffic raw data [12–14]. Despite certain datasets' excellent accuracy, researchers revealed that CNN extracts features mostly from packet sizes rather than the encrypted load data itself [15]. CNN can also, in fact, perform efficiently with images derived from packet lengths and packet time stamps [16]. As a result, researchers were naturally considering extracting characteristics directly from the sequence of packet sizes, and the Long Short-Term Memory network (LSTM) [17], which was designed to extract features from sequence data and is extensively used in the field of NLP, is used to traffic categorization [18].

3 Network Behavior Analysis Based IoT Devices Classification

3.1 Motivation an Analysis

Although encryption of IoT traffic hinders the classification of IoT devices. the network behaviors of IoT devices such as log uploading, command receiving are reflected in the network behavior of which destination and how to access. This inspired us to identify IoT devices from the perspective of network behavior.

IP addresses themselves contain enough information for traffic classification [3]. However, there is a difficulty with using IP addresses as features. First, IP addresses may change over time as a result of load balancing or server migration. Classifiers based on IP addresses are likely to be overfitted and incapable of dealing with changing IP addresses. Second, various IoT devices can share common

Table 1. Preliminary experiment result on destination sequence of session in 30 min

ID	Devices	Precision	Recall	F1-score
0	Non-IoT	1.0000	0.9984	0.9992
1	Smart camera	0.9984	0.9996	0.9990
2	Temperature sensor	0.9556	0.8945	0.9240
3	Smart bulb	0.9695	0.9974	0.9832
4	Smart socket	0.1500	0.0327	0.0538
	Accuracy	0.9907		

destinations such as gateways, DNS servers, CDN servers and cloud servers. IoT devices of the same manufacturer may also share manufacturer-specific destinations.

Devices connected to specific IP addresses are expected to perform specified tasks. When a device connects to a different destination than it did previously but for the same purpose, the IP addresses before and after the modification are related and will have similar characteristics and tend to be clustered in the same IP pool. Consequently, the strategy for changing IP addresses is to divide IP addresses into clusters according to certain rules and then classify devices according to which cluster they access.

Table 2. Sessions of the DstIP of the Confused Devices

dstIP	Description	Sessions	
		Smart bulb	Smart socket
255.255.255.255	Broadcast	52	14
192.168.1.1	Gateway	3316	13
111.231.160.125	Tencet cloud	61	24
121.5.96.248	Founder bandwidth	6564	191
42.192.30.165	Yovole cloud	11	0
47.110.145.119	Alibaba cloud	3278	0

We carried up a preliminary experiment in which different devices were classified based on the destination sequence of sessions over 30 min. The result as shown in Table 1. demonstrates that this behavior characteristic can identify between non-IoT and IoT devices, as well as between devices of various brands, but that distinguishing between devices of the same manufacturer is challenging.

Table 2 shows the destination IPs of the devices. IoT devices connects to the broadcast destination to make themselves found by users, to the gateway to attain DHCP, DNS and NTP service, and to the destination of cloud service providers for their custom functions such as logging and remote controlling. It

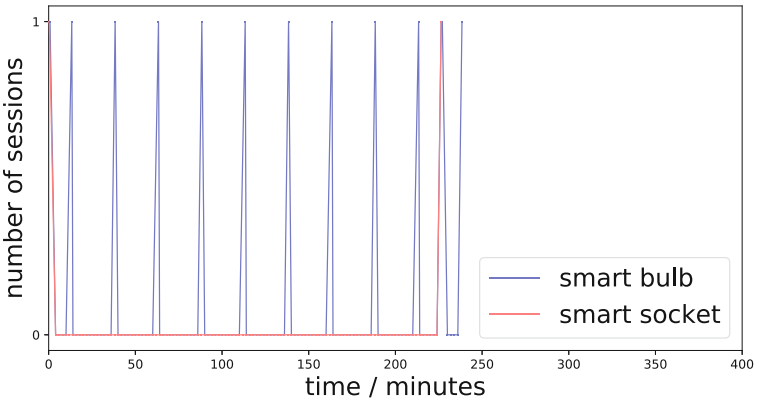

Fig. 1. Session behavior of confused devices

can be seen that the target IP of smart socket is identical to the destination of smart bulb, and the number of sessions of smart socket is much smaller than that of smart bulb, causing the deep learning model to mistake smart socket traffic for smart bulb traffic.

The difference in the number of sessions reveals the temporal behavior difference of the two devices. As shown in Fig. 1, there is a huge difference between the two devices in the frequency of establishing new sessions. In the preliminary experiment the smart socket's target address sequence can be as short as one or two sessions in a short period of time, making it difficult for the model to use the sequence's behavior information and forcing it to rely solely on the information contained in the IP address, directing the result to the smart bulb.

Although different devices may share common destinations, the network behavior of devices varies depending on their functionality and hardware, which may reflect on the order and temporal behavior pattern of destinations. Therefore, not only the combination of IP pool, but the order of IP pools and session interval that devices access, was considered.

3.2 Overview

Because the network behavior of devices is reflected in sessions, our framework is designed to extract behavior information from the sequence of sessions to classify devices. To achieve this goal, this framework needs to extract the features containing behavior information from the raw traffic from the sessions, then extract the behavior features and classify devices with a classifier. The overall framework of network behavior analysis based IoT devices classification is shown in the Fig. 2. The framework is mainly composed of four parts: session feature extraction, session feature preprocess, sequence feature extraction and multilayer perceptron (MLP) [19] classifier.

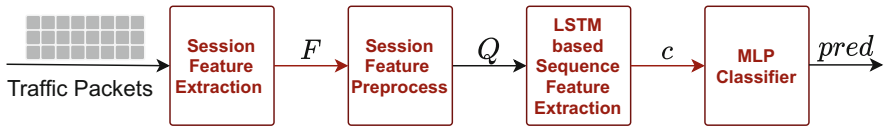

Fig. 2. System architecture of network behavior analysis

The session feature extraction module extracts session features F directly from the raw traffic data. Then $\boldsymbol{F}$ is processed by the session feature preprocess module into vectors Q for the following model fitting.

The LSTM based sequence feature extraction module generates the feature vector $\boldsymbol{c}$ of behavior, and the multilayer perceptron classifier can predict the IoT devices as *pred* from the feature vector $\boldsymbol{c}$.

3.3 Session Feature Extraction

The session feature extraction module is to extract the basic features for the following analysis as shown in Fig. 3.

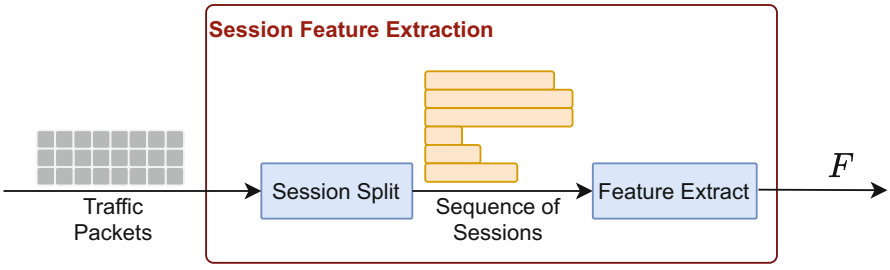

Fig. 3. Process of session feature extraction module

To begin with, the traffic is separated into sessions based on five tuples $(srcIP, srcPort, dstIP, dstPort, protocol)$, and the sessions are ordered by their begin time. Then the sequence of session features $\boldsymbol{F}$ can be abstracted as

$$\boldsymbol{F} = [F_1, F_2, ..., F_n] \tag{1}$$

where the length n depends on the amount of sessions and the i-th session's feature F_i can be represented as:

$$F_i = [f_i^1, f_i^2, ..., f_i^m] \tag{2}$$

representing the combination of m features extracted from each session.

In our work the destination and the begin-time interval are chosen to feature the network behavior. They are easily to obtained that only the first packets

of each session are needed. Sampling and processing of the full sessions can be avoided, which greatly reduces the difficulty for deploying on gateways in the future. Therefore in this case, F_i and $\boldsymbol{F}$ can also be represented as:

$$F_i = [\tau_j, d_j] \tag{3}$$

$$\boldsymbol{F} = [\boldsymbol{\tau}, \boldsymbol{d}] \tag{4}$$

where τ_j and d_j are the j-th session's begin-time interval and destination IP address respectively. Other session features such as volumes and average packet sizes can also be applied for further describe the network behavior.

3.4 Session Feature Preprocess

The function of this module is to preprocess the raw feature $\boldsymbol{F}$ into the sequence of numerical vectors Q for the following model training as shown in Fig. 4.

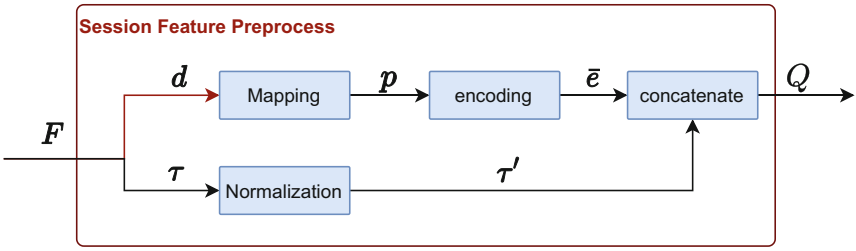

Fig. 4. Process of session feature preprocess module

For the numerical feature begin-time interval $\boldsymbol{\tau}$, normalization is applied and results in $\boldsymbol{\tau'}$. And for the destination IP addresses $\boldsymbol{d}$, we choose to map them into IP pools $\boldsymbol{p}$ and encode the IP pools into vectors $\bar{\boldsymbol{e}}$.

The destinations are clustered to the IP pools as:

$$p_i = Map(d_i) \tag{5}$$

$$\boldsymbol{p} = Map(\boldsymbol{d}) = [p_1, p_2, ..., p_n] \tag{6}$$

where d_i is the i-th destination, p_i is the i-th IP pool and the mapping function $M(\cdot)$ is determined by the result of IP clustering method such as [3] or the existing IP address database. The output of the IP clustering method determines the mapping function. IP address pools do not need to be labeled because the correlation between p_i will be learned during training.

Then p_i is encoded into vector $\boldsymbol{e}_i$:

$$\boldsymbol{e}_i = E(p_i) \tag{7}$$

$$\bar{\boldsymbol{e}} = E(\boldsymbol{p}) = [\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n] \tag{8}$$

where encoder method $E(\cdot)$ can be one-hot or embedding. All the preprocessed session features are finally concatenated into a vector $\boldsymbol{q}_i$ before fed into the sequence feature extraction module:

$$\boldsymbol{q}_i = Cat(\boldsymbol{e_i}, \tau_i) \tag{9}$$

$$Q = [\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_n] \tag{10}$$

3.5 LSTM Based Sequence Feature Extraction

We designed an LSTM with attention mechanism-based sequence feature extraction model. Its structure is shown in the Fig. 5, which is mainly composed of an attention module and an LSTM module.

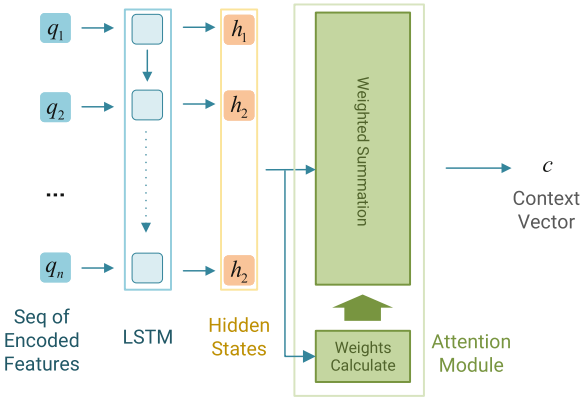

Fig. 5. LSTM based sequence feature extraction

As shown in the previous section, the input of the model is a sequence of encoded session features. The sequence of $[\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_n]$ is fed to an LSTM module, then the output for each step is:

$$[\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_n] = LSTM(Q) \tag{11}$$

where h_i is an m dimension vector that contains the information of $\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_i$. But $\boldsymbol{q}_i$ gives the most influence on $\boldsymbol{h}_i$, so $\boldsymbol{h}_i$ is chosen to represent the feature of $\boldsymbol{q}_i$.

To conclude the class of the sequence we need to calculate a vector to represent the features of the whole sequence, and that is the work of attention module. $\boldsymbol{h}_i$ is fed to the weight calculation module to determine the weight w_i,

representing the importance of the i-th IP pool to feature the whole sequence. Then the context vector

$$\boldsymbol{c} = \Sigma w_i \boldsymbol{h}_i = [c_1, c_2, ..., c_m] \tag{12}$$

represents the sequence as the output of attention module and is encoded with the network behavior feature of the whole sequence.

The attention layer has two functions. First, it can help the LSTM layer to synthesize information between sequences within a long-time interval, and alleviate the lack of LSTM's ability to recognize long sequences. Second, it can show the focus of the model every time it extracts feature, increasing the interpretability of the network. In this article, attention can point out the sessions that the model focuses on when making predictions. We can use this to investigate whether the neural network has learned a reasonable model and prepare for future work.

3.6 Multilayer Perceptron Classifier

Multilayer perceptron classifier predicts the scores of devices from the context vector $\boldsymbol{c}$. As shown in Fig. 6, The MLP is composed of multiple fully connected layers, and learns a non-linear function for classification:

$$f(\cdot) : R^m \rightarrow R^k \tag{13}$$

by training on a dataset where m is the dimension of for input and k is the dimension of output. Between the input and output layer, there can be one or more non-linear layers called hidden layers. Hidden layer consists of a set of neurons and each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation followed by a non-linear function. the output layer receives the values from the last hidden layer and transforms them into output values $\boldsymbol{s}$, representing the score of IoT devices in our situation.

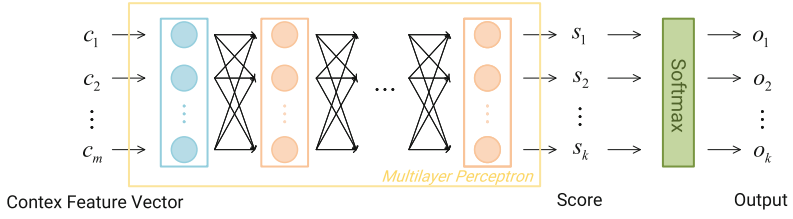

Fig. 6. Multilayer perceptron classifier

Finally the scores are normalized by the softmax function:

$$\sigma(\boldsymbol{s})_i = \frac{e^{s_i}}{\sum_{j=1}^{K} e^{s_j}} = o_i \tag{14}$$

to approximate the probability of IoT devices.

MLP is chosen to facilitate the training of neural network. After the LSTM model learned the ability to encode behavior features into context vector $\boldsymbol{c}$ during training, other classifiers such as Random Forest can also be applied to IoT devices classification.

4 Evaluation

4.1 IoT Testbed Construction and Traffic Collection

Our testbed constitutes a virtual gateway router, a virtual PC, an AP and four different IoT devices as in Table 3 and Fig. 7, all deployed on the CENI platform [20]. The virtual router with VSR1000_H3C-CMW710-E0618-X64-3 firmware installed serves as the gateway of the LAN. The WAN interface is linked to the public Internet and WLAN of the AP is connected to IoT devices. Besides, a LAN interface is connected to a Virtual Windows PC for controlling, monitoring, and traffic dumping via SSH. Additional package *tcpdump* is installed to dump the traffic of devices in the testbed. The IoT devices in the testbed include Tuya smart lightbulb, Tuya smart socket, Vemsee temperature sensor, and JOOAN wireless camera. Some other non-IoT devices connected to the testbed are smartphones and laptops.

Table 3. Devices in the Testbed

Device	Model	Firmware	Manufacturer	Description
Virtual router	–	VSR1000_H3C-CMW710	H3C	Gateway, traffic dumping
Virtual PC	–	Windows10 18362.1082	–	Saving captured traffic
Temperature sensor	DK-300C4-WS-WIFI	–	VMS	IoT device
Smart socket	YKYC-001	V1.1.1	Tuya	IoT device
Smart bulb	XDUO-S1	V1.1.7	Tuya	IoT device
WIFI Camera	JA-C10R	03.03.30.33	JOOAN	IoT device
Laptop	Xiaoxin Air 14ITL 2021	Windows10 18362.1082	Lenovo	Non-IoT device
Smart phone	GM1910	Hydrogen OS 10.0.10.GM21	OnePlus	Non-IoT device

The traffic in the testbed was captured via the *tcpdump* tool on the router, then it was delivered to the virtual PC through the SSH tunnel and eventually restored by the *dumpcap* tool of Wireshark.

The collecting of traffic lasted from May 26th, 2021 to September 15th, 2021. During this period, over 20000 flows and 500,000 packages are collected.

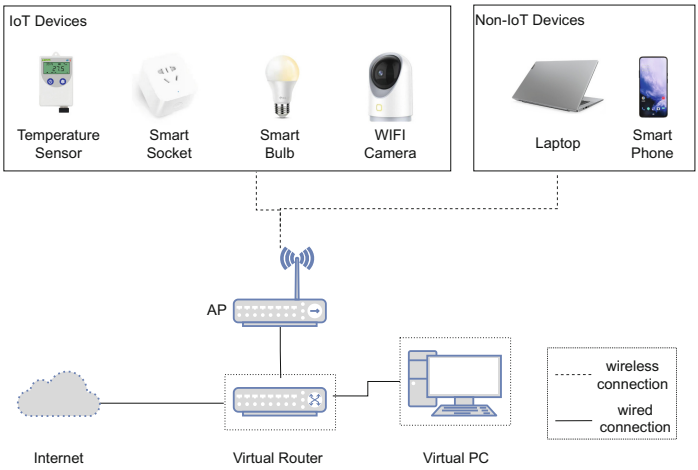

Fig. 7. IoT environment testbed

4.2 Experimental Settings

The algorithm of CCIT [1] is implemented for comparison because CCIT is one of the most classical methods in the field of IoT devices classification. CCIT extracts the sleep time, active volume, average packet size, mean rate, peak/mean rate, active time, No. of servers, No. of protocols, unique DNS request, DNS interval and NTP interval as characteristic then apply Random Forest algorithm as classifier. CCIT deals with the same task of our method and is reported to reach over 95% accuracy.

Our neural network model was implemented using pytorch. To map the IP address into pools the public IP address attribution database [21] was applied. It should be emphasized that although the IP address database can provide the geographic location of each IP address, it was only used to cluster the IP addresses without introducing their geographic location information. We used embedding layer to encode the IP pools into 3-dimension vectors because the embedding layer can learn the correlation of the pools and reflect such correlation into the distance in the feature space. The implemented model was trained on the CENI [20] platform with virtual machine of GeForce GTX 1080.

Because the number of sessions generated by different devices varies greatly as in Fig. 8, we copy multiple copies of the subset of devices with smaller number of instances during training to balance the data, so that the amount of data instances of each device is roughly the same. This is not data amplification, but a way to change the weight of the devices. During verification, we also calculate the evaluation criterions with the expanded dataset to reduce the impact of the results of devices with large amount of instances on the overall results. Our framework is denoted as NBA in the following description.

We evaluated the performance of classifying specific devices of the methods base on the precision, recall and F1-score. We also use the accuracy (the ratio of

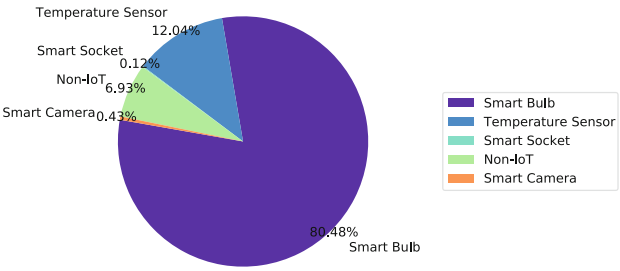

Fig. 8. Distribution of different devices in the dataset

all the correctly predicted instances and all the instances) to measure the overall performance. the definitions of the criterions are as follows:

$$precision = \frac{TP}{TP + FP} \tag{15}$$

$$recall = \frac{TP}{TP + FN} \tag{16}$$

$$F1 = \frac{precision \cdot recall}{precision + recall} \tag{17}$$

where TP, FP, TN, and FN are instances of true positive, false positive, true negative, and false negative respectively. The performance of models was evaluated by the average of each criteria in the 5-fold cross validation. Due to the long time for IoT devices to generate sessions, the most time-consuming step in our method comes from data caption rather than data processing and result inference. Therefore, we only use the classification criteria to measure our proposed algorithm.

4.3 Experimental Result and Analysis

The evaluation results are shown in Table 3. The result of preliminary experiment is also included for comparison and indicated as Pre.

Table 4. Experiment results on precision, recall and F1-score

ID	Devices	CCIT			Pre			NBA(dst)			NBA(dst+interval)		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
0	Non-IoT	0.9838	0.9750	0.9793	**1.0000**	0.9984	0.9992	0.9999	0.9990	0.9995	0.9998	**1.0000**	**0.9999**
1	Smart camera	0.9703	0.9894	0.9798	0.9984	0.9996	0.9990	**0.9993**	0.9999	**0.9996**	0.9988	**1.0000**	0.9994
2	Temperature sensor	0.9817	0.9831	0.9824	0.9556	0.8945	0.9240	0.9998	**1.0000**	0.9999	**1.0000**	**1.0000**	**1.0000**
3	Smart bulb	**0.9958**	0.9942	**0.9950**	0.9695	**0.9974**	0.9832	0.8031	0.9755	0.8804	0.9865	0.9973	0.9918
4	Smart socket	0.9909	**0.9871**	0.9889	0.1500	0.0327	0.0538	0.9713	0.7706	0.8581	**0.9987**	0.9868	**0.9926**
	Accuracy	0.9836			0.9907			0.9476			**0.9967**		

Classification of IoT Devices and Non-IoT Devices. NBA with only destination as features is capable of accurately classifying IoT and non-IoT devices with precision and recall more than 0.99. This is due to the significant differences in IP accessing behavior between IoT and non-IoT devices. Non-IoT devices, such as laptops and smartphones, always connect to a variety of servers, depending on the Web apps on the platform. Such difference reflects in the sequence of the destination and can easily captured by the LSTM module. CCIT can also classify IoT devices and Non-IoT devices in precision over 0.98, but it also gets confused in some cases due to the overlapping of some features of IoT devices and Non-IoT devices. For example, the mean rate of IoT devices is usually lower than that of Non-IoT devices, but it can also reverse when Non-IoT devices are not active and IoT devices are active. When users remotely monitor through the smart camera, the camera can generate traffic in pretty high mean rate.

Classification of IoT Devices from Different Manufacturers. Among the IoT devices of the dataset only the smart socket and smart bulb are manufactured by the same company, while the rest are made by separate companies. Because servers from different manufacturers are often installed at separate locations, the results of preliminary experiments shown in Table 4 indicate that IP addresses can effectively differentiate IoT devices from different manufacturers. The results for CCIT reveal that statistical features change significantly for various IoT devices, although not as much as the address. The sequence of IP addresses is acquired in the preliminary experiment based on a predefined length of time, which results in the short sequence length of some data instances for equipment with less traffic, resulting in poor classifier performance. NBA with destination secures data sequence length by acquiring a fixed length sequence, which results in a data instance that requires more device traffic but increases the classifier's performance greatly. NBA with destinations and session intervals combines the benefits of CCIT and IP address, thus it has the ability in differentiating between IoT devices from various manufacturers with almost 100% precision and recall, better than CCIT.

Classification of IoT Devices from Same Manufacturer. The recognition ability of the classifier for various IoT devices from the same manufacturer can be reflected in the results for smart bulb and smart socket. Previous investigations as shown in the result of smart socket and smart bulb in Table 1 and Table 4 have demonstrated that it is hard to discern various devices by IP address alone since the server deployment addresses of the same manufacturer are identical. The NBA with destination has helped to solve this problem. Because the behavior of various devices visiting each target address is different in order, even though the target servers are the same, this performance is improved when the data sequence length is guaranteed, but not as good as CCIT. That is because this element has no effect on CCIT. We have to point out that when the session-begin-time interval is considered, the result of smart bulb is satisfactory with an 0.9918 F1 score under the influence of the smart socket from its same manufacturer.

This demonstrates that statistical characteristics can differentiate various devices from the same manufacturer, and we will investigate combining such features into our frame in the future work.

Therefore we can conclude that the NBA with destination and session-begin-time interval supplemented the behavioral information in time, allowing it to perform at the CCIT's level, in the IoT devices from the same manufacturer, reflecting in the result of smart socket and smart bulb in Table 4.

4.4 Summary

The framework design of NBA is expansionary and modular. Its behavior features mainly come from the sequence of sessions, so all session features can be introduced. The extraction of behavior features can also be replaced by other models that can extract sequence information.

By introducing the behavioral characteristics of IP address and session time interval, the NBA achieves accurate recognition of IoT devices with an overall accuracy of over 99.9%, which is higher than CCIT.

Clustering IP addresses is an a priori information that the model incorporates. This priori information can aid the trained model in slowing down its failure because both unsupervised clustering methods and manually labeled IP address databases can be updated in real time.

5 Conclusion

In this study, we investigated the topic of IoT device classification from IoT traffic by developing a smart environment using IoT devices. Current statistic feature based methods may be confused by traffic camouflage technologies. We proposed network behavior analysis (NBA) framework based IoT devices classification algorithm and compared it with statistics-based methods in the evaluation. The NBA can learn the behavior characteristics from multiple session features. We investigated the destination features to solve the traffic camouflage and introduce session interval to distinguish different devices of the same manufacturer. The result shows that the NBA can distinguish IoT devices and Non-IoT devices in high precision and has great potential in combining with statistics features. Our framework can not only classify IoT devices with encrypted traffic, but also take few computing resources in the traffic extracting phase. Moreover, our framework can flexibly configure different features and classifier, therefore there is great potential to be studied.

Our future work will be expanded from the following aspects. First, we will add more IoT devices to the testbed, including multiple devices of same manufacturer to obtain more data to verify the our algorithms. Secondly, there is less work on IP address clustering at present, so we will explore better IP address clustering methods to strengthen the performance of our algorithm. Finally, we will consider applying more efficient deep learning model as the backbone to fit IoT traffic.

Acknowledgment. This work was supportedinpart by the National Key RDProgramof China under Grant 2020YFA0711400, inpart of Key Science and Technology Project of Anhui under Grant 202103a05020007, inpart by the Key RDProgramof Anhui Province in 2020 under Grant 202004a05020078, inpart by the China Environment for Network Innovations (CENI) under Grant 2016-000052-73-01-000515.

References

1. Sivanathan, A., et al.: Characterizing and classifying IoT traffic in smart cities and campuses. In: 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 559–564 (2017)
2. Wang, N., Chen, Y., Xiao, Y., Hu, Y., Lou, W., Hou, T.: Manda: on adversarial example detection for network intrusion detection system. In: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications, pp. 1–10 (2021)
3. van Ede, T., et al.: Flowprint: semi-supervised mobile-app fingerprinting on encrypted network traffic. In: Network and Distributed System Security Symposium (NDSS), vol. 27
4. Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., Lloret, J.: Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. IEEE Access **5**, 18042–18050 (2017)
5. Sivanathan, A., et al.: Classifying IoT devices in smart environments using network traffic characteristics. IEEE Trans. Mob. Comput. **18**(8), 1745–1759 (2019)
6. Bikmukhamedov, R.F., Nadeev, A.F.: Lightweight machine learning classifiers of IoT traffic flows. In: 2019 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO), pp. 1–5 (2019)
7. Junges, P.M., François, J., Festor, O.: Passive inference of user actions through IoT gateway encrypted traffic analysis. In: 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 7–12 (2019)
8. Das, A.K., Pathak, P.H., Chuah, C.N., Mohapatra, P.: Uncovering privacy leakage in BLE network traffic of wearable fitness trackers. In: Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications, HotMobile 2016, pp. 99–104, New York, NY, USA (2016). Association for Computing Machinery
9. Deri, L., Martinelli, M., Bujlow, T., Cardigliano, A.: nDPI: open-source high-speed deep packet inspection. In: 2014 International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 617–622 (2014)
10. Liu, J., Fu, Y., Ming, J., Ren, Y., Sun, L., Xiong, H.: Effective and real-time in-app activity analysis in encrypted internet traffic streams. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017, pp. 335–344, New York, NY, USA (2017). Association for Computing Machinery
11. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
12. Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R., Saberian, M.: Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft. Comput. **24**(3), 1999–2012 (2019). https://doi.org/10.1007/s00500-019-04030-2
13. Wang, W., et al.: HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. IEEE Access **6**, 1792–1806 (2018)

14. Wang, W., Zhu, M., Wang, J., Zeng, X., Yang, Z.: End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 43–48 (2017)
15. Tong, X., Tan, X., Chen, L., Yang, J., Zheng, Q.: BFSN: a novel method of encrypted traffic classification based on bidirectional flow sequence network. In: 2020 3rd International Conference on Hot Information-Centric Networking (HotICN), pp. 160–165 (2020)
16. Shapira, T., Shavitt, Y.: Flowpic: encrypted internet traffic classification is as easy as image recognition. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 680–687 (2019)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
18. Liu, C., He, L., Xiong, G., Cao, Z., Li, Z.: FS-Net: a flow sequence network for encrypted traffic classification. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, pp. 1171–1179 (2019)
19. Ruck, D.W., Rogers, S.K., Kabrisky, M., Oxley, M.E., Suter, B.W.: The multilayer perceptron as an approximation to a Bayes optimal discriminant function. IEEE Trans. Neural Networks **1**(4), 296–298 (1990)
20. China environment for network innovations (CENI). http://ceni.ustc.edu.cn/
21. Zx ipv6 address query. http://ip.zxinc.org/