An IoT Device Identification Method based on Semi-supervised Learning

Linna Fan*†‡, Shize Zhang*†, Yichao Wu*†, Zhiliang Wang*†, Chenxin Duan*†, Jia Li§ and Jiahai Yang*†
* Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China
† Beijing National Research Center for Information Science and Technology
‡ School of Information and Communications, National University of Defense Technology, Xi'an, China
§ National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China
Email: fln19@mails.tsinghua.edu.cn, zsz16@mails.tsinghua.edu.cn, wu-yc19@mails.tsinghua.edu.cn
wzl@cernet.edu.cn, dcx19@mails.tsinghua.edu.cn, lijia@cert.org.cn, yang@cernet.edu.cn

***Abstract*—With the rapid proliferation of IoT devices, device management and network security are becoming significant challenges. Knowing how many IoT devices are in the network and whether they are behaving normally is significant. IoT device identification is the first step to achieve these goals. Previous IoT identification works mainly use supervised learning and need lots of labeled data. Considering collecting labeled data is time-consuming and cannot be scaled, in this paper, we propose an IoT identification model based on semi-supervised learning. The model can differentiate IoT and non-IoT and classify specific IoT devices based on time interval features, traffic volume features, protocol features and TLS related features. The evaluation in a public dataset shows that our model only needs 5% labeled data and gets accuracy over 99%.**

***Index Terms*—IoT, identification, semi-supervised learning**

I. Introduction

Technology of Internet of Things (IoT) provides a vast market space for equipment manufacturers, Internet Service Providers (ISP) and application developers [1]. The variety of IoT devices bring convenience to a person's life and are widely used in our home and city [2]. However, the rapid proliferation of IoT devices also brings some trouble. First, it introduces increasing operational and management challenges. Network managers usually have no idea how many IoT devices are in the network. Second, IoT devices are becoming the focus of attackers. It is well recognized that IoT devices are by their nature easier to infiltrate [3].

Identifying different IoT devices and monitoring their status to ensure they are behaving normally is significant for asset management and IoT security.

Recently, researchers of IoT device identification mainly leverage supervised machine learning methods and reach accuracy over 99% [4]–[16]. These methods need numerous labeled data for training. However, collecting numerous labeled data is arduous and time-consuming, which cannot be scaled to the environment with more IoT devices.

To solve this problem, in this paper, we propose an IoT device identification model based on semi-supervised learning. In semi-supervised learning, we only have a few labeled data, and we want to infer labels of other data through the few labeled data and numerous unlabeled data. The challenges of semi-supervised learning applying in IoT device identification come from three aspects. First, the features chosen should differentiate different devices as much as possible, because we must rely on a few labeled data and numerous unlabeled data. Second, in practice, IoT devices have different running stages and present different features, which may impact accuracy. Besides, some network actions like scanning may also affect the traffic of IoT devices and therefore change the features of the device. At last, in the dataset, there are also several non-IoT devices. They may have very different features and affect the classification result.

Facing the challenges, we propose a semi-supervised model. To separate different devices as much as possible, the model takes features including time interval features, traffic volume features, protocol features and TLS (Transport Layer Security) related features as input. To solve the problem of feature fluctuation better, we make the transformed features form a single and compact cluster per class, which can be separated directly. At last, to mitigate the impact of non-IoT devices, we make our model differentiate IoT and non-IoT and identify specific IoT devices based on multi-task learning.

In summary, the paper's contributions are: (1) We propose features that can differentiate different devices as much as possible. (2) We propose a model based on CNN (Convolutional Neural Network) and multi-task learning. It can differentiate IoT and non-IoT and the specific class of IoT devices with only a few labeled data. Besides, it makes the features after dimension reduction to form a single and compact cluster per class, mitigating the impact of feature fluctuation. (3) We evaluate our model on a public dataset and get accuracy over 99% with only 5% labeled data.

The rest of the paper is organized as follows: Section II discusses related work of IoT devices identification. Section III presents overview of the proposed method. Section IV illustrates the feature extraction. Section V presents the proposed method. Section VI evaluates the proposed model. At last section VII is the conclusions.

978-3-903176-31-7 © 2020 IFIP. This work is supported by the National Key Research and Development Program of China under Grant No. 2017YFB0803004, National Engineering Lab for Next Generation Internet Technologies (No. NGIT2019004) and Natural Science Basic Research Program of Shanxi (2020JM-710).

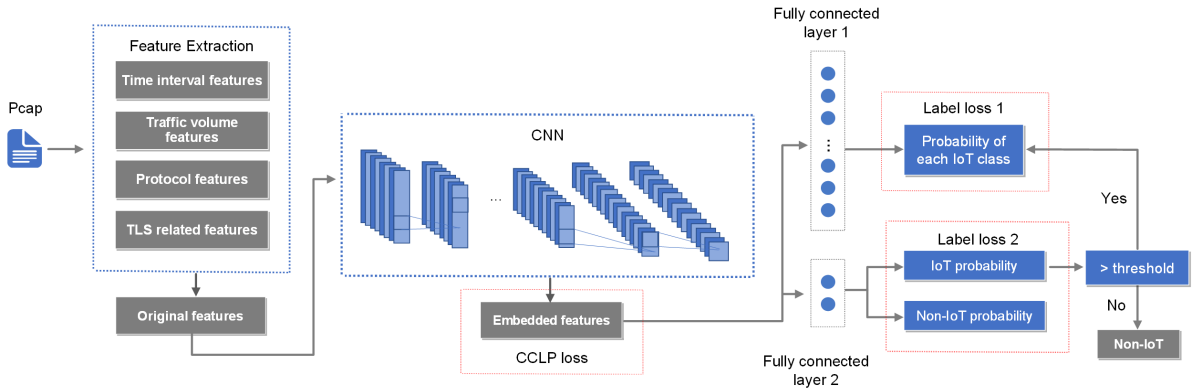

Fig. 1. The architecture of the proposed method.

II. Related work

Network traffic classification has been a topic of interest from the early stage of the Internet [17] and many researchers are focusing on general Internet traffic [18]–[21]. These years traffic analysis of IoT devices is becoming the research hotspot with the widespread IoT devices.

IoT device identification methods can be divided into active methods [11], [22]–[27] and passive methods [4]–[16], [28], [29]. Compared with active scanning, the passive method extracts IoT fingerprints from network traffic and does not impose a great impact on the network and devices. In this section, we mainly focus on passive methods. Passive methods can be classified into rule-based methods and machine learning-based methods.

A. Rule-based method

H. Guo et al. proposed an IoT device classification method that can be seen as the rule-based method [28]. The author purchased ten devices and observed their server IP or server name in DNS (Domain Name System) requests. Then they use these features to find new devices through network traffic. This method cannot be scaled and devices of the same manufacturer are thought to belong to the same class, which is not consistent with our goal.

B. Machine learning-based methods

A. Sivanathan et al. are the first to systematically profile, characterize and classify IoT devices in smart environments [4]. They use Random Forest to classify feature vectors extracted from network traffic. The features include time features, traffic volume features and protocol features. Through this method, they reached 95% accuracy classification.

After the work of A. Sivanathan [4], continuous works also extract time interval, traffic volume or protocol features from network traffic, and use machine learning models such as Adaptive Boosting (AdaBoost), K Nearest Neighbors (KNN) or random forest to get accuracy over 90% [5]–[10], [12], [30], [31].

Considering the powerful expression ability of neural networks, M. Lopez-Martin. [32] and J. Ortiz et al. [13] use neural network e.g. CNN, RNN to extract features and get accuracy over 95%.

Besides, there are also some works based on binary classifiers built for each class [14]–[16]. Their accuracy can range from 81.5% to 99.3%.

The references above are all based on supervised learning. They need numerous labeled data for training. There are also some works based on unsupervised learning [29], [33]. Samuel Marchal et al. proposed AUDI [33], which extracts periodic features from network traffic and uses KNN for clustering. Besides, A. Sivanathan [29] proposed an unsupervised one-class clustering method for each device based on PCA (Principal components analysis) and K-means. It solved the problem of model updating autonomously, but building model for each device has a relatively high cost and there are only 12 IoT devices in the evaluation.

In this paper, considering the disadvantages of supervised learning in large-scale IoT environment, we want to use a few labels to get high accuracy and propose an IoT device identification model based on semi-supervised learning.

III. Overview of the proposed method

In this section, we propose the method based on semi-supervised learning. The architecture of the method can be seen in Fig. 1.

Using “Pcap” files as input and extracting time interval features, traffic volume features, protocol features and TLS related features, we can get the original features used for classification. The original features are in high dimensional space. We need to transform the original features into lower-dimensional features. We use CNN to transform the original features into high-level features and realize dimension reduction.

Finally, we want to get the final results from the embedded features. To weaken the impact of non-IoT devices, we use multi-task learning [34]. So after CNN, the embedded features are fed into two fully connected layers respectively. The output of the fully connected layer 1 is the probability of each IoT class. The output of the fully connected layer 2 is the probability of IoT and non-IoT. If the probability of IoT

exceeds a threshold we set, we determine the IoT specific class according to the output of fully connected layer 1. Otherwise, we classify the corresponding traffic into a non-IoT device.

Considering feature fluctuation in the IoT environment, to improve the performance of our model, we adjust the parameters of the model according to two goals. One is making the output of the labeled data and the ground truth label consistent. This can be achieved by minimizing specific IoT class loss (Label loss 1) and IoT/non-IoT loss (Label loss 2). The other goal is to make the features after dimension reduction form a single and compact cluster per class, facilitating the classification of the two fully connected layers. The corresponding loss is called CCLP (compact clustering via label propagation) loss [35]. The detail is illustrated in Section V.

By minimizing label loss and CCLP loss, our model tends to compact the embedded features of the same class and make the prediction of labeled data consistent with its actual labels.

IV. Feature Extraction

Feature extraction is the first step of our method. The features extracted should differentiate different devices as much as possible. In this section, we propose four kinds of features and extract features of the traffic in 30 minutes. The 30 minutes is a time window.

A. Time interval features

In the traffic, we observed that different devices have different time intervals when sending or receiving packets. For each device, we extract time intervals between two adjacent packets in a time window and get a series of values $X = \{x_1, x_2, \ldots, x_n\}$. To mine its pattern sufficiently, we compute the statistical features such as minimum value, first quartile, second quartile, third quartile, maximum value, mean, variance, skewness [36] and kurtosis [37] of X. Fig. 2 shows the probability distribution of the maximum of time intervals for incoming traffic of three devices in dataset [9] as an example. Our subsequent probability distribution figures of features also use dataset [9] for presentation. From Fig. 2, The three devices have different characteristics in this feature. Besides, time intervals of some devices are more regular but others are not. To measure the disorder degree of time intervals, we compute the entropy of X. Considering the time interval is continuous, we divide the range between minimum and maximum into ten bins and count elements of X falling into each bin. We mark the counts of each bin as $c_1, c_2, \ldots, c_{10}$. Then the entropy can be expressed as:

$$entropy = -\sum_{i=1}^{10} p_i \log p_i \tag{1}$$

$$p_i = \frac{c_i}{\sum_{i=1}^{10} c_i} \tag{2}$$

Also, we observed that some devices send lots of packets in a short time after the idle stage, such as cameras, which makes time intervals change dramatically. To measure this pattern, we

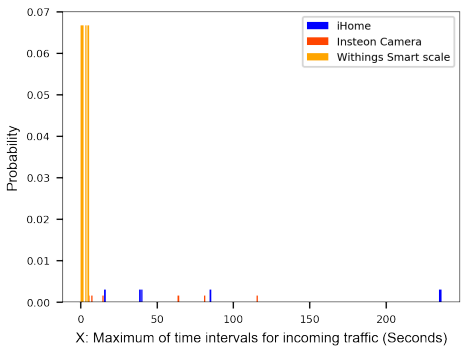

Fig. 2. The probability distribution of maximum of time intervals for incoming traffic.

use stationary, which is a fundamental feature of time series data. Here we compute the augmented Dickey-Fuller (ADF) test results for time intervals. ADF test is a common statistical test to evaluate whether a given time series is stationary or not. ADF tests the null hypothesis that a unit root is present in a time series sample [38]. The null hypothesis essentially asserts non-stationarity. Through ADF test, we can get P-value, ADF value, value at 1%, 5%, 10% level of significance leading to rejection of null hypothesis. Further, some time series may be non-stationary, but the difference of it may be stationary. As complementation, we also add ADF features of first-order difference, second-order difference of time intervals.

At last, considering different devices usually have different characteristics in NTP and DNS, we extract the minimum, the maximum, average value of NTP time intervals and DNS intervals, and compute the flow duration of a device as the complementation of time interval features.

For the above features, we compute the results of the incoming traffic, outgoing traffic and bidirectional traffic respectively.

B. Traffic volume features

Besides time intervals, we observed that the traffic volume of different devices also has its characteristics. For each device, we extract the packet length value in the time window and get a series of values $X = \{x_1, x_2, \ldots, x_n\}$. Like time interval features, we also compute the minimum value, first quartile, second quartile, third quartile, maximum value, mean, variance, skewness, kurtosis of X.

In addition, because some devices send large packets in a short time, which makes packet length change dramatically, we also use ADF test to evaluate the stationarity. We compute the ADF features of X, ADF features of the first-order difference of X, ADF features of the second-order difference of X. The detail has already been described in Section IV-A. Finally, we compute the flow rate, packet number, bytes, bytes/(packet number) of incoming traffic and outgoing traffic.

C. Protocol features

Different devices usually use different protocols in the network, transport, and application layers and have different

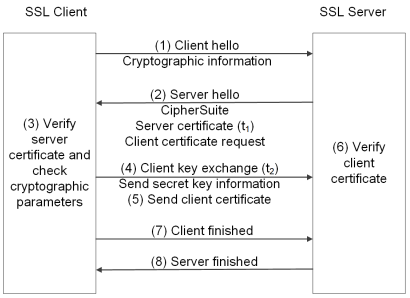

Fig. 3. TLS handshake process.

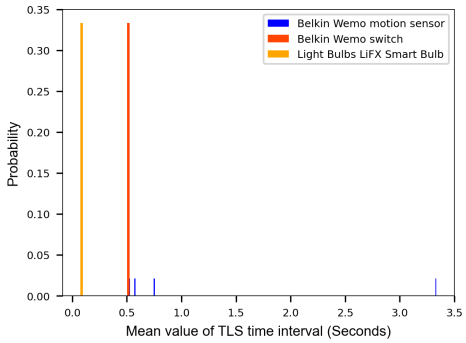

Fig. 4. The probability distribution of time difference related to TLS of devices.

protocol flags. Therefore, we extract the count of ICMP (Internet Control Message Protocol) packets, IPv4, IPv6, TCP (Transmission Control Protocol), UDP (User Datagram Protocol), NTP (Network Time Protocol), DNS, DHCP (Dynamic Host Configuration Protocol) packets of network traffic for each device as features. Also, we record the minimum, the maximum, average value of TTL (Time To Live), windows size and don't fragment (DF) ratio, which is the ratio of packets count whose DF is 1 and all packets count. In addition, we observed that different devices usually use different ports. So we split the port number range 0 to 65535 to 4 bins, 0-500, 500-1023, 1024-49151, 49152-65535 and record the occurrence of source port and destination port in each bin. Finally, considering domain names in DNS requests are usually distinguished between different devices, we extract the top ten domain names, record their occurrence for each device, compute the entropy of domain name occurance, and the number of DNS query packets as its features.

D. TLS related features

Many IoT devices use TLS/SSL protocol to communicate with their respective servers on the Internet [39]. The process of TLS handshake can be expressed as Fig. 3.

In TLS process, the client initiates the handshake by sending a "client hello" message to the server in step (1). After that, the server sends "server hello" and the server's SSL certificate to the client in step (2). In step (3), the client verifies the server's certificate and encrypts a random number using the public key stored in the server's certificate and sends it in step (4). In the dataset [9], the packets were captured at the gateway. We record the packet timestamp in step (2) that server sends its certificate to a device as t_1 and the packet timestamp in step (4) that client sends the encrypted number to the server as t_2 . During this period, the two packets only traverse the intranet and the IoT device needs to verify the server's certificate and encrypts a random number. IoT devices usually have simple hardware and software. Most of them are designed to finish certain tasks. So different devices usually have different $t_2 - t_1$. We use this time interval and TLS handshake count as features related to TLS. Some devices do not have TLS handshake in a time window. If there is not TLS handshake in a time window, we replace the time interval with -1 instead. Fig. 4 is the mean value of time interval related to TLS for three devices having TLS handshake. It can be seen that they have different characteristics in this feature.

V. Proposed Model

From feature extraction described in Section IV, we can derive the original features with 219 dimensions for devices. In this section, we will illustrate how to get the final device class through the model in detail.

As illustrated in Section III, our model consists of CNN and two fully connected layers as shown in Fig 1. CNN can transform the original features into lower-dimensional embedded features. Then the embedded features are sent into two fully connected layers to get the final result.

Our CNN model consists of six convolutional layers, two max-pooling layers and one average-pooling layer. It can transform the original features into lower-dimensional embedded features. Then the embedded features are sent into two fully connected layers. The fully connected layer 1 corresponds to specific IoT class and fully connected layer 2 has two neurons corresponding to IoT and non-IoT. Finally, if the IoT probability computed through fully connected layer 2 exceeds the threshold we set, the final class of the data is determined by the output of the fully connected layer 1. Otherwise, we think it corresponds to a non-IoT device. Considering the features of IoT devices may fluctuate during running, to make our model have good performance, we adjust the parameters of the model according to two goals. One is making the prediction of labeled data and the actual label consistent. We call the loss label loss. It includes specific IoT class loss and IoT/non-IoT loss. In Fig. 1, they are Label loss 1 and Label loss 2. Concurrently, the embedded features of the same class should be close to each other, facilitating the classification of the fully connected layers. To achieve this goal, we apply the method in [35]. The corresponding loss is called CCLP loss.

CCLP loss can be computed through the following process [35]. First, we sample a labeled batch $(\mathbf{X}_L, \mathbf{Y}_L)$ of size N_L and an unlabeled batch $\mathbf{X}_U$ of size N_U. $N_L \ll N_U$. Sending them into the CNN, we get the embedded features $\mathbf{Z}_L$ and $\mathbf{Z}_U$. Let $\mathbf{Z} = [\mathbf{Z}_L, \mathbf{Z}_U]^T$.

Then we can compute the transition matrix $\mathbf{H}$ through $\mathbf{Z}$. Elements of $\mathbf{H}$ that correspond to labeled samples L and

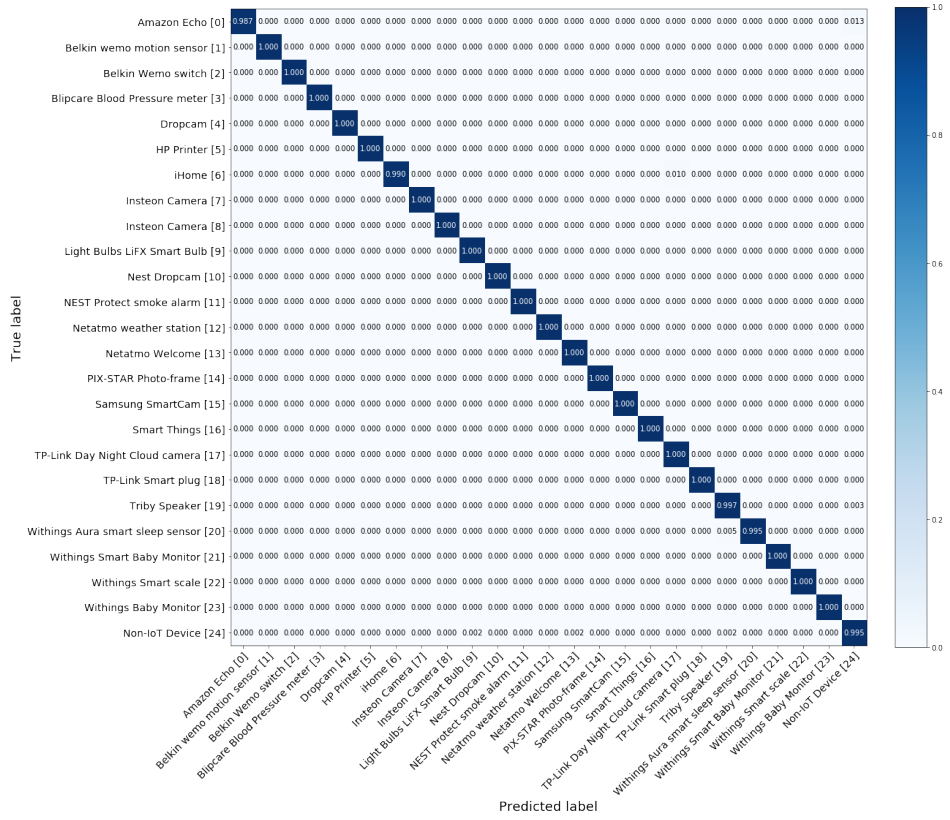

Fig. 5. Confusion matrix of the test set (8% labeled data, and the threshold is 0.7).

unlabeled samples U are arranged so that

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{LL} & \mathbf{H}_{UL} \\ \mathbf{H}_{LU} & \mathbf{H}_{UU} \end{bmatrix} \tag{3}$$

$H_{i,j}$ stands for the probability of a Markov process to start from data i and transits to data j in one step. We want $\mathbf{H}$ to approach an ideal transition matrix $\mathbf{T}$, which is the transition matrix computed through Label Propagation (LP) algorithm proposed by Kondor [40]. $\mathbf{T}$ can be computed through (4). C is the number of device classes.

$$T_{ij} = \sum_{c=1}^{C} \phi_{ic} \frac{\phi_{jc}}{m_c}, m_c = \sum_{i=1}^{N} \phi_{ic} \tag{4}$$

In (4), $\boldsymbol{\Phi} = [\mathbf{Y}_L, \boldsymbol{\Phi}_U]^T$. $\boldsymbol{\Phi}_U$ is the class posteriors for unlabeled data computed through Label Propagation (LP) algorithm [40], which is shown in (5).

$$\boldsymbol{\Phi}_U = (\mathbf{I} - \mathbf{H}_{UU})^{-1} \mathbf{H}_{UL} \mathbf{Y}_L \tag{5}$$

T_{ij} measures the transition probability between i and j. It encourages transition in the same class and discourages the transition between different classes. Then S step CCLP loss can be computed according to (6).

$$\mathcal{L}_{CCLP} = \frac{1}{S} \sum_{s=1}^{S} \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} -T_{ij} \log H_{ij}^{(s)} \tag{6}$$

$$\mathbf{H}^{(s)} = (\mathbf{H} \bullet \mathbf{M})^{s-1} \mathbf{H} \tag{7}$$

$$\mathbf{M} = \boldsymbol{\Phi} \boldsymbol{\Phi}^T \tag{8}$$

The smaller $\mathcal{L}_{CCLP}$ is, the more likely that data transits within the same class. Minimization of this loss can make the same class data compact to each other.

After getting CCLP loss and label loss, we do backpropagation to adjust the neurons' weight. Through label loss minimization and CCLP loss minimization, the model tends to compact the same class data, as well as making labels of labeled data and prediction consistent.

VI. Evaluation

For evaluation, we use the public dataset of [9]. In [9], the authors established a "smart environment" containing different IoT devices. They collected network traffic of 20 days in the environment and stored it in pcap files. In the traffic, we found 24 unique IoT devices and we will evaluate our model on this dataset. We extracted features illustrated in Section IV from the traffic for each device.

In the experiment, we divide the dataset into training set and testing set according to 7:3. In the training set, labeled data takes up 8%, which is selected randomly. Considering the data belonging to different classes is unbalanced, to get better performance, we do upsample for classes that have fewer labeled data from the randomly selected 8% labeled data. According to the proposed model in Section V, we can get the prediction of test set data. The confusion matrix of the test

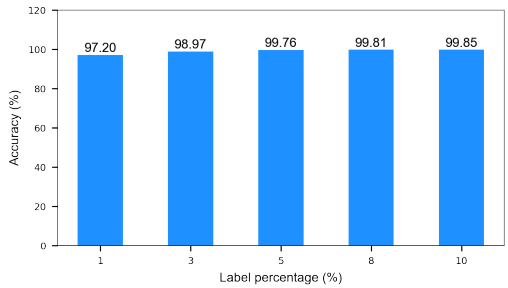

Fig. 6. Accuracy of different percentage of labeled data.

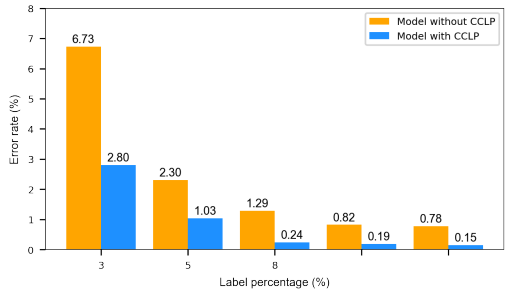

Fig. 7. Error rate comparison between the model with CCLP and model without CCLP under different label percentage.

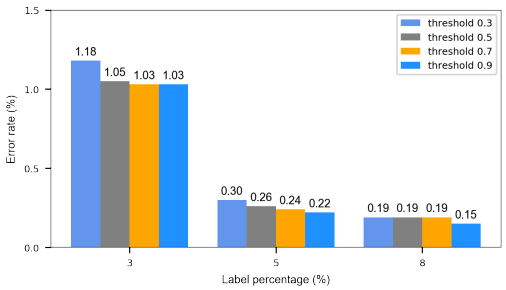

Fig. 8. Error rate of threshold 0.3, 0.5, 0.7, 0.9 under 3%, 5%, 8% label.

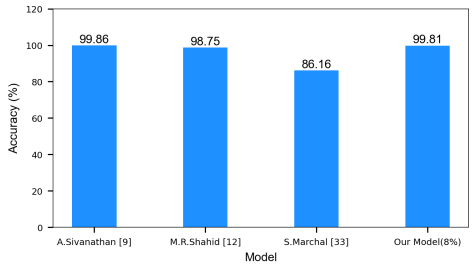

Fig. 9. Model comparison.

set is shown in Fig. 5. In the experiment, the threshold to 0.7. From Fig. 5, we find the proposed model can predict accurately for each device. The average accuracy reaches 99.81%. The lowest accuracy also reaches 98.7%, which belongs to Amazon Echo. This manifests the proposed model has high accuracy in each class.

We also test the accuracy under different percentages of labeled data. The result can be seen in Fig. 6. From Fig. 6, we find if the percentage of labeled data exceeds 8%, the accuracy promotion is not apparent. Even in the 1% labeled percentage, the accuracy can also reach 97.20%. It indicates that the features we extracted can differentiate different devices well and the model is robust.

Besides, to test performance promotion of CCLP, we compare the model with CCLP loss and the model without CCLP loss. The error rate comparison under different label percentages is shown in Fig. 7. It can be concluded that the model with CCLP has a lower error rate in each label percentage. The error rate reduces 3.93% under 1% label.

Considering different threshold may affect the prediction, we compare the error rate of the proposed model under different label percentages of different thresholds. If the probability of IoT is higher than the threshold, we then determine its specific IoT class. Otherwise, we will classify it as a non-IoT device. The comparison is shown in Fig. 8. From Fig. 8, we can conclude that a higher threshold can help decrease the error rate.

Finally, we compare the proposed model with supervised learning models, including A. Sivanathan [9], M. R. Shahid [12] and unsupervised learning method S. Marchal [33]. The outcome is shown in Fig. 9. From Fig. 9, the A. Sivanathan [9] has 99.86% accuracy. M. R. Shahid [12] has 98.75% accuracy, S. Marchal [33] has 86.16% accuracy and our proposed model has an accuracy of 99.81% under 8% label percentage and 0.7 threshold. We can conclude that the accuracy of the proposed model with 8% labeled data is higher than M. R. Shahid [12], S. Marchal [33] and comparable with A. Sivanathan [9]. That is to say, the proposed model with a few labeled data can get comparable accuracy compared with the supervised model.

VII. Conclusions

Considering supervised learning used in IoT device identification needs to collect numerous labeled data and time-consuming, in this paper, we proposed an IoT identification model based on semi-supervised learning. The method is based on CNN and multi-task learning, which can identify IoT/non-IoT as well as the specific device class. We evaluated the proposed model on a public IoT traffic dataset and get accuracy over 99% with only 5% labeled data. In the future, we will research IoT device identification with an unknown number of classes.

Acknowledgment

This work is supported by the National Key Research and Development Program of China under Grant No. 2017YFB0803004, National Engineering Lab for Next Generation Internet Technologies (No. NGIT2019004) and Natural Science Basic Research Program of Shanxi (2020JM-710).

REFERENCES

[1] M. A. Kuypers, T. Maillart, and E. Paté-Cornell, "An empirical analysis of cyber security incidents at a large organization," *Department of Management Science and Engineering, Stanford University, School of Information*, vol. 30, 2016.

[2] X. Fan, C. Xiang, C. Chen, X. Song, P. Yang, L. Gong, P. Nanda, and X. He, "Buildsensys: Reusing building sensing data for traffic prediction with cross-domain learning," *IEEE Transactions on Mobile Computing*, 2020.

[3] S. Notra, M. Siddiqi, H. H. Gharakheili, V. Sivaraman, and R. Boreli, "An experimental study of security and privacy risks with emerging household appliances," in *2014 IEEE conference on communications and network security*. IEEE, 2014, pp. 79–84.

[4] 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 *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2017, pp. 559–564.

[5] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Iotsense: Behavioral fingerprinting of iot devices," *arXiv preprint arXiv:1804.03852*, 2018.

[6] A. Bremler-Barr, H. Levy, and Z. Yakhini, "Iot or not: Identifying iot devices in a short time scale," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–9.

[7] S. A. Hamad, W. E. Zhang, Q. Z. Sheng, and S. Nepal, "Iot device identification via network-flow based fingerprinting and learning," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2019, pp. 103–111.

[8] N. Msadek, R. Soua, and T. Engel, "Iot device fingerprinting: Machine learning based encrypted traffic analysis," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2019, pp. 1–8.

[9] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying iot devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2018.

[10] M. Skowron, A. Janicki, and W. Mazurczyk, "Traffic fingerprinting attacks on internet of things using machine learning," *IEEE Access*, vol. 8, pp. 20 386–20 400, 2020.

[11] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "Gtid: A technique for physical device and device type fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 519–532, 2014.

[12] M. R. Shahid, G. Blanc, Z. Zhang, and H. Debar, "Iot devices recognition through network traffic analysis," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 5187–5192.

[13] J. Ortiz, C. Crawford, and F. Le, "Devicemien: network device behavior modeling for identifying unknown iot devices," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, 2019, pp. 106–117.

[14] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2177–2184.

[15] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. D. Guarnizo, and Y. Elovici, "Detection of unauthorized iot devices using machine learning techniques," *arXiv preprint arXiv:1709.04647*, 2017.

[16] Y. Song, Q. Huang, J. Yang, M. Fan, A. Hu, and Y. Jiang, "Iot device fingerprinting for relieving pressure in the access control," in *Proceedings of the ACM Turing Celebration Conference-China*, 2019, pp. 1–8.

[17] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Muller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1135–1156, 2013.

[18] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 2005, pp. 50–60.

[19] M. Iliofotou, M. Faloutsos, and M. Mitzenmacher, "Exploiting dynamicity in graph-based traffic analysis: techniques and applications," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, 2009, pp. 241–252.

[20] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli, "Revealing skype traffic: when randomness plays with you," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, 2007, pp. 37–48.

[21] R. Ferdous, R. L. Cigno, and A. Zorat, "On the use of svms to detect anomalies in a stream of sip messages," in *2012 11th International Conference on Machine Learning and Applications*, vol. 1. IEEE, 2012, pp. 592–597.

[22] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, 2005.

[23] S. Zander and S. J. Murdoch, "An improved clock-skew measurement technique for revealing hidden services." in *USENIX Security Symposium*, 2008, pp. 211–226.

[24] X. Feng, Q. Li, Q. Han, H. Zhu, Y. Liu, J. Cui, and L. Sun, "Active profiling of physical devices at internet scale," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2016, pp. 1–9.

[25] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 1093–1110.

[26] X. Feng, Q. Li, H. Wang, and L. Sun, "Acquisitional rule-based engine for discovering internet-of-things devices," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 327–341.

[27] K. Yang, Q. Li, and L. Sun, "Towards automatic fingerprinting of iot devices in the cyberspace," *Computer Networks*, vol. 148, pp. 318–327, 2019.

[28] H. Guo and J. Heidemann, "Detecting iot devices in the internet (extended)," *USC/ISI Technical Report ISI-TR-726 July*, 2018.

[29] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Inferring iot device types from network behavior using unsupervised clustering," in *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. IEEE, 2019, pp. 230–233.

[30] A. Aksoy and M. H. Gunes, "Automated iot device identification using network traffic," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[31] J. Bao, B. Hamdaoui, and W.-K. Wong, "Iot device type identification using hybrid deep learning approach for increased iot security," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 565–570.

[32] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[33] 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, 2019.

[34] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017.

[35] K. Kamnitsas, D. C. Castro, L. L. Folgoc, I. Walker, R. Tanno, D. Rueckert, B. Glocker, A. Criminisi, and A. Nori, "Semi-supervised learning via compact latent space clustering," *arXiv preprint arXiv:1806.02679*, 2018.

[36] D. P. Doane and L. E. Seward, "Measuring skewness: a forgotten statistic?" *Journal of statistics education*, vol. 19, no. 2, 2011.

[37] K. P. Balanda and H. MacGillivray, "Kurtosis: a critical review," *The American Statistician*, vol. 42, no. 2, pp. 111–119, 1988.

[38] R. I. Harris, "Testing for unit roots using the augmented dickey-fuller test: Some issues relating to the size, power and the lag structure of the test," *Economics letters*, vol. 38, no. 4, pp. 381–386, 1992.

[39] B. Anderson and D. McGrew, "Identifying encrypted malware traffic with contextual flow data," in *Proceedings of the 2016 ACM workshop on artificial intelligence and security*, 2016, pp. 35–46.

[40] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," 2002.