AutoML in IoT device classification

Chenghao Du
Department of Electrical and Computer Engineering
New York University

Introduction

Based on IoT Analysis' latest report, IoT devices grew from 3.6 billion in 2015 to 11.7 billion in 2020(Sinha, 2021). In the meantime, attacks on IoT devices become more and more often(Antonakakis *et al.*, 2017; Lyu *et al.*, 2017). Although researchers tried to identify IoT devices based on the network traffic,they focused on semantic representations in the network packets and have been trapped into feature engineering(Huang *et al.*, 2020; Feng *et al.*, 2018). Holland et al. designed a new method to automatically analyze network traffic. They implemented nPrint, a network packet standardization tool, and integrated with automated machine learning to eliminate feature engineering and model selection(Holland *et al.*, 2021). However, Holland didn't evaluate their model on any IoT data set. In this Deep Learning course project, the team wants to apply nPrintML on IoT network traffic data set and to answer the following question(Ren *et al.*, 2019):

Research Question

- What model(s) would have the highest accuracy for each test case?
- Which model has a better performance in terms of accuracy, training time, training set size?

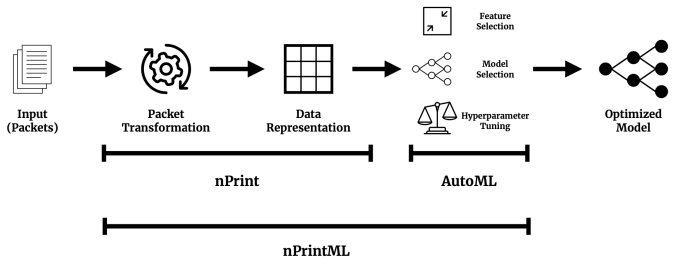

Figure 1: Model Structure

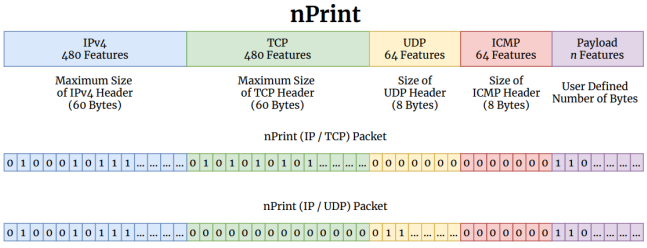

Figure 2: nPrint, the complete, aligned packet representation. Headers that do not exist in the packet being transformed are zero filled, while headers that exist but are not of maximum size are zero padded for alignment across nPrints.

Dataset

The original data collected by Ren's team in 2019(Ren *et al.*, 2019). They collected network traffic packets for 35 distinct IoT devices in England, and 46 distinct IoT devices in America. They also provided the devices' name and network condition for the data. So we assume that the data is well-labeled. For each device, they also designed different scenarios. For example, they collect Apple TV's network traffic under three situations: Power On/Off, Interact with Siri, and Browse Menu. Since we only need one pcap file for each device, our team write a simple script to select the first pcap file for each device. And instead of using the raw network packets data, we use nPrint to do prepossessing which can normalize the data packets to the same size(Holland *et al.*, 2020).

Model

As we mentioned in the previous section, the input will be normalized network packets like Figure 2. The output will be a predicted device name/type based on the test network packet. Figure 1 shows the general pipeline of this project. nPrinML adopts AutoGluon as the Auto Machine Learning tool.All tools can be downloaded on this page: https://nprint.github.io/.

Auto Machine Learning

Inside Auto machine Learning model, we decide to use three models: Random Forest, Extra Tree, and KNN(K Nearest Neighbors). And we use weighted ensemble to get the final prediction. Decision Tree is good at high dimensional features, but it usually overfits the data it is learning from because it learn from only one pathway of decisions.We decide to use Random Forest and Extra Tree to reduce the overfitting problem.

Result

We designed four experiments: Encrypted/Unencrypted Data prediction, Binary Classification between large/small data set, Binary Classification with/without MAC Address, and Multiple Devices Classification with different feature selection. All the experiments' data, model and code are included in this Github repository: https://github.com/serenitydu/nPrintML-on-IoT.

Encrypted/Unencrypted Data prediction

In this experiment, the input data contains 4921 network traffic data. We only use the IPv4 header part to train the model. For input X, any filed labeled with 1 means that the filed has been used in the IPv4 header. Any filed labeled with 0 means that the filed has not been used. And if that filed doesn't exit, nPrint will fill it with -1. For input Y, 1 means it is an encrypted data packet, and 0 means it is not in this data set. As the result showing in Figure 3, the accuracy is close 100%. But if we change the attention part from IPv4 header to Ipv6 header or Ethernet header, all the encrypted the data will be misclassified as unencrypted data. The reason is because these network packets never use the IPv6 field, and the input X only contains 0 and -1. Similarly, the Ethernet headers are either filled with all 0s or all 1s. So the Auto Machine Learning Model cannot make predict if we choose to two network keywords.

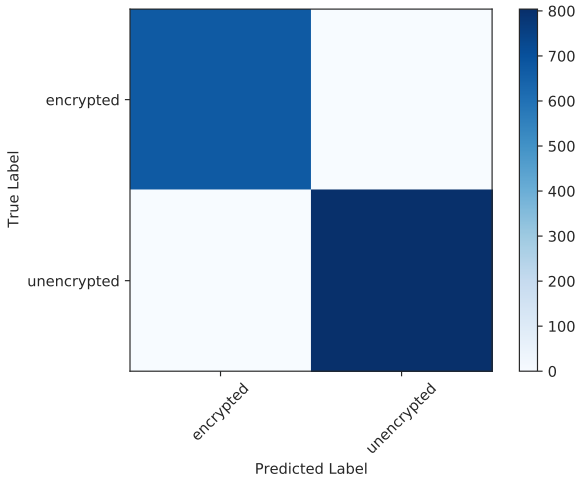

Figure 3: Encryption IPv4 Confusion Matrix

	Header	Feature Size	Time	Accuracy
Dataset A (Large)	Ethernet	38864	6m54.024s	100.00%
	IPv4	166560	NA	NA
Dataset B (Small)	Ethernet	1120	9.460s	100.00%
	IPv4	4800	18.043s	100.00%

Table 1: Binary Classification between Large and Small Data Set

	Header	Accuracy
Dataset B (Small with MAC)	Ethernet	100.00%
	IPv4	100.00%
Dataset C (Small without MAC)	Ethernet	60.00%
	IPv4	100.00%
Dataset D (Large without MAC)	Ethernet	79.00%
	IPv4	NA

Table 2: Binary Classification between Large/Small Data Set With/Without MAC Address

Binary Classification in large/small data set

In this experiment, we want use nPrintML to classify network traffic between Google Home and Amazon Echo Dot. Both of them are popular home assistants and have similar interactive functions. Dataset A contains 46 pcap files for Amazon Echo Dot and 47 pcap files for Google Home. The pcap files are captured every 15 seconds, and the numbers of packets in each pcap file are different. Dataset B contains 30 pcap files for both device and each pcap file has exactly 10 packets. Table 1 shows the result by using Ethernet header or IPv4 header as the feature. We notice that as the amount of packets increases, the feature size increases dramatically. This will cause the training time to increase exponentially. In conclusion, if user has fully access to the network traffic(e.g. MAC Address, IP Address, IP Port, etc.), using small data set can achieve the same prediction accuracy as using larger data set. And using small data set can save much more time in training than using large data set.

Binary Classification with/without MAC Address

In the second experiment, we assume that user has fully access to the network traffic. But in real world, most IoT devices are hidden behind NAT(Huang *et al.*, 2020). To simulate this situation, we remove the source and destination MAC Address in Dataset A and B to get the new Dataset C and D. The accuracy drops from 100% to 60% for small data set, and drops from 100% to 79% for large data set. Figure 4 shows that almost all the packets have been misclassified as Amazon Echo Dot. Figure 5 shows that some Echo Dot packets have been mis-classified as Google Home. nPrintML only uses Random Forest Model to make prediction for the second and the third experiment. In the feature, our team will add more Models in Binary Classification experiments. In conclusion, with limited access to the network traffic, using large data set can get a better result.

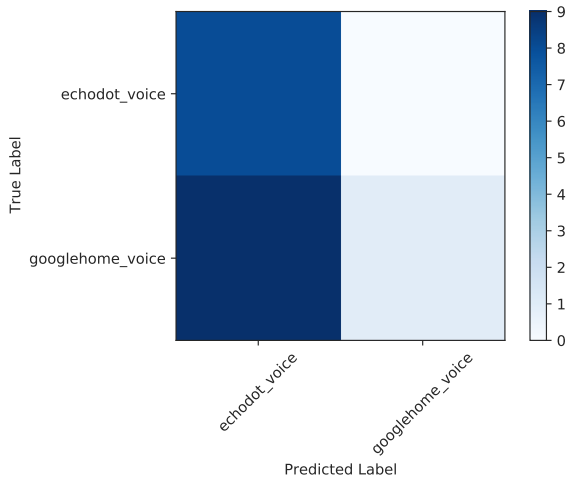

Figure 4: Binary Classification Small Data Set without MAC Confusion Matrix

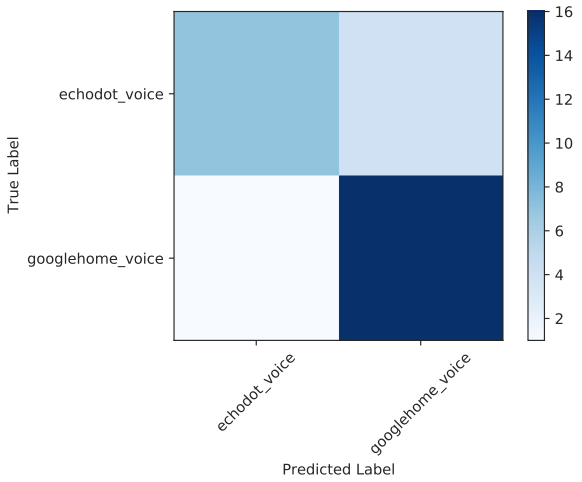

Figure 5: Binary Classification Large Data Set without MAC Confusion Matrix

Header	Feature Size	Average Accuracy
Ethernet	1120	100.00%
IPv4	4800	93.00%
Ethernet + IPv4	5920	100.00%
TCP	4800	92.50%
UDP	640	21.83%
TCP + UDP	5440	90.67%

Table 3: Multiple Devices Classification with Different Features

Multiple Devices Classification with different feature selection

In this experiment, we construct a data set with 6 different IoT devices. Four of them are smart TVs: Apple TV, Amazon Fire TV, Roku TV, and Samsung TV. Two of them are home assistants: Google Home and Amazon Echo Dot. Each device has 30 distinct pcap files and each pcap file contains 10 packets. Table 3 shows the accuracy with different feature selection. Although the accuracy reaches to 100% by using Ethernet header or Ethernet+IPv4 header, the main reason is because each device has a distinct MAC Address and the AutoML model will increase the weights for this feature in training. But these devices also have different IP address, Figure 6 shows that using IPv4 header as the feature only end up with 93% accuracy. This is because the IPv4 header contains much more sub-features(e.g Frag Offset, TTL, Protocol, etc.) than Ethernet header has, so that the IP address may not dominant the whole prediction process. Figure 7 is another example to prove that feature selection is important. In this case, most packets have been mis-classified as Roku TV. Although Roku TV is the the only device contains UDP packets in the training data set, the accuracy for this device only reaches 18% as showing in Figure 8.

Feature Work

In the feature, the team will test nPrintML on data sets without MAC Address and IP Address. And we will also apply nPrintML on real world IoT network traffic behind NAT.

References

[Antonakakis *et al.*, 2017] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. Understanding the mirai botnet. In *26th {USENIX} security symposium ({USENIX} Security 17)*, pages 1093–1110, 2017.

[Feng *et al.*, 2018] Xuan Feng, Qiang Li, Haining Wang, and Limin Sun. Acquisitional rule-based engine for discovering internet-of-things devices. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 327–341, 2018.

[Holland *et al.*, 2020] Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. nprint: A standard data representation for network traffic analysis. *arXiv preprint arXiv:2008.02695*, 2020.

[Holland *et al.*, 2021] Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. New directions in automated traffic analysis. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 3366–3383, New York, NY, USA, 2021. Association for Computing Machinery.

[Huang *et al.*, 2020] Danny Yuxing Huang, Noah Apthorpe, Frank Li, Gunes Acar, and Nick Feamster. Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(2):1–21, 2020.

[Lyu *et al.*, 2017] Minzhao Lyu, Dainel Sherratt, Arunan Sivanathan, Hassan Habibi Gharakheili, Adam Radford, and Vijay Sivaraman. Quantifying the reflective ddos attack ca-

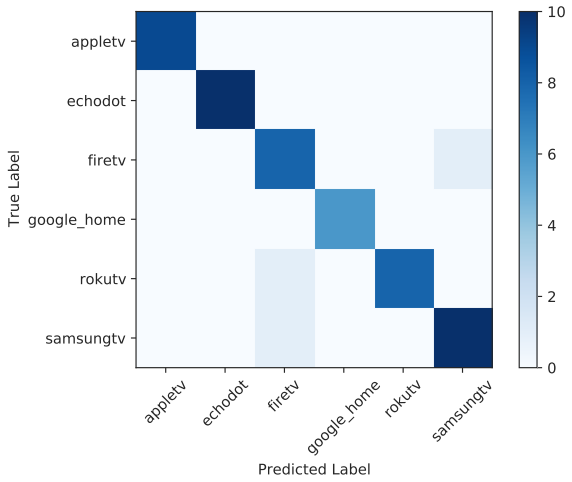

Figure 6: Multiple IoT Devices Classification IPv4 Confusion Matrix

pability of household iot devices. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 46–51, 2017.

[Ren *et al.*, 2019] Jingjing Ren, Daniel J Dubois, David Choffnes, Anna Maria Mandalari, Roman Kolcun, and Hamed Haddadi. Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach. In *Proceedings of the Internet Measurement Conference*, pages 267–279, 2019.

[Sinha, 2021] Satyajit Sinha. The rise of the iot semiconductor, Apr 2021.

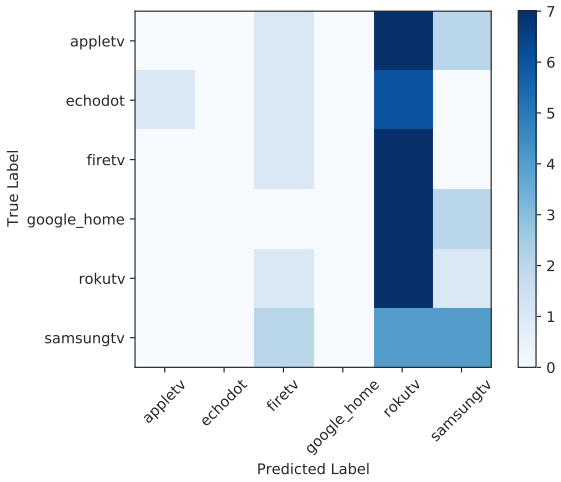

Figure 7: Multiple IoT Devices Classification UDP Confusion Matrix

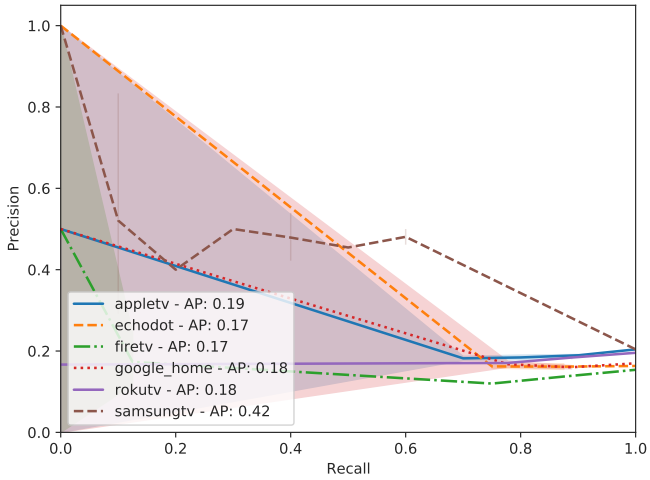

Figure 8: Multiple IoT Devices Classification Prediction Accuracy