Identification of IoT Devices by Means of Machine Learning Algorithms

Sirui Yi[1], Pengfei Gao[2], Xiang Wang[3] and Zhiang Hu[4]

[1]University of Wisconsin Madison

[2]Chongqing University of Technology

[3]Wuhan University of Science and Technology

[4]Wuhan University of Science and Technology

Keywords: Internet of Things; Machine Learning; Smart Home; Device Identification

Abstract: The rapid development of Internet of Things (IoT) not only brings lots of benefits and convenience to people, but also poses many problems and threats to people's daily life. Among many challenges dealing with those problems, the most significant one is that it is difficult for people and even organizations to identify the types of devices connected to their internet. Under such circumstances, this research was aimed at using several machine learning (ML) algorithms, including Random forests, K-nearest neighbors (KNN), and Adaboosting, to address this challenge. This research mainly concentrated on the 46 kinds of pcap files of different devices in 1GB dataset provided by Professor Nick Feamster from the University of Chicago. To begin with, each pcap file was read as dataframe, then labeled and finally saved as csv files. Then all the data were divided into two categories: trainset (80%) and test set (20%). Finally, the method of grid research was used to select the best combination of features and parameters of each ML algorithm to achieve the best performance. Overall, the trainset and test set accuracy of our IoT device identification model is up to 99.9% and 99.2%, respectively.

1. Introduction

Computer and network communication technology have been developing and progressing by leaps and bounds over the past few decades. Inevitably, an increasing number of network related industries and applications have emerged, and the Internet of things (IoT) has therefore come into being. Internet of things (IoT) is a technology that allows the connection and communication of various physical things in our daily life by means of wireless internet. This enables the exchange and transmission of information and data between objects and external environment and gives individuals instant access to these information and data [1], thus permitting individuals to perform effective remote monitoring and control over these objects. With the help of internet, IoT devices could perform tasks individually without human intervention, which greatly facilitates people's lives [2] as well as promotes the working efficiency.

However, there are some deficiencies correlated with IoT. One of the biggest drawbacks is that the types of IoT devices connected to the network might not be even known by their organizations in the future [3]. Fortunately, there are many differences in network traffic between IoT devices and non-IoT devices such as the way of communication and the patterns of network traffic [4]. Under such circumstances, this research was aimed at making use of 3 types of supervised learning algorithms, including Random forests, K-nearest neighbors (KNN), and Adaboosting, to identify what types of devices are connected to a network by analyzing the network traffic data.

It is investigated that only few research studies had focused on using machine learning algorithms for IoT device identification. In addition, among those studies, very few of them used more than one ML algorithm and achieved good accuracy; in contrast, our research group used three types of algorithms and the overall accuracy is satisfying.

The remaining part of this paper is organized as follows: Section 3 introduces the processing of data; Section 4 introduces the materials and methods of this experiment; Section 4 presents the experimental results and analysis of their results; Sections 6 is the conclusion.

2. Processing of Data

Firstly, all the data in .pcap files are re-written into .csv files for usage. Then, because for some of the devices, the amount of data is much less than those for other devices, we decide not to use all data for our project; instead, we randomly select 50 .csv files from the folders for each device if the total number of .csv files is larger than or equal to 75, and randomly select 50 rows of data from each .csv files chosen if the total number of rows in it is larger than or equal to 75. So, the maximum possible number of samples we would choose from a folder is 74*74 = 5476. In practice, the maximum number of samples we choose from a single folder is less than 3000. Since the smallest dataset in all kinds of device holds a sample size of 491, and most of the datasets holds a sample size between 2000 and 3000, we could assume that the data are uniformly distributed, which would not suffer from skewed classification. Finally, training sets and testing sets are split in each dataset, but not in the aggregated dataset, which would prevent from a high variance or bias by skewed classification again.

3. Materials and Methods

In this work, three ML algorithms were used to distinguish IoT devices from other devices: Random Forests, K-nearest neighbor (KNN) and Adaptive boosting.

3.1 Random Forests

Random Forests uses randomly selected subsets of features at each node in each tree, which could be regarded as bagging inside trees. Also, since bootstrap samples are fitted to the decision trees inside random forests, it makes sure that the training sets are not overlapping [5]. The significant parameters in random forests are:

1. Bootstrapped or not
2. The maximum depth of each tree
3. The minimum size of leaf nodes
4. The number of estimators

For the grid search, the value or range for each of the parameters listed above are:

1. Bootstrap: True
2. Max_depth: select from 15 to 29
3. Min_samples_leaf: select from 5 and 10
4. N_estimator: select from 30, 100 and 300

3.2 K-nearest Neighbor

In KNN algorithm, the selected neighbors are all correctly classified objects. In the decision-making of classification, this method only depends on the category of the nearest sample or samples to determine the category of the samples to be classified [5]. The important parameter in KNN is the number of neighbors, k. For the grid search, the range for the number of neighbors is selected from the odd numbers from 3 to 29.

3.3 Adaptive Boosting

The basic concept of Adaptive Boosting is to boost many “weak learners” to “strong learners”. Initially, it has a weight vector, where the elements inside the vectors sum to 1 and are all equal to each other. Then, the misclassified training examples would gain more weights, while the correctly classified examples would be less weighted. After a preset number of rounds, Adaptive Boosting predicts the result by majority voting on trained classifiers [5]. The parameters we care about in Adaptive Boosting are:

1. Number of estimators
2. Learning rate
3. Algorithm

For the grid search, the value or ranges for each of the parameters listed above are:

1. N-estimators: 100

2. Learning rate: select from 1, 1.5, 2, and 2.5
3. Algorithm: SAMME (for classification)

3.4 Grid Search Method

This method is designed for hyper-parameter optimization. It uses exhaustive searching by fitting models with all combinations of values in specified parameters and for the parameters not specified, it uses the default value. It has to be performed under performance metric such as cross validation on the training set [7]. In this project, the parameters in grid search are set as follow for each of the three models described before:

1. Number of folds for cross validation set to 10
2. Number of jobs set to -1
3. Verbose set to 2

3.5 Sequential Feature Selection

This method is designed for selecting the best subset of features that could be used to predict the labels most accurately. It uses the idea of greedy search, and it could reduce a feature set that initially has high dimension to that of a lower dimension. The selected features are considered to be the most relevant to the training data [5]. In this project, we would first do pipelining with the best parameters selected from Grid Search method and use the pipeline model to do sequential forward selection. The parameters in sequential forward selection are set as follow for each of the three models described before:

1. Number of feature set to the total number of features, 7
2. Forward set to True
3. Floating set to False (not using SFFS here)
4. Verbose set to 2
5. Scoring method set to accuracy
6. Number of folds for cross validation set to 10

3.6 A Brief Description for the Whole Process

For each of the three algorithms (Random Forests, KNN, Adaptive Boosting), Grid Search is performed to choose the best hyper-parameter for each of them respectively; then, using the hyper-parameters selected, pipeline models are created for sequential forward selection, which is used to get the best subset of features for each model. At last, the training set accuracies and testing set accuracies are calculated using the models using the best hyper-parameters and the best subset of features. Finally, the best model is chosen after considering overfitting in the models.

4. Results and Discussion

4.1 Accuracy

Due to the setting of random state, data for training and testing were randomly chosen each time. As a result, although different accuracies for each model might be achieved every time we ran the code, the overall difference is relatively small. It could be clearly seen from Table 1 that the overall result of this research is very successful with the accuracy of each algorithm over 98%. The K nearest neighbor classifier performed very good results with the accuracy up to 97.98%, suggesting that the sample set has good typicality. The random forests classifier performed even better, achieving an accuracy of 98.13%. Adaboosting classifier performed the best, which achieved accuracy over 99.17%. It is noteworthy that the difference between trainset and test set for each algorithm is not very significant, indicating there is no obvious overfitting in our models.

Table 1 Accuracy for Random Forests, KNN and Adaptive Boosting on test data

Algorithm Type	Testing accuracy (%)	Training accuracy (%)
Random Forest	98.13	98.78
K-nearest neighbor	97.98	98.92

Adaptive Boosting	99.17	99.99

4.2 Selection of Hyper-parameters and Subsets of Features

The accuracy of the model depends on a very significant factor: parameters of the ML algorithm. In order to select the best parameters, the method of grid research was used. The parameters chosen for each algorithm and corresponding scores are shown in table 2.

Table 2 Accuracy for Random Forests, KNN and Adaptive Boosting on test data

Algorithm Type	Parameters setting	Score (%)
Random Forest	'bootstrap': True, 'max_depth': 27, 'min_samples_leaf': 5, 'n_estimators': 200	97.76
K-nearest neighbor	'n_neighbors'=3	97.77
Adaptive Boosting	'algorithm': 'SAMME', 'learning_rate': 2, 'n_estimators': 100	99.12

Another significant factor that may influence the result of the research is the number and the type of features selected in the model. There are in total 20 features for selection; nevertheless, in order to avoid useless work or unreasonable feature selection, we focused on 7 significant features in this project. The index and names of significant features correspond as follows (For simplicity, indices are used subsequently to represent the selected features). The relationship between the performance and the number and the type of features selected is shown in Table 4.

Table 3 Index and the corresponding feature name in Dataframe

Index	Feature Name
0	ip_dst_int
1	ip_src_int
2	length
3	port_dst
4	port_src
5	time
6	protocol_new

Table 4 Relationship between the performance and the number and the type of features selected.

Algorithm Type	Feature_idx	Average score (%)
Random Forest	(5)	94.71
	(3,5)	95.19
	(3,4,5)	96.91
	(1,3,4,5)	97.49
	(0,1,3,4,5)	97.69
	(0,1,2,3,4,5)	97.54
	(0,1,2,3,4,5,6)	97.30
K-nearest neighbor	(5)	95.08
	(5,6)	96.36
	(0,5,6)	96.31
	(0,1,5,6)	97.63
	(0,1,3,5,6)	96.93
	(0,1,3,4,5,6)	96.87
	(0,1,2,3,4,5,6)	94.19
Adaptive Boosting	(5)	95.53
	(3,5)	98.21

	(3,4,5)	98.73
	(1,3,4,5)	98.93
	(0,1,3,4,5)	99.01
	(0,1,3,4,5,6)	99.05
	(0,1,2,3,4,5,6)	99.02

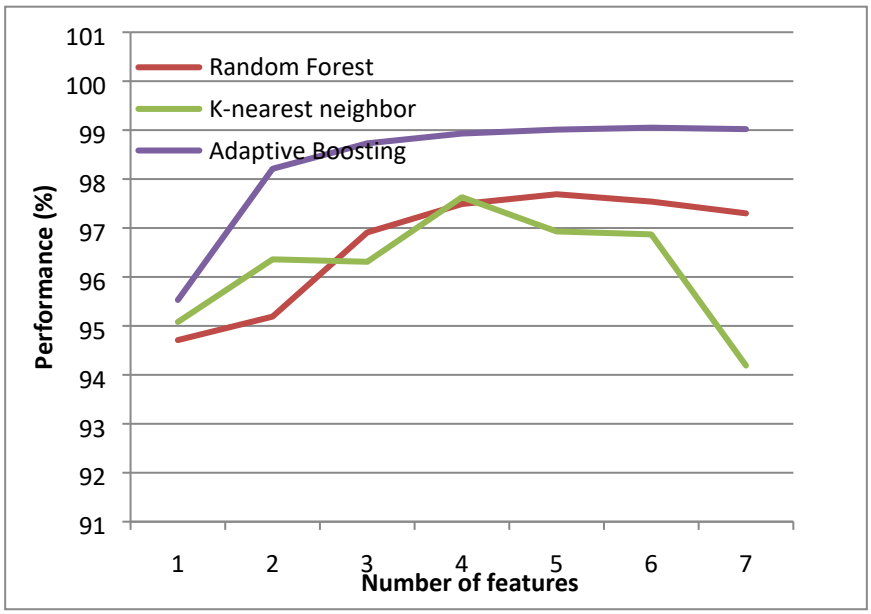

Figure 1 Relationship between number of features and performance for three ML algorithms

It could be clearly seen from Table 4 that the random forests algorithm reached its best performance of 97.69% when features 0, 1, 3, 4, 5 were selected. K-nearest neighbor classifier achieved its best accuracy of 97.63% when selecting features 0, 1, 5 and 6. For Adaboosting, it reaches the best accuracy of 99.05% if all the 7 features were chosen. It could be seen that the performance is largely influenced by the subset of features selected and though intuitively, the more features selected, the higher accuracy, but in actual practice, we could see that sometimes the most optimal result comes not from the whole set of features. Therefore, it seems that it is significant and practical to carefully choose the type and the number of features for each model to enhance the performance of IoT device identification in reality.

It is inspired by the research result that by using these three basic ML algorithms with low-dimensional features and a medium-sized dataset, the differentiation between IoT devices from other devices is already accurate enough. Therefore, we could imagine if other machine learning algorithms such as Support Vector Machines or Ordinal Logistic Regression were applied, or a bigger dataset was used, the accuracy may be further promoted.

4.3 Choice of Model

According to the above analysis, it is decided that Adaptive Boosting is the best model for this project.

5. Conclusions

Though it seems good that the Adaptive Boosting would get a very high accuracy on both training set and testing set and the accuracy difference is very low, there is still chance that it suffers from overfitting since the training set accuracy of which is 99.99%, which is very close to 1. This indicates an optimistic bias and might cause the future data to fit bad on it. For the unseen data in this project, which is the testing set, no pessimistic bias is observed, which is good, but it could not account for the variance in the training set, so that when this model is generalized for future data, the generalization error might be large.

References

[1] Bandyopadhyay, D. & Sen, J. Wireless Pers Commun (2011) 58: 49.

[2] Burhan, M., Rehman, R., Khan, B. and Kim, B. (2018). IoT Elements, Layered Architectures and Security Issues: A Comprehensive Survey. Sensors, 18(9), p.2796.

[3] Meidan, Y., Bohadana,M. Shabtai, A. ProfilIoT: A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis.

[4] Doshi, R., Apthorpe, N., Feamster, N. (2018). Machine Learning DDoS Detection for Consumer Internet of Things Devices. Available from:

[5] Sebastian, S. (2015). Python Machine Learning, 1st Edition.