IoT Botnet Malware Classification Using Weka Tool and Scikit-learn Machine Learning

Susanto
Faculty of Computer Universitas Bina Insan/Faculty of Engeenering Universitas Sriwijaya
Lubuklinggau, Indonesia
susanto@univbinainsan.ac.id

Deris Stiawan*
Faculty of Computer Science Universitas Sriwijaya
Palembang, Indonesia
deris@unsri.ac.id

M. Agus Syamsul Arifin
Faculty of Computer Universitas Bina Insan/Faculty of Engeenering Universitas Sriwijaya
Lubuklinggau, Indonesia
mas.arifin@univbinainsan.ac.id

Mohd. Yazid Idris
School of Computing, Universiti Teknologi Malaysia,
yazid@cs.utm.my

Rahmat Budiarto
College of Computer Science and IT, AlBaha University, Saudi Arabia
rahmat@bu.edu.sa

***Abstract*— Botnet is one of the threats to internet network security—Botmaster in carrying out attacks on the network by relying on communication on network traffic. Internet of Things (IoT) network infrastructure consists of devices that are inexpensive, low-power, always-on, always connected to the network, and are inconspicuous and have ubiquity and inconspicuousness characteristics so that these characteristics make IoT devices an attractive target for botnet malware attacks. In identifying whether packet traffic is a malware attack or not, one can use machine learning classification methods. By using Weka and Scikit-learn analysis tools machine learning, this paper implements four machine learning algorithms, i.e.: AdaBoost, Decision Tree, Random Forest, and Naïve Bayes. Then experiments are conducted to measure the performance of the four algorithms in terms of accuracy, execution time, and false positive rate (FPR). Experiment results show that the Weka tool provides more accurate and efficient classification methods. However, in false positive rate, the use of Scikit-learn provides better results.**

Keywords—classification, botnet IoT, Weka, scikit-learn, machine learning

I. Introduction

Botnet is one of the threats to internet network security [1]. Basically, "Botnet" is constructed by two terms, "Bot" for robot and "Net" for the network. Malware code is installed on computers in a network, then this computer network can be controlled by Bot-master remotely through the execution of several commands that threaten the whole computer network [2]. Botmaster carries out attacks on the network by relying on communication on the network traffic [3]. Internet of Things (IoT) network infrastructure consists of devices that are inexpensive, low-power, always-on, always connected to the network, and are inconspicuous and have ubiquity and inconspicuousness characteristics so that these characteristics make IoT devices an attractive target for botnet malware attacks [4]. Authors in [5] report that attackers use the IoT tool as part of a malware network. In 2014 it was discovered a botnet spam network sent more than 700,000 spam emails. Then in September 2016, there was an attack on Brian Krebs (krebsonsecurity.com) security blog from the IoT botnet (Mirai malware). The attack reaches 600Gbps to get access, especially to home routers, network-capable cameras, and digital video recorders, which usually have less protection than other consumer's IoT devices. In the same month, the Mirai-based attack on the French WebHost OVH broke the record for the most significant DDoS attack recorded at least 1.1 Tbps, and possibly as massive as 1.5 Tbps [6].

Since the IoT botnet network is developing rapidly and attacks are evolving to bypass existing detection systems, thus, a proper and intelligent solution to overcome the problem is required [7]. Recognizing whether an incoming packet on network traffic is a malware/attack or not, one can use machine learning-based classification methods [8], more specifically, the authors in [3] state that machine learning can be used for botnet malware classification process. Aman et al [9] support the statement. The authors analyze a large number of malware samples after malware detection, additional efforts are required to classify the malware into groups.

In this paper, the IoT botnet malware classification process uses four machine learning algorithms, namely Adaboost, Decision Tree, Random Forest, and Naïve Bayes. The authors conduct experiments to measure the performance of the algorithms in terms of the level of accuracy, execution time, and false positive rate (FPR) using the Weka and Scikit-learn analysis tools. The rest of this paper is structured as follows. In Section 2, we present and summarize papers related to machine learning tools for classification. In Section 3, we explain the proposed method, and Section 4 presents the results of the experiment and a brief discussion. Section 5 presents the paper's conclusions and discussion of further IoT botnet malware research.

II. Literature Review

Analysis tools in the process of classifying data using machine learning are quite numerous. Mahajan et al. [10] analyze malware samples and conclude that the malware detection process requires additional efforts to classify them, in the classification process using two machine learning tools namely Knime and Orange. In their experiments, the authors compare the results of malware classification using the Decision Tree classification method, Naïve Bayes, k-nearest neighbors, Random Forest, Support Vector Machine, and Neural Network. The results exhibit a comparison of the values of the confusion matrix, accuracy, and Cohen Kappa matrices. Then, research in [11] uses the Shogun toolbox in malware classification using the SVM method. Performance in term of accuracy and confusion matrix are discussed. Researchers in [12] use the Weka tool in botnets classification using J48 and Random Forest classification methods with experimental results presenting accuracy and false positive rate data. In [13], researchers use the Weka tool in classification malware using Ibk, Naïve Bayes, Support Vector Machine, and Decision Tree classification methods and evaluate the ROC curve and false positive rate. Then, research in [14] detects malware using the Weka tool. classification methods used are J48, J48graff, LADTree, NBTree, Random forest and Reptree. The experimental results present execution time, accuracy and false positive rate. Researchers in [15] analyze the memory and CPU usage to detect malware using the Weka tool with the Naive Bayes classification algorithm, Logistic Regression, and J48 Decision Tree. the experimental results display precision, recall, F-measure, and the number of features. The author in [16] uses the Weka tool in classification malware using Random Forest classification method with experimental results presenting detection rate, precision, F-measure, and time complexity. In [17], researchers detect malware in embedded systems using the Weka tool and Support Vector Machine, Bayesian Network, Neural Network, Decision Tree, and Rule-Based classification algorithms. the experimental results display execution time and accuracy. Researchers in [18] detect malware in large scale traffic using the Weka tool and Support Vector Machine, and Logistic Regression classification algorithms. The experimental results consider the false positive rate, true positive rate, and accuracy. Run-time malware detection on hardware using the Weka tool and Bayesian Network classification algorithm, J48, Jrip, MLP, Oner, Reptree, SGD, and SMO classification algorithms was carried out by researchers in [19]. Experimental results showing latency, accuracy, and overhead area. Identifying Botnet-IoT attacks on traffic for the internet of things smart city uses machine learning with the Weka tool and Bayesian Network, C4.5, Naive Bayes, Random Forest, and Random Tree classification algorithms was conducted by researchers in [20]. The experimental results provide accuracy, precision, recall, true positive rate, and execution time[19]. In [20], authors analyze device behavior, from device CPU usage and temperature to memory consumption, to detect IoT botnets using Python's Scikit-learn library. The classification algorithms used are Elliptic Envelope, Isolation Forest, Local Outlier Factor, and One-Class Support Vector Machine (OSVM). The experimental results present precision, recall, specificity, accuracy, AUC, and F1-score. In [21], researchers conduct malware detection using a large set of datasets in identifying malware variants and classifying them using k-NN method, Scikit-learn and Python library. The experimental results present a true positive rate, a false positive rate and an ROC. Lastly, researchers in [22], perform malware detection on executable files using the Scikit-learn and Python library with Random Forest classification algorithms, Xboost, decision tree, k-NN, and Neural Network. The experimental results include accuracy and execution time.

III. Proposed Method

A. Dataset and Tools

The dataset N_BaIoT used in this study has an unbalanced class distribution of traffic and attacks captured from a testbed network traffic. Nine IoT devices are attached to the network, i.e.: two doorbells, thermostats, baby monitors, four security cameras, and webcams. [23]. The dataset consists of normal traffic and attack traffic, i.e.: Mirai and Bashlite [24], which has 115 features including five main features namely channel, host-MAC & IP, network-jitter, host-IP, and socket. The traffic is extracted from IoT testbed network traffic based on five-time windows, i.e.: 1min, 10sec, 1.5 sec, 500 ms, and 100 ms. Besides, this dataset has a packet size, of outbound with a statistical mean and variance; count packages with a statistical value; jitter packages with statistical mean, variant, and value; and packet size of both inbound and outbound with statistical correlation coefficient, covariance, magnitude, and radius. The N_BaIoT has a dataset of 555,932 Benign, 2,838,272 Bashlites and 3,668,420 Mirai records. In the classification experiment using Weka and Scikit-learn the same amount of data was used. The researchers only used 20% of the N_BaIoT dataset that was taken flat each class from each device.

The experiments are run on a computer with Intel Core i7-9750H, 2.6 GHz processor with 16GB RAM, and Windows 10 64-bit OS.

B. Machine Learning Tools

1) Scikit-learn

Scikit-learn is an open-source library in Python [25]. Scikit learn library can be used for data processing, dimensionality reduction, classification, regression, clustering, and model selection with the evaluation results can be in the form of execution time, accuracy, confusion matrix, false positive rate, false negative rate, precision, recall, and others. In this experiment, the machine learning tools used were Scikit-learn 0.22.1 for the classification of IoT botnet malware.

2) Weka

Weka is an open-source software created by Waikato University, New Zealand [26]. Weka is a machine learning tool that can be used for data processing, visualization, classification, regression, clustering, and feature selection with evaluation results in the form of execution time, accuracy, confusion matrix, false positive rate, false negative rate, precision, recall, and others. In this experiment, the machine learning tool used was Weka 3.8.4 for the classification of IoT botnet malware.

C. Classification Method

The classification method used to perform the analysis using Weka and Scikit-learn tools is Adaboost, Decision Tree, Random Forest, and Naïve Bayes.

1) Random forest

Random forest is an ensemble learning method used for classification and regression [27]. The random forest has advantages: a low number of control parameters and models; resistance to over-installation; there are no requirements for feature selection because they can use a large number of potential attributes. Besides, random graphics also have some disadvantages, such as low model interpretability, loss of performance due to related variables, and dependence on random generators from implementation [3].

2) Decision Tree

Decision trees are tree-like structures that have leaves, which represent classifications and branches, which in turn represent the conjunctions of features that lead to that classification. The advantage of decision tree classification is the expression of intuitive knowledge, high classification accuracy, and simple implementation. The main disadvantage is that for data, including categorical variables with some different levels, the acquisition value of information tends to support features with more levels [3].

3) Naïve Bayes

The Naive Bayes algorithm performs classification tasks in the field of machine learning. It can classify very well on datasets even though it has large records with multi-class and binary class problems. The naïve bayes algorithm has the advantages of being simple, fast, and measurable. It can be used for continuous and binary values, and multi-nomial distributed attributes. It can be built with a very simple model for small and large datasets. For attributes that are not relevant are also not sensitive. On the other hand, Naïve Bayes classifier has the disadvantage of being unable to find relationships between attributes because all attributes are considered irrelevant; There is a possibility of a "zero conditional probability problem" if the attribute class has zero frequency data items; That is not suitable for regression problems [28].

4) Adaboost

Adaptive Boosting (AdaBoost) is one of the most popular algorithms used to reduce over-fitting problems, inherent in machine learning [3]. AdaBoost provides a very simple and useful method for generating classifications. Have a performance that depends on the diversity among classification classes as well as the performance of each classification class. The existing AdaBoost algorithm focuses on the problem of error minimization [29].

IV. Result

In this paper, the authors use Weka and Scikit-learn tools for the classification of IoT botnet malware. Evaluation results are analyzed using parameters of accuracy, execution time, and false positive rate.

A. Scikit-learn

1) Accuracy

By using Scikit-learn, the highest accuracy in IoT botnet malware classification with random forest classification method achieves an accuracy of 99.99%. Then, the Adaboost classification method achieves an accuracy of 99.92%, while the Decision Tree classification method has an accuracy of 98.53 %. The lowest accuracy is achieved by the use of naïve bayes classification method with an accuracy value of 82.35%. A comparison of accuracy using scikit-learn is presented in Figure 1.

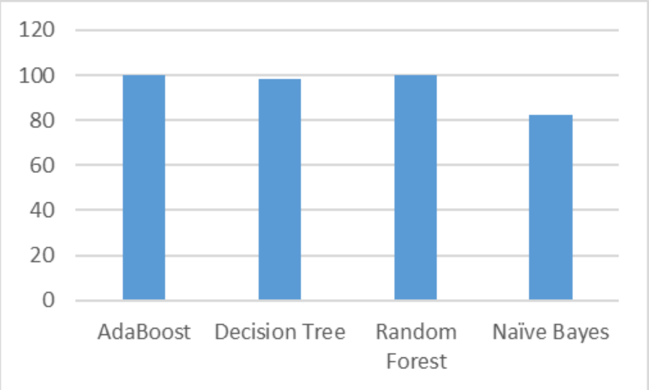

Fig. 1. Comparison of classification accuracy using scikit-learn

2) Execution Time

For Scikit-learn machine learning, the fastest execution time in classifying malware is achieved by the use of decision tree classification method with an execution time value reaching 49.88s, followed by naïve bayes classification method with an execution time of 102s. Next, is adaboost classification method, which has an execution time of 1079.86s. The longest execution time is for random forest classification method, which is 1912s execution time. A comparison of accuracy using scikit-learn is presented in Figure 2.

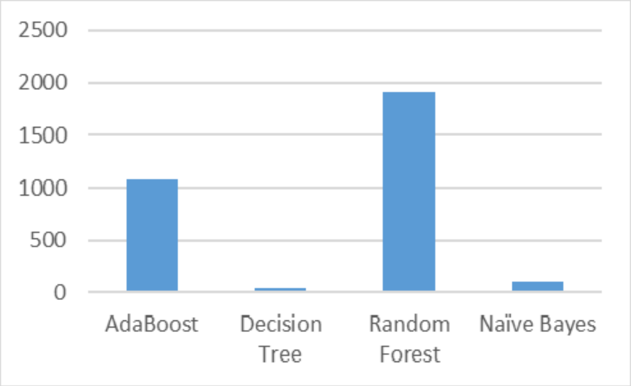

Fig. 2. Comparison of execution time classifications using Scikit-learn

3) False Positive Rate

Furthermore, Scikit-learn machine learning gives the lowest FPR value in classifying IoT botnet malware when we use random Forest classification method where the FPR value reaches 0, followed by adaboost classification method with an FPR of 0.0001 and then decision tree classification method has an accuracy of 0.0008. The highest value of FPR is for naïve bayes classification method with an accuracy value of 0.0014. A comparison of FPR using Scikit-learn is presented in figure 3.

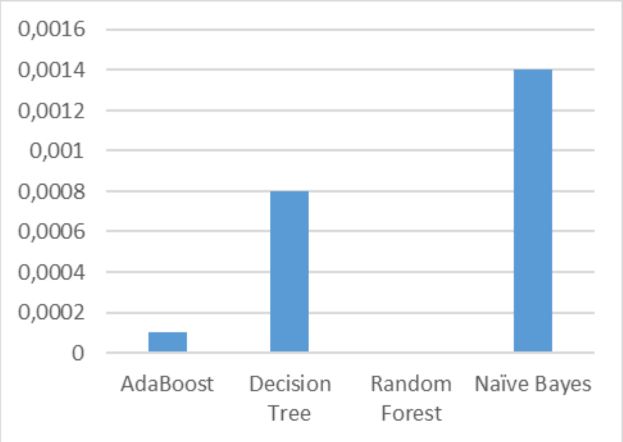

Fig. 3. Comparison of FPR classification values using Scikit-learn

B. Weka

1) Accuracy

When we use Weka, the highest accuracy in IoT botnet classification is for Random Forest classification method with an accuracy value reaching 100%. Next is Decision tree classification method with an accuracy of 99.99%, followed by adaboost classification method which has an accuracy of 96.18%. The lowest accuracy is for naïve bayes classification method with an accuracy value of 90.22%. A comparison of accuracy using Weka is presented in Figure 4.

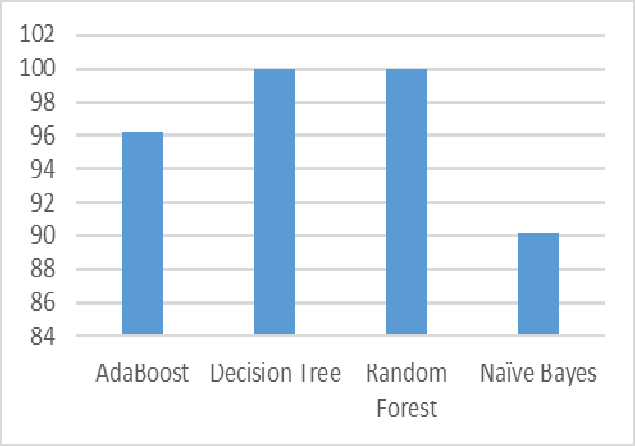

Fig. 4. Comparison of classification accuracy using Weka

2) Execution Time

The fastest execution time in IoT botnet malware classification using Weka is for naïve bayes classification method with an execution time of 41.96s. Then, adaboost classification method with an execution time of 594.07s, followed by decision tree classification method has an execution time of 893.16s. The longest execution time is using the random forest classification method with an execution time of 1417.75s. A comparison of accuracy using Weka is presented in Figure 5.

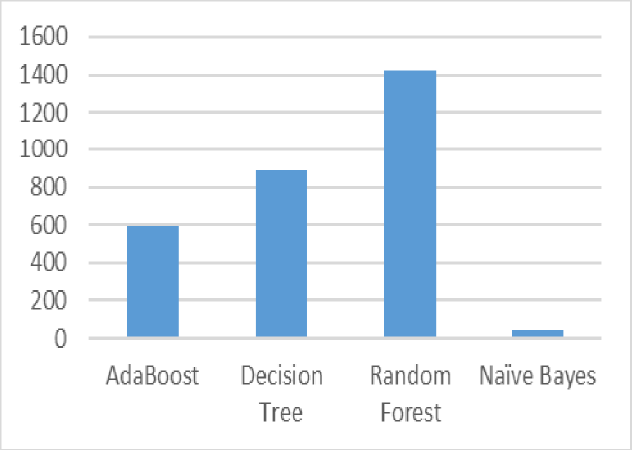

Fig. 5. Comparison of execution time classification using Weka

3) False Positive Rate

The lowest FPR value in Weka is the use of random Forest and Decision Tree classification method with an FPR value reaching 0. Next is adaboost classification method with an FPR of 0.035 and then the highest FPR is for naïve bayes classification method with an accuracy value of 0.078. A comparison of FPR using Weka is presented in Figure 6.

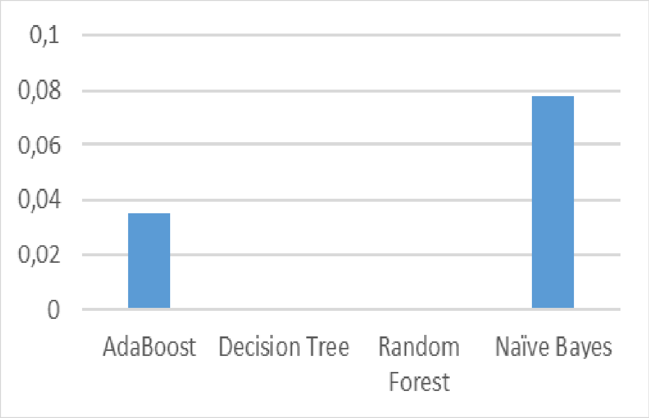

Fig. 6. Comparison of FPR classifications using Weka

C. The Comparative Study

1) Accuracy: the results of the comparison of the classification accuracy level presented in Table 1 show that by using the Weka tool, overall, the accuracy of the random forest, decision tree, and naïve bayes classification methods is higher than using Scikit-learn. However, the adaboost classification method has higher accuracy using Scikit-learn compared to Weka. From the results of average accuracy, the Weka tool is better than Scikit-learn.

TABLE I. COMPARISON ACCURACY CLASSIFICATION OF SCIKIT-LEARN AND WEKA

Classification	***Scikit-learn***	***Weka***
Random Forest	99,99%	100%
Decision Tree	98,53%	99,99%
Adaboost	99,92%	96,18%
Naïve Bayes	82,35%	90,22%
Average	95,26%	96,70%

2) Execution Time: The comparison of the execution time for classification in Table 2 shows that the use of the Weka tool with Random Forest, Adaboost, and Naïve Bayes classification methods gives faster execution time compared to the use of Scikit-learn. However, in the decision tree classification method the execution time using scikit-learn tool is faster than Weka. From the results of the average execution time, the Weka tool is faster than scikit-learn.

TABLE II. COMPARISON OF EXECUTION TIME SCIKIT-LEARN AND WEKA

Classification	***Scikit-learn***	***Weka***
Random Forest	1912	1417,75
Decision Tree	49,88	893,16
Adaboost	1079,86	594,07
Naïve Bayes	102	41,96
Average	785,935	736,735

3) FPR: From the results of the FPR comparison from the classification in Table 3, it can be seen that by using Weka and Scikit-learn tools the FPR value of the random Forest classification method are the same, then the adaboost and naïve bayes classification methods have a lower FPR value for Scikit-learn compared to Weka. However, by using the decision tree classification method the FPR values on the Weka tool lower than using scikit-learn. From the results of the average false positive rate, the Scikit-learn tool is lower than Weka.

TABLE III. COMPARISON OF FPR SCIKIT-LEARN AND WEKA

Classification	***Scikit-learn***	***Weka***
Random Forest	0	0
Decision Tree	0,0008	0
Adaboost	0,0001	0,035
Naïve Bayes	0,0014	0,078
Average	0,0023	0,113

V. CONCLUSION AND FUTURE WORK

Machine learning techniques help significantly in analyzing and predicting botnet malware on IoT. This study has compared two machine learning tools used in botnet malware data classification with different classifier algorithms. From the experimental results, it appears that the Weka and Scikit-learn tools have advantages and disadvantages in classification accuracy, execution time, and FPR. Overall, using the four classification algorithms Weka tool provides more accurate and efficient classification methods. However, in false positive rate the use of Scikit-learn provides also better results. In the future, authors consider researching on detection methods that can reduce the false positive rate, speed up execution time, and improve accuracy.

REFERENCES

[1] M. Stevanovic and J. M. Pedersen, "An analysis of network traffic classification for botnet detection," *Int. Conf. Cyber Situational Awareness, Data Anal. Assess.*, pp. 1–8, 2015, doi: 10.1109/cybersa.2015.7361120.

[2] N. S. Raghava, D. Sahgal, and S. Chandna, "Classification of Botnet detection based on botnet architechture," *Proc. - Int. Conf. Commun. Syst. Netw. Technol. CSNT 2012*, pp. 569–572, 2012, doi: 10.1109/CSNT.2012.128.

[3] A. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," *IEEE Commun. Surv. Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016, doi: 10.1109/COMST.2015.2494502.

[4] S. S. Chawathe, "Monitoring IoT networks for botnet activity," *NCA 2018 - 2018 IEEE 17th Int. Symp. Netw. Comput. Appl.*, pp. 1–8, 2018, doi: 10.1109/NCA.2018.8548330.

[5] E. M. Karanja, S. Masupe, and J. Mandu, "Internet of Things Malware : A Survey," *Int. J. Comput. Sci. Eng. Surv.*, vol. 8, no. 3, pp. 1–20, 2017, doi: 10.5121/ijcses.2017.8301.

[6] E. Bertino, "Botnets and Internet," *Computer (Long. Beach. Calif).*, pp. 76–79, 2017, doi: 10.1109/MC.2017.62.

[7] R. Alhajri, R. Zagrouba, and F. Al-Haidari, "Survey for Anomaly Detection of IoT Botnets Using Machine Learning Auto-Encoders," *Int. J. Appl. Eng. Res.*, vol. 14, no. 10, pp. 2417–2421, 2019.

[8] T. T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Commun. Surv. Tutorials*, vol. 10, no. 4, pp. 56–76, 2008, doi: 10.1109/SURV.2008.080406.

[9] N. A. F. A. Y. S. F. Shahzad, "A Hybrid Approach for Malware Family Classification," *Conf. Pap. Commun. Comput. Inf. Sci.*, no. June, pp. 181–189, 2017, doi: 10.1007/978-981-10-5421-1.

[10] S. A. Ginika Mahajan, Bhavna Saini, "Malware Classification Using Machine Learning Algorithms and Tools," *Proc. 2nd Int. Conf. Trends Electron. Informatics, ICOEI 2018*, pp. 1007–1012, 2018, doi: 10.1109/ICOEI.2018.8553780.

[11] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov, "Learning and classification of malware behavior," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5137 LNCS, pp. 108–125, 2008, doi: 10.1007/978-3-540-70542-0_6.

[12] G. Fedynyshyn, M. C. Chuah, and G. Tan, "Detection and Classification of Different Botnet C & C Channels," pp. 228–242, 2011.

[13] J. Z. Kolter and M. A. Maloof, "Learning to detect malicious executables in the wild," *KDD-2004 - Proc. Tenth ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vol. 7, pp. 470–478, 2004.

[14] M. N. P. Khodamoradi, M. Fazlali, F. Mardukhi, "Heuristic metamorrphic malware detection on statistics of aseembly instructions using classifiction algorithms," *CSI Int. Symp. Comput. Archit. Digit. Syst.*, 2015.

[15] J. Milosevic, M. Malek, and A. Ferrante, "A friend or a foe? Detecting malware using memory and CPU features," *ICETE 2016 - Proc. 13th Int. Jt. Conf. E-bus. Telecommun.*, vol. 4, no. Icete, pp. 73–84, 2016, doi: 10.5220/0005964200730084.

[16] T. Wüchner, M. Ochoa, and A. Pretschner, "Robust and effective malware detection through quantitative data flow graph metrics," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9148, pp. 98–118, 2015, doi: 10.1007/978-3-319-20550-2_6.

[17] H. Sayadi, H. Mohammadi Makrani, O. Randive, S. P. D. Manoj, S. Rafatirad, and H. Homayoun, "Customized Machine Learning-Based Hardware-Assisted Malware Detection in Embedded Devices," *Proc. - 17th IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. 12th IEEE Int. Conf. Big Data Sci. Eng. Trust. 2018*, no. only 4, pp. 1685–1688, 2018, doi: 10.1109/TrustCom/BigDataSE.2018.00251.

[18] M. Chandramohan, H. B. K. Tan, L. C. Briand, L. K. Shar, and B. M. Padmanabhuni, "A scalable approach for malware detection through bounded feature space behavior modeling," *2013 28th IEEE/ACM Int. Conf. Autom. Softw. Eng. ASE 2013 - Proc.*, pp. 312–322, 2013, doi: 10.1109/ASE.2013.6693090.

[19] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Futur. Gener. Comput. Syst.*, vol. 107, pp. 433–442, 2020, doi: 10.1016/j.future.2020.02.017.

[20] V. H. Bezerra, V. G. T. da Costa, S. Barbon Junior, R. S. Miani, and B. B. Zarpelão, "IoTDS: A one-class classification approach to detect botnets in internet of things devices," *Sensors (Switzerland)*, vol. 19, no. 14, pp. 1–26, 2019, doi: 10.3390/s19143188.

[21] M. Alazab, "Profiling and classifying the behavior of malicious codes," *J. Syst. Softw.*, vol. 100, pp. 91–102, 2015, doi: 10.1016/j.jss.2014.10.031.

[22] N. Kumar, S. Mukhopadhyay, M. Gupta, A. Handa, and S. K. Shukla, "Malware classification using early stage behavioral analysis," *Proc. - 2019 14th Asia Jt. Conf. Inf. Secur. AsiaJCIS 2019*, pp. 16–23, 2019, doi: 10.1109/AsiaJCIS.2019.00-10.

[23] Y. Meidan *et al.*, "N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, 2018, doi: 10.1109/MPRV.2018.03367731.

[24] H. Bahsi, S. Nomm, and F. B. La Torre, "Dimensionality Reduction for Machine Learning Based IoT Botnet Detection," *2018 15th Int. Conf. Control. Autom. Robot. Vision, ICARCV 2018*, pp. 1857–1862, 2018, doi: 10.1109/ICARCV.2018.8581205.

[25] "scikit-learn user guide," 2020.

[26] R. R. Bouckaert, E. Frank, R. Kirkby, P. Reutemann, A. Seewald, and D. Scuse, "WEKA Manual for Version 3-7-2," 2002.

[27] H. El Merabet and A. Hajraoui, "A survey of malware detection techniques based on machine learning," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 1, pp. 366–373, 2019, doi: 10.14569/IJACSA.2019.0100148.

[28] O. Obulesu, M. Mahendra, and M. Thrilokreddy, "Machine

Learning Techniques and Tools: A Survey," *Proc. Int. Conf. Inven. Res. Comput. Appl. ICIRCA 2018*, no. Icirca, pp. 605–611, 2018, doi: 10.1109/ICIRCA.2018.8597302.

[29] T. K. An and M. H. Kim, "A new Diverse AdaBoost classifier," *Proc. - Int. Conf. Artif. Intell. Comput. Intell. AICI 2010*, vol. 1, pp. 359–363, 2010, doi: 10.1109/AICI.2010.82.