An Architecture for Analysis of Mobile Botnet Detection Using Machine Learning

Ashok Patade(✉) and Narendra Shekokar

DJSCE, University of Mumbai, Mumbai, India
{ashok.patade,narendra.shekokar}@djsce.ac.in

Abstract. The smart-phone has become a critical cybernetic victim especially of cellular botnets. The current exploration examines mobile botnet attacks for Android smart-phone launched from a Windows based PC and detects those attacks using an ensemble machine learning classification algorithm. This investigation is to breach the gap to develop malware framework in perspective of headers' field examination of PE format with Machine Learning algorithm used. The Homogeneous architecture model proposed in this investigation chooses the most representative subset of the features toward accurate classification of botnet traffic. The ensemble classification technique used in this architecture consists of four machine learning algorithms namely; Random Forest, Gradient Boosting Algorithm, Extreme learning machine and eXtreme Gradient Boosting. After exhaustive literature survey, it is concluded that the architecture proposed in this investigation is unique homogeneous combination of four algorithm mention above. The efficiency of the proposed architecture is evaluated on existing data-set CLaMP (Classification of Malware with PE headers). It has been evaluated on a ClaMP data-set to achieve better botnet detection accuracy relative to its peer techniques.

Keywords: Mobile botnet attack detection · Malware analysis · Cyber-security · Ensemble learning

1 Introduction

In the domain of cyber security a botnet is defined as a interconnected web of compromised or susceptible PCs or electronic gadgets which are controlled or can be accessed remotely by the botmaster using a Command and Control (C&C) channel. This C&C channel can be operated by means of bluetooth, WiFi or SMS. The compromised framework is named as a bot which has vulnerabilities and makes ready for the intrusion of the botmaster. The C&C channel can be utilized for correspondence between the botmaster and the individual bots in the botnet. In the event that the gullible client is utilizing the cell phone, a significant number of the time one keep messages in the message box which is as of now read. An aggressor uses such as of now read messages to perform assaults on such cell phones. Utilizing this system an aggressor can take classified data like financial balance, individual contact and utilize this data for the malevolent exercises.

The current investigation proposes a homogeneous architecture for mobile botnet detection on android phones using an ensemble machine learning model. The aim is to choose the most representative subset of the features or attributes from the network traffic generated by a compromised bot and utilize it toward accurate classification or estimation of mobile botnet traffic. The efficacy of the technique is underlined by conducting broad experiments on existing dataset viz. CLaMP (Classification of Malware with PE headers). The ensemble classification model used in the proposed architecture uses non parametric probabilistic frameworks that have gained significance in the field of predictive analytics. Their mathematical models are described in Sect. 4.

We are explaining basic of the mobile botnet and feature selection in machine learning in this Section. In Sect. 2 the literature survey is presented in which findings of the mobile botnet attacks and use of machine learning algorithm in the mobile botnet detection using feature selection are presented. In Sect. 3 the proposed system for detecting mobile botnet attack launched from Windows PC to target Android phones is described. The mathematical model for the detection mechanism of this architecture using machine learning classification is presented in this Section. Section 4 presents the results of the architecture measured on the CLaMP (Classification of Malware with PE headers) datasets against existing techniques. The conclusion of this investigation is presented in Sect. 5.

1.1 Basic of Botnet and Mode of Communication and Propagation for Mobile Botnet

The term Botnet is a combination of robot and network. The botnet is available in various architectures like centralized, peer to peer, hybrid etc. Botnets can be outlined utilizing Personal PCs and in addition cell phones. A versatile botnet, much the same as a PC botnet, is a sort of malware that runs naturally once introduced on a gadget without portable antivirus programming. As the botnet develops, each tainted cell phone gets added to a system of bots overseen by a botmaster cybercriminal.

1.2 Background of Machine Learning

Only two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

Arthur Samuel (1959) characterizes Machine learning as, "Field that enables PCs to learn without being expressly programmed" [12,13].

- **Supervised learning:** It encourages the PC to accomplish something, at that point let it utilize it's recently discovered information to do learning [7,10,14].

- **Unsupervised learning:** Unsupervised learning let the PC figure out how to accomplish something, and utilize this to decide structure in information [11,17,18].
- **Semi-Supervised Machine Learning**
  Issues where you have a lot of information (X) and just a portion of the information is marked (Y) are called Semi-Supervised Machine learning issues [15,16,19].

Machine learning algorithm used for classification of malware mobile botnet with feature selection are briefly explained below:

Random Forest. Random forest is a troupe technique which is built by a few choice trees to vote on the order (or relapse) assignment and created by (Breiman 2001). (Horning 2010) In Random forest, part traits are picked arbitrarily, so the connection between's trees is diminished bringing about enhanced forecast precision. In Random forest calculation, the quantity of trees and the highlights to build trees are picked by the client. Be that as it may while preparing Random forest in Weka, there is no need to pick the quantity of highlights, Weka satisfies this activity by thinking of some as capacity foundation (Ali, Khan, Ahmad and Maqsood 2012).

Gradient Boosting Algorithm. Slope boosting [6] is a champion among the most serious strategies for building perceptive models. Boosting left whether a frail learner can be changed to end up better. Leeway of the angle boosting structure is that another boosting estimation should be deduced for each adversity work. Choice trees are used as the feeble learner in slope boosting.

Extreme Learning Machine. The core of Extreme learning machine [5], known as ELM, is that the learning parameters of nodes, including input weights and inclination, are discretionarily doled out and require not be tuned while the yield weights can be sensibly controlled by the fundamental summed up in reverse engendering. The main parameter should have been characterized is the quantity of nodes. Contrasted and other customary learning calculations for SLFNs, ELM gives to a great degree quicker learning rate, better speculation execution and with slightest human intercession.

EXtreme Gradient Boosting. XGBoost [3], is an updated passed on tendency boosting library expected to be significantly successful, versatile and flexible. It realizes machine learning computations under the Gradient Boosting framework. XGBoost gives a parallel tree boosting (generally called GBDT, GBM) that handle various data science issues in a snappy and correct way.

1.3 Feature Selection Methods

Aside from models with implicit component determination, most methodologies for decreasing the quantity of indicators can be set into two primary classifications. Utilizing the wording of [9]:

– *Wrapper* strategies assess various models utilizing methods that include and additionally expel indicators to locate the ideal mix that boosts show execution.
– *Filter* techniques assess the importance of the indicators beyond the prescient frameworks and therefore display just the indicators consisting of a certain category of measure [25].

1.4 PE Header and Export Table

In this subsection, a delineation of the Windows smaller executable (PE) header of combined executables and the Export table is given.

PE is the neighborhood Win32 record sort out. Each win32 executable (beside VxDs and 16-bit DLLs) utilizes PE record orchestrate. 32bit DLLs, COM archives, OCX controls, Control Panel Applets (.CPL records) and .NET executables are all PE sort out. Undoubtedly, even NT's piece mode drivers use PE archive compose. It is basic to consider Windows PE archive for two reasons: Adding code to executables (e.g. keygen imbuement or including handiness) and physically emptying executables. With respect to the last said, most shareware nowadays comes "stuffed" in order to give an extra layer of security.

In a stuffed executable, the import tables are typically pulverized and data is consistently mixed. The packer inserts code to empty the archive in memory upon execution, and a short time later bounces to the principal area reason for the record (where the primary program truly starts executing). In case we make sense of how to dump this memory region after the packer completed emptying the executable, in any case we need to settle the regions and import tables before our application will run.

PE Header

Any parallel executable record (paying little respect to operating systems prominently present in electronic gadgets such as Unix or Windows) needs to join a header to depict its arrangement: e.g., the base area of its code region, data portion, and the once-over of limits that know how to be carried from the executable, et cetera. Right once the record is performed by the working structure, the OS essentially scrutinizes this header info, and after that piles the twofold data from the archive to occupy the substance of the code/data bits of the area space aimed at the relating system. Exactly when the archive is logically associated (i.e., the structure calls it depend on upon are not statically associated in the executable), the OS needs to count on its import table to make sense of wherever to find the areas of these structure purposes.

Most twofold executable records on Windows seeks after the going with organization: **DOS Header** (64 bytes), **PE Header, sections** (code and data).

DOS Header starts with charm number 4D 5A 50 00, and the former 4 bytes is the region of PE header in the twofold executable record. Distinctive arenas are not all captivating. The PE header holds in a general sense more information and all the all the extra stimulating. In Fig. 1, you find the organization of PE Header.

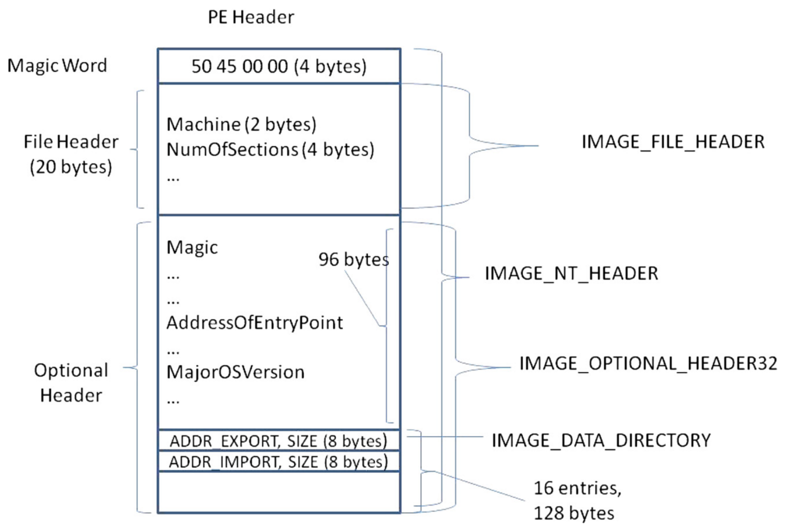

Fig. 1. Structure of PE header

Export Table

The primary area of IMAGE_DATA_DIRECTORY of the voluntary header field has facts of the fare table. By Fig. 1, it is derived that the 4 bytes arranged at PE 0x78 (i.e., balance 120 bytes) is the relative area (concerning DLL base area) of the fare table, and the accompanying byte (at balance 0x7C) is the proportion of the fare table.

1.5 Homogeneous Ensemble Model

Homogeneous ensemble models use multiple models of the same type for classifying or regressing the result. The intuition behind such a technique is to simultaneously use multiple weak predictors to build a strong predictor. Ensemble methods are of two types: Homogeneous and heterogeneous. Usually ensemble models use bagging "bootstrap resampling" method for Ensemble learning. A final meta model is build from initially models by polling, averaging etc. Every individual model uses various preparing sets, by using bootstrap, it utilizes numerous variants of preparing set. Such a group meta calculation of machine learning is made for enhancing the exactness of machine learning calculation for both relapse and grouping.

2 Related Work

In this section we are focusing on the work which is closer to the current research exertion.

[21] and Storm, separates itself from different types of malware (infections, Trojan ponies, worms) by its capacity to set up a control channel that enables its tainted customers to work as an organized group, or botnet.

Some security scientists planned assault methodologies as mention in [4], utilizing SMS benefit messages known as "Quiet SMS" to the objective gadget to execute the DOS (Denial of Service) assault.

Open source community is active to investigate assault analysis on mobile platform. Open source programming ventures for correspondence framework development including OpenBTS and OpenBSC [1,27].

[23] and Conficker is regarded as a significant framework for performing malicious activity. In the age of internet based honeypots, the malware named conficker has been regarded as one of the most harmful and malicious cyber threat that has affected PC and electronic gadgets worldwide.

The iKee [22] bot is one of the most recent contributions in cell phone malware, focusing on jailbroken iPhones. While its execution is straightforward in contrast with the most recent age of PC-based malware, its suggestions exhibit the potential augmentation of crimeware to this significant new wilderness of handheld customer devices.

Lastly, Researchers in [24], present multi day vulnerabilities and shortcomings they found in the Short Message Service (SMS) convention, which permit the inserting of high limit clandestine channels.

2.1 Literature Review of Highlight Choice and Machine Learning

The accompanying area discusses commitment of [8,20,26] in determination of various system activity highlights utilized in versatile botnet discovery and how unique machine learning calculations used to accomplish the best arrangement precision.

[26], investigate the viability of various blend of highlights to accomplish the best order exactness. They benchmark stream based factual highlights effectively utilized in the current examinations and break down their relative viability. They utilized three element choice techniques, for example, eigenvalue or spectral techniques such as Correlation Attribute Identification (CFS), Principal Component Analysis (PCA), and Minimum repetition most extreme importance (mRMR), to kill less discriminative highlights from other applicant highlights. In spite of the fact that their last list of capabilities demonstrated a high recognition rate of with 99% on a preparation dataset which incorporates modest number of botnets, they accomplished 75% identification precision on a testing dataset which contains considerably more unique kinds of botnet.

[8], examine in their investigation, the impact of the choice of various system activity stream exporters. To achieve this, they assess five distinctive activity stream exporters; Maji et al., Tranalyzer and Netmate utilizing five unique classifiers C4.5, SVM, ANN, Bayesian systems, and Naive Bayes. They direct a progression of analyses on open botnet datasets. As indicated by their test results, the best arrangement exactness is accomplished by utilizing Tranalyzer with C4.5 classifier.

In [20], featured imperative highlights utilized in versatile botnet Detection, subsequent to applying highlight choice on the extricated include set, three surely understood machine learning calculations to be specific; SVM, Random Forest and Logistic Regression are assessed to accomplish best botnet discovery exactness.

They utilize three component determination strategies to pick the most agent subset of the highlights toward precise grouping of botnet activity.

In our understanding, but for (Markel and Bilzor 2014), no further malware framework work has been ended in perspective of headers' field examination. In view of our writing audit we have investigate our discoveries in the accompanying area.

2.2 Review and Analysis

Mobile Botnets have been castoff for a variety of malicious doings and they have ripe and have become exceptionally cultured over the years. Understanding and disassembling these nets needs vigorous investigation and collaboration between private and government areas. There are very few attempts to detect and evaluate mobile botnet attacks using Machine Learning. PE file format on Windows OS is first time evaluated in current research work with Machine Learning algorithm: Random forest, Stochastic Gradient boosting, Extreme learning machine and XGBoost. A representative set of features to identify mobile botnets is built from varying network traffic features. An ensemble machine learning model is used in the proposed architectures to detect mobile botnet for android smartphone.

2.3 Problem Statement

Public WiFi networks are increasing in the current age of digitization for smart phones users. However, such networks offer benefits as well as challenges from the perspective of cyber security. They offer an anonymous channel for launching cyber attacks over the internet. Hence, it is necessary for securing smart phones which are getting attacked by public WiFi networks by building an efficient detection framework that can aid the system administrators of such public WiFi networks. The homogeneous architecture proposed in this inquiry aims to offer such a framework. The current inquiry focuses on mobile botnet attacks launched from Windows PC on remote Android phones. The scope of the investigation is defined with these restrictions as such category of attacks are increasing in scale and frequency.

3 Proposed Architecture for Analysis of Mobile Botnet Detection Using Machine Learning for Android

Thus, in the preceding sections the background information of predictive analytics/machine learning, mobile botnets, PE file format, feature selection and ensemble models was provided. The homogeneous architecture is built from the assimilation of these concepts. It selects or leads to the identification of the most representative subset of the features or variables (dependent variables) toward accurate classification (ground truth labeling) of mobile botnet traffic on android phones. The extensive experiments are performed on existing dataset viz. CLaMP (Classification of Malware with PE headers). The ensemble model used in the proposed architecture consists of four well-known machine learning algorithms namely; Random Forest, Gradient Boosting Algorithm, Extreme learning machine and eXtreme Gradient Boosting. The homogeneous architecture uses this for the first time to achieve best mobile botnet detection accuracy.

3.1 Ensemble Model for Classification

A Ensemble model is worked with the end goal of machine learning. The segments of this gathering are arrived at the midpoint of for getting the meta model.

3.2 Detection Mechanism - Mobile Botnets

In the first stage, numerous features as API calls, strings, also code behaviours are takeout statically as well as dynamically to arrest the appearances of the file examples for dissimilar mode of statement and spread for Mobile Botnet. In the second stage, smart methods of ML classification are used to mechanically classify the file examples interested in dissimilar classes/objects built on the examination of feature representations. Note that these Machine Learning malware indicators for botnet mostly vary on the feature demonstration and the employed machine learning techniques.

In these patterns, the uncovering is typically a pipeline course as revealed in below figure. Figure 2 displays the general procedure of malware detection using Machine Learning practises using PE file format.

- **Windows PE file Network Capture.** This step use Python scripts to collect the PE file samples from the network. This sample used to create the dataset.
- **Embedded Feature Extraction.** A python script as mentioned in the [2] which extract all the values from all three key PE headers. DOS_Header, FILE_HEADER AND OPTIONAL_HEADER. If any error occurs then the values will be allocated as zero for that header. Many PE files don't have DOS_Header then all the header will be assigned '0'.
- **ML Classification Model Construction.** It evaluated the performance of four non-parametric supervised learning algorithm namely Random Forest, Gradient Boosting Algorithm, Extreme learning machine and eXtreme Gradient Boosting using "Caret" package.

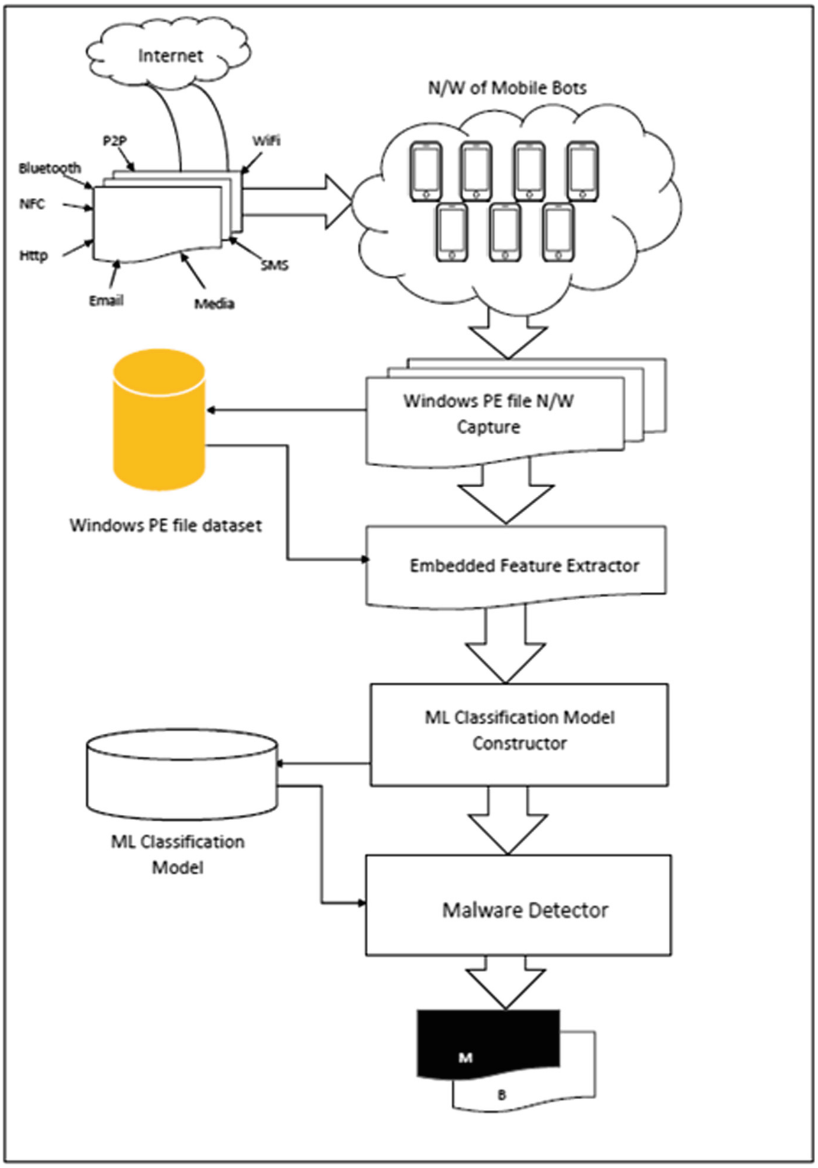

Fig. 2. Process of Malware detection using ML classification & feature selection algorithm

- **ML Classification Model.** Final model is a homogeneous model of the above four techniques.
- **Feature importance**. homogeneous model will analyze the most important features for the Malicious PE files for Mobile Botnet detection.

4 Experimental Study

In this Section the aftereffects of the proposed demonstrate is contrasted with benchmark strategies to feature its adequacy.

4.1 Experimental System

To approve the proposed thought, a test situation is made with Windows 10 Pro Operating framework running on Intel(R) Core(TM) i7-5660 CPU @2.60 GHz CPU and 16 GB of essential storage and 512 GB of auxiliary storage.

4.2 Performance Metrics

For assessment purposes, the accompanying traditional measures appeared in Table 1 are utilized to assess the completing of classification-constructed malware discovery.

4.3 Description of the Dataset - ClaMP (Classification of Malware with PE Headers)

In the study, Dataset - ClaMP (Classification of Malware with PE headers) is used. The dataset is generated with the tool - "PEFile" created by Ero Carrera [2]. A detailed description of the used dataset is provided here (Fig. 3):

Raw Set. Rough features list of abilities is made by expelling headers from all fields (segments/header information) of three important headers (DOS header, File Header and Voluntary header, plus customary and Windows-explicit arenas) in each Portable Executable record.

Integrated Set. The organized rundown of capacities is made by joining a couple picked unrefined features and a game plan of decided highlights in which 28 are same as in rough rundown of capacities, 26 boolean (categorical) highlights are made by developing individual pennants of Characteristics and DLL Characteristics from the File header and Voluntary header Windows explicit and 14 remain deduced highlights.

Table 1. Measures of classification-based malware detection performance

Measures	Specification
True Positive (TP)	Quantity of file examples accurately categorized as malicious
True Negative (TN)	Quantity of file examples accurately categorized as benign
False Positive (FP)	Quantity of file examples incorrectly categorized as malicious
False Negative (FN)	Quantity of file examples incorrectly categorized as benign
TP Rate (TPR)	TP/(TP + FN)
FP Rate (FPR)	FP/(FP + TN)
Accuracy (ACY)	(TP + TN)/(TP + TN + FP + FN)

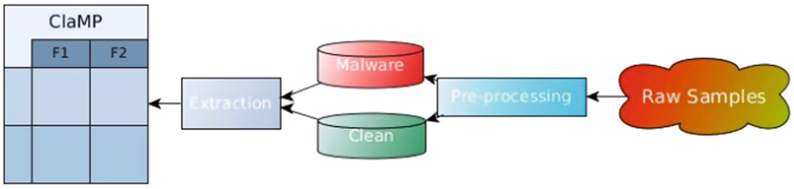

Fig. 3. CLaMP (Classification of Malware with PE headers)

4.4 Comparison with Previous Works

The proposed homogeneous architecture is evaluated against the below baselines techniques:

David et al. (2016) have discussed the sorted out examination of numerous arenas of PE header and communicated that seeing these highlights boost the malware distinguishing proof frequency.

Bai et al. (2014) devise API and DLL requests close by numerous unrefined estimations of PE headers. Examination through (Bai et al. 2014) effort won't state flawless virtues or blames of the anticipated exertion yet aimed at evaluated results (Tables 2 and 3).

Table 2. Comparison of proposed homogeneous architecture with baselines on CLAMP (Raw dataset)

Technique	Sensitivity	Specificity	Precision
David et al. (2016)	0.86	**0.98**	0.95
Bai et al. (2014)	0.89	0.96	**0.96**
Markel and Bilzor (2014)	**0.936**	0.95	0.93
Proposed technique	**0.936**	0.92	0.91

Table 3. Comparison of proposed homogeneous architecture with baselines on CLAMP (Integrated dataset)

Technique	Sensitivity	Specificity	Precision
David et al. (2016)	0.95	**0.986**	0.91
Bai et al. (2014)	0.86	0.90	0.82
Markel and Bilzor (2014)	0.93	0.94	0.90
Proposed technique	**0.968**	0.96	**0.93**

5 Conclusion

We have studied and demonstrated the efficacy of the proposed architecture for Mobile botnet attack detection launched from Windows platform to android phones. We have proposed an architecture that chooses the most representative subset of the features toward accurate classification of botnet traffic. The homogeneous architecture uses a ensemble learning algorithm that is trained on header field values only. These header field values are collected from Windows PE file. Accuracy as well as several performance metrics have been used to evaluate the performance of proposed technique with respect to several state of the art baselines. Analysis of the results show that the proposed framework of the current inquiry outperforms the other frameworks on several measures or otherwise achieves comparable performance.

References

1. Burgess, D.A., Samra, H.S., et al.: The OpenBTS project (2008). http://openbts.sourceforge.net, http://openBTS.org
2. Carrera, E.: x5: Reverse engineering automation with Python
3. Chen, T., He, T., et al.: XGBoost: extreme gradient boosting
4. Croft, N.J., Olivier, M.S.: A silent SMS denial of service (DoS) attack (2007)
5. Ding, S., Zhao, H., Zhang, Y., Xu, X., Nie, R.: Extreme learning machine: algorithm, theory and applications. Artif. Intell. Rev. **44**(1), 103–115 (2015)
6. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**, 1189–1232 (2001)
7. Gentyala, V.: Geolocation events based auto theme changer for browsers. Int. J. Adv. Res. Comput. Sci. **4**(3) (2013)
8. Haddadi, F., Zincir-Heywood, A.N.: Benchmarking the effect of flow exporters and protocol filters on botnet traffic classification. IEEE Syst. J. **10**(4), 1390–1401 (2016)
9. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: Machine Learning Proceedings 1994, pp. 121–129. Elsevier, Amsterdam (1994)
10. Nerurkar, P.: Review of data storage by fusion drive in MAC. Int. J. Adv. Res. Comput. Sci. **4**(3), 256–259 (2013)
11. Nerurkar, P., Bhirud, S.: Modeling influence on a social network using interaction characteristics. Int. J. Comput. Math. Sci. **6**(8), 152–160 (2017)

12. Nerurkar, P., Chandane, M., Bhirud, S.: Community detection using node attributes: a non-negative matrix factorization approach. In: Verma, N.K., Ghosh, A.K. (eds.) Computational Intelligence: Theories, Applications and Future Directions - Volume I. AISC, vol. 798, pp. 275–285. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-1132-1_22
13. Nerurkar, P., Chandane, M., Bhirud, S.: A comparative analysis of community detection algorithms on social networks. In: Verma, N.K., Ghosh, A.K. (eds.) Computational Intelligence: Theories, Applications and Future Directions - Volume I. AISC, vol. 798, pp. 287–298. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-1132-1_23
14. Nerurkar, P., Pavate, A.: Study of AngularJS: a client side Javascript framework for single page applications. Int. J. Contemp. Res. Comput. Sci. Technol. **1**(4), 92–96 (2015)
15. Nerurkar, P., Pavate, A., Shah, M., Jacob, S.: Analysis of probabilistic models for influence ranking in social networks. In: Iyer, B., Nalbalwar, S.L., Pathak, N.P. (eds.) Computing, Communication and Signal Processing. AISC, vol. 810, pp. 215–223. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-1513-8_23
16. Nerurkar, P., Pavate, A., Shah, M., Jacob, S.: Performance of internal cluster validations measures for evolutionary clustering. In: Iyer, B., Nalbalwar, S.L., Pathak, N.P. (eds.) Computing, Communication and Signal Processing. AISC, vol. 810, pp. 305–312. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-1513-8_32
17. Nerurkar, P., Shirke, A., Chandane, M., Bhirud, S.: Empirical analysis of data clustering algorithms. Procedia Comput. Sci. **125**, 770–779 (2018)
18. Nerurkar, P., Shirke, A., Chandane, M., Bhirud, S.: A novel heuristic for evolutionary clustering. Procedia Comput. Sci. **125**, 780–789 (2018)
19. Pavate, A., Nerurkar, P., Ansari, N., Bansode, R.: Early prediction of five major complications ascends in diabetes mellitus using fuzzy logic. In: Nayak, J., Abraham, A., Krishna, B.M., Chandra Sekhar, G.T., Das, A.K. (eds.) Soft Computing in Data Analytics. AISC, vol. 758, pp. 759–768. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-0514-6_72
20. Pektaş, A., Acarman, T.: Effective feature selection for Botnet detection based on network flow analysis (2017)
21. Porras, P., Saidi, H., Yegneswaran, V.: A multi-perspective analysis of the storm (Peacomm) worm. Technical report (2007)
22. Porras, P., Saïdi, H., Yegneswaran, V.: An analysis of the iKee.B iPhone Botnet. In: Schmidt, A.U., Russello, G., Lioy, A., Prasad, N.R., Lian, S. (eds.) MobiSec 2010. LNICST, vol. 47, pp. 141–152. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17502-2_12
23. Porras, P.A., Saidi, H., Yegneswaran, V.: A foray into Conficker's logic and rendezvous points (2009)
24. Rafique, M.Z., Khan, M.K., Alghatbar, K., Farooq, M.: Embedding high capacity covert channels in Short Message Service (SMS). In: Park, J.J., Lopez, J., Yeo, S.-S., Shon, T., Taniar, D. (eds.) STA 2011. CCIS, vol. 186, pp. 1–10. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22339-6_1
25. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics **23**(19), 2507–2517 (2007)
26. Samani, E.B.B., Jazi, H.H., Stakhanova, N., Ghorbani, A.A.: Towards effective feature selection in machine learning-based botnet detection approaches. In: 2014 IEEE Conference on Communications and Network Security, pp. 247–255 (2014)
27. Welte, H.: Report of OpenBSC GSM field test August 2009, HAR2009, Vierhouten, The Netherlands (2009)