National College of Ireland

A Comparative Analysis of Base Learning and Ensemble Learning for Botnet Detection

MSc Internship
Cyber Security

Sheriff Agboola
x18123171

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Sheriff Agboola
Student ID:	x18123171
Programme:	Cyber Security
Year:	2019
Module:	MSc Internship
Supervisor:	Vikas Sahni
Submission Due Date:	12/12/2019
Project Title:	A Comparative Analysis of Base Learning and Ensemble Learning for Botnet Detection
Word Count:	5886
Page Count:	19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	10th December 2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Comparative Analysis of Base Learning and Ensemble Learning for Botnet Detection

Sheriff Agboola
X18123171
MSc Internship in Cyber Security

Abstract

The proliferation of botnets is one of the challenges faced in the realm of cyber security. Botnet are used in perpetuating malicious activities such as stealing personal data, password and sensible information belonging to organisations, click-fraud and performing Distributed Denial of Service (DDoS) attacks, sending of unsolicited emails etc. In this study, we compare the base learning model and ensemble machine learning model using the brute force search technique.Our approach focuses on four of the most commonly used machine learning methods namely Support Vector Machine (SVM), Decision Tree, Random Forest and AdaBoost in detecting the Internet Relay Chat (IRC) botnet which is one of the oldest and most used type of botnet. The experimental results show that the detection accuracy of Random Forest is better than the rest of the machine learning methods, achieving an accuracy of 99.6% on a balanced dataset and a false positive of 0.2% on known IRC botnet dataset.

1 Introduction

Botnets have always been a daunting threat to cybersecurity and are progressively advanced as detection resistant as they control a huge number of computing devices in participating in malicious activities. The internet of things has become part of our daily life ranging from smartwatches to self-driving cars that are connected to the internet thereby enabling the transferring of data automatically between objects automatically without the intervention of humans[1]. IoT-analytics in 2018, states that the total number of IoT devices is estimated at about seven billion devices connected to the internet[2]. However, the rapid increase of this new technology which has been accepted widely by everyone is being targeted, exploited and made vulnerable by cybercriminals, using them for malicious purposes such as adding them to a Botnet and utilizing them in spamming, phishing and launching of distributed denial of services (DDoS). Mirai, a ground-breaking attack within the past decade gained access to over 300,000 devices and about 49,657 distinct IP addresses located within 164 countries by exploiting the weak points within these devices such as inadequate security updates to legacy and non-legacy devices, weak user credentials, and many more[3]. According to the European Union agency for cybersecurity (ENISA), reported how Deutsche Telekom one of the largest internet providers in

[1] https://www.hcltech.com/technology-qa/what-is-an-iot-device
[2] https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/
[3] https://www.imperva.com/blog/malware-analysis-mirai-ddos-botnet/

Europe suffered a major outage when an estimated number of 900,000 home routers were exploited and compromised by variants of Mirai[4]. Also, the post-Deutsche Telekom incident was followed by an attack on the United Kingdom's post office and internet provider home routers by the same variant of the Mirai botnet which was later used to launch a DDoS on December 2016 against a bitcoin company located in the united kingdom with traces showing an estimated percentage of over 99% belonging to infected home routers in the United Kingdom[5].

1.1 Motivation and Project Background

The rate at which we rely on the internet is increasing daily, so are the challenges involving the confidentiality, integrity and most importantly the security of a user's data. The rise of the internet, internet-based applications, as well as IoT devices over the years has experienced a great expansion in which they are a crucial part of everything we do ranging from services such as the agricultural, finance, medical and many more. The use of these services causes several security challenges in terms of malicious or illegal activities which is made feasible due to use of malicious software. The evolution of the Internet and the various services associated with it has brought about a great amount of development as malicious software are able to effectively and efficiently perform illegal activities that compromises the user's system and put the security of the user's data at risk. Within the last decade, malware has evolved by advancing its method of propagation and the ability to withstand the effort of shutting them down.

These incidents set out the need for proper mitigation against the proliferation of botnet attacks using machine learning.

1.2 Research Question

How can a botnet be identified and detected using supervised machine learning algorithms (Support Vector Machine, Random Forest, Decision Tree and, AdaBoost) by providing a comparison between the individual supervised learning models in terms of efficiency, accuracy, and performance ?

1.3 Research Objectives

In order to solve the need for proper mitigation against the proliferation of botnet attacks, the following goals are stated and implemented:

- A critical review of literature on botnet detection.
- Implement and evaluate the result of linear support vector machine model.
- Implement and evaluate the result of Random Forest model.
- Implement and evaluate the result of Decision Tree model.
- Implement and evaluate the result of AdaBoost model.
- Comparison of developed models.

[4] `https://www.enisa.europa.eu/publications/info-notes/mirai-malware-attacks-home-routers`
[5] `https://www.imperva.com/blog/new-variant-mirai-embeds-talktalk-home-routers/`

The rest of this study is structured as follows: In section 2, we present related works done concerning botnets and the various mitigation strategies carried out against botnets and based on the conclusion from this section, Section 3 provides the methodology that will be used for this study.In section 4 we provide the design specification for this project. Section 5, we implement our botnet detection models, section 6 contains the evaluation of the supervised machine learning models performance and we conclude in section 7.

2 Related Work

One of the aims set for cyber and network security is the creation of techniques which are competent enough for the detection and eventually the take down of these botnet threats. Many theories have been proposed by various researchers with the aim of addressing this malicious and illegal activities perpetuated by botnets. This section contains related study that has been carried out involving botnets, its propagation, various botnet detection strategies as well as the latest research milestones of botnet based on the technology of machine learning.

2.1 Background

Alieyan et al.[1], defines a botnet as a software program that is mostly used in infecting computing systems and are also called bots. This malicious program is popularly known for carrying out malicious goals [2]. The infected computers controlled by the botmaster which uses this well organised and highly coordinated platform in executing illegal and malicious activities. Furthermore, a botnet ranges from home network to educational and corporate networks and covering several autonomous systems managed and operated by various internet provider [3]. According to Securelist [6] by Kaspersky, claim that there was an increase in botnets in 2018 discovering above 38% of new variant of botnets which was not encountered in the previous year and this shows the potential threat of these malicious software have passively or actively towards the systems connected to the internet. Hence, with the number of bots within the botnet network it possesses the computational power to carry out malicious activities such as click fraud, phishing, identity theft and mostly the initiation of Distributed Denial of Service(DDoS) attacks[1],[4],[5]. This shows that botnets are one most powerful malicious tools in executing malicious and illegal activities by cyber criminals and for it to be mitigated properly, the neutralisation of botnets should be made feasible through methods that can detect the presence of botnets, analyse and put in place the relevant countermeasures needed.

2.2 Botnet life-cycle

Feily, Shahrestani, and Ramadass [6] categorises botnet operation into three modes of operation: the infection mode, the communication mode and the attack mode. The infection mode involves compromising of the systems which are vulnerable through the installation of malicious software hence, they become part of the bot network and each of the computer within this network is called a zombie. Further research made by Stevanovic and Pedersen [3], claim that the infection mode can be divided into a primary infection which compromises the computing systems through a malicious software which is made

[6] `https://www.securelist.com/bots-and-botnets-in-2018/90091/`

feasible by downloading it from illegitimate websites. The sole purpose of the primary infection is to aid in delivering of the binary used by the botnet and the secondary infection uses the network privileges on the victim's computer to download the full binary from an external source.

The second mode is the communication mode which contains how the command and control server (C2) interact with the compromised systems such as the regular update on the current status of each bot, the C2 server passes out information from the botherder[7]. The C2 server is the main unit for botnets and a great number of defence-in-depth is put in place to protect the server from being compromised through a single point of failure by the law enforcement since it serves as the middle man between the bots and the botmaster.Communication takes place between the server and the bots through the C2 channel which deploys methods such as the IRC(Internet Relay Chat), HTTP (Hypertext Transfer Protocol) or it secure form and many more.

The third mode involves using the bot for malicious purposes ranging from the act of disrupting legitimate services, personal information theft as well as fraudulent act such as the click-fraud. Also, Stevanovic and Pedersen [3], claim that there are no major differences between these modes as the are interconnected and the botnet infection cycle is a continuous process.

2.3 Botnet Architecture

The architecture implemented by botnets can be classified into three types which are centralised, decentralised and the hybrid architecture[8]. The centralised botnet architecture uses the one-to-many system where all the bots communicate with a single command and control server or a many-to-many system where the bots communicate with a large number of C2 servers via a dedicated channel as means of communication.The centralised architecture is an efficient method where there is no third-party in the communication involving the execution and status exchanged between the C2 and the zombies.

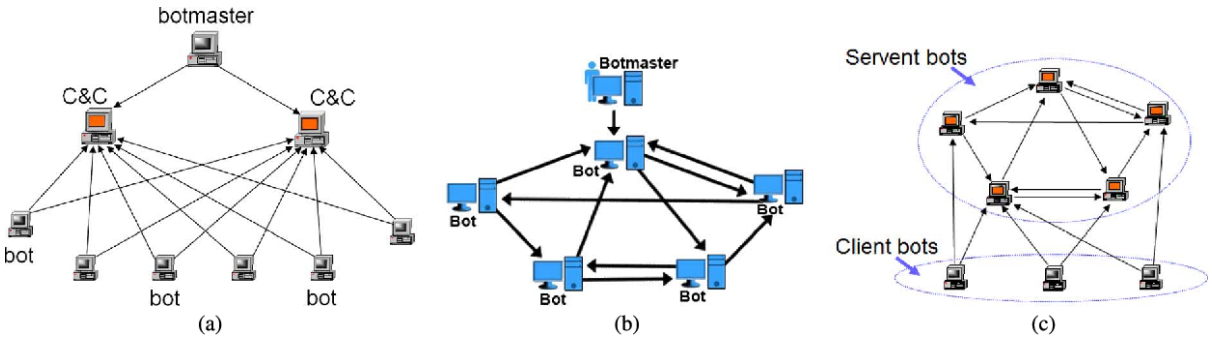

Figure 1: (a) Centralised (b) Decentralised (c) Hybrid

The decentralised botnet architecture has no command and control server as the botmaster can utilise any of the bot in sending out specific task and to gather information since there are all interconnected. Also, the decentralised architecture has an advantage of resisting botnet takedown attacks which the C2 servers are vulnerable to and its major flaw is that it lacks in terms of reliability and high latency in the communication within the bot network. Wang et al.[9], claim that the combination of both centralised and

decentralised eliminates the weakness exhibited by each of the architecture by providing a reliable transmission of commands. The hybrid architecture uses a proxy for the transmission to occur between the command and control and the bots which are connected similarly like a mesh network.Furthermore, additional layers can be put in place to protect the command and control servers at the expense of communication latency[10].

2.4 Botnet Detection

Recent research by Raghava et al.[11] and Chen et al.[12], classifies Botnets detection in general into five different categories which are signature-based, anomaly-based, DNS-based, machine learning-based and network-based detection.

The signature-based botnet detection approach was the initial method used in identifying of botnet activities and this leverages on the existing patterns of botnet discovered within a network traffic. This classical detection system acted more like an intrusion detection system (IDS) which detects malicious activities according to the signatures collected. Goebel and Holz [13], implemented a unique signature-based botnet detection technique which monitors and matches any anomaly detected within a network traffic by utilising the n-gram algorithm for scoring and analysing botnet. The merit of this method of detection is the provision of immediate classification, detection capabilities and the low rate of false positive involved. Also, the ever-evolving botnet nature gives this method a huge drawback of being unable to detect new botnets types.

Giroire et al.[14], claims that anomaly-based botnet detection tries to detect botnets in real time. This method is dependent on the abnormalities detected within a network traffic for example, the sudden increase in network traffic, abnormal system activity or high network latency. Siboni and Cohen[15] in their recent research paper, proposed an anomaly-based botnet detection system that is built on LZW[7] algorithm, a universal lossless data compression algorithm which makes it feasible to classify normal network traffic and predict the probability of new traffic to be classified. This type of botnet detection has a huge advantage in detecting botnets that are unknown and it can withstand known botnet resilience methods. The DNS-based botnet detection identifies botnets based on a specific DNS information of the bots. Raghava et al.[11], claims that the techniques implemented by this detection method shares some certain characteristics with the anomaly botnet detection and states that the anomaly detection algorithm is applied on DNS traffic.

The network-based botnet detection approach attempts to identify by inspecting network traffic in real-time and sharing few similarities to the signature-based botnet detection it is further classified into passive and active monitoring. The passive approach uses a packet inspection device to monitor incoming and outgoing network traffic with high latency in detecting botnet communication while the active approach measures the reaction of the network traffic by injecting test packets into the network.

2.5 Existing Machine Learning Methods

Machine learning algorithms are divided into three categories: Supervised, Unsupervised and the Semi-supervised learning algorithm [16]. The supervised learning approach makes use of labelled datasets in predicting the relationship between the existing and the new

[7] `https://en.wikipedia.org/wiki/Lempel%E2%80%93Ziv%E2%80%93Welch`

data. Hence, this approach is good for regression or classification purposes. Saad et al.[4] research led to a major development in the detection of peer-to-peer botnets using the network traffic behaviour in which about seventeen based features were selected and applied five supervised machine learning classifiers:Linear Support Vector Machine, K Nearest Neighbours, Gaussian Based, Naive Bayes and Artificial Neural Network and achieving an accuracy detection rate of 98% from the Linear Support Vector Machine. Beigi et al.[17], state that in their research using supervised machine learning approach had an accuracy of 99% on a unbalanced dataset and a 75% accuracy on a balanced dataset using the Decision Tree classifier C4.5 and extracting over twenty features which are further divided into time, byte, behaviour and packet.

Also, Zhao et al[18] used the Decision Tree classifier in detecting peer-to-peer botnet on a combined dataset and yielded a result of above 90% detection accuracy and a low false positive rate.Liao and Chang[19], claim that an accuracy results of about 87%, 89% and 98% and an accuracy count of 39,40 and 44 respectively was achieved during the extraction process of a peer-to-peer botnet using classifiers such as BayesNest, NaiveBayes and J48 Decision Tree using flows generated from softwares, online games and peer-to-peer bots. Kirubavathi and Anitha[20], suggest that the NaiveBayes classifier is the most suitable machine learning classifier for botnet detection while selecting four flow-based features: packet ratio ,small packet rate, bot response packet and initial packet length from six publicly available dataset achieving a false positive rate of 2% and an accuracy of 99% while comparing it with the SVM classifier and a boosted decision tree classifier of 92% and 95% respectively.

However, the unsupervised learning approach makes use of unlabelled datasets to learn about the distribution and prediction of unknown datasets and it is very efficient in carrying out tasks such as compression, feature extraction as well as clustering algorithm [16]. Li et al.[21], claim that the use of K-means unsupervised learning in conjunction with the particle swarm optimisation (PSO) approach in predicting the potential number of botnets which is based on three fundamental network behaviours of the infected botnet computers : Act, Scan and Fail behaviours, while suggesting that most approach tend to utilise the network flow attributes in conducting network analysis which has a drawback of when the network flow attributes are changed to avoid being detected or when they are encrypted.

The semi-supervised approach lies in between the supervised and the unsupervised approach and uses the unlabelled data in order to learn the probability distribution of an input space which is used in the optimisation of predicting over both the labelled and unlabelled datasets. Chen et al.[12], proposed a machine learning botnet detection approach for identifying peer-to-peer botnets by using an artificial neural network(ANN) for classification model and the application of convolutional features in extracting flow-based features from packet headers with a higher accuracy than when using the traditional flow-based extraction features achieving a detection accuracy of about 93% on known peer-to-peer botnet datasets with a false positive of above 2%.Also,Chen et al.[12] claim that their approach provides a significant confidence level when the botnet performance is enhanced with a decrease false positive rate of 1% and an increase in detection rate to above 98%

2.6 Conclusion

Based on the ever-evolving nature of botnets and the more resilience it gets we have reviewed existing approaches towards detecting this malicious software and identified loopholes in the literature and there is need to for the identification of botnets by implementing some machine learning algorithms in order to provide answers to the research question stated in subsection 1.2 and the objectives stated in subsection 1.3.In section 3, we present the methodological approach used in developing the various individual detection model to support our claim.

3 Methodology

The research methodology adopted for the purpose of this study was proposed by Fayyad et al.[22] which is popularly known as Knowledge Discovery Database (KDD) and this methodology follows a distinct process which will allow us in achieving our goal with a systematic procedure. This process consists of the following:

- Data Selection.
- Data Pre-Processing.
- Data Transformation.
- Implementation/ Data Mining.
- Evaluation.

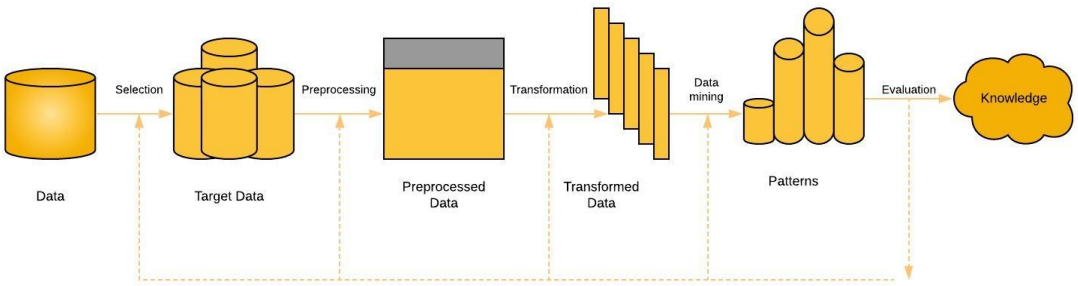

Figure 2: KDD diagram

3.1 Data Selection

Aviv and Haeberlen[23], describes the challenges in the development of a detection system is overwhelming as getting a well-trained classifier that is capable of distinguishing normal patterns and attack patterns is nearly impossible. Most datasets for studies in relation to botnet detection are limited in nature, consisting of less than four botnets sample which makes them unreliable in detecting new threats or new variants of existing threats. The aim of this study is to develop a model that can detect and predict a range of botnet attacks with a high efficiency, accuracy and performance therefore, a dataset that

represent such attacks is needed. The dataset in use for the purpose of this study is the ISCX Botnet dataset by Beigi et al.[17] which is publically available at the Canadian Institute of Cybersecurity[8] which was captured in a pcap format and converted into a csv format. The network flow traffic is bi-directional as it contains both malicious and non-malicious or benign network traffic with the malicious traffic consisting of several attacks such as Neris, Zeus, Sogou, Tbot and Smoke bot. Also, the dataset is suitable for a supervised machine learning approach as the dataset is labeled. The table below shows the comparison between existing dataset and their forensics limitation.

Dataset	Realistic Testbed Configuration	Realistic Traffic	Labeled Data	IoT Traces	Diverse Attack Scenarios	Full Packet Capture	New Generated Features
Darpa98	T	F	T	F	T	T	F
KDD99	T	F	T	F	T	T	T
DEFCON8	F	F	F	F	T	T	F
UNIBS	T	T	T	F	F	T	F
CAIDA	T	T	F	F	F	F	F
LBNL	F	T	F	F	T	F	F
UNSW-NB15	T	T	T	F	T	T	T
ISCX	T	T	T	F	T	T	T

Figure 3: Botnet dataset comparison

3.2 Data Pre-Processing

The dataset was transformed from its raw form which is a PCAP into a readable and understandable format such as CSV using flowtbag by Daniel Arndt which is made available publicly on GitHub[9], the dataset needs to be preprocessed hence, the ISCX Botnet dataset in its CSV format contains 145,700 rows from both TCP and UDP protocols and since the focus of this study is targeted at the Internet Relay Communication(IRC) botnets which utilizes the Transmission Control Protocol(TCP) streams hence, we remove all elements of the User Datagram Protocol(UDP) streams from the dataset. Secondly, in order to prevent the decrease in the accuracy of the training dataset, the dataset was checked for missing or null values and columns which meet this criteria were removed. Also,we removed botnets with different streams from the dataset as they contain botnet streams which uses other protocol streams

[8] https://www.unb.ca/cic/datasets/botnet.html
[9] https://github.com/DanielArndt/flowtbag

3.2.1 Data Labeling

The dataset is labeled with the benign traffic labeled with 0 and malicious traffic is labeled 1. Also the dataset contains botnet streams that utilizes other protocols and in order to avoid the impact these traffic will have on the classification due to their characteristics they were dropped.

3.2.2 Data Generation

The obtained dataset lacks some of the attributes needed for the purpose of this study because the dataset generator program divide packets into two types, forward and backward. Hence, we generated manually and added 11 attributes which are flow-total-bytes, flow-total-packets, flow-total-bits, bytes-per-packet, bits-per-sec, packet-per-sec, avg-var-iat, avg-iat, pct-packets-pushed, iopr and avg-payload-length.

3.2.3 Data Balancing

The malicious class is less than the benign class on a ratio of about 1:13 which makes the dataset to be unbalanced or biased as the total number of non-malicious class to be around 86695 and the malicious class with about 6379. The imbalance in the dataset can be solved by either under sampling the benign class or oversampling the malicious class hence, the former technique was selected so as to make the dataset balance 6379 of benign and malicious traffic was selected.

3.2.4 Data Segmentation

The dataset is further divided into two segment the training and the testing. The training dataset is used to build and train the classifier while the testing is used to evaluate the generalization capability of the model after building the model. The new dataset derived was split with 70% of the dataset allocated to the training data and the remaining 30% allocated to the testing data.

3.3 Data Transformation

Several data transformation techniques were deployed to ensure that the models do not overfit in respect to the training data to provide good detection accuracy on the test dataset. The method used based on the work done by previous authors were feature selection and dimensionality reduction in order to mitigate this issue. Dimensionality reduction is a method used in transforming features into lower transformation while feature selection involves the art of selecting and neglecting certain features without changing them. Feature selection it is further divided into univariate feature selection which works by using the univariate statistical test to select the best feature and the Recursive Feature Elimination (RFE) works by the elimination of the features that are of least importance and this process continues until a required condition is met and has a huge compatibility advantage of working perfectly with any machine learning model which assigns weights to features through feature importance or coefficient. For this study, the brute force search method will be used on the dataset which contains attributes created from the flow generator and that from previous studies with the aim of verifying how effective are the features historically used in this area, and also to identify if some of the features provided by the generator are effective in this process or not.

3.4 Data Mining

There are several machine learning models developed in order to achieve the perfect balance between accuracy and the model's performance and for this reason, the following machine learning models were selected: Support Vector Machine, Random Forest, Decision Tree and, AdaBoost. Support Vector Machine was selected because it achieved the highest accuracy in the experiment conducted by Saad et al.[4] while Decision Tree was used mostly by authors of previous research works. Random Forest and AdaBoost were selected because these ensembling methods have not been applied in the detection of botnets.

3.5 Evaluation

The main metrics considered for evaluating the accuracy and performance of the implemented models for this study, we are evaluating the four machine learning algorithms SVM, Random Forest, Decision Tree, and AdaBoost hence the evaluation metrics adopted are: includes sensitivity, accuracy, f score, precision and confusion metrics. Furthermore, Nguyen and Armitage[24], suggest that four parameters should be considered for the performance of a detection model which are: true positive, true negative, false positive and false negative.

3.6 Environment

The environment utilised for the purpose of this study was consistent across all stages of this research and the experiments were conducted using the python 3.5 distribution while utilising jupyter notebook and Anaconda framework as the sole integrated development environment (IDE) as they provide the user friendly platform when considering the Python environment and the packages used throughout the experiments and provides a step-by-step sections of the implemented solutions. Python language was selected above other programming language because it has an active community which offers a wide range of support in addition to it been an open source programming language. Furthermore, the experiments carried out for the purpose of this study was conducted on a 64-bit Microsoft Windows 10 pro operating system. The CPU of the system uses an AMD Ryzen 5 2500U with Radeon Vega Mobile Gfx at 2.00GHz with 8GB of RAM

4 Design Specification

4.1 Support Vector Machine Architecture

Support Vector Machine architecture similar to many machine learning models, however, it is a powerful learning model adopted for binary classification with its main goal is to locate the best hyperplane that can be used in the separation of data into classes hence, it is used to solve classification and regression problems. Also, it is one of the supervised machine learning model that which is well known and has been adopted by many authors due to its efficiency and effectiveness on datasets.

4.2 Decision Tree Architecture

A Decision tree is a supervised machine learning algorithm that deals with classification tasks. The decision tree architecture works by the creation of subgroups of the data in use and further split the result into several categories with similar groups. The C4.5 version of the decision tree has been widely used in previous studies relating to intrusion detection and botnet detection because of its efficiency in classifying malicious traffic from non-malicious traffic.

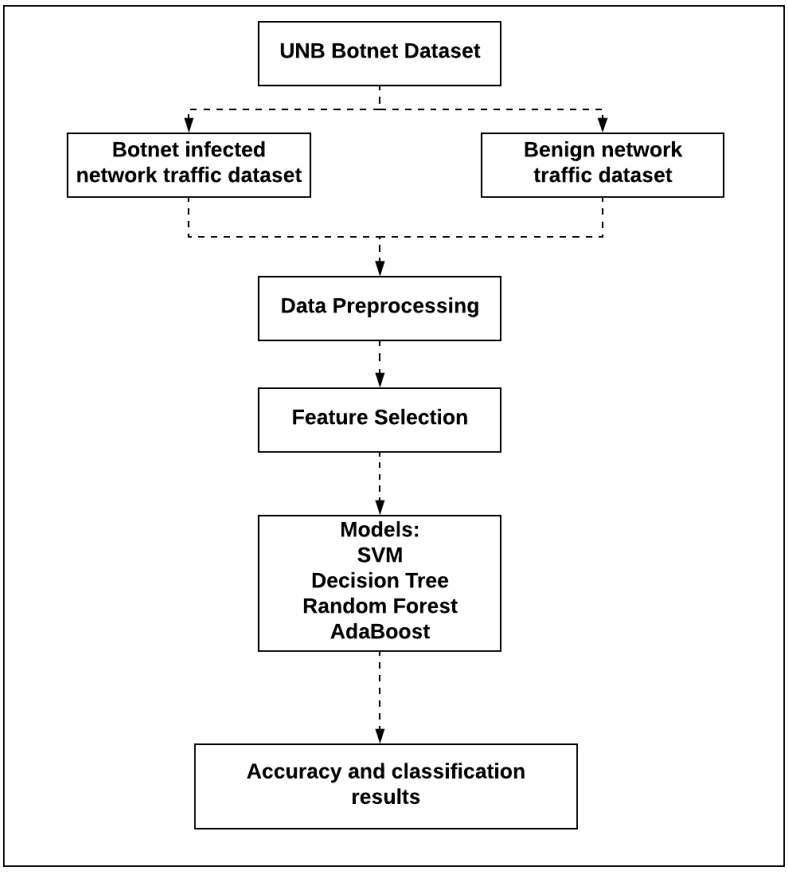

Figure 4: Design process flow

4.3 Random Forest Architecture

Random forest architecture is similar to the decision tree architecture except for the fact that it is an ensemble of decision trees with a majority voting technique used in aggregating the trees when performing classification or regression analysis. Random Forest like decision tree, is categorised under the supervised machine learning model and it requires large amount of data for it perform effectively.

4.4 AdaBoost Architecture

AdaBoost architecture is a combination of weak classifying algorithms in producing a strong machine learning classifier by selecting the training set based on the previous

outcome. A weak learning classifier often tends to classify the attributes of a data poorly, however a good accuracy score can be obtained by combining several multiple models with the selection of the training dataset at each iteration stage and applying the appropriate weight in the final voting process.

4.5 Scikit-learn Framework

The Scikit-learn framework was used extensively for the implementation of the models produced for this study, it is a combination of tools and state of the art algorithms for data mining which are efficient and yet simple. However, the merit of this python library is its simplicity and accessibility when utilising it which was a key factor when considering the proposed solution as several python packages such as matplotlib, SciPy and NumPy serves as its building block. Furthermore, scikit-learn adopts python modularity and interactivity in supplying fast and easy prototyping.

5 Implementation

The implementation of the solution proposed comprises of eight individuals files. The files consist of two dataset files,one which was generated by using the flow generator and the other generated after which the dataset was pre-processed. Two individual Jupyter notebooks with each corresponding to the data preprocessing and actual implementation of our models. Furthermore four individual text document files were utilised namely features, malicious-ips,irc_attacks, and other_botnets.

The first step in the implementation of this project was to convert our dataset from its natural form and converting it into a format that can be used by python, our preferred choice of programming language using the flow generator aforementioned in section 3.2 in the generation of our CSV file. However, the generated botnets data was still in its raw CSV form consisting of 44 attributes with no data header hence, the importation of the pandas library originally developed by wes McKinney which was used for manipulating and analysing of the CSV data imported into the python environment and appended the attributes names saved in the text document named features. The second step involves checking the imported data of null values with seven attributes were discovered to fall under this category which are:std_active, min_idle, mean_idle, max_idle, std_idle, furg_cnt, burg_cnt.

Also, the importation of the malicious_ips text document which contained two IP addresses 147.32.84.180 and 147.32.84.170 which are botnets IP addresses while the irc_attack text document contains a two-way streams that corresponds to an IRC attack hence, we labelled the data by verifying if the list of IP addresses in the malicious-ips or the irc_attacks text document is either within the source IP address or the destination IP address and then classify these flows with the label 1 as botnet or it is labelled as 0.

Thirdly, removal of other types of botnets IP address streams which are not related to IRC were removed which include the following IP addresses:147.32.84.160, 192.168.3.35, 192.168.3.25, 192.168.3.65, 172.29.0.116. Also, the dataset structure was checked to verify if the data been used is balanced or not and discovered that there was an imbalance problem with the dataset with the benign class significantly greater than the botnet class which can result in our accuracy prediction to be biased hence under-sampling the benign class adopted as stated in subsection 3.2.3. The obtained dataset lacked some of the

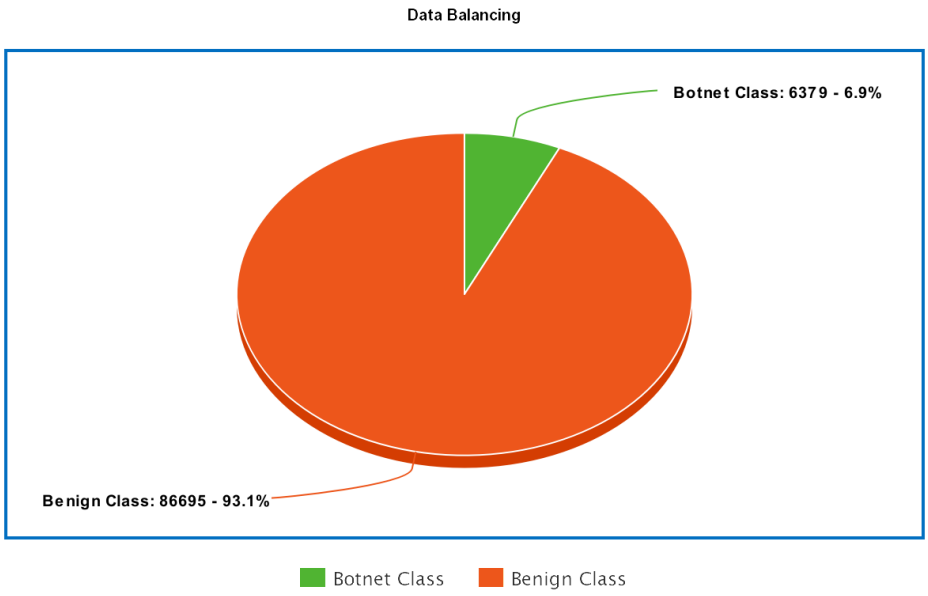

Figure 5: Malicious and benign botnet classes

attributes needed for the purpose of this study. Hence, we generated and added 11 attributes which are flow-total-bytes, flow-total-packets, flow-total-bits, bytes-per-packet, bits-per-sec, packet-per-sec, avg-var-iat, avg-iat, pct-packets-pushed, iopr and avg-payload-length. Finally,before exporting our pre-processed data, attributes with null values were dropped alongside sub-flows which are not relevant for the purpose of this study and in total seventeen attributes were removed which are: srcip, srcport, dstip, dstport, proto, std_active, min_idle,mean_idle, max_idle, std_idle,furg_cnt,burg_cnt,sflow_bpackets, sflow_bbytes, sflow_fpackets, sflow_fbytes,dscp.The data transformation technique adopted for this study is the Recursive Feature Elimination(RFE) as stated in section 3.3 above however, previous research showed the adaptation of the Principal Component Analysis(PCA) The models implemented for this study are Support Vector Machine, Random Forest, Decision Tree and, AdaBoost as discussed in section 4. The matplotlib library was of great use in designing the visualisations in the evaluation section.

6 Evaluation

In order to achieve the objectives stated in section 1.3 of this research, several experiments were carried out for the purpose of examining the accuracy and performance of the model implemented. The classification algorithms implemented as mentioned in section 4 are Support Vector Machine(SVM),Decision Tree,Random Forest, and AdaBoost. Also, the training dataset was divided into 70% for training and 30% for testing hence, the performance of the implemented algorithms will be based on confusion matrix as mentioned in subsection 3.5.The confusion matrix is based on the following measures:

- True Negative: numerical values of instances which are classified correctly as not been a member of the botnet class
- False Positive: numerical values of instances which are classified incorrectly as been a member of the botnet class

- False Negative: numerical values of instances which are classified incorrectly as not been a member of the botnet class
- True Positive: numerical values of instances which are classified correctly as been a member of the botnet class

6.1 Evaluation and Result of Support Vector Machine

The Support Vector Machine (SVM) is used in classifying classes within a dataset through which a given labelled training set utilises an algorithm and generates the result which is used to categorise new data.The result uses a hyperplane in dividing the plane where each of the classified classes falls on each side of the plane.The model was trained using several attributes and adopted the linear method for this study and achieved an accuracy result of 95.9% using the Support Vector Machine as seen in the table below.

Model	Accuracy	Precision	F-score	Recall
SVM	0.959	0.961	0.958	0.958

Figure 6: Accuracy of SVM

The model achieved a Precision score of 96.1% a F-score of 95.8% and a recall rate of 95.8% as shown above with a low false-positive of 3.5% and a true positive of 50.6%

Model	True Negative	False Positive	False Negative	True Positive
SVM	0.453	0.035	0.005	0.506

Figure 7: Confusion matrix of support vector machine

6.2 Evaluation and Result of Decision Tree

The decision tree adopts a tree-like structure of decisions calculating every possible outcome and consequences. The accuracy achieved by implementing this model is 95%, precision value of 95.2%, recall value of 94.9% and f-score value of 94.9% as seen in figure.

Model	Accuracy	Precision	F-score	Recall
Decision Tree	0.950	0.952	0.949	0.949

Figure 8: Accuracy of Decision Tree

Furthermore,The Decision tree classifier produced a false positive of 4%, a false negative of 0.9%, a true-negative of 44.7% and a true-positive of 50.2% respectively as shown in below.

Model	True Negative	False Positive	False Negative	True Positive
Decision Tree	0.447	0.040	0.009	0.502

Figure 9: Confusion matrix of Decision Tree

6.3 Evaluation and Result of Random Forest

The Random Forest is an ensemble of decision trees by applying the bagging technique. The random forest classifier is easy to implement, and outliers have little or no effect on it. The overall accuracy achieved by the random forest classifier is 99.6% with a precision value of 99.5% , a F-score value of 99.5%, and Recall of 99.5% respectively

Model	Accuracy	Precision	F-score	Recall
Random Forest	0.996	0.995	0.995	0.995

Figure 10: Accuracy of Random Forest

The Random forest classifier had false positive of 0.2%, a false negative of 0.1%, a true-negative of 48.5% and a true-positive of 51% respectively as shown in below.

Model	True Negative	False Positive	False Negative	True Positive
Random Forest	0.485	0.002	0.001	0.510

Figure 11: Confusion matrix of Random Forest

6.4 Evaluation and Result of AdaBoost

The accuracy achieved by the implementation of this model is 95%, precision value of 94.9%, recall value of 94.9% and f-score value of 94.9% as seen in figure below. However,The AdaBoost classifier produced a false positive of 4%, a false negative of 0.9%, a true-negative of 44.7% and a true-positive of 50.2% respectively as shown in below.

Model	Accuracy	Precision	F-score	Recall
AdaBoost	0.950	0.949	0.949	0.949

Figure 12: Accuracy of AdaBoost

Model	True Negative	False Positive	False Negative	True Positive
AdaBoost	0.447	0.040	0.009	0.502

Figure 13: Confusion matrix of AdaBoost

6.5 Comparison of Developed Models

All the implemented models SVM, Decision Tree, Random Forest and AdaBoost are compared below. From figure 14 the Random forest algorithm performed best in terms of accuracy, precision, recall or sensitivity, f-score, specificity and time taken to classify our instances. The support vector machine outperformed the decision tree and adaboost classifiers in accuracy, precision, recall, f-score and specificity but it is time consuming during classification time. The decision tree and the adaboost classifier produced similar result in terms of accuracy, recall, f-score and specificity with the tree classifier outperforming the adaboost in precision and runtime.

Model	Accuracy	Precision	F-score	Recall	Specificity	Runtime
SVM	0.959	0.961	0.958	0.958	0.928	1.140
Decision Tree	0.950	0.952	0.949	0.949	0.917	0.035
Random Forest	0.996	0.995	0.995	0.995	0.995	0.025
AdaBoost	0.950	0.949	0.949	0.949	0.917	0.064

Figure 14: Model Comparison

6.6 Discussion

Based on the Algorithms implemented, and the results produced for the purpose of this study, we have answered the question stated in subsection 1.2 as well as the objectives stated in subsection 1.3 have been justified and a reasonable result is achieved. The implemented models will significantly contribute to the ever-growing mitigation against botnet detection. Furthermore, this project does not address the dynamic nature of botnets which uses the peer-to-peer protocol.

7 Conclusion and Future Work

The proliferation of botnets brought about the different mitigation solution in the realm of cybersecurity and network traffic analysis has played a great part in understanding how these malicious activities are carried out. Furthermore, this detection method has been boosted significantly with the implementation of machine learning algorithms. In this study, we have attempted to show the comparison between the base learning models and the ensembling learning models in terms of a high detection accuracy and a low false positive. However, the experimental results show how effective and accurate machine learning classifiers are with the implemented models having a close accuracy detection similar to each other however, the random forest classifier achieved the highest detection rate which shows how important the random forest classifier is in botnet detection as our implementation was useful in detecting IRC botnets as this study was limited to botnets which uses this mode of communication. Future work can be done by combining several weak learning algorithms to produce a better accuracy and a low false positive value and applying the model in real time.

References

[**1**] K. Alieyan, A. ALmomani, A. Manasrah and M. Kadhum, "A survey of botnet detection based on DNS", *Neural Computing and Applications*, vol. 28, no. 7, pp. 1541-1558, 2015. Available: `https://doi.org/10.1007/s00521-015-2128-0.`

[**2**] A. Al-Nawasrah, A. Al-Momani, F. Meziane and M. Alauthman, "Fast flux botnet detection framework using adaptive dynamic evolving spiking neural network algorithm", 2018 9th *International Conference on Information and Communication Systems (ICICS)*, pp. 7-11, 2018. Available: `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8355433&isnumber=8355399`

[**3**] M. Stevanovic and J. M. Pedersen, "Machine learning for identifying botnet network traffic," Networking and Security Section, *Department of Electronic Systems, Aalborg University*, Tech. Rep., 2013.

[**4**] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, and P. Hakimian., "Detecting P2P botnets through network behavior analysis and machine learning", 2011 *Ninth Annual International Conference on Privacy, Security and Trust*, pp. 174-180, 2011. Available: `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5971980&isnumber=5971950`

[**5**] C. Haider, A. Iqbal, A. Rahman and M. Rahman, "An ensemble learning based approach for impression fraud detection in mobile advertising", *Journal of Network and Computer Applications*, vol. 112, pp. 126-141, 2018. Available: `https://doi.org/10.1016/j.jnca.2018.02.021.`

[**6**] Feily, A. Shahrestani and S. Ramadass, "A Survey of Botnet and Botnet Detection", 2009 *Third International Conference on Emerging Security Information, Systems and Technologies*, pp. 268-273, 2009. Available:`10.1109/securware.2009.48`

[**7**] G. Vormayr, T. Zseby and J. Fabini, "Botnet Communication Patterns", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2768-2796, 2017. Available: `https://ieeexplore.ieee.org/abstract/document/8026031`

[**8**] Mahmoud, M., Nir, M., and Matrawy, A. (2015). A Survey on Botnet Architec-

tures, Detection and Defences.*I. J. Network Security*, 17, pp.264-281.

[**9**] P. Wang, S. Sparks and C. Zou, "An Advanced Hybrid Peer-to-Peer Botnet", *IEEE Transactions on Dependable and Secure Computing*, vol. 7, no. 2, pp. 113-127, 2010. Available: `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4569852&isnumber=5465861`

[**10**] X. Dong, J. Hu and Y. Cui, "Overview of Botnet Detection Based on Machine Learning", 2018 *3rd International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, pp. 476-479, 2018. Available:`https://ieeexplore.ieee.org/document/8537604.`

[**11**] N. Raghava, D. Sahgal and S. Chandna, "Classification of Botnet Detection Based on Botnet Architechture", 2012 *International Conference on Communication Systems and Network Technologies*, pp. 569-572, 2012. Available: `https://ieeexplore.ieee.org/document/8455930`

[**12**] S. Chen, Y. Chen and W. Tzeng, "Effective Botnet Detection Through Neural Networks on Convolutional Features", 2018 *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp. 372-378, 2018. Available: `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8455930&isnumber=8455868.`

[**13**] j. Goebel and T. Holz, "Rishi: identify bot contaminated hosts by IRC nickname evaluation.", *In Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets (HotBots'07).*, p. 8, 2007. Available: `https://dl.acm.org/citation.cfm?id=1323136`

[**14**] F. Giroire, J. Chandrashekar, N. Taft, E. Schooler and D. Papagiannaki, "Exploiting Temporal Persistence to Detect Covert Botnet Channels", *Lecture Notes in Computer Science*, pp. 326-345, 2009. Available: `https://link.springer.com/chapter/10.1007/978-3-642-04342-0_17`

[**15**] S. Siboni and A. Cohen, "Botnet identification via universal anomaly detection", 2014 *IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 101-106, 2014. Available:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7084311&isnumber=7084286.

[**16**] J. Lee, J. Shin and M. Realff, "Machine learning: Overview of the recent progresses and implications for the process systems engineering field", *Computers & Chemical Engineering*, vol. 114, pp. 111-121, 2018. Available: `10.1016/j.compchemeng.2017.10.008`

[**17**] E. Biglar Beigi, H. Hadian Jazi, N. Stakhanova and A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches", 2014 *IEEE Conference on Communications and Network Security*, pp. 247-255, 2014. Available:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp= &arnumber=6997492&isnumber=6997445.

[**18**] D. Zhao et al., "Botnet detection based on traffic behavior analysis and flow intervals", *Computers & Security*, vol. 39, pp. 2-16, 2013. Available: https://www.sciencedirect.com/ science/article/pii/S0167404813000837?via%3Dihub

[**19**] W. Liao and C. Chang, "Peer to Peer Botnet Detection Using Data Mining Scheme", 2010 *International Conference on Internet Technology and Applications*, pp. 1-4, 2010. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber= 5566407&isnumber=5566070.

[**20**] G. Kirubavathi and R. Anitha, "Botnet detection via mining of traffic flow char-

acteristics", *Computers & Electrical Engineering*, vol. 50, pp. 91-101, 2016. Available: https://doi.org/10.1016/j.compeleceng.2016.01.012

[**21**] X. Li, J. Wang and X. Zhang, "Botnet detection technology based on DNS.", Future Internet 2017, vol. 9, no. 4, pp. 1-12, 2019. Available:https://www.mdpi.com/1999-5903/9/4/55/pdf.

[**22**] U. Fayyad, G. Piatetsky-Shapiro and P. Smyth, "Knowledge Discovery and Data Mining: Towards a Unifying Framework", *Data Mining and Knowledge Discovery*, pp. 82-88, 1996. Available: https://www.aaai.org/Papers/KDD/1996/KDD96-014.pdf.

[**23**] A. Aviv and A. Haeberlen, "Challenges in Experimenting with Botnet Detection Systems", *Proceedings of the 4th Conference on Cyber Security Experimentation and Test*, p. 6, 2011. Available: http://dl.acm.org/citation.cfm?id=2027999.2028005.

[**24**] T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning", *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 56-76, 2008. Available: 10.1109/surv.2008.080406.

[**25**] L. Rokach, "Ensemble-based classifiers,"*Artificial Intelligence Review*, vol. 33, no. 1,pp. 1–39, Feb 2010. Available:[Online]. Available: https://doi.org/10.1007/s10462-009-9124-71.