MACHINE LEARNING-BASED FRAMEWORK FOR AUTOMATIC MALWARE DETECTION USING ANDROID TRAFFIC DATA

UZOMA RITA ALO[1,*], HENRY FRIDAY NWEKE[2], SYLVESTER I. ELE[1, 3]

[1]Computer Science/Informatics Department, Alex Ekwueme Federal University, Ndufu-Alike, Ebonyi State, Nigeria
[2]Computer Science Department, Ebonyi State University, P.M.B, 053, Abakaliki, Ebonyi State, Nigeria
[3]Computer Science Department, University of Calabar, Etagbor, P.M.B 1115, Calabar, Cross River State, Nigeria
Email: [1]alo.uzoma@funai.edu.ng*, [2]henry.nweke@ebsu.edu.ng, [1, 3]myyrs2015up@gmail.com

*Corresponding Author

ABSTRACT

One of the greatest challenges facing various organizations and institutions is information security. Attackers have devised means to steals mobile user identity by developing malware that might be inadvertently installed by users due to the open source nature of android operating system causing financial loses. Although various machine learning algorithms have been proposed recently for malware detection, it is challenging to detection malicious apps with single classification model. In this paper, we propose to detect malicious apps in android traffic using four (4) different machine learning algorithms. The proposed approach was evaluated on comprehensive and publicly available dataset. The result obtained shows that decision tree and tree based ensemble algorithms produced superior results when compared with support vector machine and logistic regression models. The results suggest the impact of multiple classification algorithms to improve the performance of malware detection system. The finding can be utilized to guide security expert on the use of machine learning methods to detect malicious software.

Keywords: *Malware Detection, Machine Learning Algorithms, Mobile Network Traffic, Multiple Classifier System, Cybersecurity Systems*

1. INTRODUCTION

The global community has witnessed a significant upsurge in the amount of malware proliferation over the Internet. Report available from Anti-virus (AV)-test statistics (2018) shows an exponential growth of malwares, from 24 million in 2009 to 811.34 million in 2018. According to recent report by Per Galtan [1], the numbers of Android users attacked by banking malware has increased to 90% in 2018, with 1.8 million of such attacks being impacted between 2018 and 2019. Even though such attacks decreased from 1.8 million to 786,325 in 2016 and 515,816 in 2017, in April 2018 the number of malware attacks grew on a severely increasing trend. Kaspersky Lab's "Financial Cyber threats in 2018 report asserted that Asacub peaked more than double to nearly 60%, followed by Agent (14.28%) and Svpeng (13.31%). All three experienced volatile growth in 2018; especially Asacub as it peaked from 146,532 attacked users in 2017 to 1,125,258 users.

Malware is a software designed to infiltrate and harm a mobile or computer devices without the owner's prior knowledge. Malware detection has become a central component in information security systems. Today, due to the sophisticated method of malware techniques applied by malware perpetrators, zero-day attacks and false positives has become the most challenging problem in the malware detection area. Recent researches indicate that between 1997 and 2006, financial loss accruing from malware attack has grown from $3.3 billion to $13.3 billion [2]. Research data available also reveals that in 2016, WannaCry ransomware crypto-worm which besieged computers using the Microsoft Windows OS crashed the computers of more than 150 countries, causing financial damage resulting in loss of data organizations' data systems. Again, in 2016, Cybersecurity Ventures reported that the projected total damage caused by malware attacks was $3 trillion in 2015 and is likely to reach $6 trillion by 2021 [2, 3]. Palmer [4] also noted that a new variation and degree of an infamous banking Trojan malware with a history dated back to over a decade has emerged with a novel tactics to make sure it is harder to detect. Based on the studies by [4], the aim of the malware is to track financial information, usernames, passwords and other responsive and sensitive data. This malware, as revealed by [4], is one of the most popular forms of information-stealing malware with the objective of attacking Windows PCs and it has been in existence in variety of forms since 2017.

With the aim of combating malware attacks, various mobile applications have adopted traditional methods such as the use of antivirus software, signature-based and heuristic based methods. Antivirus is an application design to search, prevent and remove harmful application from digital devices or protect users from being infected with virus or mobile malware. However, the use of antivirus is not efficient due to continuous signature based update requirement. Moreover, mobile malware is constantly modified to evade antivirus detection [5]. Signature based method extract distinctive signature from collected malware files in order to identify malicious files [6, 7]. Nevertheless, it is easy for perpetrators to change malware signature to minimize malware detection rate. On the other hand, heuristic methods detect malware by differentiating between benign and malicious behavior. The process involves analyzing the activities of malware during run time and tagged benign or malicious depending on the behavior or pattern identified in the training data [8].

Malware detection approaches can be further divided into static, dynamic or hybrid methods [9]. Static analysis is based on the assessment of suspected code running the application without executing them. Static features such as binary sequence, sequence function calls, requested permission are extracted from the code snippets to determine whether app is malicious or not. Static analysis method is resilient to changes in malware code. However, the methods can be easily disrupted by obfuscation techniques [10]. Meanwhile, dynamic method executes code in control environment such as emulator to detect its behavior. Nonetheless, the efficiency of dynamic method depends on its ability to detect malicious behavior at runtime.

There have been increased use of machine learning algorithms for malware detection in mobile devices and network traffic [9, 11].The machine learning algorithms such as decision tree, support vector machine, K-nearest neighbors, clustering and ensemble methods have been proposed in recent time to classify mobile network traffics as benign or malicious. For instance, Nataraj et al [12] developed malware detection system using K-nearest neighbors (K-NN) and image based features extracted from malware binaries. However, it can easily be evaded by attackers due to the vast amount of redundancy in the image data.

Hou et al [13] proposed heterogeneous information network (HIN) based methods to represent the relationship between API calls and app permission for mobile malware detection. Similarly, Sun et al [14] implemented extreme learning machine based algorithm to detect mobile malware based on features such as permissions, API calls extracted from malware data. Other machine learning classifiers that have played vital roles in mobile malware detection include support vector machine [15], decision tree [16], Naïve Bayes [17], Neural Networks [18], Ensemble learning [19], Logistic Regression [20], and clustering methods [21]. Machine learning is a class of algorithms that allow models to learn from training data. Adopting machine learning models to build malware detection has proven to improve accuracy and generalization of the detection system [11].

The aim of this study is to evaluate various machine learning algorithms for detection of malware in mobile traffic data. We assess four (4) classifiers namely decision tree (DT), Support vector machine (SVM), logistic regression (LR) and Ensemble methods (Adaboost and RUSboost) using 12, 551 malware samples from android network traffic [22]. The paper presents an extensive evaluation to determine the effectiveness of the above classification algorithms to detect whether a mobile android traffic is *Benign* or *Malicious.* In the nutshell, the contributions of this paper are as follows:

- Comprehensive review of conventional machine learning based malware detection methods;
- Develop a machine Learning based framework for malware detection using different classifiers;

- Evaluation of four (4) machine learning classifier (DT, SVM, LR and ensemble methods) for malware detection;
- The proposed evaluation method employed 13 mobile malware features extracted from Android network traffics generated using mobile applications;
- Extensive evaluation of various model parameters and detection speed to ensure efficiency.

The remainder of the paper is organized as follows. Section 2 provides brief overview of related works on malware detection using machine learning. Section 3 presents the proposed malware detection system. Section 4 introduces the experimental settings. Section 5 presents the results while sections 6 concludes the paper.

2. REVIEW OF RELATED WORKS

This section highlights relevant literatures on android based malware detection methods. The section provide overview of various malware detection methods, their strength and weakness. Summary of machine learning approaches for malware detection is presented in Table 1.

2.1. Review of Malware Detection Methods

Meticulously, malware has been the most potent and devastating of major tools used by perpetrators to initiate a wide variety of security attacks, such as identity theft, theft of confidential and private information, unauthorized networks penetration, and destructions of critical infrastructures [2, 3]. Consequently, to combat malware in android devices and mobile phone networks, methods such as installation of Antivirus, signature-based and heuristic based methods have been implemented by security experts to minimize the impact of malware in android devices. However, issues such as continuous signature update requirement, easy modification by perpetrators to evade detection have greatly limited efficient performances of these traditional malware detection approaches [5, 6, 8]. In the past decades, studies have proposed machine learning and feature representation methods for android based malware detections. The features for malware detection can be broadly divided into static, dynamic and hybrid-based methods. Static based features for malware detection.

Static analysis is based on the assessment of suspected code running the application without executing them. Static features such as Hash values, binary sequence, N-gram, Opcode, sequence function calls, requested permission are extracted from the code snippets to determine whether app is malicious or not. Static analysis method is resilient to changes in malware code. On the other hand, dynamic methods involve execution of file system operation, and key registry changes in behavior in control environment to detect malware. However, in static analysis, assembly code is examined for to find assembly code patterns that contain malicious activity. This process is time consuming and difficulty. Moreover, static analysis can be easily disrupted by obfuscation techniques [10]. Code Obfuscation enable attackers to implement code encryption, program code reordering and dead code insertion to evade malware analysis [23]. Moreover, the efficiency of dynamic method depends on its ability to detect malicious behavior at runtime. To resolve the problem inherent in tradition malware analysis require careful and automatic feature representation using machine learning methods.

Recent studies in malware detection and classification has shown the important of machine learning to categorize sample android data into malicious or benign [23]. Machine learning uses mathematical, statistical and artificial intelligence techniques to automatically learn from sample data, and have been widely used in the field of network and mobile security such as intrusion detection systems, phishing, and malicious code snippets. The major advantage of machine learning algorithms such as decision tree, support vector machine, Logistic regression, Naïve Bayes, Neural Networks and Ensemble methods is their ability to efficiently detect unknown malware by automatically extracting insight from malware samples. Moreover, machine learning methods can be generalized to unknown malware variations [5, 9, 10, 12].

2.2. Machine Learning Approaches For Malware Detection

Machine learning methods involve various stages such as data collections, pre-processing, feature extraction and normalization, modeling and evaluation. These stages are deployed to train malware detection system. In this section, we briefly review different machine learning algorithms that have played key role in malware detection.

Recently various studies have attempted to implement machine learning algorithms for malware detection [24]. Chumachenko [25], researched on machine learning methods for malware detection and classification. The goal of the study was to verify the more efficient feature extraction, feature representation, and classification methods that can result in the best accuracy when used on Cuckoo Sandbox. The study evaluated several machine learning methods using 1,156 files of 9 malware families of different types and 984 benign files of various formats. The result showed that classification based on machine learning are efficient for classification of various android based malware. Similarly, Saxe et al [26] investigated the impact of Deep Neural Network on Malware Detection Using Two Dimensional Binary Program Features. The study showed that the system achieved a 95% detection rate at 0.1% false positive rate (FPR), with an experimental rate of dataset of more than 400,000 software binaries.

In a related research, Cuan et al [27] developed malware detection in PDF files using machine learning. The study implemented a support vector machine learning (SVM) with a gradient descent to counter the impact of android based malware attacks. The result showed that the SVM was able to detect 99.7% of malware. Other studies [28, 29] have also shown promising results in malware detection using machine learning methods where the focus include visualization of malware samples, external host monitoring, and genetic algorithm based feature extraction and representation. Varieties of other machine learning algorithms such as decision tree, K-nearest neighbors, Support vector machine, Naïve Bayes, Random forest, Logistic Regression, clustering algorithms, and Deep learning algorithms have also been implemented for malware detection and classification. These machine learning models that have played pivotal role in malware detection are briefly explained below.

2.2.1. Decision Tree

Decision tree is a machine learning algorithms that is made up of different node segments. To implement malware detection system using decision tree, the malware data sample is recursively partitioned into node segments that include the root node, internal node and leave [30]. Decision tree model is a non-parametric supervised machine learning that are mainly used for classification and regression tasks such as human activity recognition, image classification [31], and malware detection. Moreover, decision tree does not require assumption on the training features to predict if malware sample is benign or malicious using simple decision rule. The algorithm is simple to understand, computationally efficient and provide hierarchical representation of malware classes [30]. Recently, various studies [16, 32, 33] implemented decision tree for malware detection. However, issues such as overfitting, data sample variation might minimize the performance of decision tree algorithm.

2.2.2. Support vector machine

Support vector machine is a classification and regression algorithm based on statistical learning theory developed by Vapick et al [34], and can efficiently perform non-linear data classification. Support vector machine deploys hyperplane that separate the training malware sample using maximal margin [27], thereby dividing the malware classes into Benign and Malicious. Moreover, it is predominantly utilized for malware detection [11] due to its diversity, robustness and ability solve small scale or high dimensional data. For instance, Jingmei et al [35] proposed incremental learning method based on multi-class support vector machine for malware detection by leveraging 600 malware sample data. The proposed method was able to learn new sample and detect if such malware belong to new class or old class of malware. Then, it update the all old classification plane for every malware samples. This helps to minimize prediction errors. Other studies [15, 36]. However, Support vector machine require high and extensive data pre-processing to achieve optimal detection accuracy, and it is sensitive to missing values [9].

2.2.3 Logistic regression

This is a simple but effective machine learning model for classification and regression problem. Logistic regression is a parallelizable machine learning algorithm that provide easy interpretation of training samples feature vectors. Furthermore, it models the relationship between malware samples and classes in order to accurately detect whether such sample is malicious or not [20]. Even though logistic regression model's performance is affected by non-linear data, the algorithm have been widely applied in implementation of malware detection system due to its low computational resources requirement and simplicity [20, 37].

2.2.4 Random forest

Random forest algorithm is a collection of multiple decision tree first introduced by Breiman [38] for sample data classification and regression. Each decision tree is composed of random vectors that are identically and independently distributed while the final results are combined using voting techniques. Random forest is categorized as ensemble learning and implement in various stages which include training data collection, choosing the random decision tree, decision split computation and combining the individual decision tree using majority voting [38]. Recent years have seen researchers in mobile security demonstrating the implementation of random forest for malware detection and classification [39, 40].

2.2.5 Naïve Bayes

Naïve Bayes classifier is an effective classification algorithm for supervised learning and pattern recognition. The algorithm is computationally efficient, fast and simple to train [41]. However, Naïve Bayes based algorithm is not applicable to situation where the feature sets are correlated. Furthermore, it require extensive calculation of prior probability to achieve optimal results [9]. Naïve Bayes classifier have been extensively applied for malware detection [17, 42, 43].

2.2.6 K-Nearest neighbors

K-Nearest Neighbors (K-NN) model is a simple and efficient classification algorithm for malware detection and other classification tasks. It is a non-parametric and lazy learning algorithm based on the principle of instance learning that store malware data instances and classify new training data using similarity index measures. The commonest similarity index used in K-NN is the Euclidean distance. Moreover, K-Nearest Neighbors algorithm is effective in handling large data sample, and provide fast and accurate detection of malicious samples [12]. Studies such as [44, 45] have recently implemented K-NN for malware detection. The Major noticeable drawbacks of K-NN include high computation, and the performance is greatly affected by skewed data.

2.2.7 Deep neural networks

Artificial neural network (ANN) is a branch of artificial intelligence that apply human learning process to gain insights from data. It contain number of artificial neurons that re used to identify information store. ANN is composed of various layers such as input layer, hidden layer and output layer [46]. These layers are used to transform input data into desirable outputs. In most of the application of artificial neural networks, multiple hidden layers are stacked together to form deep neural network (DNN). The algorithm is essentially used for both data classification and prediction. Recently, deep neural network methods have been implemented for automatic feature representation using deep learning methods [47]. Deep learning algorithms automatically discover discriminant features from training data, and the approaches have been widely applied in human activity identification [48], image classification, and natural language processing [47] and recently in malware detection [49]. Various deep learning models have been investigated for malware detection and classification. These include deep neural networks, convolutional neural network, long short term memory etc. [18, 50, 51]. The main attraction of using deep learning model for malware detection lately is its ability to automatically learn high level feature vector from unlabeled data thereby produce high accuracy. However, deep neural network models require extensive hyper-parameter tuning. The use of high number of hyper-parameter increases the computation time of deep learning.

2.2.8 Multiple classifier systems (ensemble learning) methods

Major studies in malware detection have proposed multiple classifier systems (Ensemble) methods for malware detection and classification of malware families according to recent review [52]. Multiple classifier systems involves integration of individual machine learning algorithm to arrive at consensus in order to improve accuracy, robustness, and performance generalization. In this case, homogeneous or heterogeneous

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

classification algorithms thereby reduce uncertainty and ambiguity by integration of output generated by individual classification algorithms to improve the detection results of each individual classification algorithm [31]. Combination strategies employed by multiple classifier systems to combine individual classification algorithm results include majority voting, posterior probabilities, mean aggregation, weighted summation and Dempster -Shafer theory [31]. Studies [19, 53, 54] have achieved comparable results in network security and malware detection by implementation of multiple classifier systems. Major challenges in the implementation of multiple classifier system based malware detection include high computation cost, increased complexity, difficulty to implement in real time and high maintenance cost [9].

Table 1. Summary of recent machine learning algorithms implementation for malware detection, descriptions, strengths and weaknesses

Machine learning Methods	Description	Strengths	Weaknesses
Decision Tree[16, 32, 33]	Non-parametric supervised machine learning model that recursively partitions training data into various segments.	Simple to understand and interpret. Can handle samples with missing values or large scale	Prone to overfitting and does not support online learning
Support Vector machine [15, 35, 36]	Statistical learning theory based algorithm that use hyperplane to divide training data into separate classes	Essential for solving small scale and high dimensional or non-linear problems	Sensitive to small sample with missing values
Logistic Regression [20, 37]	Parallelizable machine learning algorithm that model the relationship between training data and class labels	Has less computational overhead. Moreover, it is simple and easy to implement	Difficult to solve non-linear problem
Random Forest [39, 40]	Combine multiple decision trees based on independent subsets of the training data.	Very efficient for large scale malware data. It is robust for estimating missing data.	Complex and difficult to understand as it combine multiple decision tree
Naïve Bayes [17, 41, 42, 43]	Computationally efficient model that provide independent assumption among training data and class label	Easy and quick to train, require less amount of training data and work well with noisy data	The classifier assume that data elements are independent of each other.
K-NN [12, 44, 45]	Non-parametric and lazy learning algorithm which classifies training instances by considering the class label of the k nearest training data	Important classification algorithm for implementing multi-classification problem. Does not make any assumption about the dataset.	Has high computational time, and difficult to select the appropriate value of k.
Deep Neural Networks [18, 50, 51]	Artificial intelligence approach that mimic the human brain to gain insight from data.	Produce high accuracy even for non-linear dataset with high number of input features.	Require lots of training data and hyper-parameter initialization. Moreover, it is computationally expensive to train.
Multiple Classifier Systems (Ensemble) [19, 53, 54]	Ensemble learning methods which combine multiple homogeneous or heterogeneous classifiers to improve classification accuracy and generalization.	Produce more accurate results when compared with single classification algorithms	High computation cost and increased complexity

3. PROPOSED MALWARE DETECTION SYSTEMS

This section describes the methodology adopted to develop the automatic malware detection model using android traffic data. The section is divided into subsections, and include data description, pre-processing methods, feature extraction, description of the machine learning models and evaluation of the proposed malware detection systems. Detail flow of the proposed approaches is shown in Figure 1.

3.1. Data Collection

To evaluate the proposed malware detection system using various machine learning models, we utilized data collected in the DroidCollector

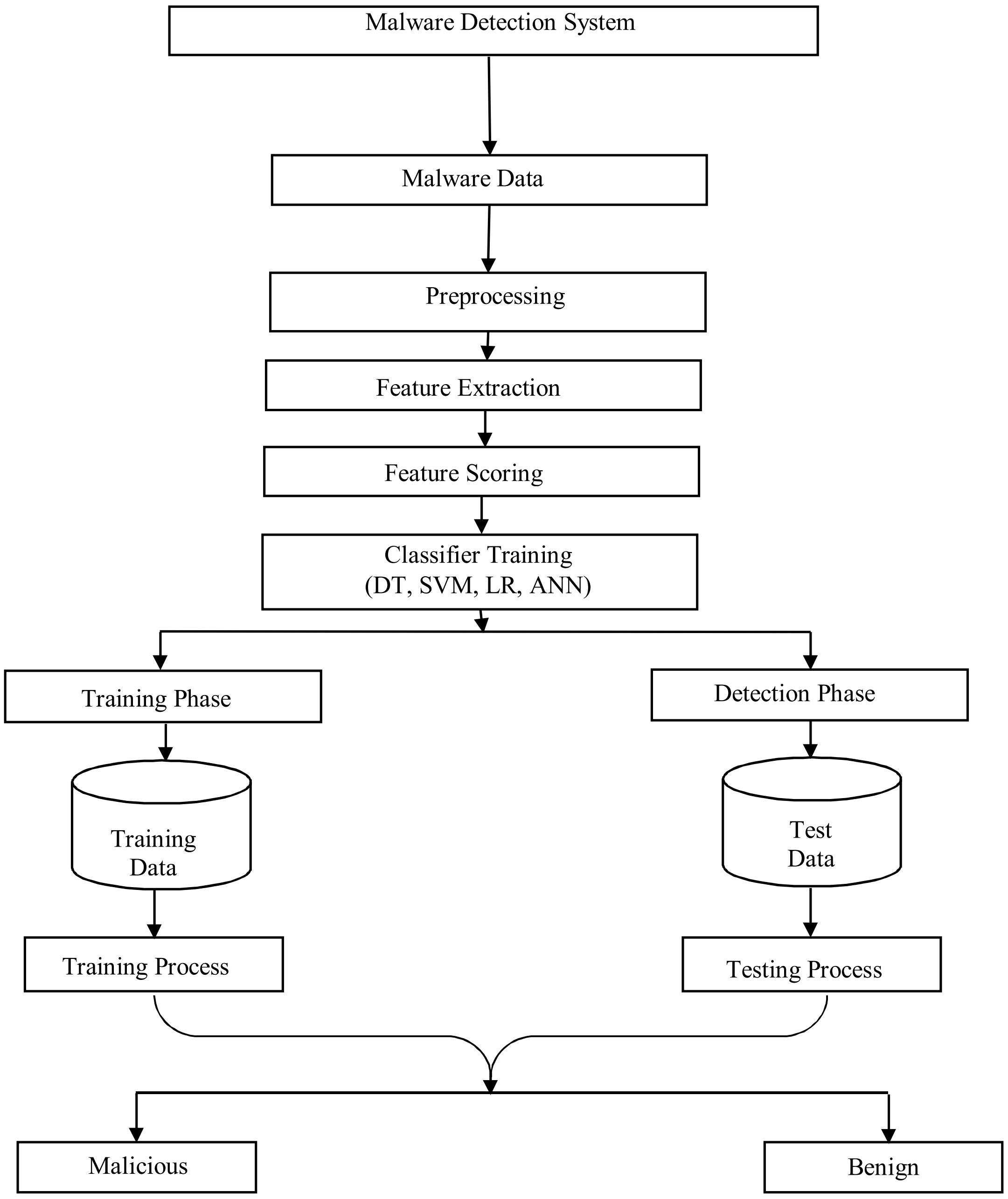

Figure 1. Proposed Malware detection System

project by Cao Dong et al [22]. The project was commissioned to collect benign and malicious traffic for the purpose of android based malware detection. The malware data contains 4,705 benign web traffic and 7,846 malicious web traffic categorised into *antivirus, communication, DailyLife, educational, finance, health and fitness apps* etc. for benign samples, and *Ackposts, Adrd, BaseBridge, DroidDreams, DroidKungFu, DroidRooter apps* etc. for malicious sample. The details of the dataset and collection protocols are explained in [22]. The dataset was downloaded from Kaggle website (https://www.kaggle.com/xwolf12/network-traffic-android-malware).

3.2. Pre-processing

During data collection, android malware data for malware detection might be unsatisfactory due to issues ranging from inconsistence specification, redundancy, missing values and imbalance distribution of the data [5]. The essence of pre-processing stage is to remove irrelevant data in order to improve the proposed malware detection models. Pre-processing techniques include replacing missing values, data cleaning, data transformation and correcting inconsistency.

In addition, redundant features increase the computation time and reduce the accuracy of malware detection algorithms. The dataset used to evaluate the proposed model contain missing values (NAN), which we replaced with previous values in the list.

3.3. Feature Extraction

In machine learning based malware detection, extraction of discriminant features for training the learning model is an important stage. Feature extraction process reduces the malware data into feature vectors that are discriminant enough to describe whether a particular mobile app is malicious or benign. In addition, features help to minimize computation time, reduce data dimensionality, remove noisy and redundant data and minimize the classification error of the learning models [11]. Feature for malware detection consist of elements such as application behaviour on the network, application behaviour on the mobile devices, app permission, Java code and certification. In this study, we utilized features based on mobile network traffics to detect if a particular mobile application is benign or malicious. The features selected are based on the previous studies [22, 55]. The features include *TCP packets, Different TCP packets, external IP addresses, Volume of byte, UDP packet, Package of source application, remote application package, byte of source application, byte of remote applications and DNS query*. Table 2 lists these features and descriptions. The features were stored as MATLAB table files, and each record contains summary data of 13 selected features of network traffics. In addition, the labelling consists of malware data categorised as *Benign* or *Malicious.*

Table 2. List Of Features For Training The Malware Detection System And Descriptions

Feature set	**Descriptions**
TCP packet	Number of TCP packet sent and received during app communication.
Different TCP packet	Total number of packets that have ports other than those exposed to TCP
External IP	Number of addresses external IP to which the application communicated with
Volume of byte	The number of bytes sent from the applications
UPD packet	Whole amount of packets transferred by UDP
Package of source application	Total number of packages sent from the application in remote server.
Remote application packets	Number of packets transferred and received from source outside the mobile apps
Byte of source application	The amount of communication in bytes between the application and server
Byte of remote application	The amount of data from server to the mobile emulator
DNS query	Number of DNS queries
Average packet rate	Average volume of packets transferred per second
Duration	The time taken to transfer a packet
DNS query time	The time taken in second to issue DNS query

3.4. Machine learning models

In order to ensure comprehensive evaluation of the proposed malware detection system, we evaluated various machine learning algorithms. The machine learning algorithms implemented include Decision tree (DT), Logistic Regression (LR), Support Vector Machine (SVM) and Ensemble Classifier. These classification algorithms are briefly explained below.

3.4.1. Decision Tree

Decision Tree (DT) is a machine learning algorithm that iteratively divides training malware data into node segments consisting of root node, internal nodes and leaves. As a non-parametric algorithm, decision tree does not require training assumption on training data. In addition, it models the non-linear relationship between training data and malware classes (*Benign* or *Malicious*). Decision tree were recently implemented for Malware detection and classification [16].

3.4.2. Logistic Regression

Logistic Regression (LR) is a Machine learning algorithm that provides easy interpretation of training data. Due to its simplicity, fastness and compactness, the algorithms have been extensively implemented for malware classification [20].

3.4.3. Support Vector Machine

Support vector machine (SVM) is a statistical learning algorithm that makes use of hyperplane to separate the training data into maximal margin. Support vector machine was developed by Vapnick et al [34] for data classification. The main advantage of the algorithm for malware detection is the ability to solve small scale, high dimensional data or non-linear problem. The algorithm was recently implemented for malware detection [15].

3.4.4. Ensemble learning algorithm

Ensemble learning algorithms are multiple classifier approach implemented to reduce data uncertainty and handle high dimensional data. Ensemble classifier integrates different or same classification algorithms to arrive at a consensus to improve accuracy and generalization of the learning algorithms [31]. In this study, we utilized Adaboost and RUSboost as ensemble learning algorithms for malware detection. Adaboost combines multiple weak decision tree algorithms into a single strong classifier. The process involves putting more weights on the difficult to classify instances. The algorithm was implemented for malware detection [9] recently. RUSboost is an ensemble algorithm for solving the problem of class imbalance by combining data sampling and boosting methods to improve malware detection algorithm [56, 57].

3.5. Evaluation

The performance of the proposed machine learning-based malware detection system was evaluated using different performance metrics. These performance metrics include accuracy, True positive rate (TPR), True Negative rate (TNR), false positive rate (FPR), False Negative Rate (FNR), ROC, Area under the Curve (AUC), Recall and Confusion matrix. Some of the performance metrics are shown in the Table 3 below with their corresponding measurement equation, and descriptions.

Table 3. Performance Evaluation Measures

Evaluation measures	**Equations**	**Description**
Accuracy	$\frac{1}{N}\sum_{i=1}^{N}\frac{(TP+TN)}{(TP+FP+TN+FN)}$	Compute the number of correctly identified malware classes out of the total number of all data instances
Recall or Sensitivity	$\frac{1}{N}\sum_{i=1}^{N}\frac{(TP)}{(TP+FN)}$	Measure the number of correctly predicted malware classes
True Negative Rate (TNR)	$\frac{TN}{TN+FP}$	Measure the ratio of benign samples wrongly identified as malware

False Negative Rate (FNR)	$\frac{FN}{FN+TP}$	The ratio of malware samples incorrectly identified as benign
True Positive Rate (TPR)	$\frac{TP}{TP+FN}$	Measure the ratio of actual malware identified as malware
False Positive rate (FPR)	$\frac{FP}{FP+FN}$	Compute the ratio of actual malware correctly identified as malware
Area under the curve (AUC)	$\frac{1}{N}\sum_{i=1}^{N} 0.5*\left[\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right]$	Measure the rate of performance Malware detection system. AUC is the plot between recall and specificity drawn from the different threshold
Note: ***TP***: is the number of malware files correctly identified as malware files, (True Positive). ***TN***: is the number of benign files correctly identified as benign files (True Negative). ***FP***: is the number of malware files incorrectly identified as benign files (False Positive). ***FN***: is the number of benign files incorrectly identified as malware files (False Negative).		

4. EXPERIMENTAL SETTINGS

This section explains the experimental process, and how the results were analyzed. The first is the initial analysis; the statistics of the raw data was examined. The original dataset contained 12,551 sample of converted CSV with 13 attributes (variable) comprising of two unique classes (Malicious and Benign) as explained in section 2.1 (data collection). To test the performance of the framework in this paper, four (4) different machine learning classifiers (Algorithms), namely: Support Vector machine (SVM), Decision Tree (DT), Logistics Regression (LR) and Ensemble Classifier were trained using MATLAB R2016a classification Learner APP. Different learning options under each classifier gallery were trained and their respective performance documented in other to find out which machine learning options gives the best model with our data, as represented in table 1. In addition, the machine learning models were implemented using the MATLAB 2016a default parameters settings.

The experimental analysis and investigation was implemented using Classification Learner APPs in MATLAB 2016 (R2016a), running on 64 bit Window 7 OS. The system is using an Intel(R) Core(TM) i3 3110M @2.4GHZ with RAM capacity of 6.00GB.

5. RESULTS AND DISCUSSION

The Four (4) machine learning algorithms Support vector machine, logistic regression, Decision tree were trained, with each classifier, except the Logistic Regression, containing different learning model options. For the purpose of defining the appropriate model that best suit our dataset, all model options under each classifier (algorithm) were trained, and their statistical measures, percentage accuracies and prediction speeds documented as represented in Table 4.

Table 4. Statistical Indicators of the Different Classification algorithms

CLASSIFIER/ ALGORITHM	MODEL/ PRESET	PREDICTION SPEED (obs/sec)	ACCURACY (%)	FDR (%)	ROC	AUC	TPR (%)	TNR (%)	FPR (%)	FNR (%)	PPV(p-value) (%)
SVM	Fine Gaussians	27000	100.0	1.0	1.0	1.0	99.9	99.9	1.0	1.0	99.9
DECISION TREE	Complete Tree	190000	100.0	0.0	1.0	1.0	100.0	100.0	0.0	0.0	100.0
ESSEMBLE CLASSIFIER	Tree based	32000	100.0	0.0	1.0	1.0	100.0	100.0	0.0	0.0	100.0
LOGISTIC REGRESSION	Logistic Regression	130000	100.0	0.0	1.0	1.0	100.0	100.0	0.0	0.0	100.0

The aim of our study is to develop a machine learning based framework for malware detection and to determine, among other measures, the quality (ROC) of different classifiers applied in the framework and to know which one of these classifiers that can produce best performance (Accuracy) in our model. Other metrics considered for the evaluation of the framework were the False Discovery Rate (FDR), which is a more conservative approach for addressing multiple tests. The FDR (q-value) is developed to control the percentage of false positives among groups of rejected hypothesis (R) that is incorrectly rejected hypothesis. Our training test for all the classifiers returned FDR (q-value) of 0.00, except for ensemble classifier that returned an FDR (q-value) of 0.22 (22%) each for *Adaboost* and RUSboost, and SVM returned an FDR (q-value) of 0.01 (1%) each for Fine Gaussians, medium SVM and coarse Gaussians. The relationship between FDR and other metrics such as PPV, TPR and TNR is illustrated in the Confusion Matrix shown in Figure 2 as generated during the data training regime.

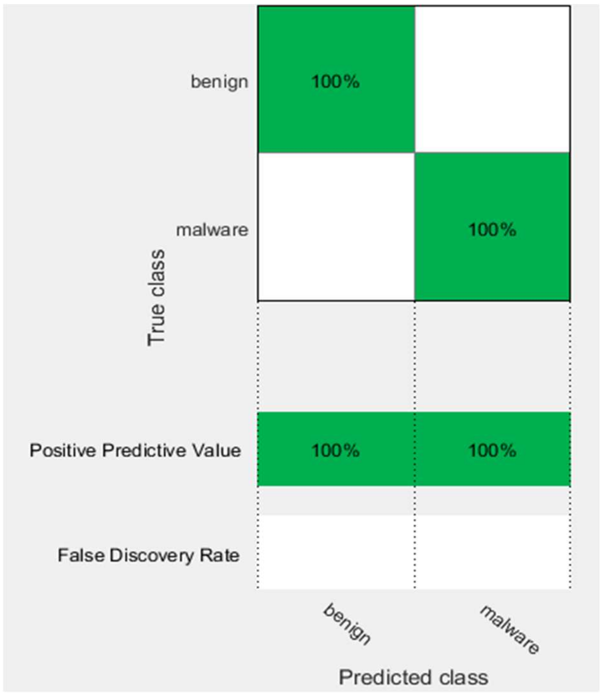

Figure 2. Confusion Matrix Indicating the PPV, FDR, TPR and TNR

5.1 Performance Evaluation of the Impact of Different Model Option in Malware Detection

The benchmark to reflect the performance of a classifier or any machine learning algorithm is its Accuracy (ACC). In this paper, the accuracy indicates the ability of the classifier to correctly predict malware samples as malware samples and benign files were predicted as benign, expressed in percentage (%), in addition to the speed of the prediction. For our training setup, as illustrated in Table 4, Figures 3 and 4 respectively, the maximum accuracy level and prediction speed of (100%, 27000 obs/sec), (100%, 190000 obs/sec), (100%, 32000 obs/sec), and (100%, 130000 obs/sec) were obtained when the dataset was trained and tested with SVM (Fine Gaussians), Decision Tree (complete tree), ensemble classifier (bagged tree), and Logistic Regression classifiers, respectively. From the result, Support Vector Machine (SVM) had 100% accuracy level like other classifiers, however it requires as little as 27000 obs/sec to train in Fine Gaussians. Consequently, SVM with Fine Gaussians is best suited for our model in terms of Accuracy and prediction speed.

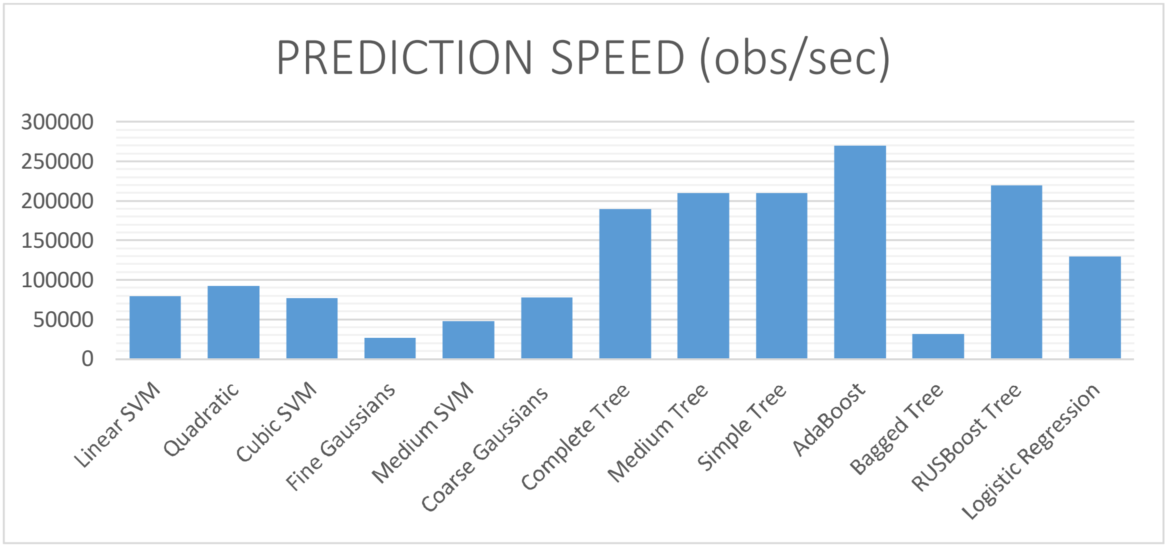

Figure 3. Prediction Speed value (obs/sec) of trained Classifiers using all Models under each Classifier

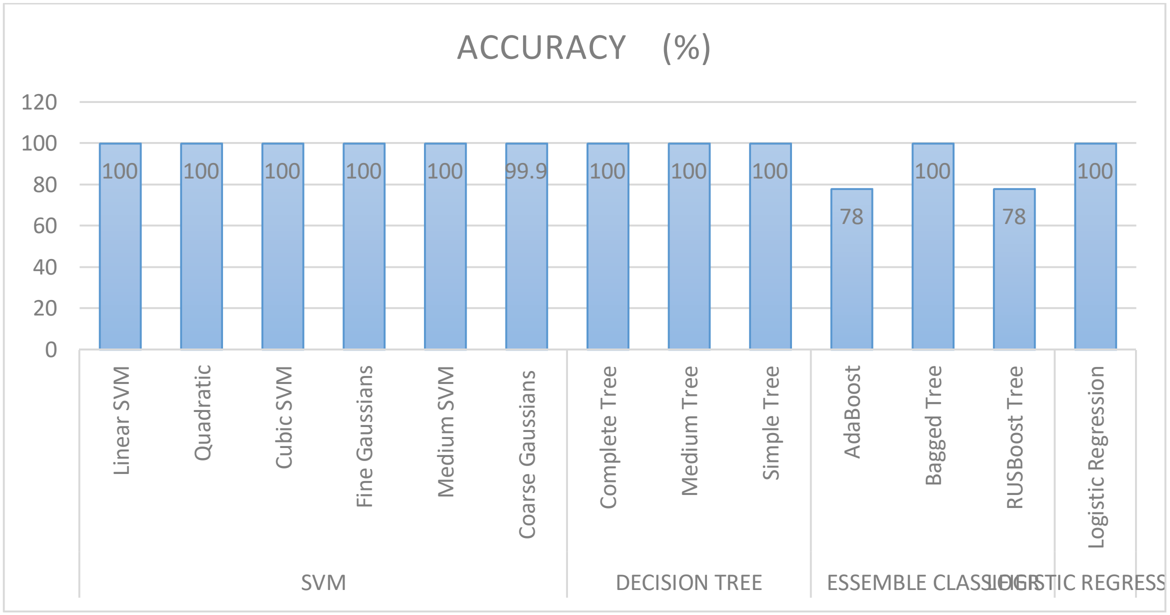

Figure 4: The Accuracy Level Value (%) Of The Trained Classifiers Using All Models Under Each Classifier

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5.2. Quality of Classifiers (ROC and AUC) in Malware Detection

This section discusses the quality of classifiers in malware detection in terms of Receiver operating characteristics (ROC) Curve and Area under Curve (AUC) as a metric to measure the quality of the classifiers. ROC applies a threshold to the outputs via the interval (0,1). ROC is defined by the ratio of the TPR, on the vertical (Y) axis and the FPR, on the horizontal (X) axis. Figure 5 represents one of the best ROC curves.

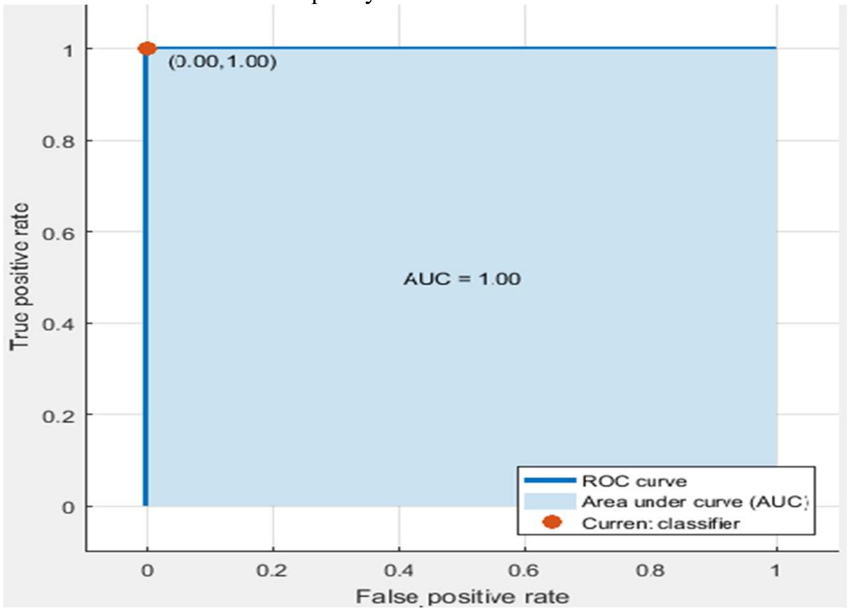

Figure 5. ROC Curve Using The SVM Model

As illustrated in Table 4, Figures 6 and 7, our model recorded ROC of 1.00 and AUC of 1.00, respectively for SVM, Decision Tree, and Logistics regression, while ensemble classifier (Adaboost) recorded an ROC 1.00 and AUC of 0.58. The ROC and AUC values of the first three classifiers are sufficient enough to make our framework a good prediction model for malware detection.

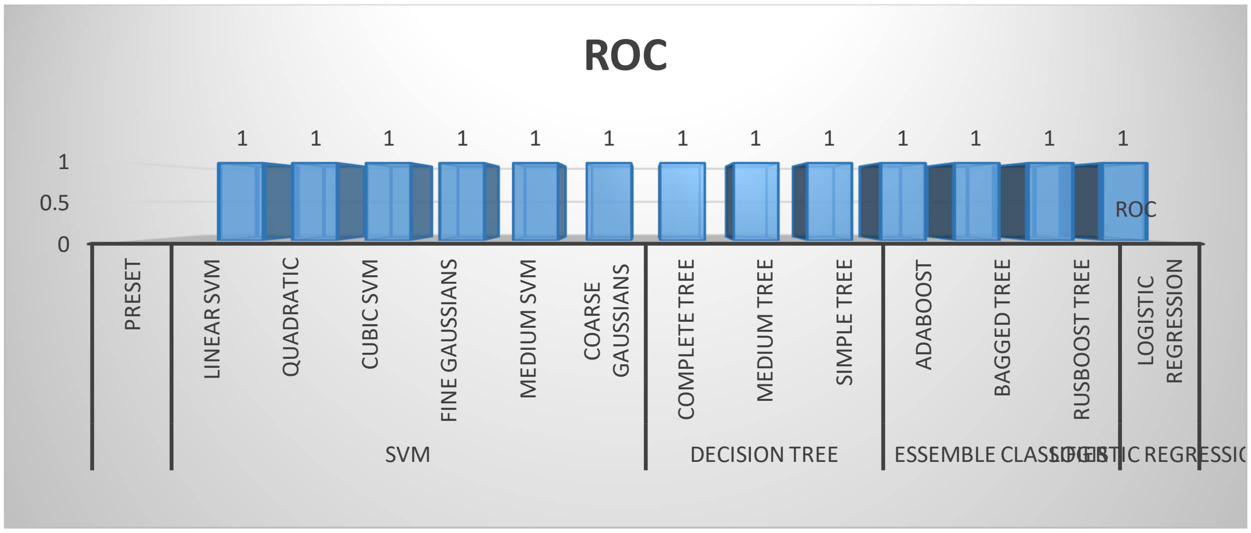

Figure 6. ROC Curve Of Values Of The Classifiers Using All Models Under Each Classifier

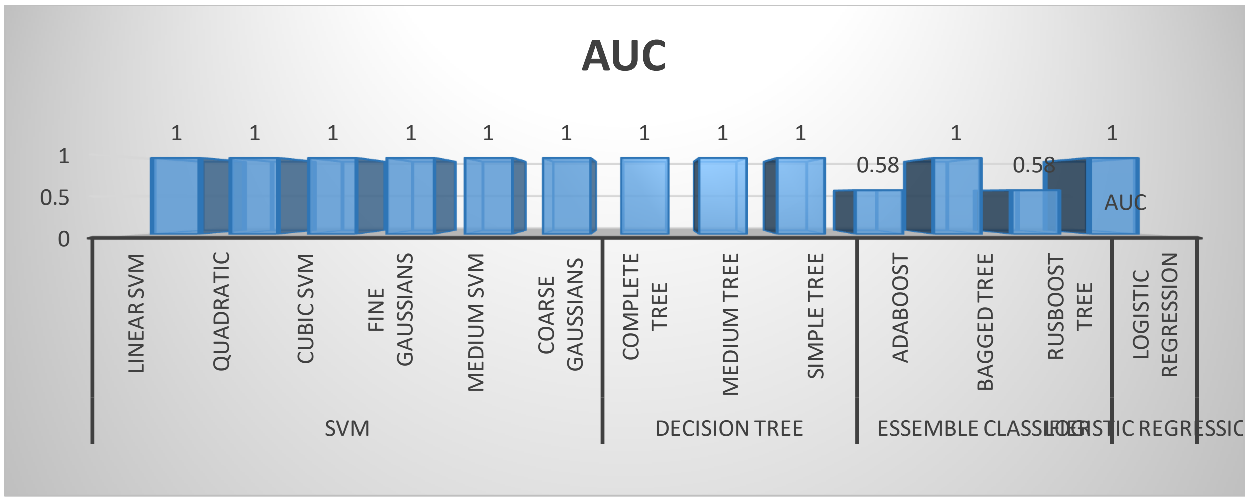

Figure 7. AUC Curve Values Of The Classifiers Using All Models Under Each Classifier

5.3. Evaluation of Positive Prediction Value (PPV) in Malware Detection

Positive predictive value (PPV) defines the probability that a malware sample with a positive screening examination is truly *Malicious*. The PPV column in Table 3 of our result describes the likelihood that those prediction that returns positive, the sample actually contain malware. From Table 3 and Figure 8, PPV of 100% (p-value = 1.00) was recorded when the system was trained with SVM (Linear, Quadratic and Cubic SVM); Decision Tree; all model under Ensemble classifier; and Logistics Regression. The result indicate that when the model was trained in SVM, DT, Ensemble classifier and Logistic regression, among those sample that returned positive, the probability of it being malware is 100%.

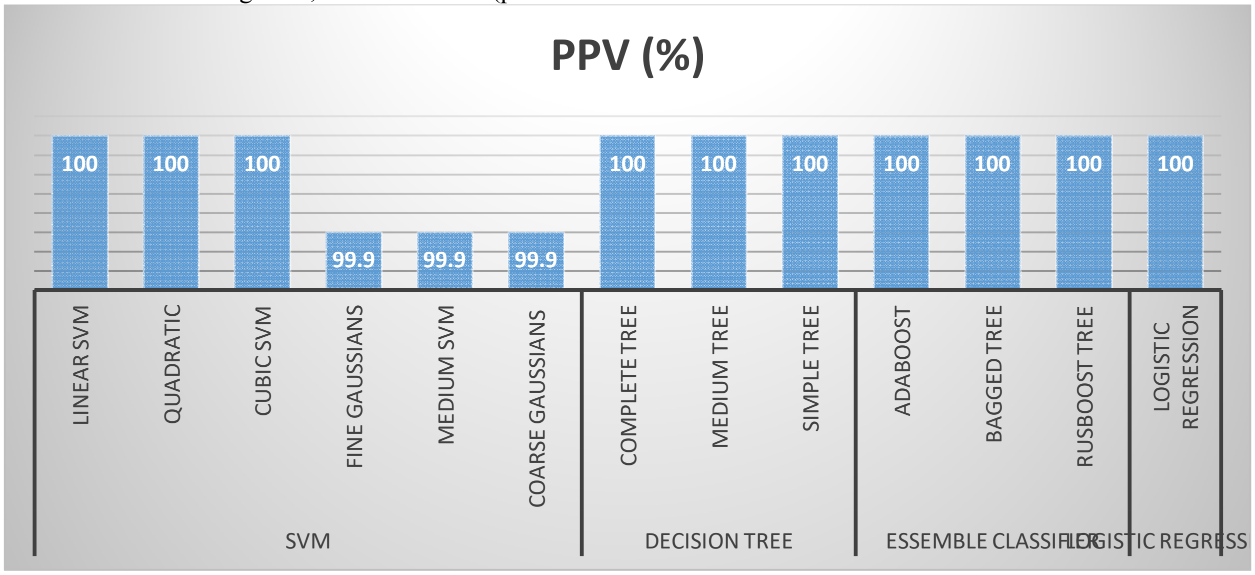

Figure 8. PPV (P-Value) Distribution Of The Trained Classifiers Using All Models Under Each Classifier

6. SIGNIFICANCE OF THE PROPOSED MALWARE DETECTION METHODS

Several studies have proposed machine learning model for malware detection using mobile apps traffic data. These models have claimed various performance accuracy ranges, however, it is difficulty to comprehensively assess the performances of these single classification algorithms. According to the no free lunch theorem of machine learning proposed in [58], no single classification algorithm is sufficient or perform best in all data mining domain. Therefore, it is importance to verify the performance of various machine learning model and then choose the best performing models. The machine learning models utilized which include decision tree, logistic regression, support vector machine and ensemble classifier achieved promising results at detecting if sample of network traffic data collected with mobile devices is malicious or benign. From the experimental results, support vector machine, logistic regression and decision tree achieved accurate and reliable results of 100% detection accuracy. The lowest results was achieved obtained by ensemble classifiers (Adaboost and RUSboost trees) that achieved detection accuracy of 78% accuracy respectively (Figure 4).

To investigate the significance of the implemented machine learning models for malware detection, we compared the results obtained with recent studies. Three (3) recent studies were chosen for this comparison. These studies implemented support vector machine [59], logistic regression [20], decision tree [32] and boosting algorithms [20]. In Leed et al [59], data permission of API of 24444 benign and 870 malicious applications were extracted as features. The features were fed as input to support vector machine for malware detection. Suhuan et al [20] evaluated logistic regression for malware detection using traffic data. Here, features such as frequency of API calls, app permission, amount of API and API names were extracted from mobile apps data containing 568 malicious apps and 566 benign apps data. In addition, the study extracted N-gram model of list of grams containing API calls sequence information were extracted from the data. Furthermore, Singh et al [32] extracted system call behaviour of 216 malicious apps and 278 benign aps to train decision tree (J48) for dynamic malware detection. The results of comparison are shown in Table 4. The comparison analysis of different studies that used similar classification models for malware detection show that our proposed outperformed recent studies in terms of performance accuracy. While [59] in their study obtained 80% accuracy, experimental results by [20] achieved 88.7% accuracy. Our study achieved approximately 100% accuracy using support vector machine outperformed detection accuracy of recent studies. The performance results obtained with our proposed evaluation models and features demonstrated that the approaches provides better accuracy, robustness and generalization when compared with recent studies.

Table 4: Comparison With Recent Similar Studies

Studies	Machine learning models	Average Accuracy
Leed et al [59]	Support vector machine	80.0%
Suhuan et al [20]	Logistic regression	88.7%
Singh et al[32]	Decision tree	97.58%
Our evaluation methods	Support vector machine	100.0%

7. CONCLUSION AND FUTURE WORKS

The objective of this paper is to advance anti malware performance and make effective and appropriate evaluation of machine learning algorithm based malware detection techniques. We evaluated four (4) machine learning models for malware detection. Different Machine Learning classifiers such as SVM, Decision Tree, Logistic Regression and Ensemble learning classifiers were applied in the framework, with different model options under each classifier trained and their performances compared based on certain statistical indicators. The performance criteria adopted to evaluate the classifiers were ROC, Accuracy (ACC), AUC, FDR, PPV, TPR, and TNR. The result of our findings shows that all the classifiers applied in the model training performed with optimum proficiency and adeptness sufficient enough to make our a good prediction model for malware classification and detection. However, the Decision tree and ensemble classifiers represented the highest and maximum level of accuracy and classification quality, with each of them recording accuracy (ACC) of 100%, ROC of 1.00, PPV of 100% (p-value = 1.0), TPR of 100%, TNR of 100% and FDR of 0.00 (q-value = 0). Hence, DT and ensemble classifiers are recommended for our framework. In future, this work would be expanded by collecting large malware traffic data and compare the results with publicly available datasets. In addition, implement multiple classifier systems and deep learning models for android malware detection and classification.

CONFLICT OF INTEREST

The authors declare no conflict of interest

REFERENCES

[1] S. Galtan, 1.8 Million Users Attacked by Android Banking Malware, 300% Increase since 2017. Bleeping Computing. Available at: https://www.bleepingcomputer.com/news/security/18-million-users-attacked-by-android-banking-malware-300-percent-increase-since-2017/. Retrieved: 9 March 2019.

[2] O. E., David and N. S., Netanyahu. “Deepsign: Deep learning for Automatic Malware signature generation and classification”. In Neural Networks (IJCNN), 2015 *International Joint Conference on, IEEE*, 2015.

[3] W. Hardy, L. Chen, S. Hou,Y. Ye, X. Li. “Dl4md: A Deep Learning Framework for Intelligent Malware Detection.” *In Proceedings of the International Conference on Data Mining (DMIN). The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing*, 2016.

[4] D. Palmer. “Point of Sale Malware Campaign Targets Hospitality and Entertainment Business.” ZDNet Security Reports. Retrieved at: https://www.zdnet.com/article/point-of-sale-malware-campaign-targets-hospitality-and-entertainment-businesses/. Date: September 13, 2019: 3.00 pm.

[5] F. A. Narudin, A. Feizollah, N. B. Anuar & A. Gani. “Evaluation of machine learning classifiers for mobile malware detection.” *Soft Computing*, Vol. *20 no.*1. 2016. pp 343-357.

[6] J. Yan, Y. Qi, & and Q. Rao. “Detecting Malware with an Ensemble Method Based on Deep Neural Network.” *Security and Communication Networks*. 2018. Hindawi. https://doi.org/10.1155/2018/7247095.

[7] S. Shalev-Shwartz, & S. Ben-David. “Understanding Machine Learning: From Theory to Algorithms.” Cambridge University Press. 32 Avenue of the Americas, New York, NY 10013-2473, USA. 2014. Available at: http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning. Retrieved: June 6, 2019:1.37 am.

[8] Z. Bazrafshan, H. Hashemi, S. M. H. Fard&A. Hamzeh, A. “A Survey on Heuristic Malware Detection Techniques.” *IKT, 2013 5th Conference on Information and Knowledge Technology*, 2013, pp. 113–120.

[9] K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun and H. Liu, "A Review of Android Malware Detection Approaches Based on Machine Learning," in *IEEE Access*, vol. 8, pp. 124579-124607, 2020, doi: 10.1109/ACCESS.2020.3006143.

[10] W. Enck, D. Octeau, P. McDaniel, S. Chaudhuri, “A study of android application security”, *In Proceedings of the USENIX security symposium*, Vol. 2, 2011, pp. 1-2.

[11] A. Feizollah, N. B. Anuar, R. Salleh, &A. W. A. Wahab. “A review on feature selection in mobile malware detection.” *Digital investigation*, Vol. *13*, 2015. pp. 22-37.

[12] L. Nataraj, S. Karthikeyan, G. Jacob, &B. Manjunath. “Malware Images: Visualization and Automatic Classification.” *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, 2011. Article No. 4.

[13] S. Hou, Y. Ye, Y. Song, and M. Abdulhayoglu, ``HinDroid: An intelligent Android malware detection system based on structured heterogeneousinformation network,'' in *Proc. ACM Conf. Knowl. Discovery Data Mining (KDD)*, Halifax, AB, Canada, 2017, pp. 1507_1515.

[14] Y. Sun, Y. Xie, Z. Qiu, Y. Pan, J. Weng, and S. Guo, ``Detecting Android malware based on extreme learning machine,'' in *Proc. IEEE 15th Int.Conf. Dependable, Autonomic Secure Comput., 15th Int. Conf. Pervas.Intell. Comput., 3rd Int. Conf. Big Data Intell. Comput. Cyber Sci.Technol. Congress (DASC/PiCom/DataCom/CyberSciTech)*, Nov. 2017, pp. 47_53.

[15] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, ``Drebin: Effective and explainable detection of Android malware in your pocket,'' in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2014, pp. 23_26.

[16] J. Li, Z. Wang, T. Wang, J. Tang, Y. Yang, and Y. Zhou, ``An Android malware detection system based on feature fusion,'' *Chin. J. Electron.*, vol. 27, no. 6, Nov. 2018. pp. 1206_1213.

[17] T. Chen, Q. Mao, Y. Yang, M. Lv, and J. Zhu, ``TinyDroid: A lightweight and efficient model for Android malware detection and classification,'' *Mobile Inf. Syst.*, vol. 2018, Oct. 2018, Art. no. 4157156.

[18] J. Yan, Y. Qi, and Q. Rao, ``Detecting malware with an ensemble method based on deep neural network,'' *Secur. Commun. Netw.*, vol. 2018, Mar. 2018, Art. No. 7247095.

[19] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, and H. Ye, ``Significant permission identification for Machine-Learning-Based Android malware detection,'' *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3216_3225, Jul. 2018.

[20] L. Suhuan and H. Xiaojun, "Android Malware Detection Based on Logistic Regression and XGBoost," *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, 2019, pp. 528-532, doi: 10.1109/ICSESS47205.2019.9040851.

[21] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, ``Crowdroid: Behavior-based malware detection system for Android,'' in *Proc. 1st ACM WorkshopSecur. Privacy Smartphones Mobile Devices (SPSM)*, Chicago, IL, USA, 2011, pp. 15_26.

[22] D. Cao *et al.*, "DroidCollector: A High Performance Framework for High Quality Android Traffic Collection," *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016, pp. 1753-1758, doi: 10.1109/TrustCom.2016.0269.

[23] J. Singh, J. & J. Singh. "A survey on machine learning-based malware detection in executable files." *Journal of Systems Architecture*, 2020. p.101861.

[24] K. Khan, A. Mehmood, S. Khan, M.A. Khan, Z. Iqbal, W.K. Mashwani. "A survey on intrusion detection and prevention in wireless ad - hoc networks," *J. Syst. Archit.* 2020. 101701, http://dx.doi.org/10.1016/j.sysarc.2019.101701.

[25] K. Chumachenko. "Machine Learning Methods for Malware Detection and Classification. Bachelor of Engineering (B Eng.) Project submitted to the Department of, Computer Engineering and Computer Science, University of Applied Sciences.

[26] J. Saxe & K. Berlin. "Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features. 2015 *10th International Conference on Malicious and Unwanted Software (MALWARE).* 20-22 Oct. 2015.

[27] B. Cuan, A. Damien, C. Delaplace and M. Valois. "Malware Detection in PDF Files Using Machine." 2018. HAL Id: hal-01704766 https://hal.archives-ouvertes.fr/hal-01704766v2. Retrieved: Retrieved: 7 March 2019.

[28] S. S. Nancy, and U. Chourasia, "A Survey over the Various Malware Detection Techniques used in Cloud Computing." International Journal of Engineering Research & Technology (IJERT). 2016

[29] P. Srivastava1, and M. Raj, "Feature extraction for enhanced malware detection using genetic algorithm." *International Journal of Engineering & Technology.* Vol.7, No. 8 2018 pp. 444-449.

[30] J. R. Quinlan "Induction of decision trees. *Mach Learning* Vol. 1. 1986. pp. 81–106

[31] H. F. Nweke, Y. W. Teh, G. Mujtaba, M. A. Al-garadi. "Data fusion and multiple classifier systems for human activity detection and health monitoring: Review

and open research directions.” *Inf. Fusion* Vol46. 2019. Pp.147–170

[32] L. Singh and M. Hofmann, ``Dynamic behavior analysis of Android applications for malware detection," in *Proc. Int. Conf. Intell. Commun. Comput. Techn. (ICCT)*, Dec. 2017, pp. 1_7.

[33] Z. Ma, H. Ge, Y. Liu, M. Zhao, and J. Ma, ``A combination method for Android malware detection based on control _ow graphs and machine learning algorithms," *IEEE Access*, vol. 7, pp. 21235_21245, 2019.

[34] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee. “Choosing multiple parameters for support vector machines.” *Mach Learning* Vol.46. 2002. Pp. 131–159

[35] J. Li, D. Xue, W. Wu, and J. Wang, “Incremental learning for malware classification in small datasets.” *Security and Communication Networks*, *2020.*

[36] J. Sahs and L. Khan, ``A machine learning approach to Android malware detection," in *Proc. Eur. Intell. Secur. Informat. Conf.*, Aug. 2012, pp. 141_147.

[37] A. Mohaisen, O. Alrawi, M. Mohaisen, “AMAL: high-fidelity, behavior-based automated malware analysis and classification” *Comput. Secur.* 52 2015 251–266, http://dx.doi.org/10.1016/j.cose.2015.04.001.

[38] L. Breiman. “Random Forests.” *Machine Learning, Vol.45, No.*1. 2001. Pp. 5-32. doi:10.1023/a:1010933404324

[39] S. Yoo, S. Kim, S. Kim, and B. B. Kang. “AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification.” *Information Sciences*, Vol. *546.* 2021. pp.420-435.

[40] A. Damodaran, F.D. Troia, C.A. Visaggio, T.H. Austin, M. Stamp, “A comparison of static, dynamic, and hybrid analysis for malware detection” *J. Comput. Virol. Hacking Tech.* Vol.13 no.1. 2017 pp. 1–24, http://dx.doi.org/10.1007/s11416- 015-0261-z.

[41] Z. Markel, M. Bilzor, “Building a machine learning classifier for malware detection”, in*: 2014 Second Workshop on Anti-malware Testing Research (WATeR),* 2014, pp. 1–4, http://dx.doi.org/10.1109/WATeR.2014.7015757, http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7015757.

[42] B. Kang, S. Y. Yerima, K. Mclaughlin, and S. Sezer, ``N-opcode analysis for Android malware classi_cation and categorization," in *Proc. Int. Conf. Cyber Secur. Protection Digit. Services (Cyber Secur.)*, Jun. 2016, pp. 1_7.

[43] A. Kapratwar, F. Di Troia, and M. Stamp, ``Static and dynamic analysis of Android malware," in *Proc. 3rd Int. Conf. Inf. Syst. Secur. Privacy*, 2017, pp. 653_662.

[44] F. Martinelli, F. Mercaldo, and A. Saracino, ``BRIDEMAID: An hybrid tool for accurate detection of Android malware," in *Proc. ACMAsia Conf. Comput. Commun. Secur. (CCS)*, Abu Dhabi, UAE, 2017, pp. 899_901.

[45] B. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, ``Android permissions: A perspective combining risks and bene_ts," in *Proc. 17th ACM Symp. Access Control Models Technol.*, New York, NJ, USA, 2012, pp. 13_22.

[46] H. S. Das, and P. Roy. “A deep dive into deep learning techniques for solving spoken language identification problems.” In *Intelligent Speech Signal Processing*. 2019. pp. 81-100. Academic Press.

[47] Y. Lecun, Y. Bengio, and G. Hinton. Deep Learning. *Nature*, 2015, 521, 436.

[48] H. F. Nweke, Y. W. Teh, M. A. Al-Garadi, and U.R. Alo. “Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges.” *Expert Syst. Appl.* 2018, Vol. 105, pp. 233–261.

[49] A. Mahindru, and A. Sangal. “Deepdroid: Feature selection approach to detect android malware using deep learning”. 2019 IEEE

[50] A. Mahindru, and A. Sangal. Perbdroid: “Effective malware detection model developed using machine learning classification techniques.” *In: A journey towards bio-inspired techniques in software engineering, Springer*, 2020 pp 103–139

[51] J. Yan, Y. Qi, and Q. Rao, ``Detecting malware with an ensemble method based on deep neural network," *Secur. Commun. Netw.*, vol. 2018, Mar. 2018, Art. no. 7247095.

[52] J. Singh, and J. Singh, “A survey on machine learning-based malware detection in executable files.” *Journal of Systems Architecture*, 2020, p.101861.

[53] T. H. Hai, and E. Huh, “Network Anomaly Detection Based on Late Fusion of Several Machine Learning Algorithms.” *International Journal of Computer Networks and Communications*, V0l. *12 no*6, 2020, pp.117-131.

[54] Y. Sun, Y. Xie, Z. Qiu, Y. Pan, J. Weng, and S. Guo, ``Detecting Android malware based on extreme learning machine,'' in *Proc. IEEE 15th Int. Conf. Dependable, Autonomic Secure Comput., 15th Int. Conf. Pervas. Intell. Comput., 3rd Int. Conf. Big Data Intell. Comput. Cyber Sci. Technol. Congress(DASC/PiCom/DataCom/CyberSciTech)*, Nov. 2017, pp. 47_53.

[55] C. C. U. López, J. S. D. Villarreal, A. F. P. Belalcazar, A. N. Cadavid and J. G. D. Cely, "Features to Detect Android Malware," *2018 IEEE Colombian Conference on Communications and Computing (COLCOM)*, 2018, pp. 1-6, doi: 10.1109/ColComCon.2018.8466715

[56] C. Seiffert, T. M. Khoshgoftaar, Van J. Hulse, and A. Napolitano. “RUSBoost: Improving classification performance when training data is skewed.” In *2008 19th international conference on pattern recognition* December 2008, pp. 1-4). IEEE.

[57] U. R. Alo, S. I. Ele, and H. F. Nweke A conceptual framework for network traffic control and monitoring using artificial neural networks. *Journal of Theoretical and Applied Information Technology*, Vol.*97 no.*22, 2019 pp.3396-3412.

[58] D. H. Wolpert and W.G. Macready. “No free lunch theorems for optimization.” *IEEE Transaction on Evolution Computing* 1(1):67–82. 1997. https://doi.org/10.1109/4235.585893

[59] M. Leeds, and T. Atkison “Preliminary results of applying machine learning algorithms to android malware detection.” In: *International Conference on Computational Intelligence (ICCI) IEEE*, Dec 2016.