IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 21, Issue 3, Ser. I (May - June 2019), PP 18-27
www.iosrjournals.org

Performance Evaluation of Machine Learning Algorithms for Detection and Prevention of Malware Attacks

Emmanuel Gbenga Dada[1], Joseph Stephen Bassi[1], Yakubu Joseph Hurcha[2] and Abdulkadir Hamidu Alkali[1]

[1]*Department of Computer Engineering, Faculty of Engineering,University of Maiduguri, Maiduguri - Borno State Nigeria.*
[2]*Department of Mathematical Sciences,Faculty of Science,University of Maiduguri, Borno state, Nigeria.*
Corresponding Author:Emmanuel Gbenga Dada

Abstract:*Malware is any type of program that is intended to wreak havoc to the computer system and network. Examples of malware are bot, ransomware, adware, keyloggers, viruses, trojan horses, worms and others. The exponential growth of malware is posing a great danger to the security of confidential information. The problem with many of the existing classification algorithms is their low performance in term of their ability to detect and prevent malware from infecting the computer system. There is an urgent need to evaluate the performance of the existing Machine Learning classification algorithms used for malware detection. This will help in creating more robust and efficient algorithms that have the capacity to overcome the weaknesses of the existing algorithms. This study did the performance evaluation of some classification algorithms such as J45, LMT, Naïve Bayes, Random Forest, MLP Classifier, Random Tree, REP Tree, Bagging, AdaBoost, KStar, SimpleLogistic, IBK, LWL, SVM, and RBF Network. The performance of the algorithms was evaluated in terms of Accuracy, Precision, Recall, Kappa Statistics, F-Measure, Matthew Correlation Coefficient, Receiver Operator Characteristics Area and Root Mean Squared Error using WEKA machine learning and data mining simulation tool. Our experimental results showed that Random Forest algorithm produced the best accuracy of 99.2%. This positively indicates that the Random Forest algorithm achieves good accuracy rates in detecting malware.*
Keywords:*Malware, classification algorithms, Random Forest, AdaBoost, Bagging, Naïve Bayes*

Date of Submission: 26-04-2019Date of acceptance: 11-05-2019

I. Introduction

The breakthrough in internet technology and computer networking have made high speed shared internet possible. The effect of this development is the daily increase in the number of computer systems that have become susceptible to malware attacks [1, 2]. The innovation has made the internet a huge storehouse where resources are virtualized and utilised to the need of users. Despite the immense benefits that the internet revolution has brought, there are numerous challenges that it also poses to the security of computer systems. The conventional computer system is entirely centered on a single host machine running operating system, while several machines connected to the host are running on the guest operating system [1]. The prevalent security threat confronting the users is the attack on a computer system by malicious programs which spread to other computers that have not been infected [3]. The threat posed by malware infections has become a major challenge in the field of computer security over the years.

The number of new malware on the internet keep on increasing at an alarming rate even as anti-virus companies are making effort to curtail the trend so as to make the vast number of computer user safe. Malware has evolved over time and is becoming more sophisticated than before. It is now more difficult to detect them. There is therefore the need to invent more efficient techniques that can detect and prevent these attacks. Malware is a malicious program which infringes on the security of a computer system in terms of privacy, reliability, and accessibility of data [3]. This trend has made academicians and industry practitioners to move from the conventional static detection techniques [4, 5] to more dynamic, sophisticated and spontaneous methods that applies accumulated malware behaviour to detect malware attacks [6, 7, 8].

A malware can simply be defined as a malicious program which the user unsuspectingly install on their machine and later these programs can begin to disrupt the proper operation of the machine or might continue unnoticed and carry out malicious actions without been noticed [9]. When the attacker gains control of the machine, he can then have access to any information stored on the machine. Some of the deceptive approaches used to install malware on the computer system through the internet include repackaging the software, update attack [10] or desire for download [11]. The attacker employs any of the methods mentioned before to create

malicious software by inserting a certain type of malware into it before uploading it to the internet. Malware can be described as various types of software which have the capacity to wreak havoc on a computer system or illegally make use of this information without the consent of the users [12]. Malware can be categorized in various types, for instance, Botnet, Backdoor, Ransomware, Rootkits, Virus, Worms, and Trojan Horse, Spyware, Adware, Scareware and Trapdoor. They are used to attack computer systems and for performing criminal activities such as scam, phishing, service misuse and root access [13].

II. Literature Review

This section discusses the different categories of malware and relevant works that have been done in detecting malware attacks in the literature.

A. Types of Malware

For some time now, different types of malware have been performing various malicious activities on computer systems. These activities range from merely displaying undesirable subject to absolutely hijacking the computer system from the user and denying them access to it. The most popular and frequently noticed malware include:

Trojan Horse: Is a program that looks harmless and helpful to users like any other authentic software. However, after opening the application, this malware distributes some other malicious codes that corrupt the files and applications installed on the computer, and also steal sensitive information such as password. Unlike computer viruses and worms, Trojans require interaction with users to reproduce themselves. This makes Trojans one of the most destructive and hazardous types of malware because it is mostly discovered after it has affected the computer system [14]. According to [15], Trojan horse can be categorised into two main groups: General Trojan and Remote-Access Trojan. General Trojans: this type of Trojans has a wide range of malicious activities. They can threaten data integrity of victim machines. They can redirect victim machines to a particular web site by replacing system files that contain URLs. They can install several malicious software on victim computers. They can even track user activities, save that information and then send it to the attacker. Remote-Access Trojans: we can claim that they are the most dangerous type of Trojan. They have the special capability which enables the attacker to remotely control the victim machine via a LAN or Internet. This type of Trojan can be instructed by the attacker for malicious activities such as harvesting confidential information from the victim machine. Examples of Trojan Horses are Remote access Trojans (RATs), Backdoor Trojans (backdoors), IRC Trojans (IRC bots), Keylogging Trojans [15].

Virus: Virus as a malware that has a self-replicating nature. It is constructed to modify or stop the functioning of a computer[16]. It multiplies by first infecting one program. It is a kind of malware that can cause serious damage varying from the computer system merely displaying arbitrary errors in making the system experience a Denial of Service (DoS) attack. What distinguishes a virus from a Trojan is the ability of a virus to duplicate itself by attaching itself to other valid software and become a part of them. Viruses are usually propagated through copying of files from one computer system to another, through websites, or e-mails that contain files that have already been contaminated with virus [14]. Also, software installed on the computer are corrupted by the viruses as a result of injecting the genuine software with malicious code and as it is executed, the virus is transmitted to other programs on the computer [17]. There are many different ways for transmitting a virus to other computers such as by sending an infected file as an email attachment or by embedding copies of infected files into a removable medium such as a CD, DVD or USB drive [17, 18]. Viruses can increase their chances of spreading to other computers by infecting files on a network file system or a file system that is accessed by another computer. One of the crucial differences between virus and worm is the capability of worm to automatically spread itself to other computers in the network by exploiting computer's security vulnerabilities. There are various classifications of a virus, they include an encrypted, polymorphic and metamorphic virus.

Adware: Is a malware whose only purpose is to show advertisements to the user. They are regarded as one of the least threatening categories of malware. Their intention is to display on the affected computer commercials which the user is likely to be attracted to, it records data from the computer such as browser and search engines histories [19]. Adware is sometimes classified as spyware subject to the seriousness of the recording. Adware, or advertising-supported software, is any software package which automatically plays, displays, or downloads advertisements to a computer. These advertisements can be in the form of a pop-up. The object of the Adware is to generate revenue for its writer. Adware, by itself, is harmless; however, some adware may come with integrated spyware such as keyloggers and other privacy-invasive software. Adware is usually seen by the developer as a way to recover development costs, and in some cases, it may allow the software to be offered to the user free of charge or at a reduced price. Conversely, the advertisements may be seen by the user as interruptions or annoyances, or as distractions from the task at hand [19].

Spyware: Is a kind of self-installing malware that execute without the user's approval. It is used to gather and track information about the person and the browsing history of a computer system. It is generally packaged together with software that is made available to users at no cost [20]. Spyware is also called rootkit because of the packaging with freeware. Spyware is a code that enables a third party to spy on a host. Spyware has been used for a variety of purposes including identity theft and theft of personal data, spying on online activities of individuals (e.g. spouses) and watching users' online activities. It is a type of malware installed on computers that collects information about users without their knowledge. The presence of spyware is typically hidden from the user and can be difficult to detect [21]. Spyware usually modifies the computer settings, leading in very sluggish connection speeds and/or loss of Internet connection. Moreover, some of the system functionality start malfunctioning thus making the computer to be very slow and several strange software are automatically installed.

Worm: Is a malware that does not attach itself to other software as it does not need a host software to fasten itself to. This is what differentiates worm from the virus. A worm normally affects its victim through the area of exposures that it can exploit. It employs various means to propagate, and corrupt other computer systems [14]. Worms have the capacity to wreck the same extent of havoc a virus will cause to an infected computer system. Worms are not parasitic in behaviour like the viruses. They are independent programs that can cause harm on their own. These worms may or may not have a payload but both types can be pretty harmful. Worms without payloads do not affect the system that it infects [16]. Whereas the worms with payload will do harm to the infected system as well. In some cases, the payload acts as a backdoor instead of making changes to the system [16]. A worm could have a very harmful effect on systems in the network, such as could consume too much system memory or system processor (CPU) and cause many applications to stop responding [22]. Some of the most famous worms include the ILOVEYOU worm which has made businesses to lose upwards of 5.5 billion dollars in damage[23].

Bot: Also known as a web robot or botnet are application software that runs automated tasks over the internet. They belong to a category of malware that allows its principal to gain access to the infected computer system. Bots can propagate through backdoors made available by a virus or worm on the victim computer. Bots are known for employing an application layer protocol that enables communication in the form of text with its principal. Distributed Denial of Service (DDoS) attacks that have the capacity to obstruct the services of the target computer by over-flooding its bandwidth or resources with requests can be launched using several bots [14].

Ransomware: Is a subcategory of malware which encrypts the files on the victim's computer or totally locked you out. It turns your files to unintelligible information and makes them useless and payment is necessitated prior to the decryption and returning of the ransomed files to the owner. They usually infect their victims through Trojan [24].

Rootkits: Are a set of software tools used by hackers to get and sustain continuous administrator-level access to a computer system so as to camouflage the changing of files, or activities of the hacker to keep the user in the dark. Rootkits are commonly linked with Trojans, worms, and viruses that obscure their presence and actions from users and other system processes [24].

Backdoor: Is a class of malware that offers a supplementary stealthy "entrance" to the system for attackers. The backdoor itself does not directly harm the system but it opens the door for attackers to wreak havoc. Due to this characteristic, backdoors are in no way used individually. Ordinarily, a backdoor is antecedent malware attack or other forms of attacks [24].

Keylogger: Also known as keystroke logging is a type of surveillance malware that once the computer is infested with it has the ability to record every keystroke make on that system. The recording is saved in a log file which is normally encrypted and sent to a specific receiver. Such information can include passwords, Band Verification Number, ATM card numbers and other confidential information [25].

B. Related Works

With the unprecedented increase in the number of malware been released on the internet, many researchers have taken it upon themselves to evaluate the performance of classification algorithms that have been used for detecting and classifying malware by using a combination of performance metrics. We, therefore, find it necessary to determine which algorithm performs best for any chosen metric to assist in the proper classification of malware. Several studies have been carried out to compare the performances of some

classification algorithms for malware detection. Classification algorithms whose performances have been so far compared include Naïve Bayes [25]. Other algorithms compared include Decision Trees [26, 27], Support Vector Machine [28, 33], Random Forests [29, 30], J48 [31], C4.5 [32], kNN [25], Multilayer perceptron [29], CART [26], Neural Network [28], IBK [33], Bayesian Network [30]. Table 1 depicts the summary of the algorithms used in previous studies.

Table 1: Summary of Related Algorithms Compared in Literature

	Algorithms															
Reference	**Naïve Bayes**	**Decision Tree**	**SVM**	**Random Forests**	**J48**	**C4.5**	**kNN**	**CART**	**Neural Network**	**IBK**	**Multilayer Perceptron**	**PART**	**Recursive ...**	**Bagging**	**Iterative Trees**	**Bayesian Network**
[25]	√		√	√	√											
[26]	√		√	√	√		√									
[27]	√	√	√					√								
[28]	√		√		√				√							
[29]	√		√		√					√						
[30]	√		√	√	√						√	√				
[31]				√			√				√					√
[32]	√			√	√						√					
[33]	√		√	√									√	√	√	
[34]	√		√	√	√	√										

An automatized system for malware behaviour analysis based on emulation and simulation techniques was proposed by [37]. The suspicious code was tested on a sandbox environment that enhances the security and reliability of the system. The performance of the system was evaluated using balanced malware datasets on Bayes, Decision Trees and Support Vector Machine classifiers. The performance metrics used include the True Positive Ratio (TPR), the False Positive Ratio (FPR), and Total Accuracy. None of the algorithms achieves 100% accuracy in classifying the dataset as either malware or not. Cuckoo Sandbox was used by [25] to determine the best feature extraction, feature representation, and classification methods that result in the best accuracy. Specifically, k-Nearest-Neighbors, Decision Trees, Support Vector Machines, Naive Bayes and Random Forest classifiers were evaluated. They used a dataset that consists of 1156 malware files of 9 families of various kinds and 984 benign files of different formats. The performance of the system is relatively low.

A static malware detection system was used by [26] to mine 247,348 executables, 236,756 malicious while benign files were 10,592. The dataset is much larger with imbalanced class representation. The performance of the system is poor. A comparative study of many feature selection techniques with four different Machine Learning Classifiers was presented by [28]. The result reveals that the performance of Principal Component Analysis (PCA) feature selection and Support Vector Machines (SVM) is superior to other classifiers. The authors in [33] proposed a classifier selection criterion that considers bounds on the performance estimates using confidence intervals in conjunction with a performance target. Experimental results showed that the performance is good, though there is a need for improvement. An investigation of Machine Learning based malware detection using dynamic analysis on real devices was presented by [29]. Dynamic features were extracted using a tool from Android phones. A comparative analysis of emulator based and device based detection using different Machine Learning algorithms was done. Experimental results demonstrated that the combination of Machine Learning and features extraction produced better performance.

An unconventional solution to evaluating malware detection using the anomaly-based approach with Machine Learning Classifiers was proposed in [30]. Among the many network traffic features, the four types chosen include basic information, content-based, time-based and connection based. Evaluation results showed that the Bayes network and random Forest classifiers produced more superior performances, with a 99.97% true-positive rate (TPR) as against the multi-layer perceptron with only 93.03% on the MalGenome dataset. Moreover, the experiment showed that the kNN classifier performed better than other classifiers compared by detecting the latest Android malware with an 84.57% true positive rate. The authors in [31] proposed the Dimensional Naïve Bayes Classification (MDNBS) and Rete algorithm to efficiently detect the malware in an Application Programmable Interfaces (APIs) and classifying its type as worms, virus, Trojans, or normal. Initially, the input dataset is preprocessed by normalizing the data, then its upper and lower boundaries are estimated during feature extraction. The results of the experiments show that the performance the proposed techniques using measures such as True Positive Rate (TPR), False Positive Rate (FPR), precision, recall, f-measure and, accuracy is good. A string based malware detector to enhance malware detection was presented in [27]. The system uses only binary information and the permissions that are normally used by the anti-virus

engines, in order to provide a scalable solution based on Machine Learning. The performance of the system as evaluated by using 5000 malware and 5000 benign-ware, and compare the results with 56 Anti-Virus Engines from VirusTotal.

The researchers in [32] proposed an analysis system to detect lexical and string obfuscation in Java malware. They recognized a set of features that typify obfuscated code, and use it to train a Machine Learning classifier to differentiate between obfuscated and non-obfuscated malware. Experimental results using a dataset of 375 malware samples containing 182927 strings and 12721 Java classes produced 99% classification accuracy of 99%. The robustness of the system was tested using the chi-squared statistic for each feature. Table 2 summarises the different performance measures used in previous works.

Table 2: Summary of related Performance Metrics used for Comparison in Literature

	Performance Metrics																		
	Reference	**Correctly Classified instances**	**Incorrectly Classified instances**	**Kappa Statistics**	**Random Forests**	**Mean Absolute Error**	**Root Mean Square**	**Relative Absolute Error**	**Root Relative Squared Error**	**TPR**	**FPR**	**Precision**	**Recall**	**F-Measure**	**ROC Area**	**Runtime**	**Classification Accuracy**	**Detection Rate**	
	[25]	√								√	√	√	√				√		
	[26]	√									√						√		
	[27]		√							√	√	√	√				√		
	[28]																		
	[29]									√	√	√	√						
	[30]									√	√	√	√	√					
	[31]										√	√	√					√	
	[32]										√	√	√	√		√	√		
	[33]										√			√	√				
	[34]	√	√									√	√						

III. Materials and Methods

Three stages were involved in the performance evaluation of the various Machine Learning classifiers considered in this study. The phases are Dataset Preparation, Pre-Processing and Application of different Machine Learning algorithms on the ClaMP (Classification of Malware with PE headers) dataset files [34]. The dataset has a total of 5184 instances, which contain 2683 Malware, and 2501 Benign. The dataset has 55 features. The ClaMP dataset [36] is converted into .arff format (a format compatible for the file) supported by the WEKA Machine Learning simulation environment for input data that was used for the analysis. To do a satisfactory classification of the ClaMP dataset [36] , J45, LMT, Naïve Bayes, Random Forest, MLP Classifier, Random Tree, REP Tree, Bagging, AdaBoost, KStar, SimpleLogistic, IBK, LWL, SVM, and RBF Network were utilized and a 10 folds cross-validation was employed in this study. The reason for opting for 10 folds was because of outputs generated from extensive tests on different datasets with erratic learning modus operandi that have proved beyond reasonable doubt that 10 is the most appropriate number of folds needed to obtain the optimal estimate of error [35]. To carry out cross-validation, a particular number of folds is selected, the data is randomly subdivided into 10 segments in which the class is denoted in almost the same size when compared to the complete dataset. Each segment is held out sequentially and the learning method trained on the nine-tenths that remain; afterward, its error rate is processed on the holdout set. Consequently, the learning process is executed 10 times on different training sets. Conclusively, the mean value of the 10 error evaluation are selected as the general error evaluation. To effectively compare the performance of the different classifiers, percentage split was used on the dataset. This permits the extraction of certain percentage of the data for assessment. A percentage split of 66% split was used for this study.

IV. Experimental Results And Discussion

Experiments were conducted using the complete dataset with 10 folds cross-validation and 66% split. The performance of each Machine Learning classifiers was evaluated in terms of Accuracy, Precision, Recall, Kappa Statistics, F-Measure, MCC, Receiver Operator Characteristics Area, and Root Mean Squared Error. The experimental result is depicted in table 3.

Table 3: Results of Accuracy, Precision, Recall, Kappa Statistic, F-Measure, MCC, ROC Area, and RMSE

S/N	Algorithm	Accuracy		Precision		Recall		Kappa Statistics		F-Measure		MCC		ROC Area		RMSE	
		10 folds	66% split	10 folds	66% split	10 folds	66% split	10 folds	66% split	10 folds	66% split	10 folds	66% split	10 folds	66% split	10 folds	66% split
1	J48	0.978	0.976	0.978	0.976	0.978	0.976	0.956	0.952	0.977	0.976	0.956	0.953	0.985	0.984	0.142	0.146
2	LMT	0.983	0.981	0.983	0.982	0.983	0.982	0.966	0.963	0.983	0.982	0.966	0.964	0.991	0.993	0.125	0.125
3	Naïve Bayes	0.652	0.643	0.783	0.777	0.652	0.644	0.323	0.297	0.615	0.599	0.427	0.406	0.961	0.958	0.580	0.586
4	Random Forest	**0.992**	**0.988**	**0.993**	**0.988**	**0.992**	**0.988**	**0.985**	**0.976**	**0.993**	**0.988**	**0.985**	**0.976**	**0.999**	**0.999**	**0.105**	**0.118**
5	MLP Classifier	0.973	0.971	0.973	0.971	0.973	0.971	0.946	0.942	0.973	0.971	0.947	0.942	0.993	0.991	0.149	0.151
6	Random Tree	0.961	0.946	0.962	0.946	0.961	0.946	0.923	0.892	0.962	0.946	0.923	0.893	0.964	0.950	0.194	0.227
7	REP Tree	0.969	0.966	0.970	0.967	0.969	0.966	0.939	0.932	0.970	0.966	0.940	0.933	0.987	0.983	0.163	0.174
8	Bagging	0.978	0.967	0.978	0.968	0.978	0.967	0.956	0.934	0.978	0.967	0.956	0.935	0.996	0.996	0.132	0.150
9	AdaBoost	0.922	0.927	0.924	0.928	0.922	0.928	0.844	0.855	0.922	0.928	0.846	0.855	0.978	0.971	0.241	0.246
10	KStar	0.874	0.867	0.886	0.878	0.874	0.868	0.749	0.736	0.874	0.867	0.760	0.746	0.929	0.915	0.326	0.346
11	SimpleLogistic	0.969	0.961	0.970	0.961	0.969	0.961	0.939	0.922	0.970	0.961	0.939	0.922	0.996	0.994	0.149	0.171
12	IBK	0.979	0.975	0.980	0.976	0.979	0.976	0.959	0.951	0.980	0.976	0.960	0.952	0.980	0.976	0.141	0.155
13	LWL	0.897	0.887	0.909	0.900	0.897	0.888	0.793	0.774	0.897	0.887	0.805	0.787	0.956	0.955	0.282	0.293
14	SVM	0.956	0.953	0.957	0.953	0.956	0.953	0.913	0.906	0.957	0.953	0.914	0.907	0.956	0.953	0.207	0.216
15	RBF Network	0.690	0.662	0.776	0.776	0.690	0.662	0.362	0.315	0.654	0.620	0.458	0.417	0.779	0.869	0.421	0.429

A. Accuracy

The Accuracy is performance metrics that are used to express the percentage of correct predictions. It does not take into consideration the true positives and negatives separately. This is the basic reason why accuracy alone cannot be used to determine the performance of a model. Other performance metrics apart from the accuracy are required to be used. The value of 1 indicates the best accuracy. From the experimental results of various classifiers in this study, the best Accuracy is 0.992 generated when the 10-fold cross-validation was used on Random Forest classifier while the worst was 0.652 produced when 66% split was used on the Naïve Bayes classifier. Figure 1 and Table 3 shows the Accuracy of each classifier.

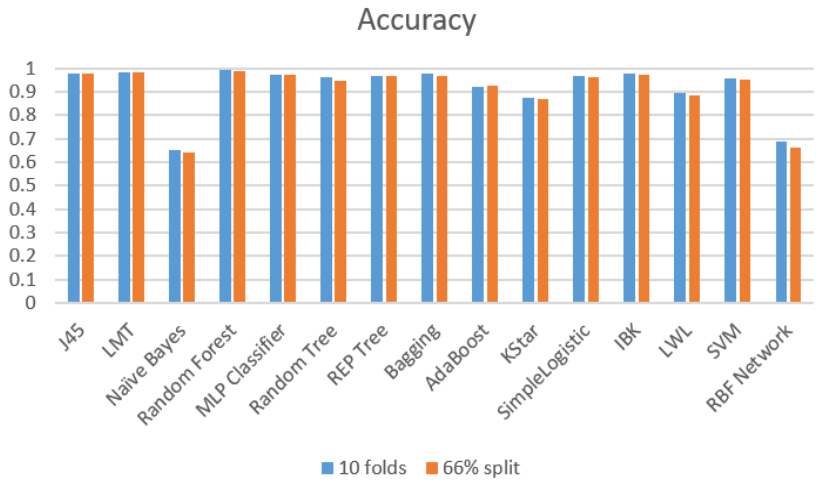

Figure 1: Comparison of Accuracy

B. Precision and Recall

Precision, which is also known as positive predictive value, returns the rate of relevant results rather than irrelevant results. It is a small percentage of important recollected instances, while recall is the fraction of relevant instances that are recollected. The recall is the sensitivity for the most relevant result.

Precision and recall depend on an understanding and measure of relevance.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = TPR = \frac{TP}{TP+FN} \quad (2)$$

The precision and result of the different classifiers are depicted in Table 3 and figures 2 and 3. The highest precision and recall values of 0.993 and 0.992 respectively were produced when 10 fold cross-validation was done on Random Forest. Figures 2 and 3 show the precision and recall of various classifiers used in this study.

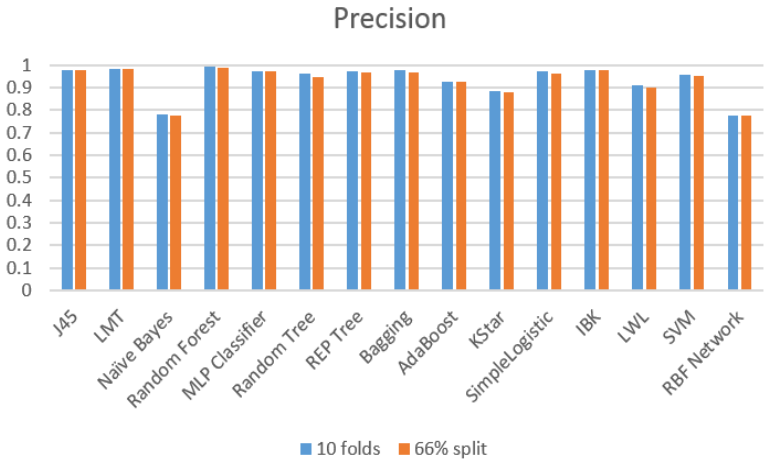

Figure 2: Comparison of Precision

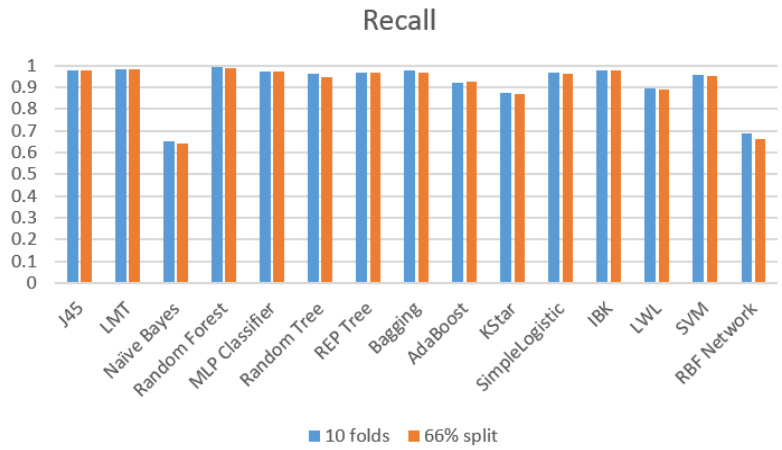

Figure 3: Comparison of Recall

C. Kappa Statistics

Kappa statistic is a performance metric that compares an observed accuracy with an expected accuracy (random chance). It reflects the degree of agreement between the true classes and the classifications. The kappa statistics value of 1 is the highest indicating complete agreement. In this study, the highest kappa characteristics is 0.985 which was produced when the test was conducted on Random Forest with 10 folds cross-validation. Table 3 and figure 4 shows the respective kappa statistics for each classifier.

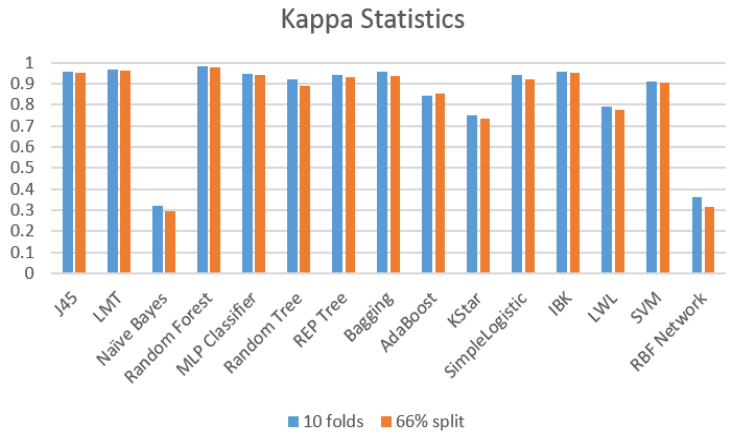

Figure 4: Comparison of Kappa Statistics

D. F-Measure

F-Measure is the value that estimates the complete performance of the system by uniting precision and recall into a single number. The highest value of 1 specifies the best result.

$$F - measure = \frac{2\ x\ Recall\ \ x\ Precision}{Recall\ + Precision} \quad (3)$$

Our experimental result in table 3 and figure 5 shows that Random Forest has the highest F-measure of 0.993.

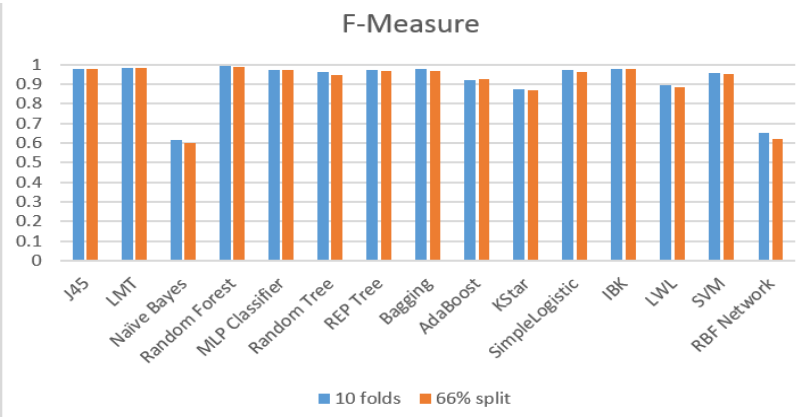

Figure 5: Comparison of F-Measure

D. ROC Area

The ROC (AUC) Area of a classifier is the probability of the classifier ranking a randomly chosen positive instance higher than a randomly chosen negative instance. A ROC value of 1.0 represents a perfect prediction, ROC value of 0.9 indicates excellent prediction, ROC of 0.8 depicts good prediction, ROC of 0.7 is a mediocre prediction, while ROC of 0.6 symbolises a poor prediction. Figure 6 depicts the areas under ROC curves of classifiers used in this study with Random Forest achieving the best performance with 0.999 while RBF Network has the poorest performance with 0.779.

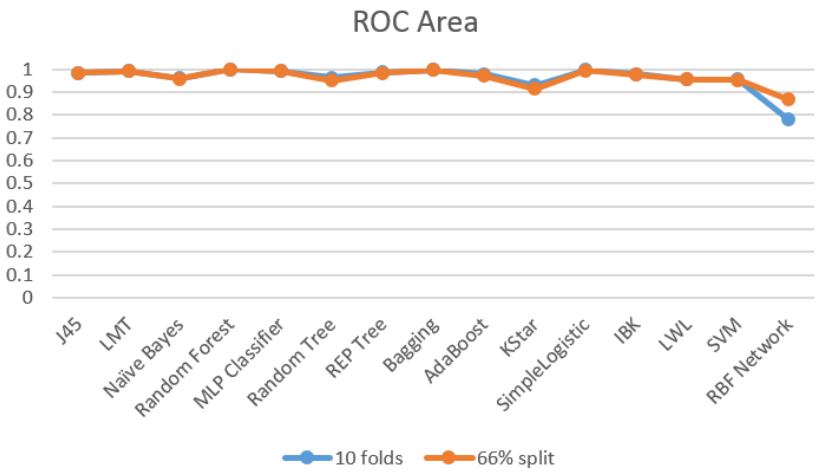

Figure 6: Comparison of ROC Area

E. Matthew Correlation Coefficient (MCC)

The true and false positives cannot be adequately described using one indicator, the Matthews Correlation Coefficient (MCC) have proved to be the best general measure [34]. MCC is a performance metric that measures the properties of the two-class problem. It takes into consideration the true and false positives and negatives. It is a balanced metric, even when the classes are from dissimilar sizes. The formula below can be used to compute the value for MCC:

$$MCC = \frac{(TPxTN)-(FNxFN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (4)$$

when the output is +1 it represents the best prediction, while −1 signifies a complete disagreement. Table 3 and figure 7 shows the MCC for each classifier underconsideration. Random Forest classifier produced the best MCC value of 0.985 while Naïve Bayes generated the worst result of 0.427.

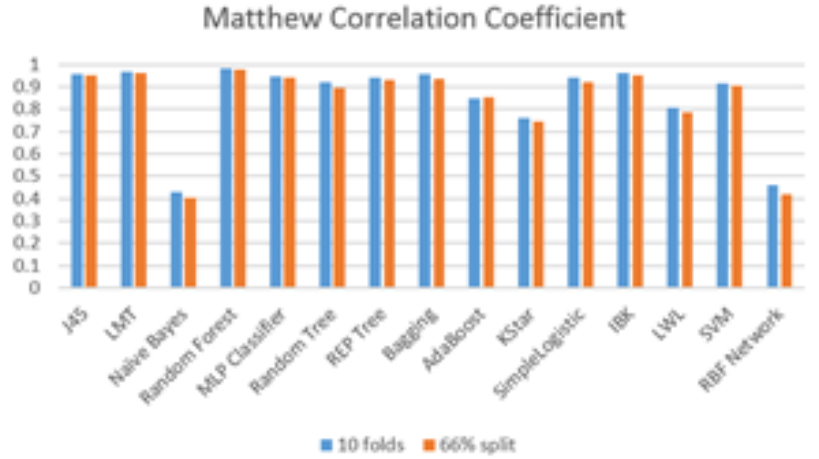

Figure 7: Comparison of MCC

V. Conclusionand Recommendations

This paper presents a comparative study of malware detection using fifteen different Machine Learning algorithms. Some of the state-of-the-art models such as J45, LMT, Naïve Bayes, Random Forest, MLP Classifier, Random Tree, REP Tree, Bagging, AdaBoost, KStar, SimpleLogistic, IBK, LWL, SVM, and RBF Network were used in the study and their statistical results presented. From the experimental results obtained from running the various classification using 10-fold cross-validation and 66% split test, it has been demonstrated that some unpopular algorithms perform relatively well on the ClaMP dataset [36] on WEKA. It becomes apparent from our study that Random Forest is the best classifier among the fifteen (15) classifiers considered. Experimental results indicated that even with less feature selection used, the Random Forest classifier with 0.992 performs comparatively better in malware classification, much better than the popular classification algorithms such as SVM with 0.956 accuracy, AdaBoost with accuracy of 0.922, Bagging with 0.978, J48 with 0.978, Naïve Bayes with 0.652, and Multilayer Perceptron classifier with 0.973. We recommend

that more publicly available malware datasets be used to evaluate the performance of other Machine Learning algorithms using different Data Mining and Machine Learning tools such as RapidMiner.

References

[1]. Sanjay Chakrabortya and LopamudraDey. A rule-based probabilistic technique for malware code detection. Multiagent and Grid Systems – An International Journal, IOS Press, 12, 2016, pp. 271–286 271. DOI 10.3233/MGS-160254

[2]. Y. Zhou, Z. Wang, W. Zhou, and X. Jiang. Hey, you, get off of my market: Detecting malicious apps in official and alternative android markets. in NDSS, vol. 25, no. 4, 2012, pp. 50–52.

[3]. D. Keragala. Detecting malware and sandbox evasion techniques, SANS Institute InfoSec Reading Room, 2016. URL: https://www.sans.org/reading-room/whitepapers/ forensics/detecting-malware-sandbox-evasion-techniques-36667.

[4]. Sharif, M., Yegneswaran, V., Saidi, H., Porras, P., and Lee, W. Eureka: A framework for enabling static malware analysis. In Computer security-ESORICS 2008, pages 481- 500. Springer.

[5]. Moser, A., Kruegel, C., and Kirda, E. Limits of static analysis for malware detection. In Computer security applications conference, ACSAC 2007. Twenty-third annual, 2007, pages 421-430.

[6]. Egele, M., Scholte, T., Kirda, E., and Kruegel, C. A survey on automated dynamic malware-analysis techniques and tools. ACM Computing Surveys (CSUR), 2012, 44(2):6.

[7]. Ahmad, S., Ahmad, S., Xu, S., and Li, B. Next generation malware analysis techniques and tools. In Electronics, Information Technology and Intellectualization: Proceedings of the International Conference EITI 2014, Shenzhen, 16-17 August 2015, page 17. CRC Press.

[8]. Gorecki, C., Freiling, F. C., Kuhrer, M., and Holz, T. Trumanbox: Improving dynamic malware analysis by emulating the internet. In Stabilization, Safety, and Security of Distributed Systems, Springer, 2011, pages 208-222.

[9]. Jyoti Malik and RishabhKaushal. Credroid: Android Malware Detection By Network Traffic Analysis, ACM PAMCO'16, July 05 2016, Paderborn, Germany, DOI: http://dx.doi.org/10.1145/2940343.2940348

[10]. L. Tenenboim-Chekina, O. Barad, A. Shabtai, D. Mimran, L.Rokach, B. Shapira and Y. Elovici. Detecting Application Update Attack on Mobile Devices through Network Features. In INFOCOM 2013.

[11]. MahinthanChandramohan and HeeBengKuan Tan. Detection of Mobile Malware in the Wild. In Computer, ieeexplore.ieee.org, (Sept. 2012) vol. 45, pp. 65-71.

[12]. MohamadBaset. Machine Learning for Malware Detection. MSc. Dissertation, School of Mathematical and Computer Sciences, Heriot-Watt University, 62 pages, 2016.

[13]. T. G. M. Van Erp, T. D. Cannon, H. L. Tran, A. D. Wobbekind, M. Huttunen, J. Lonnqvist, J. Kaprio, O. Salonen, L. Valanne, V. P. Poutanen, C. G. Standertskjold-Nordenstam, A. W. Toga, and P. M. Thompson. Genetic influences on human brain morphology, in IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2004., April 2004, Vol. 1, pp. 583–586

[14]. Cisco (2015). What is the difference: Viruses, worms, trojans, and bots? Online. http://www.cisco.com/web/about/security/intelligence/virus-worm-diffs.html.

[15]. Zeidanloo H. R., Tabatabaei S. F., Amoli P. V. and Tajpour A. All About Malwares (Malicious Codes), 2015.

[16]. Vemparala S. Malware Detection using Dynamic Analysis. Master's Theses. San Jose State University, 2015. http://scholarworks.sjsu.edu/etd_projects/403

[17]. Q. Jamil and M. A. Shah. Analysis of Machine Learning Solutions to Detect Malware in Android. The sixth international conference on innovative computing technology (INTECH 2016), pp. 226 – 232, 2016.

[18]. Saradha R. Malware Analysis using Profile Hidden Markov Models and Intrusion Detection in a Stream Learning Setting. Master's Thesis. Fac. of Engineering. Indian Institute of Science, 2014.

[19]. Radu-Stefan Pirscoveanu. Clustering Analysis of Malware Behaviour. Master Thesis Department of Electronic Systems at Aalborg University, 139 pages, 2015.

[20]. Symantec. What are malware, viruses, spyware, and cookies, and what differentiates them? 2009. Online. http://www.symantec.com/connect/articles/what-are-malware-viruses-spyware-and-cookies-and-what-differentiates-them.

[21]. Brunton F. and Nissenbaum H. Political and Ethical Perspectives on Data Obfuscation, 2012, pp. 164-188.

[22]. Damodaran A., Troia F. D., Corrado V. A., Austin T. H. and Stamp M. A Comparison of Static, Dynamic, and Hybrid Analysis for Malware Detection, 2015.

[23]. Gadhiya S. and Bhavsar K. Techniques for Malware Analysis. International Journal of Advanced Research in Computer Science and Software Engineering. Vol. 3, Issue 4, 2013, pp. 972-975.

[24]. Zimba, A., Wang, Z., & Chen, H. Multi-stage crypto ransomware attacks: A new emerging cyber threat to critical infrastructure and industrial control systems. ICT Express, 2018.

[25]. KaterynaChumachenko. Machine Learning Methods for Malware Detection and Classification. Bachelor's Thesis in Information Technology. University of Applied Sciences, 93 pages, 2017.

[26]. Baldangombo, U., Jambaljav, N. &Horng, S.J. A Static Malware Detection System Using Data Mining Methods. International Journal of Artificial Intelligence & Applications, 4(4), p.113, 2013. Available at:http://arxiv.org/abs/1308.2831.

[27]. Alejandro Martın, Hector D. Menendez, and David Camacho. String-based Malware Detection for Android Environments. Studies in Computational Intelligence 678, pp. 99-108, 2017. DOI 10.1007/978-3-319-48829-5_10.

[28]. Ban Mohammed Khammasa, AlirezaMonemia, Joseph Stephen Bassia, IsmahaniIsmaila, Sulaiman Mohd Nora, Muhammad NadzirMarsono. Feature Selection and Machine Learning Classification for Malware Detection. JurnalTeknologi 77:1, 2015, pp. 243–250.

[29]. Mohammed K. Alzaylaee, Suleiman Y. Yerima and SakirSezer. EMULATOR vs REAL PHONE: Android Malware Detection Using Machine Learning, IWSPA 2017 Proceedings of the 3rd ACM International Workshop on Security and Privacy Analytics, co-located with CODASPY'17, pages 65-72, Scottsdale, Arizona, USA - March 24 - 24, 2017. DOI: 10.1145/3041008.3041010.

[30]. FairuzAmalinaNarudin, Ali Feizollah, Nor BadrulAnuar and Abdullah Gani. Evaluation of Machine Learning classifiers for mobile malware detection. Soft Comput. 2014, DOI 10.1007/s00500-014-1511-6.

[31]. M. Asha Jerlin and K. Marimuthu (2018) A New Malware Detection System Using Machine Learning Techniques for API Call Sequences, Journal of Applied Security Research, 13:1, 2018, pp. 45-62, DOI: 10.1080/19361610.2018.1387734.

[32]. Renuka Kumara, Anand Raj EssarVaishakh . Detection of obfuscation in java malware. International Conference on Information Security and Privacy (ICIS2015), 11-12 December 2015, Nagpur, INDIA. Procedia Computer Science 78, 2016, pp. 521 – 529.

[33]. Anshuman Singh, Sumi Singh, Andrew Walenstein and ArunLakhotia. Deployable Classifiers for Malware Detection Conference Paper in Communications in Computer and Information Science March 2012, DOI: 10.1007/978-3-642-29166-1_34

[34]. M. W. Powers (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness& Correlation" Journal of Machine Learning Technologies, vol. 2, no. 1, pp. 37–63, 2011.
[35]. I. H. Witten and F. Eibe (2005). Data mining : Practical Machine Learning tools and techniques, 2nd ed. Morgan Kaufmann Publishers, 2005.
[36]. ClaMP: Classification of Malware with PE headers (2016). available at https://github.com/urwithajit9/ClaMP/blob/master/README.md
[37]. J. Devesa, I. Santos, X. Cantero, Y. K. Penya, and P. G. Bringas, "Automatic behaviour-based analysis and classification system for malware detection." ICEIS (2), vol. 2, pp. 395–399, 2010.

IOSR Journal of Computer Engineering (IOSR-JCE) is UGC approved Journal with Sl. No. 5019, Journal no. 49102.

Dada*et al.* "Performance Evaluation of Machine Learning Algorithms for Detection and Prevention of Malware Attacks" IOSR Journal of Computer Engineering (IOSR-JCE) 21.3 (2019): 18-27.