Data and Metadata. 2025; 4:1258
doi: 10.56294/dm20251258

ORIGINAL

Android traffic malware analysis and detection using sprint machine learning technique

Análisis y detección de malware en el tráfico de android mediante la técnica de aprendizaje automático sprint

Rajkumar[1] ✉, Gobinath R[2], J Thimmiaraja[3], K.Sathesh Kumar[1], Viji C[1], A Mohanraj[4]

[1]Alliance School of Advanced Computing, Alliance University. Bangalore, India.
[2]Department of Computer Science, Christ University. Bangalore, India.
[3]Department of Information Technology, Dr Mahalingam College of Engineering and Technology. Pollachi, India.
[4]Sri Eshwar College of Engineering. Coimbatore, India.

Cite as: Rajkumar, Gobinath R, Thimmiaraja J, Sathesh Kumar K, Viji C, Mohanraj A. Android traffic malware analysis and detection using sprint machine learning technique. Data and Metadata. 2025; 4:1258. https://doi.org/10.56294/dm20251258

Submitted: 27-06-2025 **Revised:** 08-08-2025 **Accepted:** 12-11-2025 **Published:** 13-11-2025

Editor: Dr. Adrián Alejandro Vitón Castillo

Corresponding Author: Rajkumar ✉

ABSTRACT

The recent history has seen cybercrime seize the notorious position of stealing business and personal information of individuals. This risk is because of vulnerability in security defense and the explosion in the use of interconnected devices. Malware is created in various types; they include viruses, worms, trojans, ransomware, spyware, and adware. Among all the malware, the Android malware is the worst because it can destroy the device by invading the privacy of a person and stealing his/her information. The already adopted anti-virus and anti-malware software require periodic updates and have no ability to detect the new and advanced malware. Our paper, therefore, proposes the malware analysis and systematic classification using machine learning algorithms. In our study, we associate the effectiveness of Deep Neural Network, Recurrent Neural Network, and Secure Program Identification classifiers with the features of Ack-post, Finance, Moghave, Send pay and Chess to pre-train and test the models. We have obtained a classification accuracy of 99,5 % with the proposed SPRINT classifier, 98 percent with DNN, and 96 percent with RNN, through our experiment. Such an outcome highlights the superiority of the proposed SPRINT classifier to the rest in its ability to classify the malware with machine learning algorithms. With the help of the proposed approach, new and advanced malware instances can be timely and effectively identified and classified, which would contribute to the improved security of Android devices and eliminate possible cyber threats.

Keywords: Malware Detection; Machine Learning; Malware Variants; Malware Classifications.

RESUMEN

En la historia reciente, el cibercrimen ha adquirido una notoriedad alarmante al robar información empresarial y personal. Este riesgo se debe a la vulnerabilidad de las defensas de seguridad y al auge en el uso de dispositivos interconectados. El malware se presenta en diversas formas, incluyendo virus, gusanos, troyanos, ransomware, spyware y adware. Entre todos los tipos de malware, el de Android es el más peligroso, ya que puede dañar el dispositivo al invadir la privacidad del usuario y robar su información. El software antivirus y antimalware ya existente requiere actualizaciones periódicas y no tiene la capacidad de detectar el malware nuevo y avanzado. Por lo tanto, este trabajo propone el análisis y la clasificación sistemática del malware mediante algoritmos de aprendizaje automático. En nuestro estudio, asociamos la efectividad de los clasificadores de Redes Neuronales Profundas (DNN), Redes Neuronales Recurrentes (RNN) e Identificación Segura de Programas (SPRINT) con las características de Ack-post, Finance, Moghave, Sendpay y Chess para preentrenar y probar los modelos. En nuestro experimento, obtuvimos una precisión

de clasificación del 99,5 % con el clasificador SPRINT propuesto, del 98 % con DNN y del 96 % con RNN. Este resultado demuestra la superioridad del clasificador SPRINT sobre los demás en su capacidad para clasificar malware mediante algoritmos de aprendizaje automático. Gracias a este enfoque, se pueden identificar y clasificar de forma oportuna y eficaz las nuevas instancias de malware, lo que contribuirá a mejorar la seguridad de los dispositivos Android y a eliminar posibles ciberamenazas.

Palabras clave: Detección de Malware; Aprendizaje Automático; Variantes de Malware; Clasificaciones de Malware.

INTRODUCTION

The 21 st century businesses are transforming like a whirlwind, most of the brands are pursuing digital advertising rather than conventional marketing. This is attributed to the fact that people buying products online(smartphones) have been on the rise over the years. This enables digital marketing to be easier to use in introducing a product to a large population. Online advertisements are available on the web pages or mobile applications.[1,2] At the same time internet advertisement is prone to considerable amount of fraud. Adware is a computer program that transcends what is acceptable during the freeware and shareware software. Whereas freeware and shareware programs could potentially contain advertisements as one of the methods of generating revenue, adware goes a step further and shows the user too many or annoying advertisements.[2]

Adware operates automatically without the awareness of the user in order to enhance its advertising that is called malware by certain programs.[3] Android is one of the dominant mobile operating systems with a huge market share. It is estimated that there are more than 1 billion smartphones sold with the Android operating system that have been sold to customers and it is believed that the number of applications downloaded by the Google Play Store application alone are 65 billion. Moreover, there is a rise in attack by malware on consumer electronics due to the growing use of smart digital devices and wide-ranging software. There are various mobile applications that have been developed to suit different needs. Mobile phones are no longer merely a means of reaching an individual but it is applied in other areas like economic, commercial, social, professional, and school activities.[4] Smartphones are capable of conducting web services such as the purchase of e-ticket, shopping and e-platforms. A lot of individuals are relying on smart devices in case of bank transfers, and this aspect exposes individuals to leak personal and financial information. To minimize the risk of malware attacks, there is a need to identify and prevent such attack through the development of effective and reliable methods.[5,6] To ensure protection in communication, the key goal is to use methodologies to protect the equipment against harmful programs and data which are three major security properties in securing data ie., confidentiality, integrity, and availability.

In order to determine the distinction between the malware and friendly apps classification strategies were developed so as to monitor the actions of the android user. The classification of apps is done based on the consideration of whether the apps have active or passive attributes. The effectiveness and accuracy of prediction by a classifier is greatly determined by the integrity of the features under analysis. Android based classification technologies are not accurate enough. The state-of-the-art classification techniques based exclusively on the isolated static features have the high rate of false-negative (FN) output. These features are based on data-studies and control flow studies. False-negatives are higher in programs with hybrid characteristics. Adaptive classification strategies include bigger false positive (FP) at the expense of having the right rates of FN and FP. This FP rate[7,8] demonstrates that a certain part of benign applications is labeled as dangerous. This research paper aims at analyzing true keywords, classifying malware and clean adwares as well as finding out algorithms which can reach more accuracy.

The capability of learning the samples and effectively studying the diverse adware problems manually with time which puts in place the extensive application of classification techniques. Alternatively, automated algorithms are attacked to provide an outsider with access to manipulate the inputs of the algorithms to get the desired output. The scholars employed supervised learning to detect android adwares on both static and dynamic information.[1] Manifest file is used to collect the fixed attributes as compared to the network trace that collects dynamic attributes. Addressing these parameters, it is possible to classify benign and adware into some families. The techniques used by researchers in this study are neural methods and random forests, AdaBoost, and SVM machine learning. The binaries detection task was proved to be more difficult than the classification of adware into several types.

The Android malware systems that have machine learning algorithms struggle with the issue of correct malware detection and classification because of the ever-changing nature of malware. A gap in the available studies is that no full datasets are provided that reflect precisely the diversity of Android malware. Often these datasets represent only a small part of the total malware situation and it can be hard to extrapolate

the findings and assess the performance of machine-learning algorithms in practice. It must also have more advanced machine learning models that can be able to identify and detect dynamic and obfuscated malware because it focuses on the use of static analysis, which has a restricted capacity to detect. More sophisticated models could be required with analysis of dynamic analysis and behavioral analysis that could enhance the accuracy.

With the increasing number of mobile devices and the volume of mobile data, there is a need to come up with the algorithm that can easily and fast detect the malware without causing excessive pressure on the system resources. Hence, there is a necessity to enhance machine learning algorithms in terms of scalability and efficiency in large-scale malware detection. Android malware detection requires the emphasis on the interpretability of machine learning models. When no one can interpret the machine learning model, it might be challenging to determine why a machine learning model made a particular prediction, which can complicate the process of locating and correcting the mistakes or biases within the model. It is essential to come up with both accurate and transparent ML models in order to create confidence in machine learning-oriented malware detection systems.

Farther research on malware detection could be biased on certain kinds of malware and not reflect the actual case which would restrict the generalization of the results found in previous studies. They also failed to test the performance of the models on unknown malware which is a very important element in the effectiveness of any malware detection system. The method of the study might not be applicable to bigger datasets or complicated malware. The current issue with the malware detection is acute since some of them cannot distinguish between harmless and harmful contents of an android application, and many consume much time to be detected.

According to past research studies with little adherence to machine learning (ML) processes, there were deep learning (DL) models like convolutional networks (CNN), recurrent neural networks (RNN) and long-short-term memory networks (LSTM),[9,10,11] malware datasets in relation to desktops. However, most of them covered the datasets related to mobile-related malware. The malware attacks were detected through machine learning and deep learning models. Vinayakumar et al.[12] used Ember malware dataset which consisted of 70 140 malicious and 69 869 benign software rewrites. Their studies utilized many machine learning procedures and deep learning models such as K-Nearest Neighbor(KNN), Support Vector Machines,(SVM) Random Forests(RF), AdaBoost, Logistic Regression(LR), Naive Bayes(NB) and Deep Neural Network(DNN) among others. The 200 epochs pre-trained models that make use of Adam optimization algorithm. The long-short term memory model was good in terms of accuracy of 98,9 %.

Recently, in 2020 Jeon[13] proposed a malware detection system, whose methodologies are the deep learning. The convolutional encoder was used in translating the opcode sequence of the executable files of windows. Then malware were detected using RNNs and achieved a detection rate of 96 % and true positive rate of 95 %.

In the study by Yazdinejad et al.[14], they used 200 benign and 500 malware re-codes to be used in the process of both benign and harmful activity. They used the LSTM model to develop a malware detection system that was 10-fold cross-validated on the data that they got. Their research achieved 98 percent precision.

Mahindru et al.[15] suggest a framework to detect malware or benign files in the android application with a small number of criteria collection. This framework was used to test thirty types of android apps. The model can categorize malware files and all the benign apps into non-malicious using nonlinear ensemble decision tree, forest approach, multilayer perceptron, deep neural network, and farthest first clustering with 98,8 percent accuracy. Therefore, this framework can be spread into the collection of multiple android applications that could host soft computing models that may result in the increased chance of detecting malware.

Ma et al.[16] proposed an algorithm that breaks down the android applications and generates a CFG(context-free grammar) separately and identifies the android malware. They produced three dissimilar data on the basis of CFG (Context-Free Grammar) of the disassembled Android applications. These data were concerned with the usage, frequency and order of system APIs (Application Programming Interfaces) usage in the applications. To find the harmful threats in the emerging applications these researchers constructed an API sequences dataset and built two-class classification model to each dataset. They have matched the performance of these three models with accuracy, recall and F-score using standard classification measures and achieved a maximum accuracy of 98,98.

Cai et al.[17] have developed a technique named JOWM to give weight to various features, and they have also developed a system named JOWMDroid to detect malware in Android applications through five stages of the process of creating a malware signature using only the methods of a static analysis of the applications. The former will be achieved by first extracting the original features of APK files in eight categories. The next step is to select the most significant features with the IG technique. This is followed by assigning an initial weight to each of the chosen features by three machine learning models. Five weight mapping functions are set to alter original weights. Lastly, the parameters of the weight mapping function and the classifier are maximized with the DE method. They further matched four weight-conscious classifiers and four state of the art feature weighting methods behaviors through JOWM. The fundamental weights were doing well as represented by

the observational data. They added further that by adjusting the weight mapping and setting the parameters together, the parameter-based weight mapping resulted in a considerable enhancement of malware detection. This strategy is superior to the performance in the comparison of the four cutting edge approaches. In addition, they discovered the optimal settings simultaneously through a combination of the optimization. A proper feature weighting method on weight-aware classifiers is found to be more successful than the weight-unaware classifiers.

Fallaha et al.[18] explored numerous methods of ML techniques such as learning the characteristics counting, type of ML techniques used, the volume of recorded information, the ability to identify the malware family and the ability to understand the new family of malware. They also compared techniques sensitivity using decision tree, random forest, KNN, linear regression, SVM, MLP, and Gaussian naive Bayes, as the number of attributes. In the implementation of the ideal condition in the previous evaluation, this algorithm failed to work well in the identification of the family of malware. This way ML algorithms have been characterized as limited in learning.

Every day, cyber attacks are on the rise due to the rising presence and reliance of digital smart devices. To minimize and monitor the malware activities numerous techniques have been presented with the help of a sufficient amount of big data tools and ML techniques. It is more time-consuming, however, although nascent malware can be detected with the usual ML. The standard engineering methods become unnecessary, and their importance is decreased by the advanced machine learning algorithms such as DL. In the present study, we have examined different methods of detection and classification of malware based on ML and DL to test the samples of harmful intentions.[19]

Since the fast growing rate of web applications is leading to a menace of becoming vulnerable to malware on the web that offers a means through which malware creators can streamline malware tools.[20] In this research we are aiming at comprehending the performance of machine learning in analyzing and estimating the malware classifier. They investigated the six classifiers of machine learning methods and reclaimed PE file and library information features with the latent analysis. The machine learning systems are proposed to make sure that existing files are not infected by harmful files. Random forest method is suggested to use in the classification of the data because its accuracy in the case of the experiment is 99,4. This finding confirms that PE library is appropriate to be used in the process of a static analysis since it focuses on extremely narrow features that may enhance the detection of malware. The general benefit is that users can check before opening a file whether malicious software has been installed on their computer without their awareness.[21]

Research problem: Android devices Malware has become a major issue of concern to consumers and organizations, and this can cause spoiling of the device, theft of information, loss of data, and breach of privacy. The conventional anti-malware software fails to detect new and sophisticated malware. Thus in order to classify and study malware in a systematic way there is a necessity to propose new ways of classifying malware.

Research motivation: The absence of knowledge in security and the proliferation of more network devices result in the proliferation of cyber threat through the formation of different types of malware. The malware attack in Android device brings about serious problems hence the current anti-malware programs are not enough to address the problem. Our study aims at offering an effective machine learning algorithm based approach to detect and categorize malware.

Research gap: although several machine learning techniques have been explored, none of them has all the specifications in the classification and detection of malware. Earlier studies focus on the comparison of behavior of various classifiers thus, there is need to extract the best features in order to train the model. This current study does not provide a full dataset that can designate the various types of malware; this is not managed to generalize the findings and determine the effectiveness of machine learning algorithms in real-life scenarios. To eliminate this gap our research. To seal these loopholes, our study would suggest an innovative approach where a set of features are utilized to train and test the models and to evaluate the performance of a few classifiers, to refine the accuracy of malware detection and classification of the Android devices.

METHOD

The proposed methodology involves a literature review, a thorough examination of current literature, and an interview with three participants of the pandemic to understand how they perceive the incident.<|human|>3. Proposed methodology introduction: The proposed methodology will include literature review, in-depth analysis of existing literature, and an interview with three pandemic participants to learn the way they interpret the incident.

Data set

Android is the most widely used mobile operating system which is open source and it is flexible to install third party applications without centralized control across the world. The effect of this technological impact is malware attacking open-source targets. Despite the security measures that Android uses, the new malicious activities as well as advanced and generated malicious activities alongside the vulnerabilities created means

that new strategies and framework needs to be introduced to improve the security. In order to prevent malware attacks, researchers and developers, therefore, presented a number of security protection solutions based on the use of static analysis, dynamic analysis, and artificial intelligence tools. Data-based analytical models are utilized to identify the insights that could be used to forecast the malicious activities, and the emergence of data science in cybersecurity. The paper recommends the application of network layer properties to develop machine learning models to identify the malware files with the use of datasets that can be found in the research community. Some of the elements included in the dataset are source/ remote application packets, TCP packets, distributed TCP ports, external IPs, volume bytes, UDP packets and TCP URG packets.

The malware is a malicious software that is known as the backdoor and it allows access to a network or computer without the user being aware. To a great extent, the backdoor malware is introduced into the system by phishing email, social engineering, and harmful software vulnerabilities. Once in the device, this malware allows the attackers access to the device and they steal out information and install more malware remotely so that they can utilize it in doing malicious activities. This virus is difficult to locate and fix because it brings about a severe issue to the accessed network and system.

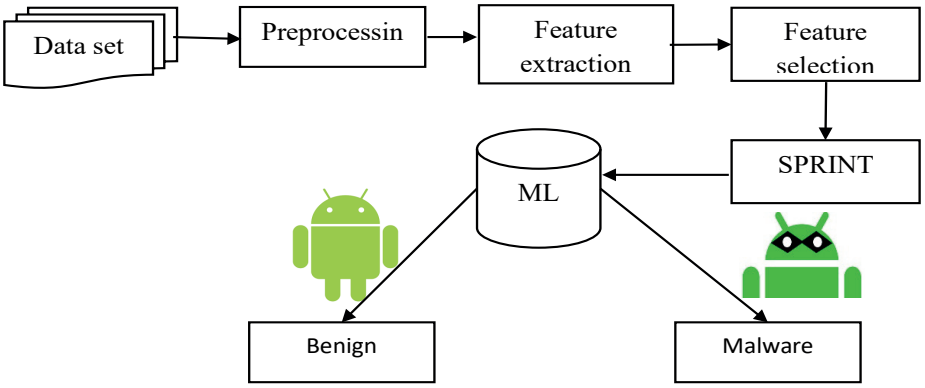

Figure 1. Proposed architecture

Chess Android malware disguises itself as a chess application on Android devices, which carries a malicious code to harm the targeted device. This is a malware program that has been installed in the machine either via third party application stores, malicious web sites or emails. This malware can steal personal information, show unwanted advertisements, automatic malware downloads or even take control of the intended device. It is also capable of doing phishing and DDoS(distributed denial of service)attacks. To avoid this malware attack, one can prevent it by downloading the apps in the credible sources like the Google play store and ensure the software in the devices is regularly updated to maintain the devices up to date.

Finance android malware is aimed at attack on financial transactions in Android OS devices. The malware gets into the devices via phishing mail, third-party applications stores, or fraudulent banking applications. It loots confidential financial information such as credit card details, login information and banking details. Phishing malware or ransomware attacks are also initiated by this malware. As a case in point, malware will show fake pages of login windows to cheat users into providing their login details or it will lock the device and will need a payment to unlock the lock. To avoid this malware attack, one can prevent it by downloading the apps in the credible sources like the Google play store and ensure the software in the devices is regularly updated to maintain the devices up to date.

Trojan:Android/Moghava is also a form of malware that is used in Android OS devices. This viral software also finds its way to the devices through phishing emails, malicious internet sites or deceptive applications. And leads to such malicious activities as stealing personal data, showing undesired advertisements, installing other malware, or even gaining control over an infected computer. It is quite a dangerous malware because it penetrates the security of the Android OS by using overlay attacks method. With these methods such malware presents a fake screen to cheat the users into filling their login information. The only way to avoid this malware attack is by downloading apps on the reliable resources like the Google play store and keeping the device software up to date.

In case you have a feeling that the send pay application is loaded with malware, then act as soon as possible to protect your personal data and gadget. It can steal financial data, and personal information or even steal your device.

The actions to follow in case you question the legitimacy of send pay app includes malware:

- Uninstall the application: delete the application at once off your device.
- Scan the infected device with malware: with appropriate anti-malware scan the whole device with malware that will aid in correcting threats on your device.

- Change your passwords: change the passwords of the accounts that you used to send and receive money using the app.
- Monitor your account: continue watching your financial accounts and credit reports to watch out against possible mischief.
- Report the app: report the app to the corresponding authorities as soon as possible to avoid the further threat.

One should pay a lot of attention to downloading or using financial apps. Get the applications at reputable sources and also read the reviews prior to installation.

Proposed SPRINT classifier

Identification of Android malware is done using the Sparse Representation-based INTent(SPRINT) machine learning algorithm. Such a program examines the characteristic features of Android apps which are not dynamic so as to verify the overall features of malware. These are request app permission such as API calls that the app makes and the presence of certain code patterns. The SPRINT algorithm identifies and categorizes the benign and malware files of the Android applications using a particular combination of both supervised and unsupervised machine learning methods. This was trained on a big data consisting of malware and benign Android applications. This classifier can detect a new type of malware. It is achieved through feature extraction approach that is immune to the modifications of the code structure and the techniques of obfuscation by the attackers. In this way, SPRINT classifier algorithm has passed with high precision in experiment tests of our proposed technique to detect Android malware. The major steps to be taken when using the SPRINT technique to categorize Android malware are as follows:

Data Collection: gather a sample of Android applications containing malware as well as benign applications.

Feature Extraction: select the features within each of the apps in the dataset, i.e. permissions requested, API calls made, etc. The algorithm SPRINT relies on permission, API calls and intent filters.

Data Preprocessing: preprocess the feature information, e.g. normalizing or scaling the data, and transforming categorical data to numerical data.

Training: train the SPRINT algorithm on the processed feature data and the associated labels (i.e. benign or malware). The algorithm is trained to acquire the intent patterns within the dataset differentiating the benign and malicious apps.

Testing: use a different test dataset to test the trained model to make sure it is performing appropriately in solving new, unseen apps.

Deployment: implement the developed SPRINT algorithm on Android devices to scan and classify applications installed on the device in real-time.

The feature extraction and selection

These characteristics are network traffic properties that are often deployed in network intrusion detection systems (NIDS) to categorize network traffic as either malicious or benign. One such classifier is Sprint which identifies malicious network traffic based on these features.

A short description of each attribute is given below:

External-IP: this is the IP address of a remote host on which the local host communicates. The malware can either communicate with already known malicious IP addresses or it can also be trying to conceal its actual IP address.

Source-App-Packets: the quantity of packets which are issued by the source application. Malware can also create large numbers of packets to create vulnerabilities or to interact with its command-and-control (C2) server.

Dist-Port-Tcp: is the destination port number of TCP traffic. The malware can also exploit non-standard ports to avoid detection or make use of vulnerabilities.

Dns-Query-Time: this is defined as the duration that is incurred in resolving a domain name to an IP address. Malware can keep dynamically creating C2 domain names based on domain generation algorithms (DGA), with the effect of making a large number of DNS queries.

Remote-App-Packets: the term is used to describe the number of packets that the remote application has received. The C2 server can send a high amount of packets holding commands or data to malware.

Tcp Packets: the number of packets of TCP between the local and the remote host. Malware can take advantage of a sequence of TCP packets that contains a vulnerability or be used to communicate with its C2 server.

Udp Packets: the number of UDP packets sent between the local and the remote hosts is the total number. UDP packets could be used by malware to avoid detection or simply communicate with its C2 server.

The combination of these characteristics in our research sprint is done through a machine learning algorithm that then categorizes network traffic as malicious or benign. It should be noted though that no single classifier has a hundred percent hit and a hundred percent miss rate and false positives or false negatives are possible.

Hence, it is advised to conduct several classifiers and manual analysis in order to verify the classification.

RESULTS

A confusion matrix is a clear statement of the prediction results of the classification problem. There are two types of predictions like correct or incorrect prediction. Depending on the broken down and count values, the accurate or inaccurate prediction outcome can be recognized. The best thing ict does is to display the error type committed by that particular classifier and the manner in which the error occurred with the classifier.

Description of the Terms

Positive (P): there is positive observation.
Negative (N): it does not make a positive observation.
True Positive (TP): the observation is a positive observation that is expected to be a positive observation.
False Negative (FN): is a positive observation that is forecasted negative.
True Negative (TN): it is a negative observation and is anticipated to be negative.
False Positive (FP): it is an observation that is negative, but it is predicted to be positive.

Recall

The recall is defined as the proportion of accurate positive examples divided by all positive samples. The strong recall and the minimum FN show that examples have been correctly recognized. It is calculated as below.

$$Recall = \frac{TP}{TP + FN}$$

Precision

Precision is the outcome of the proportion of the amount of duly categorized good samples to the amount of forecasted good samples. When the value is high in precision, it represents positive results. It is calculated as below.

$$Precision = \frac{TP}{TP + FP}$$

F-measure

The Recall and Precision are calculated to obtain F-measure. Instead of Arithmetic Mean, F-measure is measured with the help of Harmonic Mean. It is owing to the fact that Harmonic Mean is effective with high values. Among them, F-measure is never bigger than Recall or Precision. It is calculated as below;

$$F - measure = \frac{2 * Recall * Precision}{Recall + Precision}$$

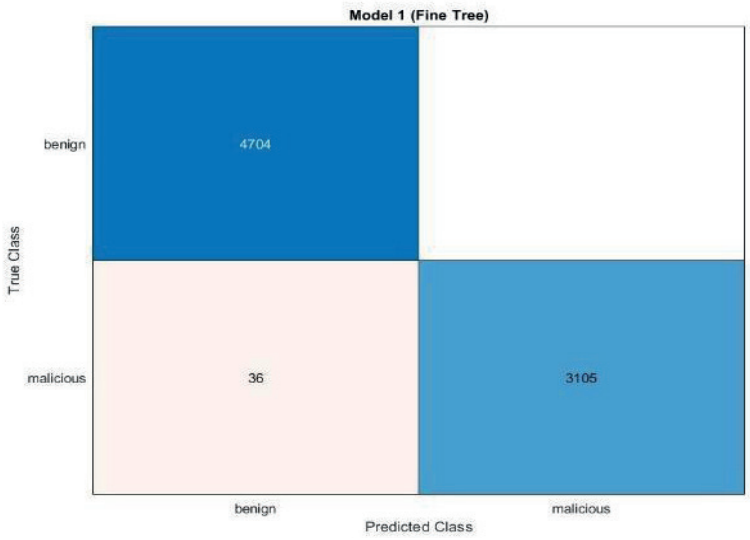

Figure 2. Number of observation malicious vs benign

The x-axis in figure 2 corresponds to the predicted values of the classes, that is, 4704 are predicted to be benign and 36 are predicted to be malicious. The y-axis, on the other hand, is the actual labels of the classes with 36 observations being actually benign, and 3105 observations being actually malicious.

Figure 3. Confusion matrix PPV vs FDR

The confusion matrix of PPV and FDR is in figure 3. Identified labels of the predicted classes are presented in the x-axis of the figure with the two classes being benign and malicious. The four metrics FDR, PPV, number of malicious observations, and number of benign observations are depicted by the y-axis.

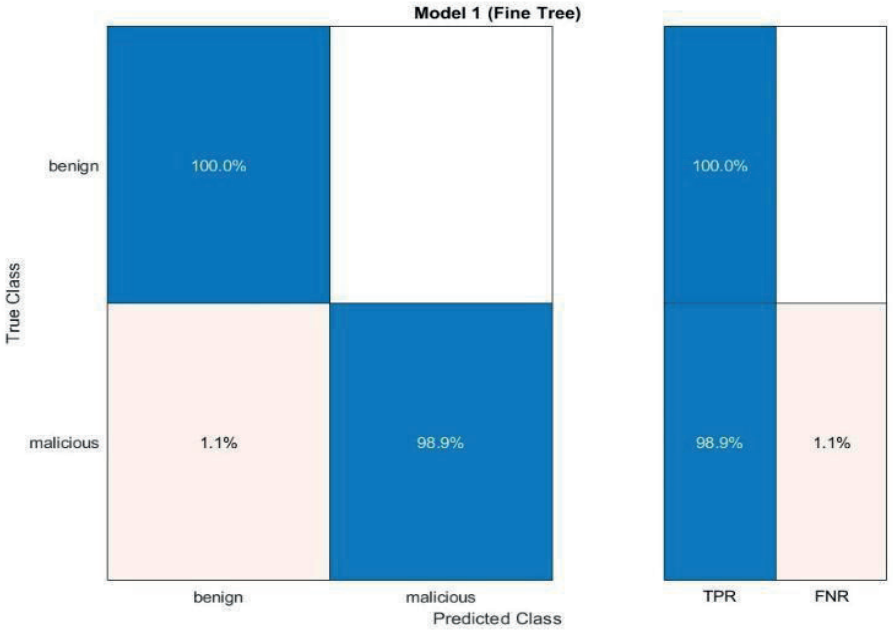

Figure 4. Confusion matrix TPR,FNR

The confusion matrix of TPR and FNR is shown in figure 4. The figure has the predicted labels of classes on the x-axis; these are benign, malicious, TPR and FNR. The two measures are displayed on the y-axis as number of malicious observations, and number of benign observations.

Scatter plot

Classification can be done by a scatter plot where the data points are plotted in a 2-D space, and the various colors or shapes are used to represent the different classes. This enables us to have a visual inspection on the association between the input features and the class labels.

In order to construct a scatter plot to be used in classification, we proceeded as follows:

Gather the data: you will require a collection of data that will be denoted by the label of the various classes it occurs.

Select the features: select two features which you want to plot on the x- and y-axis of the scatter plot.

Plot the data: plot the data on the two-dimensional space, where one of the features is plotted on the x-axis and the other feature is plotted on the y-axis.

The various classes need to be represented by use of different colors or shapes.

Interpret the plot: examine the scatter plot, whether there are patterns or relations of the input features with the class labels. To illustrate, in case the data points of one type are concentrated in a particular region of the plot, this may indicate that the input features, are very correlated with the class label.

It is important to note that scatter plots are most appropriate when the number of input features is small (i.e. two or three). In the case of datasets that use a higher-dimension input feature, plotting is occasionally hard in such a way as to be readily understood visually. Other visualization methods like dimensionality reduction methods (e.g., PCA) can also be applied in such situations in order to project the data to a lower-dimensional space, which can subsequently be plotted more simply.

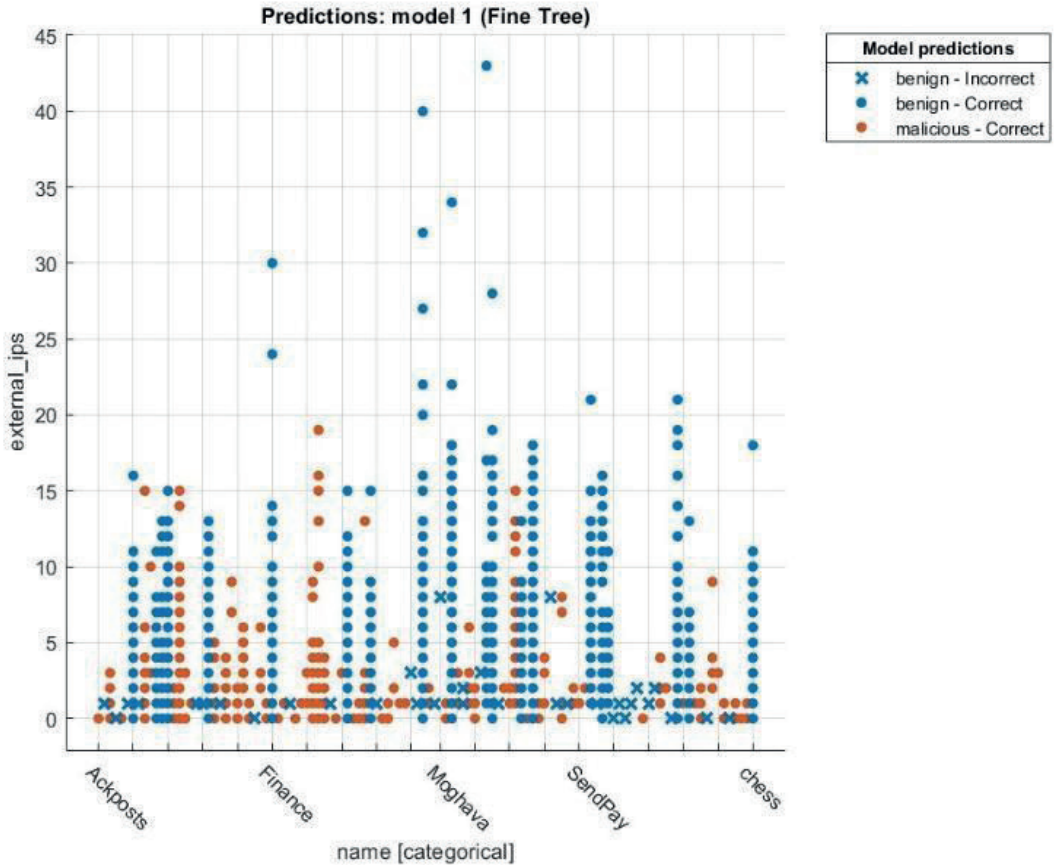

Figure 5. Scatter plot parameters name vs external-ips

The plot of the parameters of scatter plot name vs external-ips is shown in figure 5. Malware or viruses apply the External IP addresses to reach the command-and-control (C&C) servers, through which it controls the malware files and continues the malicious processes in the infected computer. Android malware usage of external IP addresses:

Communication with C&C servers: malware communicates with its C&C server to obtain commands on what to do with the stolen data and transmit them with the help of external IP addresses. These IP addresses are typically obscured or obfuscated to ensure that it becomes hard to detect the C&C server.

Exploit kits Malware infects the device with exploit kits installed in outer IP addresses and exploits the vulnerabilities of the operating system or other applications.

Proxy servers: proxy servers can be exploited by the malware to direct its traffic via another location to escape the external IP addresses.

Ad fraud: malware creates counterfeit clicks or views of advertisements to earn money to the attacker with the help of external IP addresses.

Antivirus software and ensuring that mobile software is updated can be used to block the Android malware that involves the use of outside IP addresses.

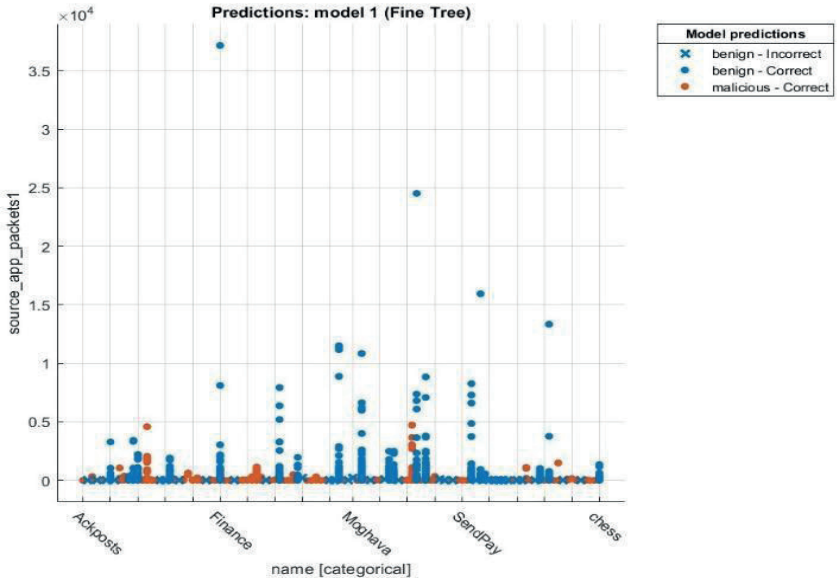

Figure 6. Scatter plot parameters vs source –app-packets

Figure 6 indicates the parameters of the parameters of the scatter plots versus source app packets. In this figure Blue cross symbolizes the benign incorrect files, blue dot symbolizes the benign correct files and orange dot symbolizes the malicious file. The source-app-packets of the android malware implement a range of methodologies such as packet sniffing, man-in-the-middle(MITM) attacks or utilization of hidden APIs to retrieve and transfer network traffic information of the targeted device. The network traffic obtained includes the sensitive data like financial data, the login credentials and confidential information. To prevent such attacks on the device one has to use a secure network connection such as encrypted Wi-Fi or VPN when accessing sensitive data.

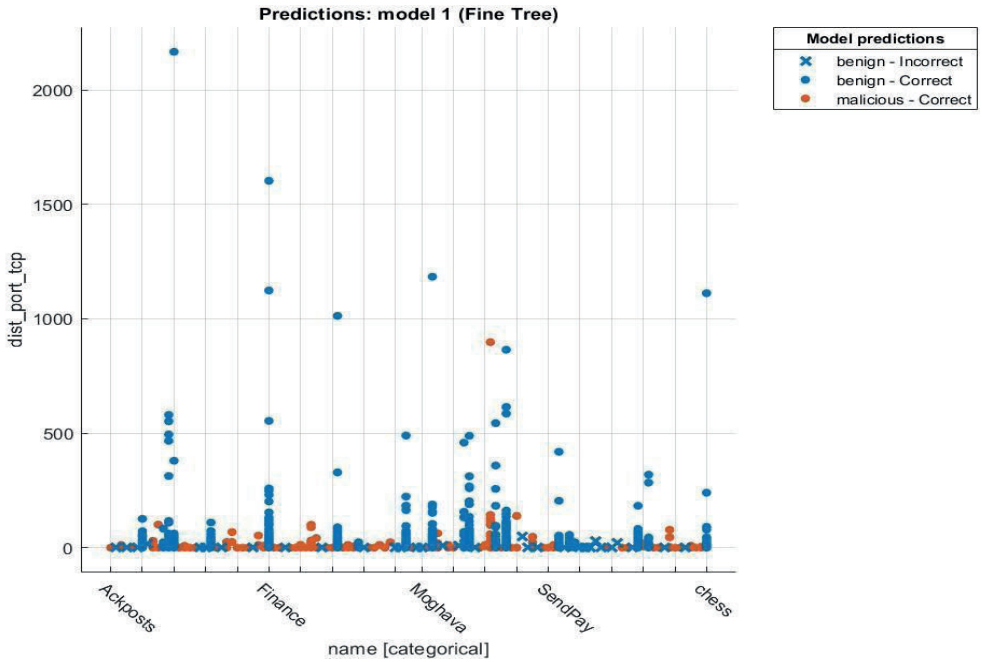

Figure 7. Scatter plot malware parameter vs dist-port-tcp

The comparison of the parameters of the scatter plot to Dist-port-tcp is presented in figure 7. The blue dot depicts the benign files that are correct, the blue cross depicts the benign files that are incorrect and the orange dots depicts the malicious file. Network communication protocols frequently make use of dist-port-tcp. Android devices are based on different methodologies when it comes to communication with its command and control (C&C) server. Network communication protocols through TCP/IP are one of the methods. Malware uses specified TCP ports to use with its network communication strategy to connect with its C&C server or other network devices. To avoid detection, the malware can either hardcode the port number or make it dynamically. This particular application of dist-port-tcp depends on the communication plan of the malware. It is important to point out that the process of Android malware network data analysis may be complex and require specific skills and equipment.

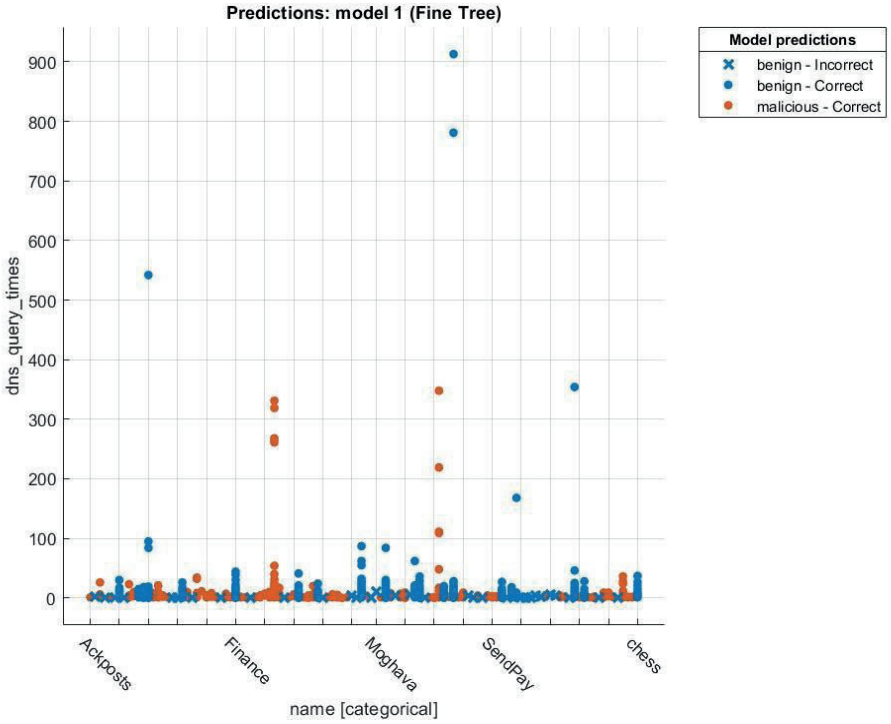

Figure 8. Scatter plot malware parameter vs dns-query-times

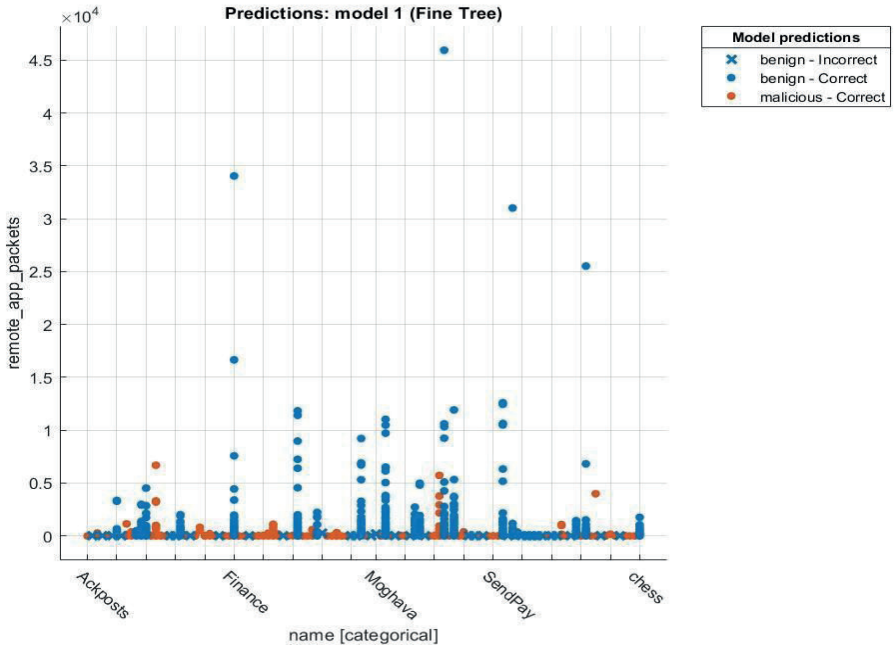

Figure 9. Scatter plot malware parameter vs remote-app-packets

Figure 8 is the comparison of scatter plot parameters against dns-query-times, dns-query-times is the time taken by malware performance of Domain Name System (DNS) query. It is an indispensable part of the internet infrastructure because it is used to translate domain name into IP addresses. The Malware file relies on the DNS queries as the communication strategy in a C and C server or to send off the victims to the malicious sites. In malware DNS query performance, security researchers gain the performance of the malware and identify the existence of the devices by the time they measure the performance. One should keep in mind that malware on Android devices can be hiding its DNS requests and other network traffic with the help of a variety of methodologies such as encrypted communication or non-standard ports of communication.

Figure 9 reproduces the set of parameters of the scatter plot malware comparison with remote-app packets.

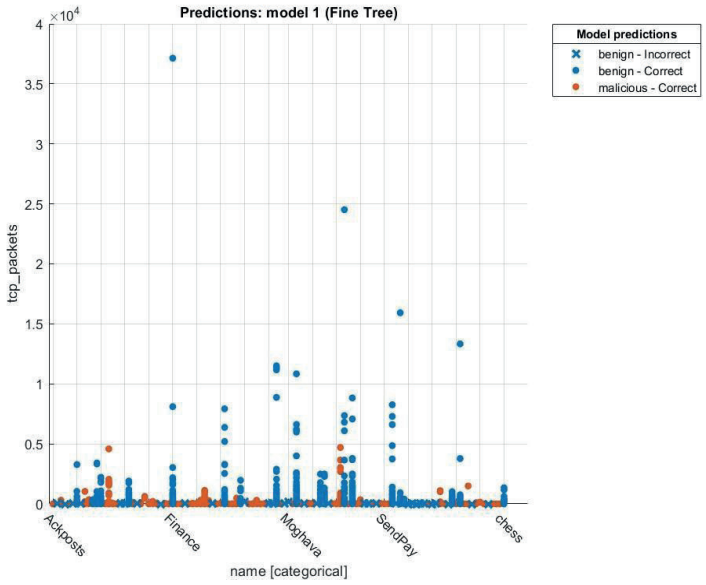

Figure 10. Scatter plot malware plots and tcp_packets

Remote-app-packets employ a large number of methods to connect with a remote server, or carry out commands remotely in malware of the android system. Some of the popular methods are:

- Command and control (C&C) servers: android malware can communicate with a command and control server and receive commands and send them back. The communication is often hidden in normal network traffic.
- Android SMS messages: ccertain Android malware can access a remote server via SMS messaging which has information or some other commands.
- Data exfiltration: android malware can steal such data as personal data contacts, call logs, and text messages and transmit them to a third party server.
- Backdoor access: the malware on Android devices can provide a backdoor accessibility to the device. This will enable an attacker to remotely execute commands as well as to steal data in the device.

The comparison in figure 10 between malware plots and tcppackets using a scatter plot. Android malware can be utilized by TCP packets to communicate with remote servers, download and upload information and execute commands. Malware attackers have been known to utilize TCP packets to send sensitive information that has been stolen by the device, including its login credentials, banking details, and personal data. The malware on the infected device also communicates with TCP packets, including to execute attacks, download and install malware, or steal more data. Android malware can also utilize other network packets like UDP, ICMP and HTTP packets depending on the functionality and purpose of the whoever is making the virus. Of particular importance is the fact that network traffic detection and analysis is also one of the main parts of malware analysis since it could provide valuable information about how the virus acts and help locate the possible vectors of attack and channels of communication.

The comparison of malware in a scatter plot and udppackets is presented in figure 11. Like in TCP, in Android malware UDP packets are used with many purposes. UDP is a connectionless protocol that is often employed in applications that need high and efficient data transfer like real time audio and video streaming, online gaming, and VoIP services. UDP packets are used by malware attackers when they are communicating with remote servers and getting instructions, uploading or downloading data or performing other malicious operations. UDP packets do not require making a connection and maintaining it with the remote server as well as they are hard

to trace and prevent and could therefore be used to evade security software. UDP packets can also be used in DDoS attacks whereby a malware infected device sends a large quantity of UDP packets to a target server causing its failure or inaccessibility.

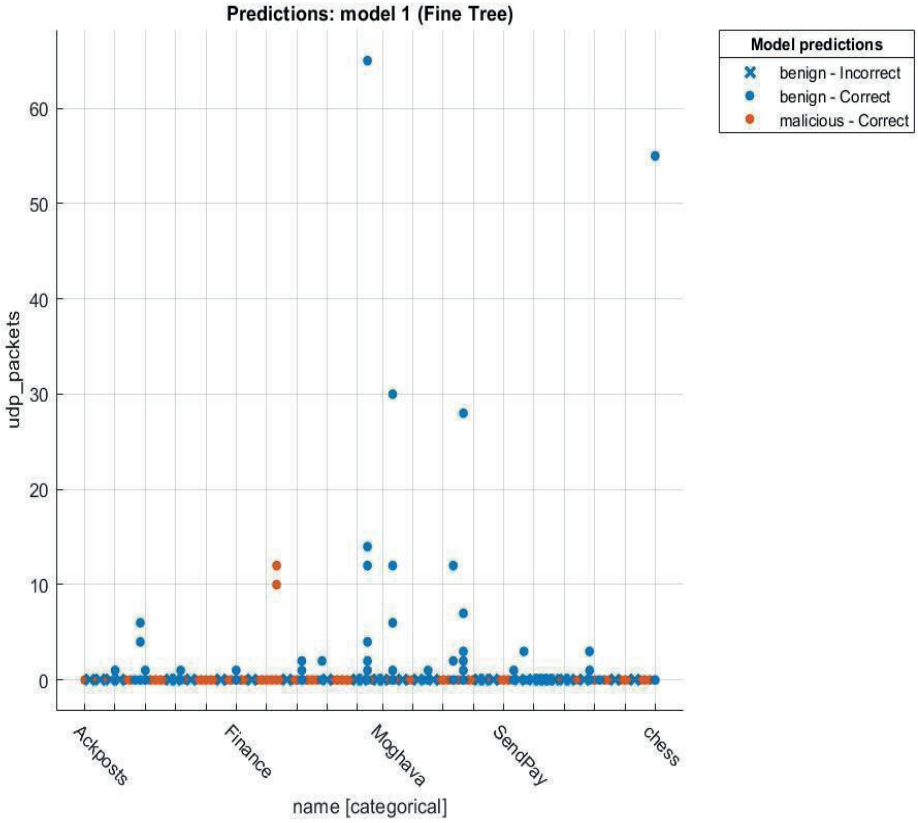

Figure 11. Scatter plot malware plots and udp_packets

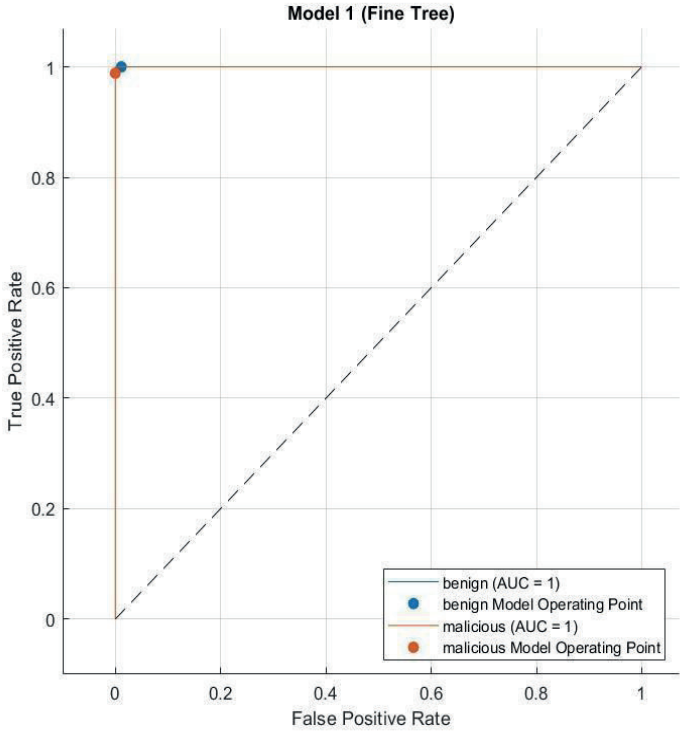

Figure 12. ROC plot false positive vs true positive rates

Figure 12 is the graph that is plotted with the graph ROC plot false positive vs true positive rate. The X-axis is selected with False positive rates that are plotted between 0 and 1 with a difference of 0,2. Blue dot refers to the benign model operating point whereas the orange dot refers to malicious model operating point.

Table 1. Simulation parameter for benign

Parameters	External-IP	Source-app-packets	Dist-port-tcp	DNS-query-time
Ack-post	16	0,3x10^4	700	650
Finance	24	3,5x10^4	1100	300
Moghave	40	1,3x10^4	1400	100
Send pay	20	1,5x10^4	400	180
Chess	18	1,4x10^4	1200	180

Table 2. Simulation values for malware

Parameters	External-IP	Source-app-packets	Dist-port-tcp	DNS-query-time
Ack-post	4	0,1x10^4	50	20
Finance	18	0,15x10^4	250	320
Moghave	14	0,4x10^4	300	300
Send pay	3	0	25	25
Chess	8	0,15x10^4	150	250

Table 1 and 2 demonstrates the values of simulation of a benign and malware dataset. It has five parameters, including External-IP, Source-app-packets, Dist-port-tcp, DNS-query-time, and Ack-post. All parameters are assigned with a particular value on different classes: Finance, Moghave, Send pay and Chess. By this observation benign files take higher values as compared to malware files. Each parameter has a unique value that is used in various classes meaning that the parameters could be applicable in differentiating among various classes of benign network communication. This table is used went it is required to simulate the malware and benign behavior of different applications.

Matlab Simulation Summary Table

Table 3. Simulation Summary table

Accuracy (validation)	99,5 %
Total cost (validation)	36 %
Predication speed	53000 obs/sec
Time	55,2 sec
Model	Hyperparameters
Maximum number of splits	100
Split criterion	Gine diversity index
Feature selection	14/14 individual features selected
PCA	Disabled
Misclassification cost	Default

Table 3 presents the findings of validation of a machine learning model. Average accuracy of the model in predicting the right class of the validation data is known to be 99,5 which means that the model is highly accurate in its predictions. The overall cost of misclassification on the validation data is also presented and is 36 percent. The outcomes of these validations are employed to evaluate the machine learning model performance on making correct predictions of the class label of the validation data. The accuracy is high as well as the misclassification cost is low which means that the model is doing well. MATLAB is applied in our study to identify the performance of the proposed SPRINT algorithm. The dataset is run in the simulation environment, which is the necessary parameters that include accuracy as a percentage, the speed of predictions, time, and model used to execute the implementation. The table above 1 gives a detailed summary of the parameters that would be used in the proposed execution.

Table 4. Accuracy Comparison Table

Classifier	Accuracy
Proposed SPRINT	99,5 %
RNN	96 %
DNN	98 %

Table 4 is a comparison of SPRINT algorithm suggested with Ada boost with random forest and SVM, KNN. The comparison work is implemented in MATLAB and thereafter analysed on basis of accuracy. The presented algorithm, the SPRINT, has achieved the highest accuracy of 99,5 and 96 and 98 with RNN and DNN respectively.

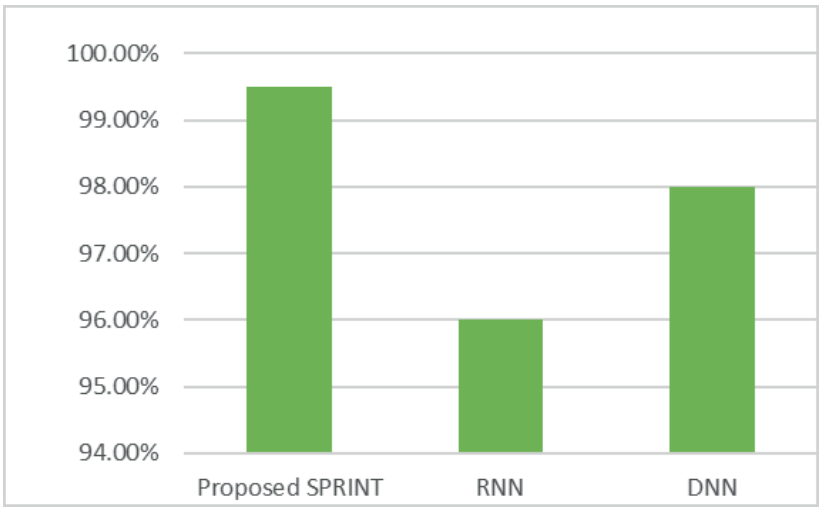

Figure 13. Accuracy comparison

According to figure 13, the bar graphical illustration corresponds to the accuracy of three distinct classifiers Proposed SPRINT, RNN and DNN. Proposed SPRINT has been reported to have the highest accuracy of 99,5 that is higher than the two classifiers. This implies that the RNN and DNN classifier are also doing well in classifying the correct class labels of the test set, but not so accurate as the Proposed SPRINT classifier. Therefore Proposed SPRINT classifier is highly precise in its predictions and it could probably classify most of the data of the test set correctly.

CONCLUSIONS

The issue of cyber threats has been on the increase in the current digital era and malware is one of the major security threats facing both the individual and the company. Newer and more sophisticated forms of malware may not be controlled by the traditional anti-malware systems therefore requiring more complex methods. The usage of the machine learning algorithms DNN, RNN, and SPRINT classifiers that are used to detect Android devices malware is the topic of discussion in this paper. Our proposed SPRINT classifiers achieved a classification accuracy of 99,5 percent in the experiment. Therefore, our offered methodologies can be utilized to detect and categorize new and sophisticated Android malware that will result in securing the safety of Android devices and prevent future cyber-attacks. The research of machine learning algorithms is a prospective domain that can aid to create more effective and efficient solutions to cybersecurity.

BIBLIOGRAPHIC REFERENCES

1. Suresh, S.; Di Troia, F.; Potika, K.; Stamp, M. An analysis of Android adware. Journal of Computer Virology and Hacking Techniques 2019, 15, 147–160.

2. Yilmaz, S.; Zavrak, S. Adware: a review. International Journal of Computer Science and Information Technologies 2015, 6, 5599–5604.

3. Freeman, L.A.; Urbaczewski, A. Why do people hate spyware? Communications of the ACM 2005, 48, 50–53.

4. Shefali, S.; Jolivot, R.; Choensawat, W. Android malware classification based on mobile security framework. IAENG International Journal of Computer Science 2018, 45, 514–522.

5. Altaher, A.; Mohammed, O. Intelligent hybrid approach for Android malware detection based on permissions and API calls. International Journal of Advanced Computer Science and Applications 2017, 8, 60–67.

6. Ndagi, J.Y.; Alhassan, J.K. Machine Learning Classification Algorithms for Adware in Android Devices: A Comparative Evaluation and Analysis. In Proceedings of the 2019 15th International Conference on Electronics, Computer and Computation (ICECCO), Abuja, Nigeria, March 2019; pp. 1–6.

7. Yerima, S. Android Malware Dataset for Machine Learning 2 figshare. Dataset. 2018.

8. Alazab, M.; Layton, R.; Venkataraman, S.; Watters, P. Malware detection based on structural and behavioral features of API calls. 2010.

9. Jeon, S.; Moon, J. Malware-detection method with a convolutional recurrent neural network using opcode sequences. Information Sciences 2020, 535, 1–15.

10. Yazdinejad, A.; HaddadPajouh, H.; Dehghantanha, A.; Parizi, R.; Srivastava, G.; Chen, M.-Y. Cryptocurrency malware hunting: A deep recurrent neural network approach. Applied Soft Computing 2020, 96, 106630.

11. Darabian, H.; Homayounoot, S.; Dehghantanha, A.; Hashemi, S.; Karimipour, H.; Parizi, R.M.; Choo, K.K.R. Detecting cryptomining malware: A deep learning approach for static and dynamic analysis. Journal of Grid Computing 2020, 18, 293–303.

12. Qiu, J.; Zhang, J.; Luo, W.; Pan, L.; Nepal, S.; Xiang, Y. A survey of android malware detection with deep neural models. ACM Computing Surveys 2020, 53, 1–36.

13. Vinayakumar, R.; Alazab, M.; Soman, K.; Poornachandran, P.; Venkatraman, S. Robust intelligent malware detection using deep learning. IEEE Access 2019, 7, 46717–46738.

14. Mahindru, A.; Sangal, A.L. MLDroid: framework for Android malware detection using machine learning techniques. Neural Computing & Applications 2020, 33, 5183–5240.

15. Ma, Z.; Ge, H.; Liu, Y.; Zhao, M.; Ma, J. A combination method for android malware detection based on control flow graphs and machine learning algorithms. IEEE Access 2019, 7, 21235–21245.

16. Cai, L.; Li, Y.; Xiong, Z. JOWMDroid: android malware detection based on feature weighting with joint optimization of weight-mapping and classifier parameters. Computers & Security 2021, 100, 102086.

17. Fallah, S.; Bidgoly, A. Benchmarking machine learning algorithms for malware detection in smartphones. Jordanian Journal of Computers and Information Technology 2019, 5, 1.

18. Mao, Y.; Yang, Y. A wrapper feature subset selection method based on randomized search and multilayer structure. BioMed Research International 2019, 9864213, 1–9.

19. Gavriluţ, D.; Cimpoesu, M.; Anton, D.; Ciortuz, L. Malware detection using machine learning. In Proceedings of the 2009 International Multiconference on Computer Science and Information Technology, Mragowo, Poland, 12–14 October 2009; pp. 735–741.

20. Pavithra, J.; Josephin, F.J.S. Analyzing various machine learning algorithms for the classification of malwares. IOP Conference Series: Materials Science and Engineering 2020, 993, 012099.

21. Mohanraj, A.; Sivasankari, K. Hybrid Temporal Convolutional Networks with LSTSVM Model for Android Malware Detection Using Explainable AI. Journal of Electrical Engineering & Technology, Springer Nature Singapore, 1-11.

22. Mohanraj, A.; Sivasankari, K. Android traffic malware analysis and detection using ensemble classifier. Ain Shams Engineering Journal, Elsevier, 103134.

FINANCING

No financing.

CONFLICT OF INTEREST

None.

AUTHORSHIP CONTRIBUTION

Conceptualization: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Data curation: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Formal analysis: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Research: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Methodology: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Project management: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Resources: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Software: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Supervision: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Validation: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Display: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Drafting - original draft: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.
Writing - proofreading and editing: Rajkumar, Gobinath R, J Thimmiaraja, K.Sathesh Kumar, Viji C, A Mohanraj.

https://doi.org/10.56294/dm20251258 ISSN: 2953-4917