International Journal of Engineering Sciences & Emerging Technologies, Oct. 2023.
ISSN: 22316604 Volume 11, Issue 2, pp: 55-61 ©IJESET

MALWARE DETECTION IN ANDROID MOBILE PLATFORM USING MACHINE LEARNING ALGORITHMS

Richa Gupta, Sherin Zafar, Imran Hussain, Siddhartha Sankar Biswas

Department of CSE, SEST, Jamia Hamdard, New Delhi, India
richagupta@jamiahamdard.ac.in
sherin.zafar@jamiahamdard.ac.in
ihussain@jamiahamdard.ac.in
ssbiswas@jamiahamdard.ac.in

ABSTRACT

Android is the foremost prevalent portable working framework and with more than 3 billion dynamic clients. Android has ended up one of the hot targets for malware in later a long time. This malware incorporates spyware, adware, keeping money malware etc. The number of malware assaults and the complexity of assaults is expanding day by day. In this extend, we attempt to come up with a quick and dependable way to identify the malware by analyzing the malware statically and powerfully. We change over the malware tests into grayscale pictures and extricate highlights from these pictures for the inactive examination. Energetic investigation is done by extricating highlights based on the applications framework calls done amid its run time. Authors in this research study utilize diverse machine learning calculations which take these highlights as an input to recognize the malware tests from the generous tests. Utilizing both inactive and energetic investigation authors perform a crossover analysis to grant an improved exactness at identifying the malware.

.

KEYWORDS: *Android, Machine Learning, Python, Security, malware.*

1. INTRODUCTION

The number of smartphone gadgets in utilize is expanding exponentially. It has gotten to be a fundamental contraption of necessity because it is utilized for numerous day-to-day errands like advanced instalments, GPS route, communication etc. The Android working framework holds 70 percent of the total portable advertise and the number of other contraptions based on android like keen TVs are also expanding each year. With this colossal share, android includes a high-security risk in case of a cyber assault, compromising the security of millions of individuals. Malware is computer program that's basically planned to the private data display on the gadget or harm the gadget. The Android working framework is designed to be adaptable which opens it to more dangers. Users are able to introduce modern applications from any source. The foremost utilized source for android apps Google PlayStore is the most conveyance vector for most android malware. 67 percent of the malevolent app introduces come from the Google play store. One such later malware is the BRATA malware which can remotely get to a user's gadget and take cash utilizing managing an account apps by showing fake login pages. After taking it resets the android gadget to plant settings taking off no follow of the malware. A few of the prevalent sorts of android malware are:

• Adware: This promotion malware application tosses undesirable promotions on the client screen and produces income.

• Ransomware: It makes the gadget blocked off by scrambling all records on the portable. In arrange to get to the gadget, it inquires for colossal sums of cash.

DOI: 10.5281/zenodo.10434257

• Spyware: This program application once introduced can take delicate data. Ordinary applications inquire the client for the camera, mouthpiece consents, etc but spyware is introduced without the user's consent.

• Backdoor: Backdoors give a covered-up door for aggressors to get to the gadget. Aggressors can inaccessibly control the victim's gadget.

• Worm: The most point of such malware is to make duplicates of itself and spread to other gadgets. It may too contain a few malevolent codes.

Since the number of malwares is expanding day by day, the Android malware location strategies are continually advancing to keep pace. Android malware location methods can be classified into two particular categories: Inactive investigation and energetic investigation. The inactive examination employments the data dwelling inside the apk record to classify the app as malware or generous. The data analyzed is primarily the source code, consents, components, etc. Inactive investigation can encourage be classified into distinctive investigations such as signature/pattern-based, resources-based, components based, permissions-based, etc. In energetic investigation, the energetic behavior of the malware is observed. The energetic behavior incorporates the framework calls, organize exercises etc. Both these strategies can be combined to do a cross breed examination that employments both the data inside the apk and the energetic behavior of the malware. Machine learning has shown good comes about in taking advantage of the designs within the information and utilizing it for classification issues. It has appeared guarantees to keep pace with the ever-evolving malware as compared to the conventional malware discovery procedures. The machine learning demonstrate takes the application information as an input and distinguishes whether the application is malware or generous [1,2,3,4]. In the upcoming sections, Section 2 describes methodology of the proposed work, followed by results and simulations in Section 3 and conclusion and future work in Section 4. References are listed at the end of the paper.

2. METHODOLOGY

Authors in this research study have gathered apk files (both benign and malware) from MalDroid and Androzoo. These datasets are a huge repository of malware of all types including adware, banking, SMS malware etc. Since these datasets are huge, we will filter out a few sample apks to include all kinds of malware. We will verify the correctness of the apk files whether it is malware or benign using freely available sources like virus total. The research study has extracted the classes.dex file from the apk file using the apktool which presents the data in a more readable way, to obtain different information about the application including the permissions required, activities etc. For the static analysis, we will extract the features from the Android apk and then convert it into an 8-bit grayscale image. These images will be generated for all applications and will be used to build the machine learning model. We will compare the performance with different models like Decision Trees, Random Forest and Convolutional Neural Networks. GIST feature extractor will be used for the Decision Trees and Random Forest as CNNs can extract features by itself [5,6,7,8,9,10]. Figure.1 depicts the static analysis workflow model.

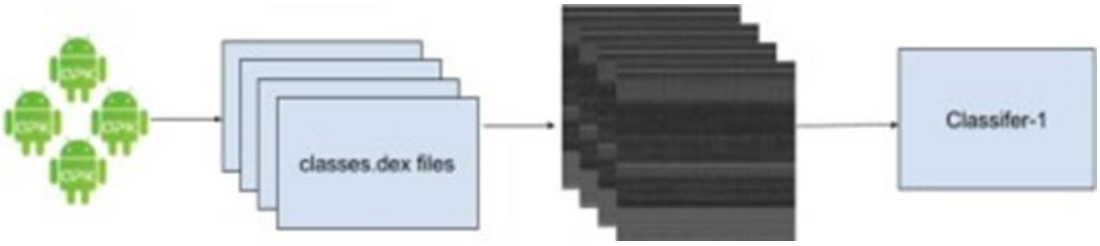

Figure1. Static Analysis Workflow

DOI: 10.5281/zenodo.10434257

For dynamic analysis we plan to run the application in a sandboxed environment to mimic the malware behaviour. We will use an android emulator to run the application and obtain the frequency of system calls used [11,12,13,14]. We will gather this data in an automated way using monkey runner to run the apks and capture the data from the emulator as depicted in Figure.2.

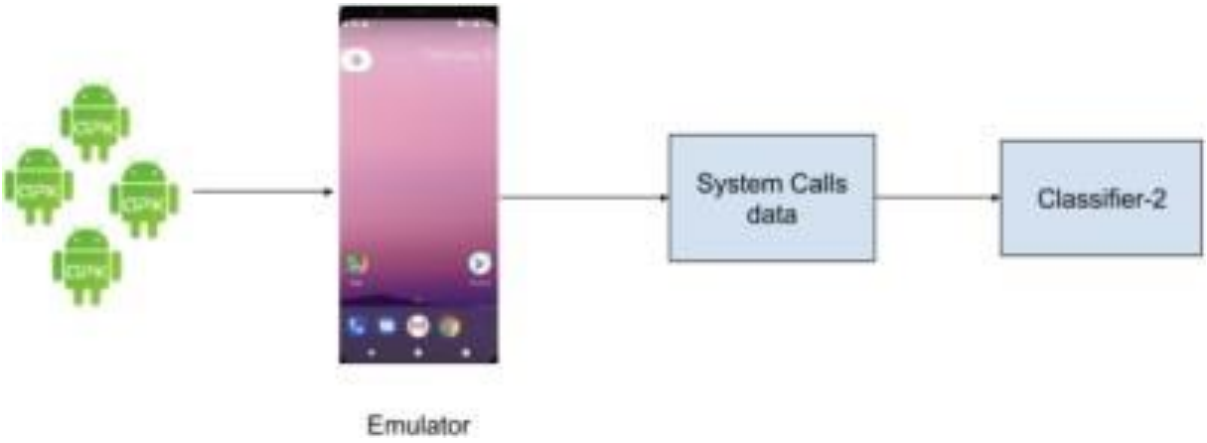

Figure.2: Dynamic analysis workflow

Authors have followed a hybrid approach by combining the scores of both the classifiers to the final prediction by also comparing the performance of this approach by using different machine learning classifiers as depicted in Figure.3.

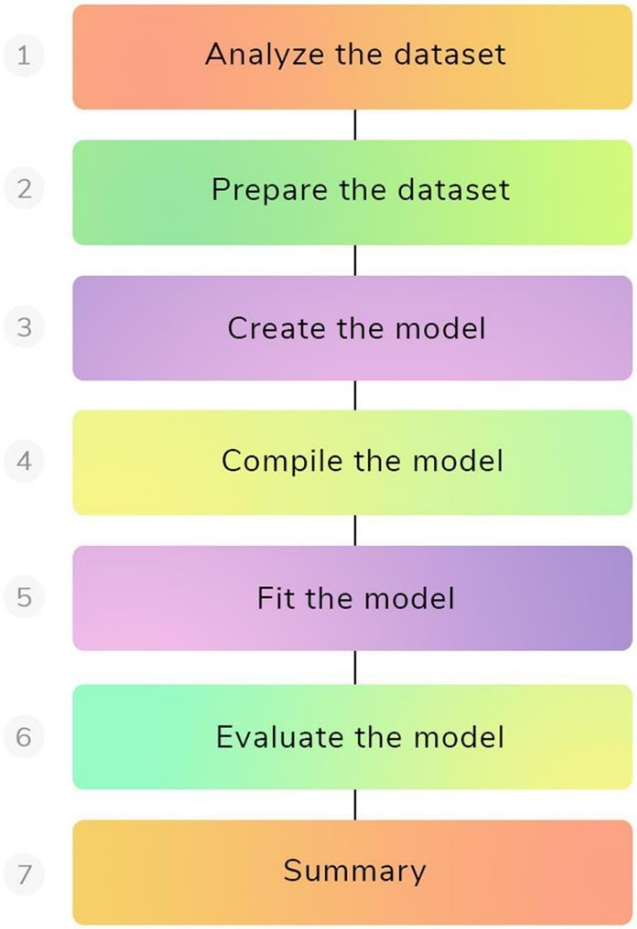

Figure.3 Data Flow Diagram

DOI: 10.5281/zenodo.10434257

3. RESULTS AND SIMULATION

This research work is implementing a machine learning classifier, we will use standard techniques to evaluate our model. K-fold cross validation is used in which data is split into k groups and k-1 groups are used for training and the remaining 1 group is used for testing. The average accuracy of all k iterations is considered as depicted in Figure.4. Confusion matrix gives an idea about the performance of our classifier. Confusion matrix will be utilized to calculate different metrics like accuracy, precision, recall etc. Values in the confusion matrix are True positives, True negatives, False positives, False negatives. Utilizing these values able to calculate exactness, exactness, review etc. Exactness isn't continuously a great degree for imbalanced information subsequently we require accuracy and review. Exactness is the proportion between Genuine positives (classified as malware and are really malware) and Genuine positives additionally wrong positives (add up to number of tests anticipated as malware within the testing information). Review is the proportion between Genuine positives (classified as malware and are really malware) and Genuine positives additionally Untrue negatives (add up to number of tests in testing information that are really malware). Both accuracy and review ought to be as tall as conceivable.

Figure.4 Fold cross validation

Accuracy = TP + TN / (TP + TN + FP + FN)

Precision = TP / (TP + FP) Recall = TP / (TP + FN)

To take these two metrics into consideration the research study has obtained the harmonic mean of precision and recall, this metric is called the F1 score. F1 score should also be as high as possible. Classifiers that have similar precision and recall values have a higher F1 score as compared to the classifiers where one is low and the other one is high. While the confusion matrix can't be described by a single number, the Mathews correlation coefficient (MCC) (also known as phi coefficient) is considered one of the best measures to describe the confusion matrix. It ranges from -1 to 1 where 1 represents the perfect prediction and -1 represents the inverse prediction. Since we plan on using 3 classifiers (Decision Trees, Random Forests, CNNs), we will also compare the performance of each classifier using the above metrics. Authors have compared the performance for each type of analysis, that is, static, dynamic and hybrid. Figures 5-10 depicts the simulation workflow of proposed work.

DOI: 10.5281/zenodo.10434257

Figure.5 Importing the dataset

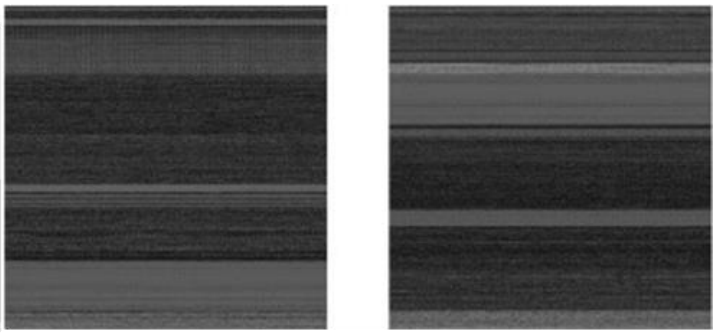

Figure.6 8-bit grayscale images of the sample malware

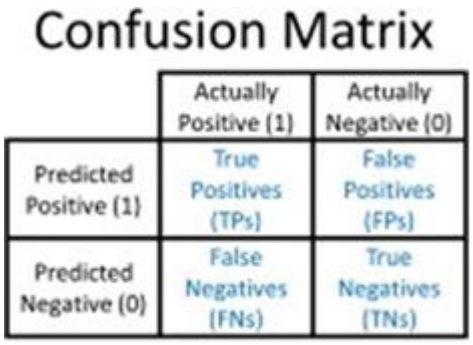

Figure.7 A sample confusion matrix

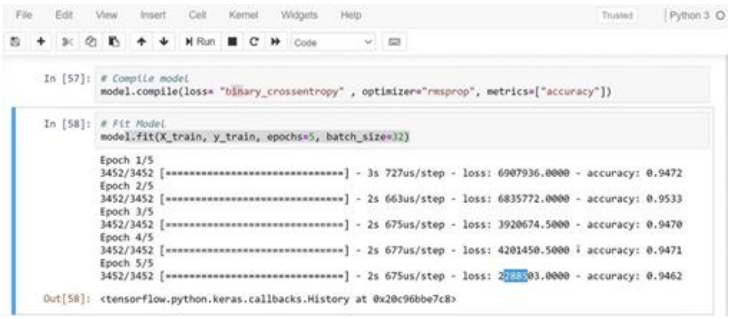

Figure.8 Loading dataset

Models' performance Comparison

Models	Train Data Accuracy	Test Data Accuracy	F1- score	Position
Random Forest	0.982 %	0.983 %	0.973 %	1st
Logistic Regression	0.70%	0.69%	0.0 %	3rd
Neural Networks	0.951 %	0.953 %	0.91 %	2nd

Figure.9 Comparing performance of all 3 models

DOI: 10.5281/zenodo.10434257

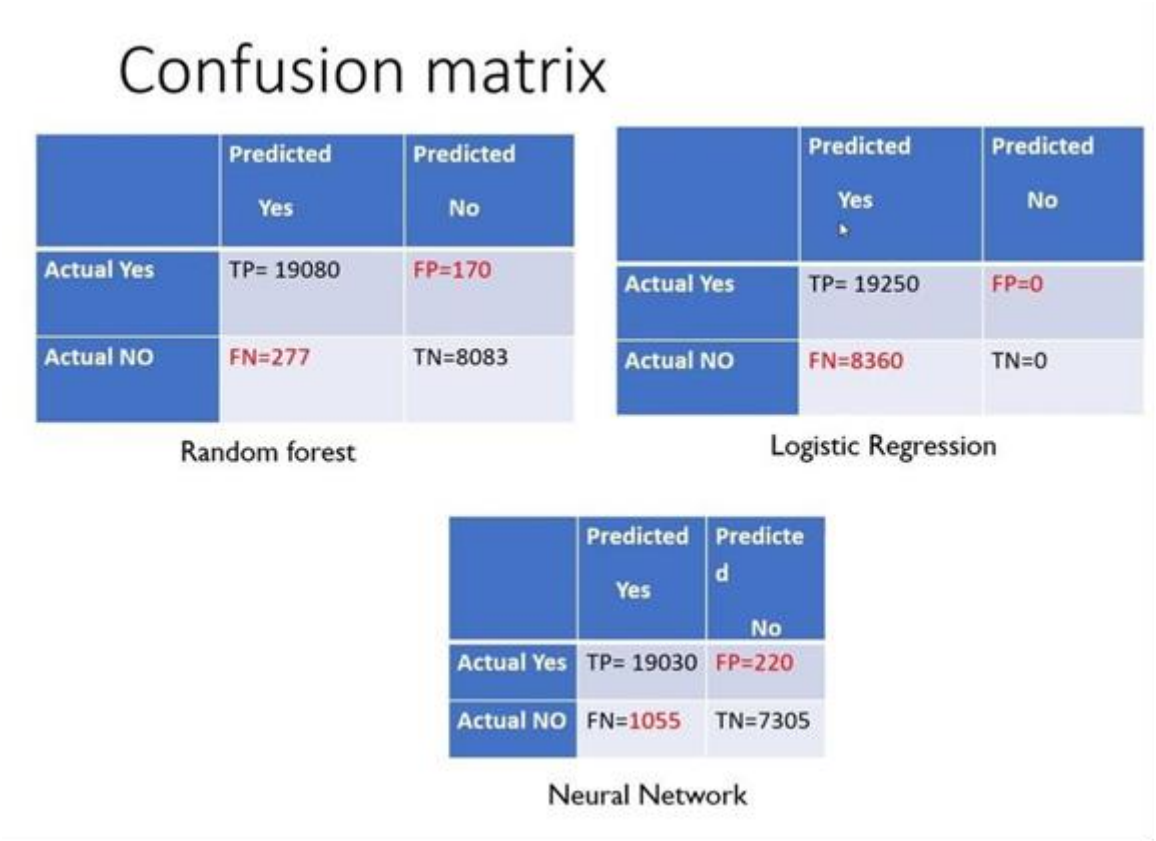

Figure.10 Finally obtained confusion matrix

4. CONCLUSION

We implemented a fast and reliable malware detection approach using machine learning. The static analysis is the first step to detect the malware, since visualization techniques combined with machine learning showed good results. We incorporated one such approach by converting the classes.dex file into image and using it for classification. Static analysis is not always sufficient to detect the malware with recent malware using code obfuscation and encryption, so we combined it with a dynamic analysis approach. We obtained the system calls data by running the application in a sandboxed environment. Individually performing static or dynamic analysis may not always provide the best results. By combining these two approaches in a single analysis known as hybrid analysis we got better results. The overall highest accuracy 97% is achieved in the detection process by our model. Using different machine learning classifiers, we compared the performance of each model by evaluating each classifier on the basis of different metrics like precision, recall, accuracy and F1 score etc

REFERENCES

[1] Prerna Agrawal and Bhushan Trivedi (2019). "A survey on android malware and their detection techniques". pages 1–6, 02. doi: 10.1109/*ICECCT*.8868951.

[2] https://dl.acm.org/doi/10.1145/2901739.2903508

[3] https://www.researchgate.net/publication/339218483_Android_Malware_Detection_through_Machine_Learning_Techniques_A_Review

[4] Daniel Arp, Michael Spreitzenbarth, Malte Hübner, Hugo Gascon, and Konrad Rieck. Drebin (2014), "Effective and explainable detection of android malware in your pocket". 02. doi: 10.14722/ndss.2014.23247

[5] https://www.tutorialspoint.com

[6] https://www.machinelearningmastery.com

DOI: 10.5281/zenodo.10434257

[7] https://www.unb.ca/cic/datasets/andmal2017.html

[8] https://www.javatpoint.com/machine-learning-random-forest-algorithm

[9] M. Al Ali (2016), Malware detection in Android mobile platforms using data mining algorithms.

[10] Z. Aung and W. Zaw (2013), "Permission-based android malware detection", International Journal of Scientific and Technology Research, vol. 2, pp. 228-234.

[11] N. Bhargava, G. Sharma, R. Bhargava and M. Mathuria (2013), "Decision tree analysis on J48 algorithm for data mining", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, pp. 1114-1119, 2013.

[12] L. Breiman (2001), "Random forests", Machine Learning, vol. 45, pp. 5-32.

[13] I. Burguera, U. Zrutuza and S. Nadjm-Tehrani (2011), "Crowdroid: Behavior-based malware detection system for Android", Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, pp. 15-26.

[14] M. Damshenasa, A. Dehghantanha, K.-K. R. Choo and R. Mahmud (2015), "M0Droid: An Android behavioral-based malware detection model", Journal of Information Privacy and Security, vol. 11, pp. 141-157.

DOI: 10.5281/zenodo.10434257