A project report on

ANDROID MALWARE DETECTION USING MACHINE LEARNING BASED ON FINGERPRINT

Submitted in partial fulfillment for the course

Malware Analysis in Data Science (CSE4053)

by

Yashika Singh (20BAI1123)
Laksha S (20BAI1186)
G Barani Raj (20BAI1172)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2023

FACULTY SIGN

Dr Jayasudha

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

DECLARATION

We hereby declare that the thesis entitled "**ANDROID MALWARE DETECTION USING MACHINE LEARNING BASED ON FINGERPRINT**" submitted by **Yashika Singh (20BAI1123), Laksha S (20BAI1186), Barani Raj (20BAI1172)** for the completion of the course, **Malware Analysis in Data Science (CSE4053)** is a record of bonafide work carried out by Yashika Singh (20BAI1123), Laksha S (20BAI1186), G Barani Raj (20BAI1172) under the supervision of Dr Jayasudha, out course instructor. We further declare that the work reported in this document has not been submitted and will not be submitted, either in part or in full, for any other courses in this institute or any other institute or university.

Place: Chennai

Date:

Signature of the Candidate

Yashika Singh

Laksha S

Barani Raj

ABSTRACT

We have suggested employing fingerprint-based ML for Android malware detection. Which places a strong emphasis on the following factors: (1) the ability to scale over a huge corpus of malware; (2) the resistance to typical obfuscation techniques; and (3) the portability across various systems and architectures. In the context of bulk and offline detection at the level of the app market, an approximative fingerprinting approach for android packaging that captures the underlying static structure of the android applications is used. On top of this fingerprinting approach, we have suggested a malware clustering framework to carry out malware clustering by constructing and splitting the similarity network of dangerous apps. Second, a rough fingerprinting method that produces Android malware activity reports by combining dynamic analysis and natural language processing methods. We have suggested a malware detection framework using machine learning classification based on this fingerprinting approach. t may be deployed on mobile devices and classifies method call sequences using machine learning. Additionally, it uses natural language processing and deep learning techniques to be resistant to popular code obfuscation tactics and adaptable to malware and operating system changes over time.

INTRODUCTION

A project aimed at identifying and detecting harmful applications on the Android platform uses machine learning (ML) based on fingerprints to identify and detect Android malware. The project uses machine learning methods to generate a fingerprint of a specific app based on its permissions, APIs, and code. Next, by comparing it to a database of fingerprints associated with known malware, this fingerprint is used to detect harmful programmes.Data gathering, feature extraction, model training, and testing are a few of the processes that the project entails.After that, the fingerprint data is used to train the ML model. The model is trained to identify the patterns and traits of existing benign applications and malware, enabling it to categorize new apps as harmful or benign. In order to assess the model's precision and usefulness in identifying malware, a different set of data is used to test it.The overall goal of the project to detect and block the installation of harmful apps on Android devices is to increase the security of Android devices. This research has the potential to increase the effectiveness and precision of Android malware detection through the use of ML algorithms and fingerprinting methods.

RELATED WORK

[1] A Survey of Android Malware Detection with Deep Neural Models

A disruptive technology called Deep Learning (DL) has altered the field of cyber research. When there is a vast amount of data available, deep learning models have many advantages over typical Machine Learning (ML) models. Due to the rapidly increasing number of Android malware, the obfuscation of Android malware, and the possible protection of enormous amounts of data assets kept on Android devices, Android malware detection or classification qualifies as a big data problem. The use of DL to Android malware detection looks like a logical choice. The selection of DL architecture, feature extraction and processing, performance measurement, and even acquiring enough high-quality data are issues faced by academics and practitioners.By thoroughly examining the most recent developments in DL-based Android malware detection and classification, we intend to overcome the problems in this survey. According to the DL architecture, which includes FCN, CNN, RNN, DBN, AE, and hybrid models, we classify the literature. With an emphasis on describing code semantics for Android malware detection, the objective is to highlight the research frontier. We also go over the difficulties facing this young discipline and offer our opinions on potential future research opportunities.

[2] Hybrid sequence-based Android malware detection using natural language processing Attackers have targeted the Android platform because of its popularity and openness. Recent years have seen a dramatic rise in Android malware, posing severe concerns to Android security. Therefore, recommending effective Android malware detection techniques is critical in thwarting malware. Recently, malware detection has relied heavily on a variety of features that machine learning has retrieved from static or dynamic analysis.Purely dynamic analysis systems cannot discover all possible code execution paths, but existing code obfuscation, code encryption, and dynamic code loading techniques can be used to hamper systems that are only reliant on static analysis. We suggest CoDroid, a sequence-based hybrid Android malware detection method that makes use of sequences of static opcode and dynamic system call, as a solution to these problems. We create a CNN-BiLSTM-Attention classifier, consisting of Convolutional Neural Networks (CNNs), the Bidirectional Long Short-Term Memory (BiLSTM), and an attention language model, and we regard one sequence as a sentence in natural language processing. We thoroughly assess CoDroid using a real-world data set and conduct in-depth analysis against other current similar detection techniques.

[3]"An Empirical Analysis of Android Malware Detection using Machine Learning Techniques" by N. Maddodi and R. V. G. Raja Kumar: This study presents a comprehensive review of the various machine learning techniques used in Android malware detection, including supervised and unsupervised methods.The study evaluates the performance of three different machine learning algorithms, namely SVM, decision trees, and Naive Bayes, using a dataset of Android applications.The authors conducted experiments to compare the accuracy, precision, recall, and F1-score of the three algorithms. They used different feature sets, including permissions, API calls, and intent actions, to train the models and evaluate their performance. The study also examines the impact of the dataset size and the class imbalance problem on the performance of the models.The results of the experiments show that SVM outperforms decision trees and Naive Bayes in terms of accuracy and F1-score, while decision trees perform better in terms of recall. The study also highlights the importance of selecting appropriate feature sets and balancing the dataset to achieve better performance.

[4]"Android Malware Detection based on Fuzzy Fingerprinting" by S. S. Tiwari and S. K. Pal: This research proposes a novel approach to Android malware detection based on fuzzy fingerprinting, which uses a fuzzy matching algorithm to compare the characteristics of an application with known malware fingerprints.The approach uses a fuzzy matching algorithm to compare the characteristics of an application with known malware fingerprints.The authors conducted experiments to evaluate the effectiveness of the proposed approach using a dataset of Android applications. The study compares the performance of fuzzy fingerprinting with other machine learning-based approaches, including SVM and decision trees, and hybrid approaches.The results of the experiments show that fuzzy fingerprinting achieves a higher detection rate compared to other approaches, with a lower false positive rate. The study also shows that combining fuzzy fingerprinting with other machine learning algorithms can further improve the detection performance.

[5]"A Survey of Machine Learning Techniques for Android Malware Detection" by F. Wang and Y. Zhang: This study provides an overview of the different machine learning techniques used in Android malware detection, including feature selection, classification algorithms, and ensemble learning methods.The study covers various machine learning algorithms, including supervised and unsupervised learning, and discusses their effectiveness and limitations.The authors provide an overview of the Android platform and its security architecture, including the different types of Android malware and their attack vectors. They then discuss the challenges of Android malware detection, such as the diversity of Android applications, the dynamic behavior of malware, and the large scale of the Android ecosystem.

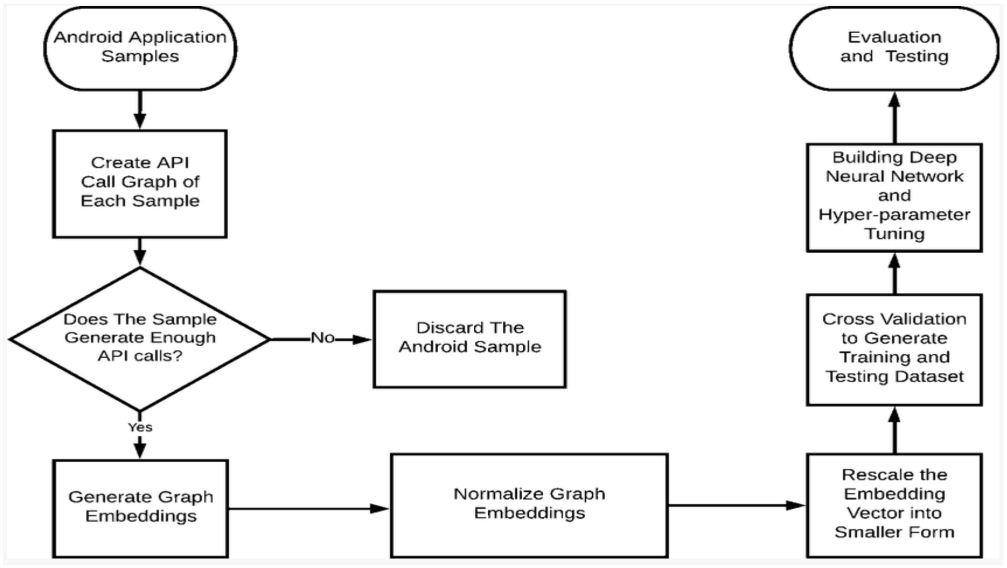
[6]"A Deep Learning Approach to Android Malware Detection" by S. Zhang, X. Wang, and Y. Liu: This research proposes a deep learning-based approach to Android malware detection using a convolutional neural network (CNN) to extract features from the raw binary code of an application.The authors use a convolutional neural network (CNN) to extract features from the Android application's code and use them to classify the application as benign or malware.The authors conducted experiments to evaluate the effectiveness of their proposed approach using a dataset of Android applications. They compared their approach with other machine learning-based approaches, such as SVM and decision trees, and demonstrated that their deep learning approach achieved higher accuracy, precision, recall, and F1-score.

[7]"Android Malware Detection using Machine Learning Techniques: A Review" by S. Arora, R. Jain, and M. Gupta: This research provides a review of the various machine learning techniques used in Android malware detection, including static and dynamic analysis, as well as hybrid approaches.The study covers various machine learning algorithms, such as decision trees, SVM, neural networks, and ensemble methods.The authors provide an overview of the Android platform and its security architecture, including the different types of Android malware and their attack vectors. They then discuss the challenges of Android malware detection, such as the diversity of Android applications, the dynamic behavior of malware, and the large scale of the Android ecosystem.

PROPOSED ARCHITECTURE

The following levels are included in the suggested layer architecture for the Android malware detection using ML based on fingerprint project:

- Data Collection Layer
- Feature Extraction Layer
- Preprocessing Layer
- Machine Learning Layer
- Testing Layer
- Deployment Layer



Android Application Samples
Create API Call Graph of Each Sample
Does The Sample Generate Enough API calls?
No
Discard The Android Sample
Yes
Generate Graph Embeddings
Normalize Graph Embeddings
Rescale the Embedding Vector into Smaller Form
Cross Validation to Generate Training and Testing Dataset
Building Deep Neural Network and Hyper-parameter Tuning
Evaluation and Testing

PROPOSED METHODOLOGY

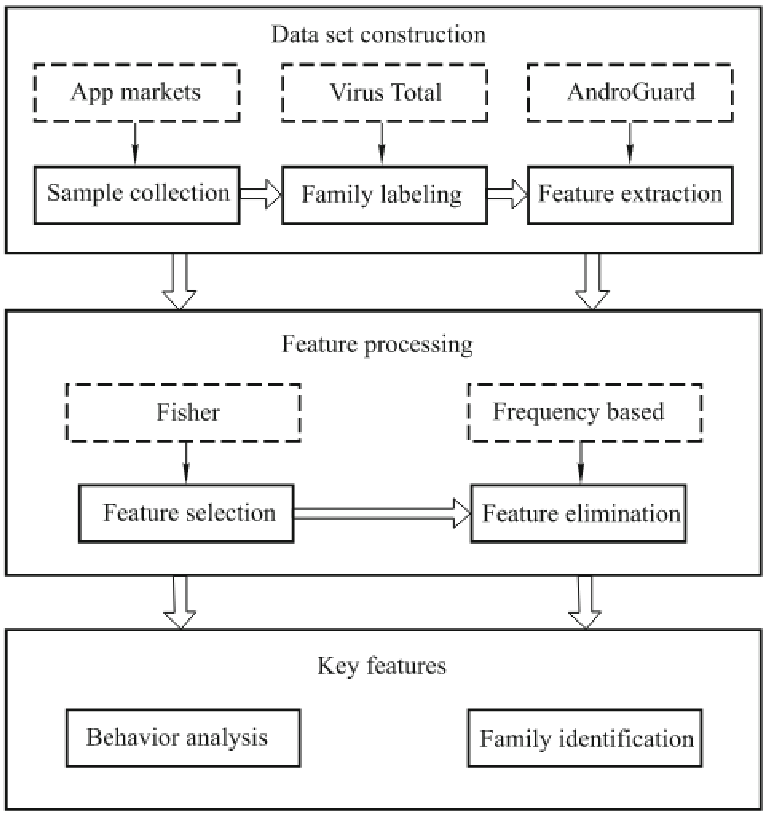
Fingerprinting involves generating the cryptographic hash values for the suspect binary based on its file content. The cryptographic hashing algorithms such as MD5, SHA1 or SHA256 are considered the de facto standard for generating file hashes for the malware specimens.

Malware HTTP request fingerprinting - But this is not for Mobile Malware through apps

Mobile Malware Detection based on Energy Fingerprints seems to be a deadend.

AndroDFA: Android Malware Classification Based on Resource Consumption as fingerprint methodology to classify Android malware into families. This approach was based on dynamic analysis and leveraged resource consumption metrics collected during app execution. Our experimental evaluation showed an accuracy of 82% with malware from the Drebin dataset and the Genymotion emulator

Fingerprinting Android malware families - We can use this method



Data set collection: Android samples are downloaded from different app markets and saved in a local database. The data preprocessing includes family labeling and feature extraction. VirusTotal is employed to scan the samples and label the family information. Then, Androguard is used to extract features and construct the static feature sets.

Feature processing: Feature selection and feature elimination are included in this stage. We use the Fisher feature selection algorithm to rank the features and select the features that have a stronger classification ability. Then, the frequency-based feature elimination algorithm is developed to identify the features that can describe the malware family characteristics.

Fingerprint construction: After feature elimination, the key features are selected. The family fingerprints are also constructed using the combined key features from different categories. Then, the features can be used to classify families and analyze behaviors.

FISHER FEATURE SELECTION:

- Feature selection is widely used for data preprocessing in machine learning and pattern recognition.
- It reduces data dimensions in order to reduce the calculation difficulty and consumption.
- Feature selection is used to find out M features to construct a feature subset from the original N features with the condition M < N.
- Therefore, the key issues are determining how to evaluate the selected feature subsets and

identifying the criteria that can be chosen to evaluate the features.

- According to the evaluation criteria, feature selection algorithms are of four categories: distance based, information-based, correlation-based and consistency based algorithms.
- Fisher discrimination theory can be used to reduce the feature dimension.
- Fisher feature selection is a distance-based technique.
- The best project direction indicates the maximum dispersion in different classes and the minimum dispersion in the same class.
- Thus, the feature at the best project direction has the best distinguishing ability

FREQUENCY BASED FEATURE ELIMINATION:

- The top ranked features from the feature selection in the sequence have a better classification ability.
- However, the selected features cannot reflect the characteristics of a particular class.
- If a selected feature is found more frequently in one class than in another class, then this feature can reflect the characteristics of the former class.
- In contrast, if the feature is equally distributed in two classes, it does not have the reflection ability. To determine the features that can reflect malware families, we propose the frequency-based feature elimination algorithm.

BEHAVIOR ANALYSIS OF MALWARE FAMILIES:

- Based on the outputs of the feature elimination, we can classify the existing malware classes into families based on their most prominent features.
- Ideally we can classify into 3 to 5 families.
- The parameters could ideally be permissions, suspicious API Calls and network information gathering leading to access of hardware components like camera, microphone, wifi, bluetooth, etc,.

ISSUES WITH THE PROPOSED MODEL

- Have to download apps and find malware existing in the apps through dynamic malware analysis manually.
- This has a huge time constraint since building this dataset is not easy.
- Setting up the sandbox environment for android malware analysis is expensive.
- The proposed model would take a lot of time to construct since more understanding of the implementation of such a sophisticated classification algorithm is required.
- Though this model is not exactly upon any modern fingerprints, here the prominent features of the malware are considered as fingerprints and the classification is carried out.

ISSUES WITH THE EXISTING MODEL

- Limited dataset: One of the main problems with the current system is the little amount of data that was utilized to train and test the ML model. Because the dataset is small, it could not accurately reflect the state of the malware landscape, which could result in malware being classified incorrectly.

- Overfitting: Overfitting occurs when a machine learning (ML) model gets overly focused on recognising the training data and is unable to generalize to new data. Overfitting can result in false positives and false negatives in the detection of Android malware, which decreases the model's efficacy.
- Costly to compute: ML-based malware detection can be expensive to compute, especially on mobile devices with constrained memory and processing capacity. This may lead to shorter battery life and slower detection times.
- Selection of Important Features: The project's fingerprinting technique's efficacy depends on the characteristics chosen. If crucial aspects are overlooked, the fingerprint could not correctly reflect the app, which could result in classification errors.
- Malware's dynamic nature: Makes it challenging to stay up to date with the most recent dangers. As a result, the ML model could be unable to identify fresh malware.

RESULT AND CONCLUSION

- Based on the issues, it is evident that this classification algorithm would take time to build but the process seems to be promising to give better results than any other way of fingerprinting the malwares.
- Future works could be to find a way or a dataset so as to carry out the initial process of creating a dataset simpler
- This way, this classification algorithm could be built easier.

DATASET:

- https://www.kaggle.com/datasets/shashwatwork/android-malware-dataset-for-machine-learning?resource=download

REFERENCES

1. Qiu, J., Zhang, J., Luo, W., Pan, L., Nepal, S., & Xiang, Y. (2020). A survey of android malware detection with deep neural models. *ACM Computing Surveys (CSUR)*, *53*(6), 1-36.
2. Zhang, N., Xue, J., Ma, Y., Zhang, R., Liang, T., & Tan, Y. A. (2021). Hybrid sequence‐based Android malware detection using natural language processing. *International Journal of Intelligent Systems*, *36*(10), 5770-5784.
3. Yerima, S. Y., Sezer, S., & Muttik, I. (2014, September). Android malware detection using parallel machine learning classifiers. In *2014 Eighth international conference on next generation mobile apps, services and technologies* (pp. 37-42). IEEE.
4. Karbab, E. B., Debbabi, M., & Mouheb, D. (2016). Fingerprinting Android packaging: Generating DNAs for malware detection. *Digital Investigation*, *18*, S33-S45.
5. Peiravian, N., & Zhu, X. (2013, November). Machine learning for android malware detection using permission and api calls. In *2013 IEEE 25th international conference on tools with artificial intelligence* (pp. 300-305). IEEE.
6. Sahs, J., & Khan, L. (2012, August). A machine learning approach to android malware detection. In *2012 European intelligence and security informatics conference* (pp. 141-147). IEEE.
7. Christiana, A., Gyunka, B., & Noah, A. (2020). Android malware detection through machine

learning techniques: A review.