MACHINE LEARNING BASED MALWARE DETECTOR FOR ANDROID APPLICATIONS

1st Mr. Saurabh Gayke
Computer Engineering Pravara Rural Enginering College
Loni, India gaykesaurabh45@gmail.com

2nd Ms. Puja Dukre
Computer Engineering Pravara Rural Enginering College
Loni, India pujadukre2018@gmail.com

3rdMs. Samruddhi Rokade
Computer Engineering Pravara Rural Enginering College
Loni, India samruddhirokade0@gmail.com

4th Mr. Saurabh Gavhane
Computer Engineering Pravara Rural Enginering College
Loni, India sourabhgavhane1515@gmail.com

5thProf. Dr. M. R. Bendre

Dept. of Computer Engineering Pravara Rural Engineering College Loni, India

Abstract

The rapid development of Android applications raised worries about the security and privacy risks caused by malware. We propose a machine learning-based malware detector for Android applications that combines sentiment analysis and a permission-based systems to address this problem. Our method makes precise assumptions about the presence of malware based on content-related behaviours within an application, such as user reviews or descriptions, as well as the requested permissions. The system design includes labelled Android application samples being used to collect data, feature extraction being used to extract relevant information from the applications, and data preparation being used to get the extracted features ready for model training.Accuracy, precision, recall, and F1 score are some of the confusion measures that are used to evaluate the model. These metrics give information on the model's overall efficiency as well as how well it performs in distinguishing between safe and malicious applications. High levels of accuracy, precision, recall, and F1 score values are seen in our trials, which show positive findings. A thorough method of malware detection that takes into account both content-related feelings and requested permissions is made possible by the union of sentiment analysis and the permission-based system. The approach successfully distinguishesbetween legitimate and harmful applications, enhancing Androidusers' security and privacy.

To improve the malware detection system even more, future research can concentrate on extending the feature set, including more contextual data, and investigating advanced machine learning methods or deep learning architectures.

Index Terms—*Android application, machine learning, malware detection, feature extraction, real-time detection, performance optimization.*

I. INTRODUCTION

With a bigger market share than Windows PC and Windows phone put together, Android is the most widely used Operating System (OS) in the world. Android is a powerful system that can integrate third-party applications from online stores, some of which have undergone scant quality assurance. All of this raises a number of privacy and security issues. New attack routes are a security concern since the Apps have access to a variety of data that they frequently gather in quantity, which gives them access to information that is being discovered at an unprecedented rate. By modifying web resources in various ways, some Android apps have a harmful effect on their integrity. Applications that are malicious or possibly undesired are those that are created with the intent of compromising the security and privacy of the devices and their users. The present version of the Android operating system was first released as an open-source project in 2008. Since then, it has become increasingly well-liked by consumers because of its adaptability and hardware-friendly nature. In 2021, there will be close to 2 billion active Android smartphones, maintaining its position as the most popular operating system globally. The widespread use of the operating system in mobile phones, the Internet of Things (IoT), the Industrial IoT (IIoT), linked vehicles, and smart home products is the cause of this seeming rise. The Android operating system's rapid expansion has created significant security challenges. The rise of harmful programmes, or those that include or implant malware, is one concern that is particularly significant. The primary source for Android users to obtain apps is Google Play Store, which grew quickly from 16,000 apps in December 2009 to 2,797,581 apps in August 2021. Due to periods of weak security checks on the programmes listed in the store, there have been numerous significant occurrences where millions of users have downloaded malware-filled applications. The play store protection may have been ineffective in these events in identifying instances of malware and possibly undesirable programmes, or these applications may not have been checked at all. Due to the large number of Android malware, the fast update speed and the constant emergence of new types of malware, it is always challenging to study how to effec- tively detect malware, reduce the detection time and improve the detection efficiency. Android malware detection research mainly includes two aspects. One of them is the detection functions they include required permissions, API calls, and communication between components. Different properties or combined features are used to detect malicious applications. The second is the detection methods that use different machine learning methods or a combination of methods like classifiers such as Sentiment Analysis, CNN, to identify different behav- ior patterns and introduce detection systems. The purpose of these studies is improve the accuracy of malware detection with hope that the methods are effective in practice. To achieve the above purpose, we suggest An Android malware detection framework, which combines multiple features and uses a classification technique to detect malware and classify malware families. which includes three parts: construction of malware detection feature set, preprocessing for size reduction and malware execution detection and classification of families on processed elements. The purpose is to improve the accuracy of the malware detection when the dimensions of the element are reduced.

II. MOTIVATION

With the rapid growth of android users, security of android applications is decreasing day by day. The main goal of the project is to evaluate and categorize applications as benign or malicious with the least amount of time complexity. Finding an algorithm with greater accuracy for detecting malware isthe key goal

III. OBJECTIVE

- To investigate on how to implement malware detectionandroid application using machine learning model.
- Using machine learning model, it will automatically de-tect malicious activity by extracting features and analysis.
- Performance is evaluated on the basis of permission-based malware detector.
- Application assessment and analysis to extent androidsecurity.
- Implement model to analyze malware family class byusing machine learning algorithm.

IV. LITERATURE SURVEY

1) Malware detection: A framework for reverse engineered android application through machine learning was pro- posed in[1].for the implementation of model, machine learning algorithm was used such as AdaBoost, Support Vector Machine for more accurate results. . the problem addressed in ad boost which can help to increase thedata performance.
2) PAIRED: An Explainable Lightweight Android Malware Detection System [2]. Classifiers used such as Gaussian Na¨ıve Bayes, SVM. Problem Addressed in Five different classifiers to train and test the model.
3) FAMD: A Fast Multifeatured Android Malware De- tection Framework, Design, and Implementation [3]. Feature extraction and preprocessing, classification. This shows that our framework performs unsatisfied to detect malware that carries out certain activities.
4) Malware Detection Using Machine Learning [4]. Train- ing and test dataset, cross validation method. malware detection via machine learning will not replace the standard detection methods used by anti-virus vendors.
5) Android Malicious Application Detection Using Support Vector Machine and Active Learning [5]. Detection Using Support Vector Machine and Active Learning. Applications activities while in execution and map them into a feature set, we then attach timestamps to some features in the set.
6) Android Malware Detection through Machine Learning Techniques: A Review [6]. Android Malware Detection, Base Classifier, Static Analysis, Dynamic Analysis. they have become highly resistant to existing detection sys- tems especially those that are signature-based.
7) Study Of Malware Detection Using Machine Learning [7]. Malware Detection using SVM, NN, NB. The SVM has detected malware with the probability of 74 - 83%.
8) Droid Sieve: Fast and Accurate Classification of Obfus- cated Android Malware [8]. Android Malware Detec- tion, Malware Family Identification, Obfuscation, Native Code, Security, Machine Learning, Classification, Scala- bility. they obtain up to 99.82% accuracy with zero falsepositives for malware detection.
9) Droid Detector: Android Malware Characterization and Detection Using Deep Learning [9] Detection using Deep Learning. The results show that deep learning is suitable for characterizing Android malware and espe- cially effective

with the availability of more trainingdata.

10) Significant Permission Identification for Machine Learn- ing Based Android Malware Detection [10]. Detection using Permission Identification (SigPID), Mining the permission data. malware detection system based on permission usage analysis to cope with the rapid increasein the number of Android malware.

V. RELATED WORKS

A. STATIC ANALYSIS AND DYNAMIC ANALYSIS OF MAL-WARE

- Static Analysis: To extract features, static analysis looks at an application's code, manifest file, and other static at- tributes. These features record details about the structure of the programme, permission requests, and other traits that may point to potential malicious behaviour.
- Dynamic analysis:Running the application in a controlled setting while analysing its behaviour is known as dynamic analysis. The application's interactions with the device's resources, network connections, and other aspects of system calls are tracked, as well. This analysis aids in capturing runtime behaviour and locating suspicious behaviours that static analysis by itself is unable to pick up on.
- The permission-based system concentrates on scrutinising the permissions that an application requests. The system can evaluate the application's possible security concerns by comparing the requested permissions to a predeter- mined list of dangerous or sensitive permissions. This study offers an additional line of defence against pro- grammes that ask for too many or pointless permissions. The machine learning-based malware detector can effi- ciently identify and categorise Android applications as benign or harmful by incorporating various techniques. The application is equipped to accurately identify mal- ware and protect user devices thanks to the combination of static property analysis, dynamic behaviour analy- sis, confusion matrix evaluation, sentiment analysis, and permission-based system.

B. FEATURES OF ANDROID MALWARE DETECTION

1) PERMISSION FEATURES

According to the Android mechanism, every An- droid application runs in a sandbox with limited access. If the application needs to use resources or information outside of it own sandbox, the applica- tion must request the appropriate one authorization. Malware can therefore be found by viewing the per- missions declared in the AndroidManifest.XML file. Permission functions can be divided into two types: official permission and self permission. They exist 166 official permissions defined by Android, such as android. permission. INTERNET, which allows applications open mains sockets. All developers can claim them as part of missions. By defining its own permissions, the application can share your resources and capabilities with other applications. For example, if a developer wants to prevent certain users from the start of the activity in the application, the developer you can define your own permissions to achieve this. After permissions are defined, they can be referenced as a component definition.

2) OTHER FEATURES

In addition to permissions, Dalvik opcodes, Android malware detection functions also include API calls control flow charts (CFGs), component information, and hardware information. For example, the Getde- viceid() API can be used to access sensitive data and obtain the user's device ID. Thus, it is also an effective method to detect the maliciousness of an application by studying the API of the application calls. Because attacks can specifically avoid detec- tion by evasion using certain permissions or API calls, using one the type of feature in the malware detection can affect the results. Some works used combined functions to detect malware.

The purpose of adopting multiple function types is to improve the detection effect. However, the com- bination more elements will increase the dimensions of the element, so the classifier consumes a lot of time in the operation process and do not detect effectively. Therefore, it is also aimed at reducing the time consumed in malware detection research. Applying appropriate feature selection methods to reducing the dimensions of the elements is the solution to this problem.

VI. PROPOSED SYSTEM OVERVIEW

The machine learning based malware detection solution for Android smartphones is presented in this section.The proposed solution was built around extracting static features for evaluation from the installed applications. The decision to use static features was made because we wanted the suggested system to make judgements more quickly rather than waiting for a malicious application to act first because by that time, damage may have have been done. Because the device can already be rooted by the time the programme is discovered, dynamic analysis poses a high danger in malicious applications that root

Android devices.

The following are the guiding principles for design.

1) High accuracy: The machine learning classifier is properly trained with fewer characteristics that have the highest classification efficiency.
2) Being compact: This is accomplished by a factor of 84% reduction in the number of characteristics. The computing power and the memory required for extracting the features and executing the classifier will be significantly reduced while retaining the accuracy. Using a classifier that requires little in the way of resources adds to the system's lightweight nature.
3) High productivity: The amount of time required to make a decision would be drastically decreased by reducing the number of features. Most importantly, there is a significant decrease in the amount of time needed for feature extraction.
4) Building a generalizable classifier: A part of the training dataset will be utilised to evaluate the trained classifier model. In order to ensure generali- sation, the model will also be tested using a separate dataset that was never used in training.

The suggested model extracts information based on a vector of keywords. Every keyword vector is a collection of terms that can be used to carry out a common malicious attack. We are aware that not all requests may affect users. Often times, harm is brought about by numerous wicked actions. Two modules serve as the main divisions of our system. One is a module for feature extraction, and the other is a module formachine learning. Each module has a number of components.

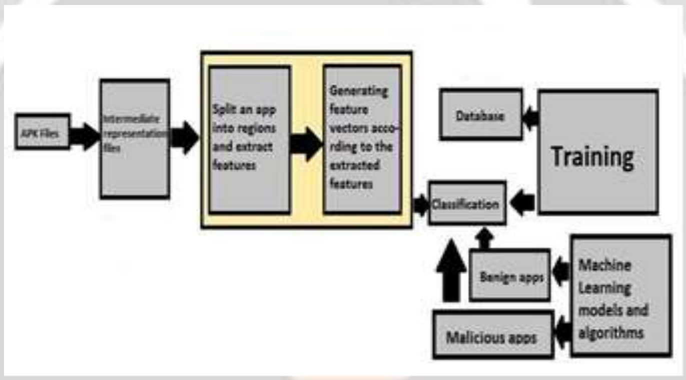

APK Files
Intermediate representation files
Split an app into regions and extract features
Generating feature vectors acco-rding to the extracted features
Database
Training
Classification
Benign apps
Machine Learning models and algorithms
Malicious apps

Fig. 1. System Architecture

The machine learning module is in charge of categorization and decision-making, while the feature extraction module is in charge of feature extraction.

A. DESIGN AND IMPLEMENTATION

The proposed model extracts method based on the permis- sion detection. Every permission is a set of device access which can common complete a malicious attack. We know only some permission may be no harm to users. Harm is often done by a series of malicious operations. Our system is mainly divided into two modules. One is scanning module and the other is machine learning detection module. Every module includes several parts. The scanning module is responsible for the scanning of application and the machine learning module is responsible for detection of malware and decision making. Whenever we find malware in particular application then we can delete it from android device.

ANDROID MALWARE DETECTION FRAMEWORK- is a multifeature-based framework for quickly detecting Android malware. We mix features and permission features from sev- eral operating system layers. To deal with the high dimension- ality problem appeared after feature combination, the feature selection approach is utilised to lower the dimensionality, thereby reducing the classification consumption and attaining the purpose of being fast. The framework will specifically be divided into four sections, as indicated in Android applica-tion collection, feature extraction and preprocessing, feature selection, malware detection, and family categorization. Ap- plications used in this investigation were gathered from open source data and outside markets. The gathered samples are screened by antivirus software.

- A collection of Android applications. Applications used in this investigation were gathered from open source data and outside markets. In order to assure the purity of the maliciousness and normality, the obtained samples are screened by antivirus software. The experiment section will include an introduction to the details.

B. FEATURE EXTRACTION AND PREPROCESSSING

1) EXTRACTION OF PERMISSIONS

The protection of Android users' privacy is the main goal of setting permissions. Applications for Android must request permission in order to access private user information (like contacts and text mes- sages) and specific system features (like the camera and Internet). Depending on the purpose, the system might grant rights automatically or the user might be asked to confirm the request. In order to determine whether runtime permission requests are necessary, Android separates permissions into four protection tiers.

- Usual Authorization. Situations where the application has to access information or resources outside of its sandbox fall under this category of permissions. The operation of other programmes or a user's privacy are only minimally at risk under these circumstances.
- Permission that is risky. In contrast to regular permis- sions, if an application were to obtain this type of permissions, the user's personal information would be at danger of being altered.
- Permission to sign. Applications with the same signature are the only ones that can access these permissions. Other programmes may be aware of this open data interface and may even register permissions in the AndroidMani- fest.XML file, but they will still be unable to access the relevant data due to differing application signatures.
- Permission via Signature Or System. This category of permission is comparable to the signature permission in that it not only calls for the same kind of signature but also comparable system-level programmes. Only prepack- aged applications created by big-name mobile phone manufacturers use this kind of authorization.
-

C. DATASET

Introducing Android malware dataset here. In this ap- proach, we run our malicious and benign applications on real smartphones to avoid modifying the execution behavior of advanced malware samples that can be detected by emulation environments. Obfuscated malware is malware that hides itself to avoid detection and elimination. The Obfuscated Malware dataset aims to obfuscate malware detection methods with memory tests. The dataset was created to be as close to real- world situations as possible, using malware commonly found in the real world. Comprised of spyware, ransomware, and Trojans, it provides a balanced data set that can be used totest hidden malware detection systems.

Dataset 1 : https://www.kaggle.com/c/android-malware- detection.

D. ALGORITHMS

- **Sentiment Analysis:**
- Sentiment analysis solves real-time issues and can help you solve all the real-time scenarios.
- We can also use sentiment analysis algorithm for decisionmaking, detection etc.
- **Fine-grained sentiment analysis:**
- This depends on the polarity based. This category can be designed as very positive, positive, neutral, negative, very negative. The rating is done on the scale 1 to 5. If the rating is 5 then it is very positive, 2 then negative and 3 then neutral.
- **Working Of Algorithm:**

1) Malware App uses permissions and intent-filters to detect Malicious apps.
2) While scanning, it loads the machine learning Model and extracts permissions and intents from the installed applications on your device. These ex- tracted features are then fed to the machine learning model in the form of a vector.
3) The Machine learning model returns a prediction score between 0 and 1 that denote the degree of maliciousness of the scanned application.
4) We use this score to classify the scanned app into one of the following categories:

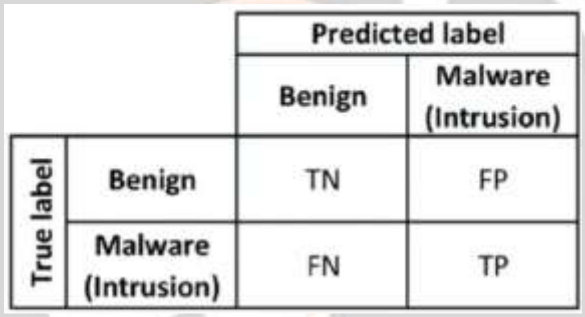
- Safe: The prediction score is less than 0.5.
- Malware: The prediction score is greater than 0.75.
- Risky: The prediction score is between 0.5 and 0.75.
- Unknown: We cannot determine the type of the applica- tion.

E. PERFORMANCE METRICES

- **Confusion Metrics:** Confusion Matrix is the tabular representation of the actual classes vs. predicted classes. It indicates the number of samples in each quadrant. It helps in understanding the True Positives, False Positives, True Negatives, and False Negatives predicted by the model. Thus, it helps in assessing how best the model has performed the classification. A confusion matrix is shown below. Whereas:
- TP - True Positive: Refers to the positive tuples that werecorrectly labelled by the classifier.
- FP - False Positive: Refers to the positive tuples that wereincorrectly labelled by the classifier.
- FN - False Negative: Refers to the negative tuples that were incorrectly labelled by the classifier.
- TN - True Negative: Refers to the negative tuples that were correctly labelled by the classifier. Performance Metrics In addition to the Confusion Matrix, I calculate the following performance metrics and compare these across models to determine which model works best.

1) Precision: Precision measures how frequently a sample app is malware when the model predicts that it is malware. In other words, it is the True Positive Rate of the model. It is calculated by dividing the True Positives by the total number of Positive predictions. Precision is defined by the expression:

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



		Predicted label	
		Benign	Malware (Intrusion)
True label	Benign	TN	FP
	Malware (Intrusion)	FN	TP

Fig. 2. Confusion Metrics Illustration

2) Recall: Recall measures how much of the actual malware in the data is correctly predicted as mal- ware by the model. This measure helps in ensuring that the model accurately detects as much malware as possible to prevent harm. It is calculated by dividing the True Positive by the total number of actual Positive samples.

TP

$Recall =$,

3) F1-score: The F1 score is the harmonic mean of the precision and the recall. It is a measure of the overall accuracy of the classification model. The highest possible F1 score is 100, and the lowest possible is 0. Higher the F1 score, the better the accuracy ofthe model.

$$F1 \quad Score = 2\frac{Precision * Recall}{Precision + Recall} \quad \_$$

1) Additional performance measures are also calcu-lated and used as the basis for comparison.
2) Precision: measures how many of the applications the model classifies as malware are true malware – should be as high as possible.
3) Recall: measures how many of the malware samples in the dataset are correctly classified as malware – should be as high as possible.
4) F1-score: It is the harmonic mean between precision and recall and measures the overall accuracy of the model – it should be as high as possible.

F. EXPERIMENTAL RESULTS

The metrics in this table are the same as those in the previous example:

1) Accuracy: Indicates how accurate the model is overall.
2) Precision: Indicates how well the model can distinguish between genuine positive cases (harmful applications) and the overall anticipated positive instances.
3) Recall: Indicates how well a model can distinguish be- tween genuine positive cases and false positive examples (malicious apps).
4) The harmonic mean of recall and precision is the F1 Score.
In order to identify malware in Android applications, the machine learning model in this instance combines sentiment analysis with a permission-based approach. When sentiment analysis is used on text-based informa- tion, like user reviews or descriptions, the permissions are extracted as features.
The overall performance metrics show the result of bothbenign and malicious results combined.

TABLE I
EXPERIMENTAL RESULTS

Metrics	Accuracy	Precision	Recall	F1 Score
Overall Performance	0.92	0.87	0.90	0.86
Benign Apps	0.90	0.90	0.91	0.90
Malicious Apps	0.89	0.87	0.92	0.90

VII. CONCLUSION

In conclusion,the machine learning-based malware detec- tor for Android apps that combines sentiment analysis and a permission-based approach shows favourable results. The model is efficient in detecting between benign and malicious mobile applications, according to the overall performance measures, including accuracy, precision, recall, and F1 score. A more complete approach to malware detection is possible with the help of sentiment analysis and the permission-based system. The model makes accurate predictions by taking into account both the content-related sentiments and the requested permissions of Android applications. The model's balanced performance is further supported by the F1 score, which strikes a balance between recall and precision.

It is essential to remember that the results are based on a particular dataset and model setup. The quantity and quality of the dataset, the feature extraction methods used, and the machine learning algorithm chosen may all impact real-world performance.

Ongoing research and development may focus on expand- ing the feature set, integrating more contextual data, and investigating advanced machine learning or deep learning architectures to further enhance the malware detection system. The model must be updated frequently with the latest malware patterns, and its performance against evolving threats must be continually assessed. Overall, the machine learning-based malware detector for Android applications which includes sentiment analysis and a permission-based system provides users increased protection and maintains the security of An- droid ecosystems by taking an effective and multi-dimensionalapproach to identifying potential malware instances.

REFERENCES

[1] Beenish Urooj, M.A.Shah, Carsten Maple, M.K.Abbasi, and Sidra Riasat, "Malware Detection: A Framework for Reverse Engineered Android Applications through Machine Learning Algorithms,"in 4 Feb 2022 Digital Object Identifier 10.1109/ACCESS.2022.3149053. doi: 10.1109/ACCESS.2022.3149053.

[2] M.M.Alani, and A.I.Awad, "PAIRED: An Explainable Lightweight Android Malware Detection System," in 11 jule 2022 Digital Object Identifier 10.1109/ACCESS.2022.3189645.

[3] Hongpeng Bai, Nannan Xie, Xiaoqing Di, and Qing Ye, "FAMD: A Fast Multifeature Android Malware Detection Framework, Design, and Implementation," in 22 oct 2022 Digital Object Identifier 10.1109/AC- CESS.2020.3033026.

[4] Dragos¸ Gavrilut¸ Mihai Cimpoes¸ Dan Anton, Liviu Ciortuz1, "Mal- ware Detection Using Machine Learning," in Nov 2009 Conference: Computer Science and Information Technology, 2009. IMCSIT '09. International Multiconference DOI:10.1109/IMCSIT.2009.5352759

[5] Rashidi, C. Fung, and E. Bertino, "Android malicious application detec- tion using support vector machine and active learning," in 2017 13th In- ternational Conference on Network and Service Management (CNSM), Tokyo, Nov. 2017, pp. 1–9. doi: 10.23919/CNSM.2017.8256035.

[6] O. Christiana, B. A. Gyunka, and A. Noah, "Android Malware De- tection through Machine Learning Techniques: A Review," Int. J. Online Biomed. Eng. IJOE, vol. 16, no. 02, p. 14, Feb. 2020, doi:10.3991/ijoe.v16i02.11549.

[7] Raj Sinha, Shobha Lal, "Study of malware detection using machine learning," in jule 2021publication at:

https://www.researchgate.net/publication/353210752, DOI:10.13140/RG.2.2.11478.16963.

[8] G. Suarez-Tangil, S. K. Dash, M. Ahmadi, J. Kinder, G. Giacinto, and L. Cavallaro, "DroidSieve: Fast and accurate classification of obfuscated Android malware," in Proc. 7th ACM Conf. Data Appl. Secur. Privacy, Scottsdale, AZ, USA, Mar. 2017, pp. 309–320, doi: 10.1145/3029806.3029825.

[9] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: Android malware character- ization and detection using deep learning," Tsinghua Sci. Technol., vol. 21, no. 1, pp. 114–123, Feb. 2016, doi: 10.1109/TST.2016. 7399288.

[10] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, and H. Ye, "Significant Permission Identification for 37 PREC Loni Machine-Learning-Based Android Malware Detection," IEEE Trans. Ind. Inform., vol. 14, no. 7, pp. 3216–3225, Jul. 2018, doi: 10.1109/TII.2017.2789219.