# SMARTPHONES LEVERAGING MACHINE LEARNING FOR ANDROID MALWARE DETECTION

[1] Jahnavi Devi Koppula, B.Tech, M.Tech, Assistant Professor, Department of CSE, Eluru College of Engineering And Technology, Duggirala, Andhra Pradesh-534004.
[2] M.H.V.S.S.Sudheesh, B.Tech, Department of CSE, Eluru College of Engineering And Technology, Duggirala, Andhra Pradesh-534004.
[3] A.R.S.Rajeswari, B.Tech, Department of CSE, Eluru College of Engineering And Technology, Duggirala, Andhra Pradesh-534004.
[4] Yesurathnam Kommera, B.Tech, Department of CSE, Eluru College of Engineering And Technology, Duggirala, Andhra Pradesh-534004.

**Abstract:** Android malware growth has been increasing dramatically along with increasing the diversity and complicity of their developing techniques. Machine learning techniques are the current methods to model patterns of static features and dynamic behaviors of Android malware. Whereas the accuracy rates of the machine learning classifiers increase with increasing the quality of the features, we relate between the apps' features and the features that are needed to deliver its category's functionality. Differently, our classification approach defines legitimate static features for benign apps under a specific category as opposite to identifying malicious patterns. We utilize the features of the top rated apps in a specific category to train a malware detection classifier for that given category. Android apps stores organize apps into different categories, for instance, 26 categories on Google Play Store. Each category has its distinct functionalities which means the apps under a specific category are similar in their static and dynamic features. In general, benign apps under a certain category tend to share a common set of features. On the contrary, malicious apps tend to request abnormal features, less or more than what is common for the category that they belong to. This study proposes category- based machine learning classifiers to enhance the performance of classification models at detecting malicious apps under a certain category. The intensive machine learning experiments proved that category-based classifiers report a remarkable higher average performance compared to non-category based..

## 1. INTRODUCTION

According to International Data Corporation (IDC), Android OS is the most popular smart phone platform with 82.2% of the market share of smart phones, while 13.9% for IOS apple in the second quarter of 2015. Statistically speaking, it is also the first targeted platform by malware authors seeking to take the control over millions of Android smart phones over the world. Due to the popularity of Android's smart phones, its apps' security is a serious issue concerning 80% of smart phones users. Android is an open source development environment that offers a rich SDK that enables developers to deploy their own apps and distribute them through Android apps centers. Android's popularity is a result of being an open source, third-party distribution centers, a rich SDK, and the popularity of Java as a programming language. Importantly, due to this open environment, malware authors can develop malicious apps that abuse the features that the platform offers or pack a legitimate app with a piece of malicious code; besides, exploiting vulnerabilities in the platform, hardware, or other installed apps to lunch malicious behaviors. Mainly, malware authors seek access confidential data of a device's user, monetary benefits via premium SMS, or joining the device to a botnet. Even legitimate apps introduce the risk of privacy-invading; Mcafee reported in Q1 2014 that 82% of Android apps track user's and 80% gather location data. Research studies in the Android malware detection field work in three approaches static, dynamic or hybrid. In static analysis, malware is disassembled into a source code from where specific features are extracted. In dynamic analysis, malware is monitored at run-time in a virtual environment. In the both approaches, machine learning algorithms have been used to build classification models by training classifiers with datasets of malware features that collected from static or dynamic analysis. The learned classification models are then used to detect malicious apps and classify them into their families. In this study, we approach the problem differently by utilizing the features of benign apps for malware detection. We relate between the features that the app requests and the common features for its category. Android apps stores organize apps into different categories; for example, Google play store organizes apps in 26 categories such as: "Health & Fitness", "News & Magazine", "Books & References", "Music & Audio", etc. Each category has its distinct functionalities which means the apps under a certain category share similar features. One group of these features are the permissions; permissions are the privileges that enable apps to access the system's resources to perform their functions. Each built-in permission is responsible for providing the capabilities to execute a particular process. Apps belong to a specific category deliver the same functionality as a result they require a common combination of permissions. For instance, apps under "Communication" category commonly request READ CONTACTS but it is uncommon if it is requested by apps under "News & Magazines". In general, benign apps under a certain

category tend to have a common set of features: permissions, intents filters, hardware components, broadcast receivers, APIs, etc. On the contrary, malicious apps tend to request abnormal features, less or more than what is common for the category that they belong to. Repeatedly from that point of view, this study proposes category-based machine learning classifiers to enhance the performance of classification models at detecting malicious apps under a certain category.

## 2. LITERATURE SURVEY

2.1 Dong-Jie Wu,Ching-Hao Mao,Te-En Wei,Hahn-Ming Lee,Kuo-Ping Wu in 2012 were proposed the Recently, the threat of Android malware is spreading rapidly, especially those repackaged Android malware. Although understanding Android malware using dynamic analysis can provide a comprehensive view, it is still subjected to high cost in environment deployment and manual efforts in investigation. In this study, we propose a static feature-based mechanism to provide a static analyst paradigm for detecting the Android malware. The mechanism considers the static information including permissions, deployment of components, Intent messages passing and API calls for characterizing the Android applications behavior.

2.2 Elmouatez Billah Karbab,Mourad Debbabi,Saed Alrabaee,Djedjiga Mouheb in 2016 were proposed the The astonishing spread of Android OS, not only in smart phones and tablets but also in IoT devices, makes this operating system a very tempting target for malware threats. Indeed, the latter are expanding at a similar rate. In this respect, malware fingerprints, whether based on cryptographic or fuzzyhashing, are the first defense line against such attacks. Fuzzyhashing fingerprints are suitable for capturing malware static features. Moreover, they are more resilient to small changes in the actual static content of malware files.

2.3 Hossein Fereidooni,Mauro Conti,Danfeng Yao,Alessandro Sperduti in 2016 were proposed the number of malware applications targeting the Android operating system has significantly increased in recent years. Malicious applications pose a significant threat to Android platform security. We propose ANASTASIA, a system to detect malicious Android applications through statically analyzing applications' behaviors.

2.4 Cheng,Xunxun Chen,Yongzheng Zhang,Shuhao Li,Yafei Sang in 2017 were proposed with the widespread use of smartphones, more and more malicious attacks happen with information leakage from apps installed on users' devices. The adversary always uses a malware as the client to take remote control of smart phones, and leverages the vulnerability of operation systems to send back the collected information without users' permissions. All the information has to be transferred by network traffic.

2.5 Rubata Riasat,Muntaha Sakeena,Abdul Hannan Sadiq,Yong-Ji Wang in 2018 were proposed Android mobile phone is one of the most usable smartphone today which makes Android the most anticipated operating system in the market. However, the increase in malware threats with the growth in Android market cannot be neglected. Android is an open source platform which gives full leverage to developers on one side, but also provides an open door to malware.

2.6 Ajit Kumar,Vinti Agarwal,Shishir K. Shandilya,Andrii Shalaginov,Saket Upadhyay,Bhawna Yadav in 2019 were proposed Android malware has become the topmost threat for ubiquitous and useful Android eco-system. Multiple solutions leveraging big data and 4 machine learning capabilities to detect android malware are being constantly developed. Too often, many of these solutions are either limited to the research output or remain isolated and unable to reach to end-users or malware researchers.

2.7 Ikram Ul Haq,Tamim Ahmed Khan,Adnan Akhunzada in 2021 IEEE Access were proposed the dramatic increase in Android-based smart devices has brought technological revolution to improve the overall quality of life and thus making it worth a billion-dollar market. Despite the huge hype surrounding Android market, the prevalent and potentially sophisticated malicious mobile malware has become a serious threat to the popular Android platform and an ideal target for varied cyber adversaries.

2.9 Zakaria Sawadogo,Jean-Marie Dembele, Gervais Mendy, Samuel Ouya in 2023 were proposed the prevalence of cyber security threats, such as the Android Zero-day vulnerability, is becoming increasingly worrisome. With the widespread use of Android-powered mobile devices, attackers are leveraging zero-day vulnerabilities to infect Android software at an alarming rate. Detecting zero-day vulnerabilities in Android applications is particularly challenging due to their unpatched and undiscovered nature, resulting in a lack of reference points for identification.

## 3. EXISTING SYSTEM

One existing work has used data processing and options generated from windows workable API calls. They

achieved sensible leads to a really giant scale dataset with concerning 35,000 transportable workable files. Another activity foot printing methodology additionally provides a dynamic approach to discover self-propagating malware. All these existing ways have basically advanced the mechanical man malware detection; however, the misuse detection isn't reconciling to the novel mechanical man malware and continually needs frequent change of the signatures. Here lies the analysis gap. In exiting system they implemented the classifiers like naive bayes and decision tree which gives the poor accuracy.

DISADVANTAGES:

• Misuse detection isn't reconciling.
• Accuracy is less.
• Mechanical man malware detection.

**4. PROPOSED SYSTEM**

In the proposed system we implement a better feature extraction technique and then we apply the genetic algorithm for feature extraction and then we use two machine learning model called as SVM and multi perception classifier for classification of android malware detection which gives the better accuracy ratio when compare to existing system.

ADVANTAGES:

- Easy to identify and block malware.
- Accuracy is more.
- Dynamic feature extraction using genetic algorithm.

5. UML DIAGRAMS

1. DATA FLOW DIAGRAM

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

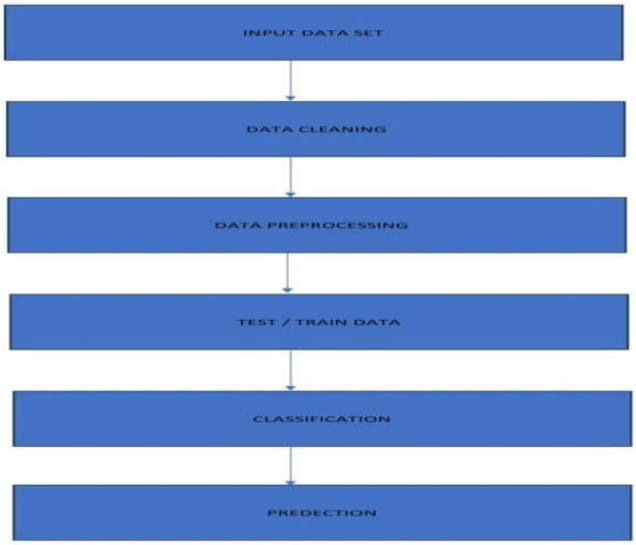


Fig 5.1 shows the data flow diagram of the project

**2. USECASE DIAGRAM:**

Use Case Diagrams are used to depict the functionality of a system or a part of a system. They are widely used to illustrate the functional requirements of the system and its interaction with external agents (actors). In brief, the purposes of use case diagrams can be said to be as follows

• Used to gather the requirements of a system.
• Used to get an outside view of a system.
• Identify the external and internal factors influencing the system.

Use case diagrams commonly contains

• Use cases
• Actors
• Dependency, generalization and association relationships.

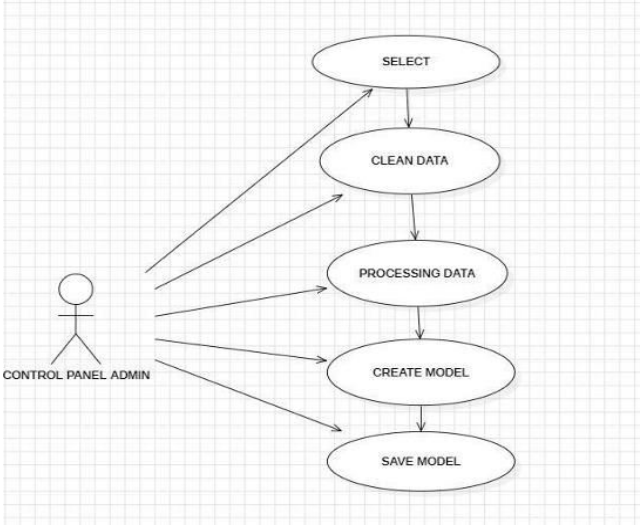


Fig 5.2 Shows the Use case Diagram for Admin Panel

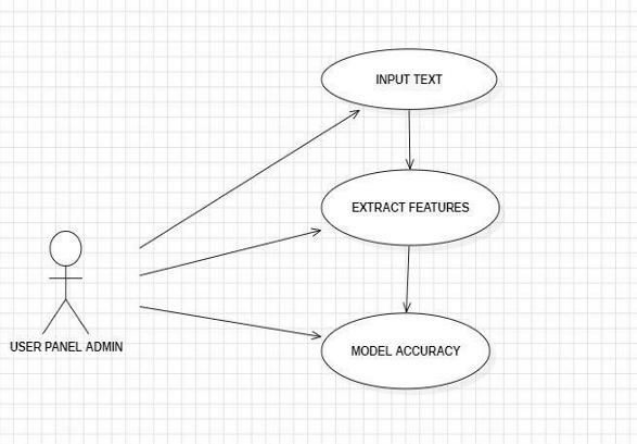


Fig 5.3 Shows the Use case Diagram for User Panel

## 6. RESULTS

The output screen of the webpage initially shows the APK classification process. • This includes uploading the chosen app file, predicting its class, and displaying model accuracy. • Meta data containing app name, target SDK version, and file size, is also provided.


Fig 6.1 shows the APK classification process

The below screen shows the webpage hosted on 127.0.0.1:5000 showcases an APK application analysis. • Users can upload files and apply a support vector machine algorithm for prediction. • Upon clicking predict the output reveals the predicted class (e.g., Malware) with a model accuracy (e.g.,89%). • The meta data includes the app name (e.g., Hanuman Temple), target SDK version (e.g.,8), and file size (e.g.,1.58 MB).

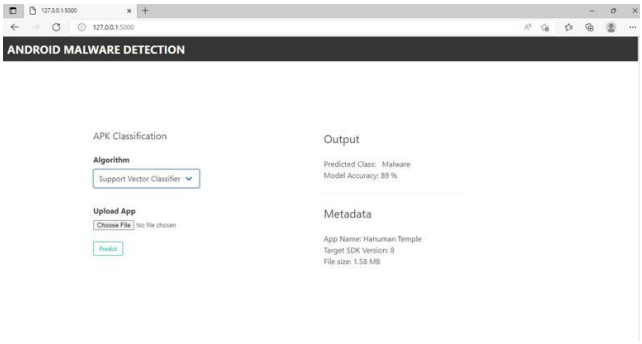


Fig6.2: shows the webpage hosted on 127.0.0.1:5000

The below screen shows the webpage hosted on 127.0.0.1:5000 showcases an APK application analysis. • Users can upload files and apply a neural network algorithm for prediction. • Upon clicking predict the output reveals the predicted class (e.g., Malware) with a model accuracy (e.g.,92.26%). • The meta data includes the app name (e.g., Hanuman Temple), target SDK version (e.g.,8), and file size (e.g.,1.58 MB).

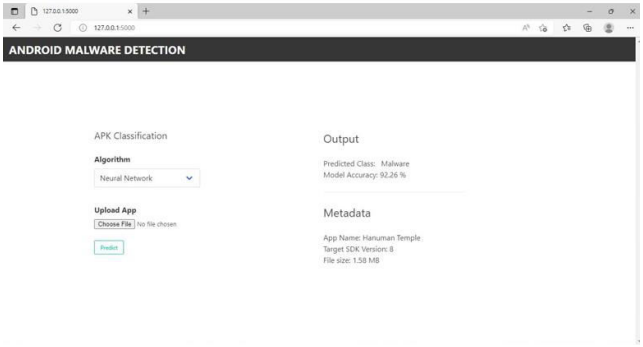


Fig 6.3 shows the webpage hosted on 127.0.0.1:5000

## 7. CONCLUSION

In our study, we propose category-based machine learning classifiers to improve the performance of the classification models. In static analysis of Android malware, machine learning algorithms have been used to train classifiers with features of malicious apps to build models that capable of detecting malicious patterns. Differently, our classification approach defines legitimate static features for benign apps as opposite to identifying malicious patterns. We utilize the features of the top rated apps in a specific category to define a profile of the common sets of features for that category. In other words, to detect whether or not the app posses the characteristics of benign, we relate between the app's features and the features that are needed to deliver the category's functionality that the app belongs to. Android stores organize apps into different categories; 26 categories on the Google Play Store, for example. In each category, the apps deliver a similar functionality as a result the they tend to request a common set of features like same permissions, APIs, hardware components, broadcast receivers, intents filters, etc. On the contrary, malicious apps tend to have abnormal features, less or more than what is common for the category that they belong to. Malicious apps can be identified by comparing between the features they request to the features that are requested by benign apps in the same category. For example, malicious apps, compared to the benign apps in the same category, tend to request over-privileged permissions, listen to specific events that broadcast by the Android system, or

using unneeded APIs for the app's category functionality that can be used to lunch malicious behaviors. We compare the performance of category-based and non-category based classi- fiers at detecting malicious apps under a specific category. To achieve this comparison, we built three datasets of apps' features: apps from all categories (allCateg), apps from "Music & Audio" category (musicCateg), and apps from "Personalization" category (personaCateg). For each dataset, we trained three machine learning classifiers: Support Vector Machines, Random Forests, and Ada Boost; the classifiers were trained with three group of features: permissions, broadcast receivers, and APIs. For testing, apps from "Music & Audio" category were tested with (musicCateg) and (allCateg) classifiers, respectively; and apps from "Personalization" category were tested with personaCateg and allCateg classifiers, as well. The category-based classifiers reported a higher performance compared to the noncategory based at detecting malicious and benign in the two categories of our study: "Music & Audio" and "Personalization".

**FUTURE SCOPE**

Our future work will consider three aspects. First, including other static features such as: functions calls in building the classification models to get a better understanding of the processes that apps may lunch in a way to increase the detection accuracy of the classifiers. Second, implementing the proposed solution on a large-scale level by building profile models for other categories and sub categories. Third, testing the feasibility of integrating our solution with dynamic detection techniques by profiling dynamic features for each category; dynamic features like system calls, network connections, resources' usage, and etc.

**8. REFERENCES**

[1]Dong-Jie Wu;Ching-Hao Mao,Te-En Wei;Hahn-Ming Lee,Kuo-Ping Wu."DroidMat: Android Malware Detection through Manifest and API Calls Tracing" in 2012 Seventh Asia Joint Conference on Information Security.

[2]Elmouatez Billah Karbab,Mourad Debbabi,Saed Alrabaee,Djedijga Mouheb."DySign: dynamic fingerprinting for the automatic detection of android malware"in 2016 11th International Conference on Malicious and Unwanted Software (MALWARE).

[3]Hossein Fereidooni,Mauro Conti,Danfeng Yao,Alessandro Sperduti."ANASTASIA: android malware detection using Static analysis of Applications"in 2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS).

[4]Zhenyu Cheng,Xunxun Chen,Yongzheng Zhang,Shuhao Li,Yafei Sang."Detecting Information Theft Based on Mobile Network Flows for Android Users "in 2017 International Conference on Networking, Architecture, and Storage (NAS).

[5]Rubata Riasat,Muntaha Sakeena,Abdul Hannan Sadiq,Yong-Ji Wang ."Onamd: An Online Android Malware Detection Approach" 2018 International Conference on Machine Learning and Cybernetics (ICMLC).

[6]Ajit Kumar,Vinti Agarwal,Shishir K. Shandilya,Andrii Shalaginov,Saket Upadhyay,Bhawna Yadav."PACE: Platform for Android Malware Classification and Performance Evaluation" in 2019 IEEE International Conference on Big Data (Big Data).

[7]Umme Sumaya Jannat,Syed Md. Hasnayeen,Mirza Kamrul Bashar Shuhan,Md. Sadek Ferdous."Analysis and Detection of Malware in Android Applications Using Machine Learning" in 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE).

[8]Ming Fan,Xiapu Luo,Jun Liu,Chunyin Nong,Qinghua Zheng,Ting Liu IEEE Transactions on Reliability ."CTDroid: Leveraging a Corpus of Technical Blogs for Android Malware Analysis" in 2020.

[9]Ikram Ul Haq,Tamim Ahmed Khan,Adnan Akhunzada IEEE Access ."A Dynamic Robust DL-Based Model for Android Malware Detection " in 2021.

[10]Zakaria Sawadogo,Jean-Marie Dembele,Gervais Mendy,Samuel Ouya."Zero-Vuln: Using deep learning and zero-shot learning techniques to detect zero-day Android malware" in 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME).