International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Protectors of the Android Domain: Research into Mobile Malware Detection and Defense

B. Bhaskar[1], Sharanya S.[2], A. Bala Murali[3], E Archana[4], Mr. T. A. Mohanaprakash[*5]

Submitted: 29/08/2023 **Revised**: 20/10/2023 **Accepted**: 02/11/2023

Abstract: Due to the ever-increasing amount of malware that can be found on mobile devices, Android malware detection has become an essential topic of study. It starts with an overview of android malware, including its many subtypes such as Trojan horses, adware, spyware, and ransomware, as well as the procedures that must be taken to avoid having malware installed on your device. In order to detect malicious programs, it performs an analysis on a number of attributes that have been collected from the application package files and system call traces of the device. The suggested solution makes use of a neural network model that was educated using a dataset consisting of both safe and harmful apps. This problem is solved with the assistance of android malware detection with the use of machine learning. In order to determine whether or not the Android application file that was uploaded includes malware or can be used safely, the system that has been presented makes use of a method for supervised machine learning that is known as a Neural Network. It gives an overview of the numerous methods that may be used to identify android malware, such as detection based on signatures, detection based on behaviors, and detection based on machine learning. Mobile devices, especially smartphones powered by Android, have emerged as indispensable aids in our day-to-day activities. On the other hand, a growth in the usage of mobile devices has resulted in an increase in the number of cyber-attacks, with malware being a serious concern. Malware designed for Android devices presents a serious threat to users of mobile devices since it has the potential to steal personal data, disrupt device functionality, and jeopardize the security of the device. As a result, it is more important than ever to identify and eliminate malware on Android devices.

Keywords: *Android malware, Malware detection, Trojan horses, Ransom ware, supervised machine learning, Personal data security.*

1. Introduction

Smartphones have quickly become indispensable in most people's lives, and the Android operating system has amassed a significant following. However, developing Android applications for third-party marketplaces involves a number of obstacles, the most significant of which is the elimination of any potential for infection. Traditional detection approaches depend on technology that is based on signatures, which is inadequate when dealing with unknown forms of malware. Analysis, both static and dynamic, of unknown harmful code in Android devices is the primary focus of the currently available detection techniques.

[1] *Assistant Professor, Department of Computer Science and Engineering, Madanapalle Institute of Technology and Science, Madanapalle – 517325, INDIA, bhaskarb@mits.ac.in*
[2] *Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, Kattankulathur, Chennai-603203, INDIA, nice_sharanya@yahoo.co.in*
[3] *Assistant Professor, Department of Computer Science and Engineering, St. Joseph's Institute of Technology, OMR, Chennai-600119, INDIA, balamura2@gmail.com*
[4] *Assistant Professor, Department of Computer Science and Engineering, Panimalar Engineering College, Chennai – 600123, INDIA, archana.athi@gmail.com,https://orchid.org/0000-0003-3601-9259*
[5] *Associate Professor, Department of Computer Science and Engineering, Panimalar Engineering College, Chennai – 600123, INDIA, tamohanaprakash@gmail.com, https://orchid.org/ 0000-0002-6885-4710*
** Corresponding Author Email: tamohanaprakash@gmail.com*

The proliferation of mobile devices has led to a rise in the number of cybercriminal activities, which in turn has put both the security and confidentiality of mobile systems in jeopardy. The analysis of program permissions is a standard way for detecting malware; however the process gets more difficult when all accessible rights are taken into consideration. As a result, we need a malware detection system that is both effective and works in real time.

Learning machine algorithms, and particularly deep learning, have been investigated as potential solutions to these problems. The proliferation of mobile apps, on the other hand, creates difficulties in terms of both scalability and accuracy. Analysis of API calls and permissions, when combined with machine learning techniques, may assist in identifying malicious as well as benign Android applications.

Because of Android's ubiquity, fraudsters are drawn to the platform, which has led to a never-ending fight against malware detection. For Android malware detection, a variety of methodologies, including static, dynamic, and hybrid methods, is used. This study focuses on detection methods that are based on neural networks that are used for Android apps. It does this by computing similarity scores between malicious and benign apps on the basis of suspicious API calls and then using those API calls as features in a feature vector. During the SVM classifier training process, potentially dangerous permission

combinations and extra characteristics are used. The system that has been presented can determine with high precision whether or not an application contains harmful code.

In conclusion, there is an urgent need for a malware detection solution for Android that is both effective and flexible. This study investigates neural network-based classification algorithms that make use of API call analysis and permission analysis to properly categorize applications as either benign or dangerous.

2. Related Works

More and more sensitive data is being processed by today's smartphones and tablets, which are equipped with a wide range of applications and services. Popular mobile platforms, such as Android and iOS, constitute an enticing target for malware authors, and the number of vulnerabilities targeting mobile devices is increasing everyday along with the general trend. The current statistics show an alarming surge in mobile malware abusing victims to earn profits, moving towards a billion-dollar sector, while experts attempt to discover alternative detection ways to combat against mobile malware. When it comes to mobile malware analysis and detection, current methods aren't always up to snuff [2] [4].The purpose of this paper is to present a well-organized summary of current research on mobile virus detection approaches, highlighting their strengths and weaknesses [1]. In recent years, Android-powered smartphone use has skyrocketed. Android's widespread appeal has attracted consumers, but the platform's popularity has also led to heightened security worries. Therefore, Android malware detection is a hot issue in the field of mobile security. With a focus on Android devices, this paper examines the most recent mobile malware assaults, vulnerabilities, detection methods, and security solutions. We have provided much comprehensive taxonomy that classifies methods for detecting mobile malware according to their analytical methodology, platform, data collecting, operational effect, acquired findings, and involvement of artificial intelligence. In order to better understand the threat landscape and the vulnerabilities that mobile malware exploits, a new taxonomy is offered. In addition, we have spoken about and categorized forensic analysis initiatives from the standpoint of mobile malware detection. We took the perspective of the invader and examined the numerous evasion tactics often used by malware developers to thwart detection. Finally, recommendations for future research are provided to assist academics and business mitigates the negative effects of these irritating attempts [2]. The number of people who own smartphones, especially those who use Android, has risen sharply in recent years. Malware detection is an increasing priority on the Android platform. Machine learning-based methods have improved malware detection accuracy more than any other available method. Therefore, machine learning algorithm malware detection has to be made available on Android smartphones. Researchers have suggested many Machine Learning methods for malware detection, each using a unique Machine Learning Algorithm like SVM, NB, or DNN. With an emphasis on Machine Learning Based classifiers, this article examines the current state of the art in Android malware detection methods [3].

The mobile ecosystem as a whole is seeing fast development in Android apps, but Android malware is also expanding at a breakneck pace. The issue of Android malware detection has been explored by a wide range of researchers, who have proposed a wide variety of hypotheses and approaches. According to the available literature, machine learning shows great promise as an efficient and effective method for identifying Android malware. However, there are studies that have looked at various problems with Android malware detection using ML. We think our study is a good supplement to the existing studies since it covers more ground. This article provides a complete overview of machine learning-based methods for detecting Android malware. The Android system architecture, security procedures, and the categorization of Android malware are briefly introduced, along with some more contexts on Android apps. Next, we zero in on machine learning to conduct a thorough review and summary of the state of the field with regards to vital factors such sample collection, data cleansing, feature selection, ML model and algorithm development, and detection efficiency assessment. Finally, we evaluate the long-term outlook for machine learning-based Android malware detection research. This survey will provide researchers with a comprehensive understanding of machine learning-based Android virus detection. It might pave the way for future studies and assist direct investigation in the sector as a whole [4]. The widespread availability of smartphones is largely due to the Android operating system. The newest innovations have down the price to where it's accessible to everyone. The rise of cybercrime on mobile devices has coincided with the rise of the Android platform. Because it runs on an open source OS, it is a frequent target of cybercriminals. The current situation of Android security is analyzed in depth in this article. This article divides Android system assaults into four categories: (i) those that target the hardware, (ii) those that target the kernel, (iii) those that target the hardware abstraction layer, and (iv) those that target applications. The research covers a wide range of security risks and countermeasures in these areas, providing a thorough examination of the fundamental issues in the Android security space. The essay also highlights the importance of app developers in creating a safer Android ecosystem. This article makes an effort to compare and contrast different

malware detection approaches with regards to their respective strengths and weaknesses. This research may aid in the development of a more complete, secure, and effective response to the challenges Android faces by providing insight into the Android security domain from several perspectives [5]. The proliferation of mobilemalware on the Android platform parallels the explosion in smartphone use and usage. Due of its dominant market share, Android is now a prime target for cybercriminals. Malware authors are increasingly drawn to Android due to the platform's openness, Android's high market share, and the rising popularity and demand for smartphones. From a scientific perspective, it is essential for an examiner to see the malicious software in action in order to grasp the extent of the threat to personal data. With so many sophisticated methods available, it's probable that no one method can detect malware effectively on its own. As a result, we have a number of options for efficient virus identification. This research focuses on the differences and similarities between static and dynamic analysis approaches to Android malware. In addition to comparing and contrasting the two main types of malware analysis, this study also covers the several sub-techniques that fall under each. In this study, a new method, hybrid analysis, is developed by fusing static and dynamic analysis, and its efficacy is evaluated in relation to that of established methods [6]. Researchers are putting more time and effort into studying how to identify malware on mobile smart devices. As mobile malware continues to spread rapidly, protecting mobile users' anonymity is more important than ever. Programming devices known as intrusion detection systems collect data, analyze it, and identify intrusions. Intrusion aversion systems (IPS) are the next generation of these systems, and they have the ability to take preventative measures. Accuracy rate is a crucial metric for evaluating the efficiency of an Intrusion Detection System. The goal of this study is to improve upon previous research by increasing the proportion of correct diagnoses while decreasing the number of false positives [7].

More and more attention is being paid by researchers to the problem of discovering mobile viruses. With the proliferation of mobile viruses, privacy for mobile users is more crucial than ever. Intrusion detection systems are computerized tools that sift through data to find evidence of hacking attempts. The next generation of such systems, known as intrusion aversion systems (IPS), may take preemptive action. When assessing the effectiveness of IDS, accuracy rate is a vital indicator. The purpose of this research is to build on past studies by raising the percentage of accurate diagnoses while minimizing the occurrence of false positives [8]. Since cellphones are becoming more and more pervasive, they have access to more private data. Advanced mobile malware, especially Android malware, may steal or use this information without the user's knowledge or permission. Therefore, it is crucial to develop efficient methods of analyzing and detecting such dangers. This article offers a thorough review of the state-of-the-art in Android malware analysis and detection methods, focusing on their ability to keep up with rapidly changing malware. This article classifies systems by approach and time period to analyze development and flaws. This paper also provides a framework for future study by discussing assessments of industry solutions, malware statistics, and malware evasion tactics [9]. Malware targeting the Android platform has grown in proportion to the platform's success. Traditional malware detectors are unable to identify these new forms of malware because of the creative methods malware authors use to construct harmful Android apps. Unknown Android malware may be detected using machine learning methods and the characteristics gathered from static and dynamic analysis of Android apps. In this study, we take a look at how different Android malware detection systems identify malware and compare them using a number of criteria. To further emphasize the prevalence of machine learning algorithms in this field for identifying Android malware in the wild, we were able to locate research work in all the Android malware detection strategies that involve machine learning [10].

3. Existing System:

Malware is a kind of cyber assault that is both widespread and devastating. Results the testing findings showed that with just 11 static features and the ExtraTree method, FG-Droid was able to attain a 97.7% area under the ROC curve (AUC) score. As a result of testing several models on the Drebin, Genome, and Arslan datasets using machine learning (ML), deep neural networks (DNNs), recurrent neural networks (RNNs), long short-term memories (LSTMs), and gated recurrent units (GRUs), this was born. This method examines an app's attributes and behavior to see whether it displays any signs of being harmful. Permissions, file system interactions, network activities, and code function procedures are all examples of things that may be examined using heuristics. To detect malicious activity, machine learning algorithms examine characteristics and trends in the application. New and changing malware strains may be detected using this strategy. Our study provides suggestions in the form of strategies to deal with rising security dangers presented by malware and reduce threat and malware infection rates based on the evaluated survey papers. Therefore, a novel approach of feature grouping was devised to create a classifier that is effective despite having little features, minimal analysis time, and high classification success.

Consequences

- It has a cumbersome interface that is difficult to

navigate.

- ❖ High Propensity to Make Mistakes
- ❖ It's a bigger hassle and uses up more supplies.

4. Proposed Methodology:

Any Android app's installation and restart activity logs, as well as any API calls made by such apps before and after a device restart, would be logged and characterized by the proposed system. For a more reliable, precise, and scalable android detection tool, it is necessary to conduct a critical examination of existing mobile malware frameworks, as suggested by the proposed model. In this work, we examine and categorize mobile malware based on privilege escalation and the objectives of their attacks. They also aid researchers in understanding the methods used by modern mobile malware to evade detection. The suggested model is almost as accurate as the current model while using much less memory and CPU time. Finally, the model's little space and time commitment makes it morally preferable to the status quo. Finally, the experimental findings are analyzed and contrasted using a variety of performance criteria.

BENEFITS:

- ❖ A simple and intuitive interface
- ❖ Improved precision and dependability
- ❖ The aesthetics of the setting
- ❖ More reliable and objective

4.1 System Architecture:

The procedure is mapped out by the system's architecture. In this case, the website does the searching and archiving for us. There, bot logic is utilized to pre-process the input and get process details. The algorithm is then applied, and further steps are conducted.

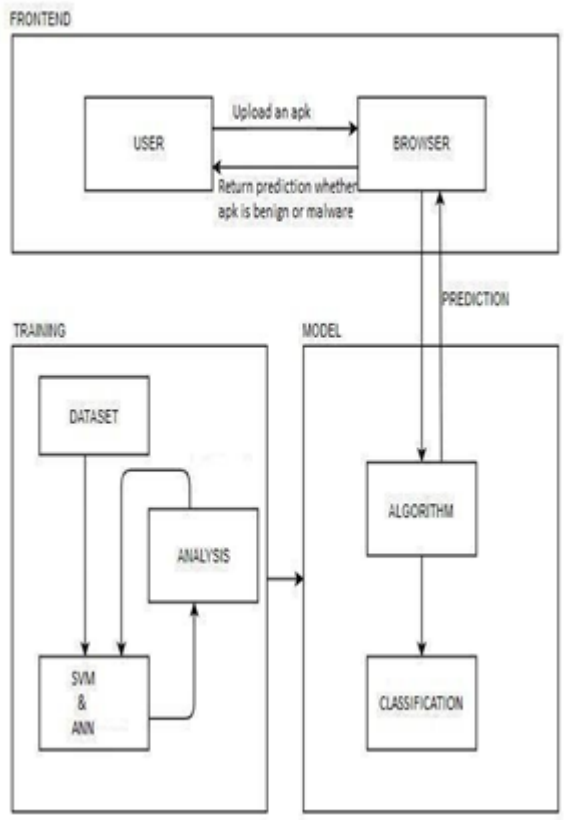

Fig1. Work Flow of the Proposed Work.

The procedure is mapped out by the system's architecture. In this case, the website does the searching and archiving for us. There, bot logic is utilized to pre-process the input and get process details. The algorithm is then applied, and further steps are conducted.

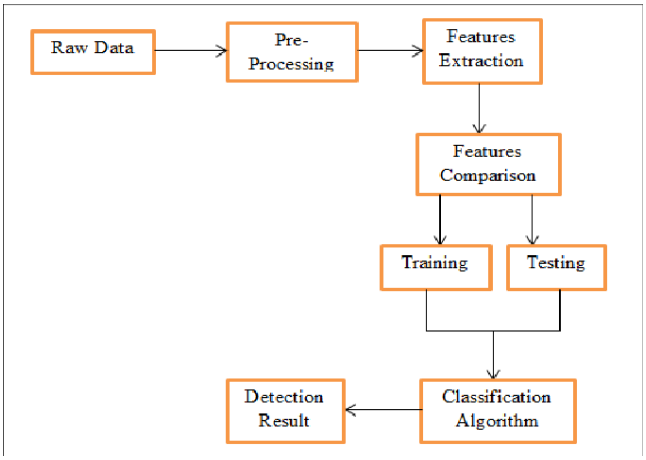

Fig2. Architecture Diagram

Data is first retrieved from APKs obtained from a Github malware repository. The data is then pre-processed before analysis. The last step is to feed the processed data into the training model. Once the training data is in place, the model is complete. The categorization result will be delivered in the testing phase using the most precise method.

3.2 SYSTEM IMPLEMENTATION:

3.2.1 Pandas:

Pandas is a very effective library for doing such things with data. Series and DataFrame are the two main new data types that are presented. Pandas is utilized to effectively manage textual data in this system. Filtering, cleaning, manipulating, and analyzing data are all made possible by its many features. Pandas can conduct a wide variety of data operations, including handling missing values, aggregations, merging datasets, and more. It is helpful for preparing textual data for machine learning algorithms since it streamlines processes like data preparation, feature extraction, and exploratory data analysis.

3.2.2 NumPy:

When working with Python for scientific purposes, the NumPy library is indispensable. In order to efficiently store and manipulate huge multi-dimensional arrays, it introduces the powerful nd array data structure. Here, NumPy is utilized to manage numerical information extracted from the processed text. It offers a wide variety of mathematical operations that may be performed numerically on arrays. Processing the numerical characteristics and feeding them into machine learning algorithms requires quick calculations, element-wise operations, arrayslicing, and reshaping, all of which are made possible by NumPy.

3.2.3 Scikit-learn:

Scikit-learn is a very well-liked machine learning package that has several useful tools and algorithms. Scikit-learn provides crucial parts of this system, including those responsible for identifying hate speech. To transform the cleaned-up text into a numerical form usable by machine learning, we use the CountVectorizer tool from Scikit-learn. It does this by converting the text into a matrix of token counts, which accurately represents the occurrence of individual words and n-grams. The Decision Tree Classifier in Scikit-learn is a supervised learning technique that uses a labeled dataset to develop a set of decision rules that may be used to categorize hate speech and abusive language.

3.2.4 Flask:

Flask is a Python web framework that is both lightweight and adaptable, making it ideal for developing both websites and APIs. Flask is used to build the hate speech detection system's user interface. Users may access the system through a web-based interface or application programming interface (API) endpoints.

Flask is responsible for directing requests and creating answers. It allows the system to take user-entered content, process it using the hate speech detection model, and provide the findings to the authors of the material. The user-facing part of the system may benefit from Flask's simplicity and adaptability.

3.2.5 APK:

The Android operating system, and a number of other Android-based operating systems, utilizes the Android Package file format with the .apk extension to distribute and install mobile applications, mobile games, and middleware.

3.2.6 Werkzeug:

To create WSGI (Web Server Gateway Interface) applications in Python, you may make use of the Werkzeug library collection. One of the most complex WSGI utility libraries, it began as a basic collection of assorted utilities for WSGI applications. The developer is free to choose their preferred approach to request processing, database connectivity, and template engines. Don't compel people to rely on each other. Features like as a debugger, request/response objects, cache control objects, cookie management, file uploads and a plethora of community-built extensions are all part of the package. To be distributed using the BSD license.

3.2.7 Androguard:

The Androguard is a reverse engineering tool for Android applications written in the Python programming language. In order to achieve this, we need to extract the app's components from their raw Android Package (.apk) files. Malware and security flaws may be tested for at this point. As long as Python is there, Androguard will run on Linux, Windows, and OSX. Please be aware that there are several dependencies involved with operating Androguard on Windows, and for the purpose of simplicity, we suggest that you utilize a Virtual Machine to run Linux instead.

3.2.8 Keras:

Keras is the TensorFlow platform's high-level API, providing a friendly, highly-productive interface for addressing issues in machine learning with a special emphasis on cutting-edge deep learning techniques. For rapidly creating and releasing machine learning solutions, it offers fundamental abstractions and building pieces.

3.2.9 Pickle:

Pickle is a module in Python used for serializing and de-serializing Python objects. This convertsPython objects like lists, dictionaries, etc. into byte streams (zeroes and ones). You can convert the byte streams back into Python objects through a process called unpickling. Pickling is also known as serialization, flattening, or marshaling.

5. Results and Discussion:

The need of efficient hate speech detection systems has only grown in the dynamic environment of internet discourse. In order to keep online communities free from damage, these technologies are essential for monitoring content and removing offending material. While progress has been made with current hate speech detection systems, there is still more that can be done to strengthen and improve upon them. This article will explore a number of methods for improving hate speech detection systems, such as using more complex NLP methods, improved feature engineering, a more diverse set of training data, and active learning tactics. These developments have the potential to lead to more dependable methods for policing objectionable language and preventing the spread of hate speech online. Advanced Natural Language Processing (NLP) methods are one of the most promising areas for enhancing hate speech detection systems. Simpler models, such those used by rule-based or shallow machine learning approaches, are often used by conventional systems. While helpful, these approaches may not be enough for identifying subtle forms of hate speech that vary greatly depending on the setting. There are more advanced NLP models that may be used to overcome this restriction, such as Long Short-Term Memory (LSTM) networks and BERT (Bidirectional Encoder Representations rom Transformers).

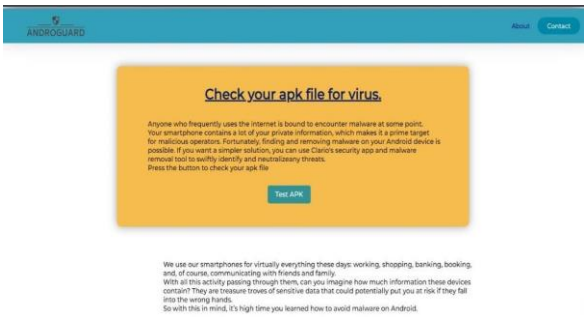

Fig3. Represents the Apk file for virus Detection.

Long short-term memory (LSTM) networks are a kind of deep learning model that performs particularly well with sequence data, such as that seen in text analysis. Sentiment analysis, text production, and language translation are just few of the areas where these models have excelled. In order to improve the accuracy of hate speech detection systems, LSTM networks are being included into them. These networks are able to better recognize the contextual links between words and phrases. Because of this enhanced comprehension, the algorithm is able to detect hate speech that would otherwise be overlooked by more simplistic approaches. In a similar vein, BERT, a cutting-edge transformer-based paradigm, has significantly advanced the science of natural language processing. The capacity of BERT to extract meaning and context from text is unrivaled. Systems designed to identify hate speech might benefit from BERT's ability to understand nuances in language including sarcasm, irony, and cultural allusions. Enhanced detection accuracy is the result of better contextual comprehension.

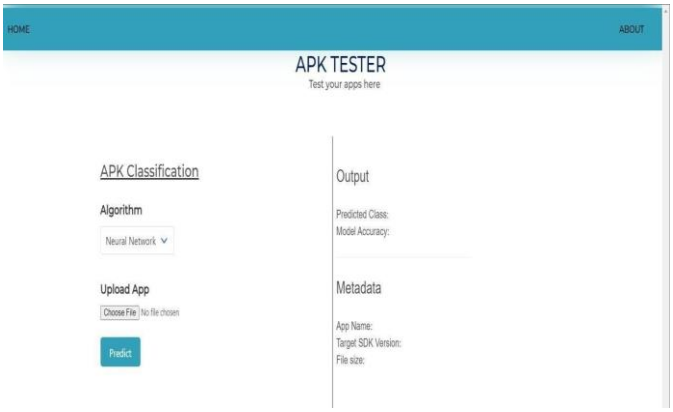

Fig4. Represents the APK Tester

Improvements may be done both with improved NLP methods and with enhanced feature engineering. Selecting and developing useful qualities that the model employs in prediction is what feature engineering is all about. Basic linguistic variables, including word frequency and length, are typically used by conventional hate speech detection systems. Hate speech, however, often employs figurative language and symbols, thus these characteristics may not be exhaustive. Researchers might look at more sophisticated feature engineering methods to overcome this restriction. Word embeddings, which map words to dense vector spaces, are one such method, as are part-of-speech tagging and named entity identification, both of which may be used to determine relevant context. Such sophisticated elements might enrich the model's representation of the text, enabling it to pick up on hints of hate speech that would otherwise go unnoticed. In addition, factors unique to online hate speech, such as the prevalence of slang and emoticons meant to express contempt, may play a significant role in boosting the reliability of the system.

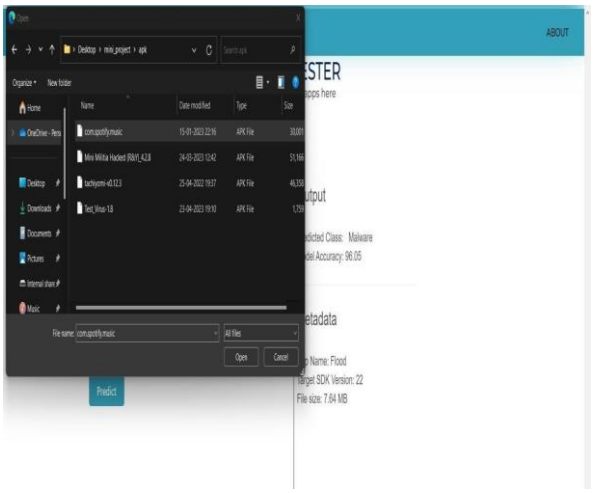

Fig5. Represents the selection of document from the file.

Diversifying the training data is also a crucial part of improving hate speech detection systems. The success of machine learning models relies heavily on the quality of their training data. It is possible that the range of languages, cultures, and settings in which hate speech might occur is underrepresented in the datasets used to train many current algorithms. Therefore, these algorithms may not be able to properly recognize hate speech outside of their training environments.

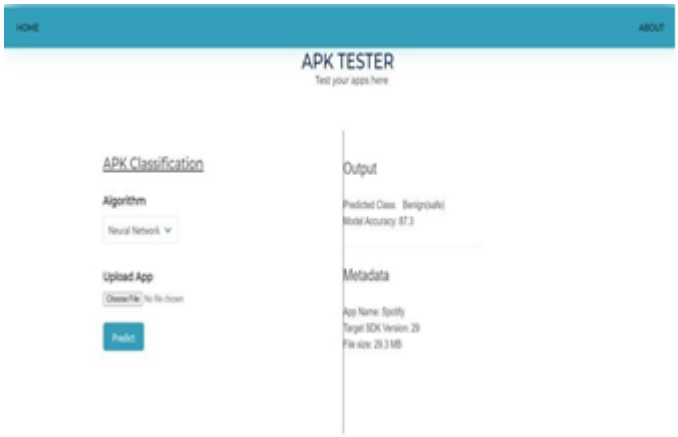

Fig6. Represents the output result after selection of the document from the file.

More varied and representative training data is needed to overcome this obstacle. This information has to represent the diverse nature of online interaction by include a wide range of languages, dialects, cultural settings, and platforms. In addition, as time passes and new cultural references and societal challenges arise, so does hate speech.

Training data should be regularly updated to ensure that the system can effectively identify new and emerging kinds of hate speech. Unfortunately, collecting large volumes of labeled training data may be a time-consuming and costly endeavor. This issue may be addressed with the use of active learning strategies. When it comes to choosing training data, active learning is a machine learning method that prioritizes difficult samples. The quantity of labeled data needed may be minimized by the strategic selection of samples that are on the model's decision boundary or those it is unsure about.

Fig7. Represents the output result after selection of the document that detects the Virus.

Active learning not only helps the model learn faster, but it also helps preserve resources. It allows the algorithm to become more responsive to shifting patterns of hate speech with less tagging effort. In reality, this means the algorithm may actively seek input from human annotators to improve its hate speech recognition skills when it meets new and possibly damaging material. In conclusion, it is crucial that hate speech detection technologies be developed and improved in the current digital environment. These systems may be made more reliable and accurate by using active learning procedures, increasing the diversity of training data, and adding sophisticated NLP techniques like as LSTM and BERT. Advanced feature engineering approaches enable richer representations of text, while more complex models allow for a more nuanced grasp of context and language. In order to keep up with the dynamic nature of online hate speech, the system is trained on a wide variety of data and uses active learning methods. Our attempts to curb hate speech and inflammatory language online must evolve in tandem with the rapid development of related technologies. By making these changes, we can look forward to improved hate speech detection algorithms that will make the internet a more welcoming and safe place for everyone to participate.

6. Conclusion:

Because of the Android operating system's meteoric rise in popularity among consumers, software developers are increasingly focusing their attention on this sector. This pattern has led to the distribution of a large number of apps that are helpful to consumers in the marketplaces. In this manner, many apps are openly shared, and a significant portion of them include the propagation of harmful

software. As a result, there is a technique of distribution that involves the inclusion of both harmful and innocuous programs. Malware that targets Android devices is among the most deadly dangers that can be found on the internet, and its frequency has skyrocketed over the last several years.

Cyber security professionals are confronted with an unsolved issue. Identifying and categorizing Android malware may be accomplished using a number of different technologies that are based on machine learning. Malware classification may be effectively carried out with the help of this project's Machine Learning Model, which makes use of feature selection and a Machine Learning Classifier. Our model has shown promising results, with an SVM Category Classification accuracy of over 93% and an ANN Category Classification accuracy of over 90.82% respectively. In the not too distant future, we want to provide an online service that, among other things, will enable users to assess whether or not a program (application) is malware before downloading it, as well as the program's category and family. A great deal of progress would be made in protecting the safety of an Android smartphone if this safeguard were taken.

Acknowledgements

No funding sources.

Author contributions

All authors are equally contributed in preparing, experimenting and reviewing the article.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] V. Kouliaridis, K. Barmpatsalou, G. Kambourakis, and S. Chen, ‘‘A survey on mobile malware detection techniques,’’ IEICE Trans. Inf. Syst., vol.103, no. 2, pp. 204–211, Feb. 2020

[2] Qamar, A. Karim, and V. Chang, ‘‘Mobile malware attacks: Review, taxonomy & future directions,’’ Future Gener. Comput. Syst., vol. 97, pp. 887–909, Aug. 2019.

[3] E. J. Alqahtani, R. Zagrouba, and A. Almuhaideb, ‘‘A survey on Android malware detection techniques using machine learning algorithms,’’ in Proc. 6th Int. Conf. Softw. Defined Syst. (SDS), Rome, Italy, Jun. 2019, pp. 110–117.

[4] H. Lubuva, Q. Huang, and G. C. Msonde, ‘‘A review of static malware detection for Android apps permission based on deep learning,’’ Int. J. Comput. Netw. Appl., vol. 6, no. 5,pp. 80–91, Sep./Oct. 2019.

[5] P. Bhat and K. Dutta, ‘‘A survey on various threats and current state of security in Android platform,’’ ACM Comput. Surv., vol. 52, no. 1, p. 21, Feb. 2019.

[6] M. Choudhary and B. Kishore, ‘‘HAAMD: Hybrid analysis for Android malwaredetection,’’ in Proc. Int. Conf. Comput. Commun. Informat. (ICCCI), Jan. 2018, pp. 1–4 [7]. D. BalaGanesh,

[7] Chakrabarti, and D. Midhunchakkaravarthy, ‘‘Smart devicesthreats, vulnerabilities and malware detection approaches: A survey,’’ Eur. J. Eng. Res.Sci., vol. 3, no. 2, pp. 7–12, Feb. 2018.

[8] P. Yan and Z. Yan, ‘‘A survey on dynamic mobile malware detection,’’Softw.Qual. J., vol. 26, no. 3, pp. 891–919, Sep. 2018.

[9] K. Tam, A. Feizollah, N. B. Anuar, R. Salleh, and L. Cavallaro, ‘‘The evolution of Android malware and Android analysis techniques,’’ ACM Comput. Surv., vol. 49, no.4, p. 76, 2017.

[10] S. K. Muttoo and S. Badhani, ‘‘Android malware detection: State of the art,’’ Int. J. Inf. Technol., vol. 9, no. 1, pp. 111–117, Mar. 2017.