JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

ANDROID MALWARE DETECTION USING ANN AND GENETIC ALGORITHM

[1]Dr.K.N.S Lakshmi, [2]Amith Prasad
[1]Professor & HOD Department of Computer Science and Engineering, [2]MCA 2nd year
[2]Master of Computer Applications,
[2]Sanketika Vidya Parishad Engineering College, Visakhapatnam, India

ABSTRACT:
The Android Malware Detection using Genetic Algorithm (Feature Selection) and Artificial Neural Network (ANN) Model project aims to develop an advanced system for detecting malware in Android applications. The project combines the power of genetic algorithms and artificial neural networks to achieve accurate and efficient malware detection. By employing a Genetic Algorithm, relevant features are selected from a large feature set, reducing dimensionality and improving classification performance. The selected features are then used to train an ANN model capable of accurately classifying Android applications as malware or benign. The project encompasses steps such as dataset preprocessing, implementing the Genetic Algorithm, training the ANN model, and evaluating its performance. The results obtained from the system can demonstrate its effectiveness in detecting Android malware and contribute to enhancing the security and privacy of Android users.

KEYWORDS:

Malware Prediction, Machine Learning, Genetic Algorithm, Artificial Neural Network (ANN), Android Dataset.

1.1 INTRODUCTION

The Android Malware Detection using Genetic Algorithm (Feature Selection) and Artificial Neural Network (ANN) Model project addresses the pressing issue of malware detection in Android applications. With the increasing popularity of Android devices, the threat of malware has become a significant concern, as malicious applications can compromise user privacy, steal sensitive information, and cause various security risks. Traditional malware detection methods often struggle to keep pace with the evolving techniques employed by malware authors, necessitating the development of more efficient and accurate approaches. This project proposes a solution that leverages the power of genetic algorithms for feature selection and artificial neural networks for classification to enhance the detection capabilities. By effectively combining these techniques, the project aims to improve the overall security and privacy of Android users by providing a robust system capable of accurately identifying and mitigating potential malware threats. The key point of the project is to help android user beware of the malicious applications on the web and protect their privacy.

1.1 SCOPE OF THE PROJECT

The scope of this project is to develop an Android malware detection system using a combination of genetic algorithms and artificial neural networks. The system aims to analyse and classify Android applications as either benign or malware based on their requested permissions. The project involves preprocessing the dataset, feature selection using genetic algorithms, training an ANN model, and integrating the system into an Android application. The scope also includes evaluating the system's performance, improving accuracy through iterative development, and considering future enhancements such as real-time detection, user feedback integration, and expanding the malware classification to different families. Additionally, the project can be extended to address cross-platform compatibility and optimize the system's performance for efficient malware detection on mobile devices.

2.1 SYSTEM ANALYSIS

Functional Analysis:
User Interface: The system should provide a user-friendly graphical interface for easy interaction and APK file selection.
APK Processing: The system should be able to extract permissions from the selected APK files using the Androguard library.
Preprocessing: The system should preprocess the extracted permissions to match the input requirements of the ANN model.
ANN Model: The system should load a pre-trained ANN model using the Keras library for predicting the nature of APK files.
Result Display: The system should display the prediction results and extracted permissions on the graphical interface.

Non-Functional Analysis:
Performance: The system should provide accurate and efficient malware detection with minimal processing time.
Usability: The graphical interface should be intuitive, easy to navigate, and visually appealing.
Security: The system should ensure the confidentiality and integrity of the analysed APK files.

The Android Malware Detection System follows a client-server architecture. The client side consists of the graphical user interface (GUI) implemented using the Tkinter library, which interacts with the user for APK file selection and displays the results. The server side includes the APK processing, permission extraction, preprocessing, and ANN model prediction modules.

By conducting a thorough system analysis, the Android Malware Detection system using Genetic Algorithm and ANN Model ensures that all aspects of the project are carefully considered and addressed. This analysis ensures the system is designed and implemented to meet the project's specific requirements effectively and efficiently while mitigating potential risks and challenges.

2.2 EXISTING SYSTEM

Signature-Based Detection:

Signature-based detection involves matching known malware signatures against the code or characteristics of Android applications. This technique relies on maintaining a database of malware signatures and comparing them with the applications being scanned. However, signature-based detection can be slow and ineffective against new or modified malware that does not match existing signatures. It requires constant updates to the signature database, making it less suitable for detecting emerging threats.

Rule-Based Detection:

Rule-based detection involves defining specific rules or patterns that indicate malicious behaviour in Android applications. These rules are typically based on known malware behaviours or suspicious patterns. However, rule-based detection can be slow and limited in its ability to detect complex or polymorphic malware that can evade predefined rules. It also requires constant updates and maintenance to keep up with evolving malware techniques.

DISADVANTAGES OF EXISTING SYSTEM:

1.Low accuracy
2.Slow and Ineffective against new or modified android applications.
3.High cost
4.High Variance.

2.3 PROPOSED SYSTEM

The proposed system utilizes Genetic Algorithm for feature selection and Artificial Neural Network (ANN) for classification. It collects a diverse dataset of Android applications, preprocesses it, and extracts relevant features like permissions and API calls. The ANN model is trained on the selected features to classify applications as benign or malicious, enhancing detection accuracy. Performance optimizations, such as parallel processing and hardware acceleration, ensure efficient execution for real-time or near-real-time malware detection. The user-friendly interface allows users to input applications and receive classification results. Detailed reports and summaries provide insights into malware classification outcomes, statistics, and performance metrics. The system emphasizes continuous updates to the malware signature database, feature selection algorithms, and ANN model for effective detection of new malware variants. By combining advanced techniques and adaptation, the system enhances Android security by identifying and mitigating potential malware threats.

ADVANTAGES OF PROPOSED SYSTEM:

1.Improved Accuracy.
2.Efficient Feature Selection.
3.User-Friendly Interface.
4.Enhanced Security and Privacy

2.4 FEASIBLITY STUDY

Technical Feasibility:

The proposed system leverages established technologies such as Genetic Algorithms and Artificial Neural Networks, which have been widely studied and implemented in the field of machine learning. The availability of tools, libraries, and frameworks for feature selection, classification, and performance optimization makes the technical implementation feasible. Additionally, the system can be developed using popular programming languages such as Python and utilize existing resources and hardware infrastructure.

Economic Feasibility:

The economic feasibility of the proposed system depends on factors such as development costs, maintenance costs, and potential return on investment. Development costs include software development, algorithm refinement, and system integration expenses. Maintenance costs involve updating malware signature databases, feature selection algorithms, and the ANN model to adapt to new threats. The return on investment can be achieved through increased security, reduced malware-related incidents, and enhanced user trust.

Operational Feasibility:

The proposed system is operationally feasible as it aligns with the goal of detecting and mitigating malware threats in Android applications. It can be integrated into existing security infrastructure or operated as a standalone solution. Operational feasibility also depends on the availability of the necessary data sources for building and updating the malware signature database, as well as ensuring the system's compatibility with different Android platforms and application versions.

Based on the feasibility analysis, the proposed Android Malware Detection System demonstrates technical feasibility, aligns with operational goals.

3 SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS (Minimum Requirement)

1.RAM:4GB+RAM

2.PROCESSOR: i3 5th Gen 2.2 Ghz

3.STORAGE: 5GB

3.2 SOFTWARE REQUIREMENTS

1.Domain: Python
2.Version: Python IDLE (3.8.0) or above
3.Code Editors: Notepad++ or Visual Studio Code Editor
4.Frameworks and Dependencies: pandas, scikit-learn, NumPy, matplotlib, joblib, tkinter
5.Operating System: Windows 10 or above

Pandas:
Pandas provide us with many Series and DataFrames. It allows you to easily organize, explore, represent, and manipulate data. Smart alignment and indexing featured in Pandas offer you a perfect organization and data labelling. Pandas has some special features that allow you to handle missing data or value with a proper measure. This package offers you such a clean code that even people with no or basic knowledge of programming can easily work with it. It provides a collection of built-in tools that allows you to both read and write data in different web services, data-structure, and databases as well. Pandas can support JSON, Excel, CSV, HDF5, and many other formats. In fact, you can merge different databases at a time with Pandas.

NumPy:
Arrays of NumPy offer modern mathematical implementations on huge amount of data. NumPy makes the execution of these projects much easier and hassle-free. NumPy provides masked arrays along with general array objects. It also comes with functionalities such as manipulation of logical shapes, discrete Fourier transform, general linear algebra, and many more. While you change the shape of any N-dimensional arrays, NumPy will create new arrays for that and delete the old ones. This python package provides useful tools for integration. You can easily integrate NumPy with programming languages such as C, C++, and Fortran code.

Joblib:
Joblib is a Python library for efficient parallel computing and data serialization. It simplifies the process of parallelizing code and provides caching mechanisms to optimize repetitive computations, making it valuable for data-intensive tasks.

Tkinter:
Tkinter is a Python library for creating graphical user interfaces (GUIs). It provides a set of tools and widgets to build interactive applications with windows, buttons, input fields, and more. Tkinter is user-friendly and widely used due to its simplicity and cross-platform compatibility.

Scikit-Learn:
Scikit Learn comes with a clean and neat API. It also provides very useful documentation for beginners. It comes with different algorithms – classification, clustering, and regression. It also supports random forests, k-means, gradient boosting, DBSCAN and others. This package offers easy adaptability. Once you get well with the general functionalities of Scikit Learn, switching to other platforms will be no problem at all. Scikit Learn offers easy methods for data representation. Whether you want to present data as a table or matrix, it is all possible with Scikit Learn. It allows you to explore through digits that are written in hands. You can not only load but also visualize digits-data as well.

4 CODE EDITORS

4.1 Notepad++

Notepad++ is a source code editor and a replacement for Microsoft Windows' default Notepad, available for free under the GNU General Public License. It supports multiple languages and is built using C++ with Scintilla, ensuring a faster execution speed and a smaller program size through optimized routines while maintaining user-friendliness. The developer of Notepad++, Don Ho, originally used JEXT but decided to build a C++ text editor with Scintilla after being dissatisfied with JEXT's poor performance. He developed it in his spare time since his company rejected the idea. Notepad++ was released in 2003 and was initially available only for Windows. Although the author considered porting it to Mac OS X and Unix platforms using wx Widgets, this idea was ultimately rejected. In 2010, the US government obliged US-based open-source project hosts to deny access to users in certain countries, which the developer felt was a violation of the free and open-source software (FOSS) philosophy. As a response, Notepad++ moved out of US territorial jurisdiction by releasing a version on Tux Family in France. Notepad++ was criticized by Lifehacker for its user interface but was still voted as the most popular text editor by its readers in 2014. In 2015, it was voted the most used text editor worldwide by Stack Overflow respondents. Notepad++ left Source Forge completely in 2015 and moved its forums to Node BB and its bug tracker to GitHub.

4.2 VISUAL STUDIO CODE EDITOR

Visual Studio Code (VS Code) is a popular and versatile source code editor that provides a rich set of features for project documentation. It offers a user-friendly interface with a wide range of extensions and plugins, making it highly customizable and adaptable to various programming languages and project requirements. With its integrated terminal, version control system, and intelligent code completion, VS Code allows developers to efficiently write and manage project documentation. It supports Markdown, a lightweight markup language, enabling easy creation and formatting of text documents. Additionally, its live preview feature enables real-time visualization of Markdown files, facilitating a seamless editing experience. Overall, VS Code provides a powerful and efficient environment for creating comprehensive and well-structured project documentation.

5 MODULE DESCRIPTION

The client-side application has one module.

1. Malware Detection: This module allows the user to upload any android application. The features of the application are extracted and displayed along with the prediction of Malware or Benign Applications.

5.1 ARCHITECTURE

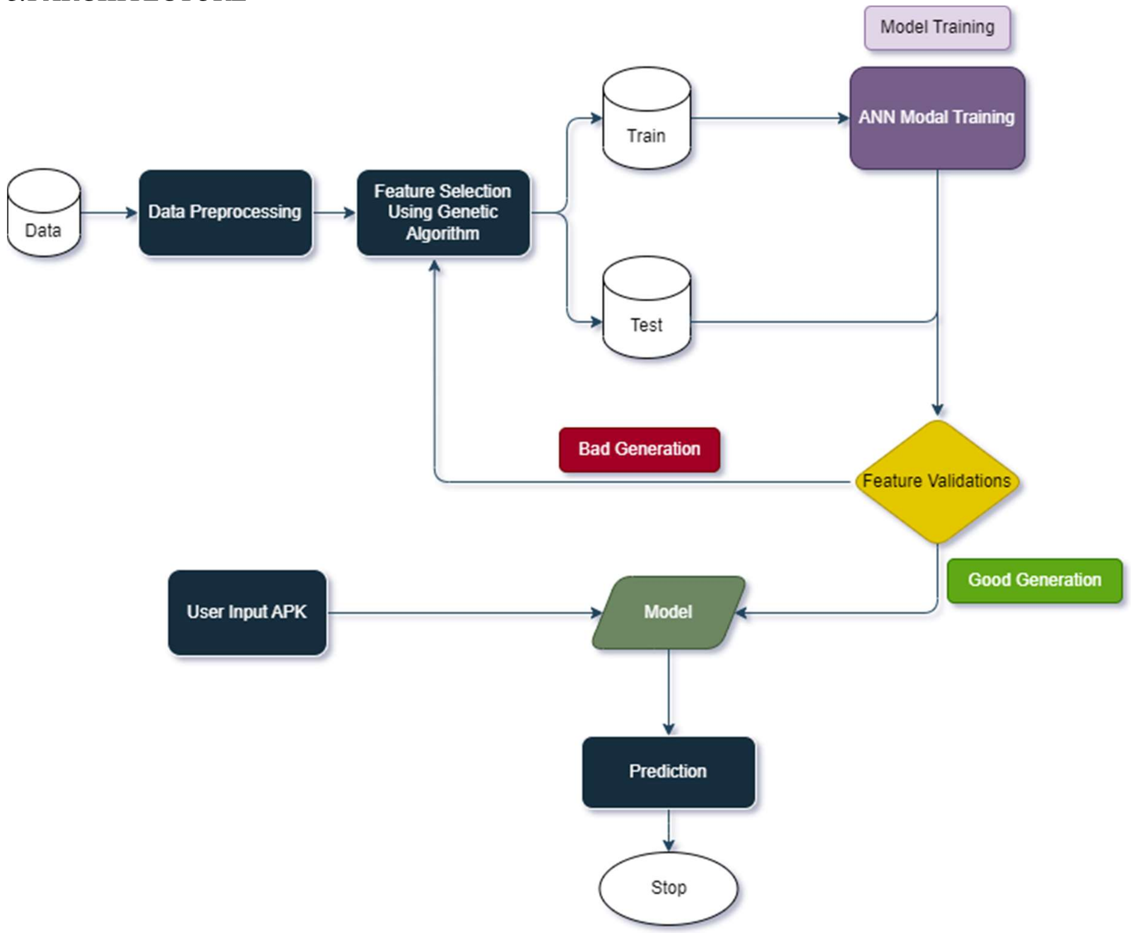
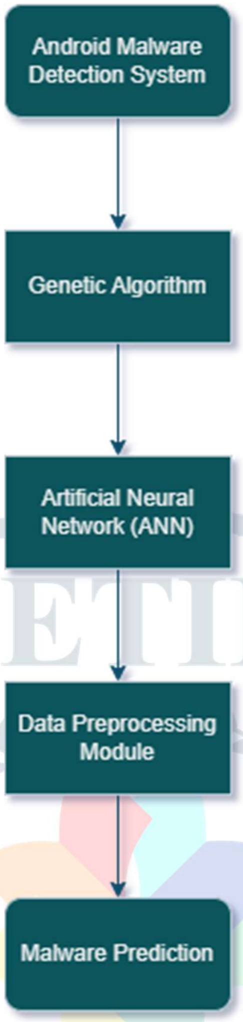


Figure 5.1: System Architecture

5.2 DATAFLOW DIAGRAM


Android Malware Detection System
Genetic Algorithm
Artificial Neural Network (ANN)
Data Preprocessing Module
Malware Prediction

THINGS IN UML

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.
There are four kinds of things in the UML:
•Structural things
•Behavioural things
•Grouping things
•A notational thing
Structural things are the nouns of UML models. The structural things used in the project design are:
First, a class is a description of a set of objects that share the same attributes, operations, relationships and semantics.

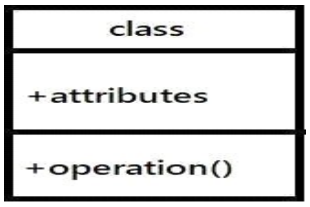


Figure 5.2: Classes

Second, a use case is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor.

Place order

Figure 5.2.1: Use Cases

Third, a node is a physical element [20] that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability

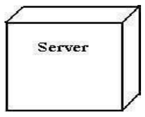


Figure 5.2.2: Nodes

Behavioural things are the dynamic parts of UML models. The behavioural thing used is:

Interaction: An interaction is a behaviour that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behaviour invoked by a message, and links (the connection between objects)

5.3 Relationships in UML

There are four kinds of relationships in the UML:
•Dependency
•Association
•Generalization
•Realization
A dependency is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing)

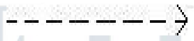


Figure 5.3: Dependencies

An association is a structural relationship that describes a set links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.

Figure 5.3.1: Association

A generalization is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).

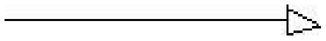


Figure 5.3.2: Generalization

A realization is a semantic relationship between classifiers, where in one classifier specifies a contract that another classifier guarantees to carry out.

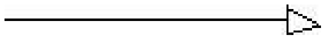


Figure 5.3.3: Realization

6 RELATED WORKS

6.1 Pooja Yadav, Neeraj Menon(2022), Efficient Net convolutional neural networks-based Android malware detection

Owing to the increasing number and complexity of malware threats, research on automated malware detection has become a hot topic in the field of network security. Traditional malware detection techniques require a lot of human intervention and resources (in terms of time and storage) as these involve manual analysis of all malware files in the application. Moreover, malware authors have developed techniques like polymorphism and code obfuscation that evade the traditional signature-based detection approaches employed by anti-virus companies. To solve this issue, malware detection enabled by deep learning (DL) methods is being used recently. This paper presents a performance comparison of 26 state-of-the-art pre-trained convolutional neural network (CNN) models in Android malware detection. It also includes performance obtained by large-scale learning with SVM and RF classifiers and stacking with CNN models. Based on the results, an EfficientNet-B4 CNN-based model is proposed to detect Android malware using image-based malware representations of the Android DEX file. EfficientNet-B4 extracts relevant features from the malware images. These features are then passed through a global average pooling layer and fed into a SoftMax classifier. The proposed method obtained an accuracy of 95.7% in the binary classification of Android malware images, outperforming the compared models in all performance metrics.

6.2 Sharfah Ratibah Tuan Mat, Mohd Faizal Ab Razak, Mohd Nizam Mohmad Kahar, Juliza Mohamad Arif, Ahmad Firdaus (2022), A Bayesian probability model for Android malware detection

The unprecedented growth of mobile technology has generated an increase in malware and raised concerns over malware threats. Different approaches have been adopted to overcome the malware attacks yet this spread is still increasing. To combat this issue, this study proposes an Android malware detection system based on permission features using Bayesian classification. The permission features were extracted via the static analysis technique. The 10,000 samples for the judgement were obtained from AndroZoo and Drebin databases. The experiment was then conducted using two algorithms for feature selection: information gain and chi-square. The best accuracy rate of detection of permission features achieved was 91.1%.

7 OUTPUT SCREENS

7.1 LANDING PAGE

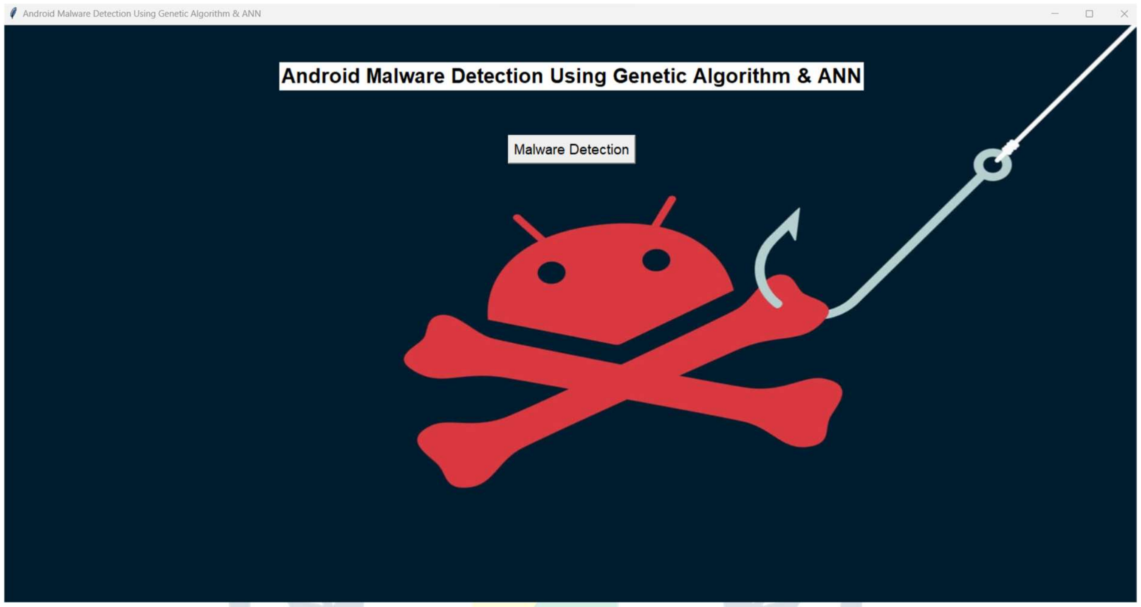
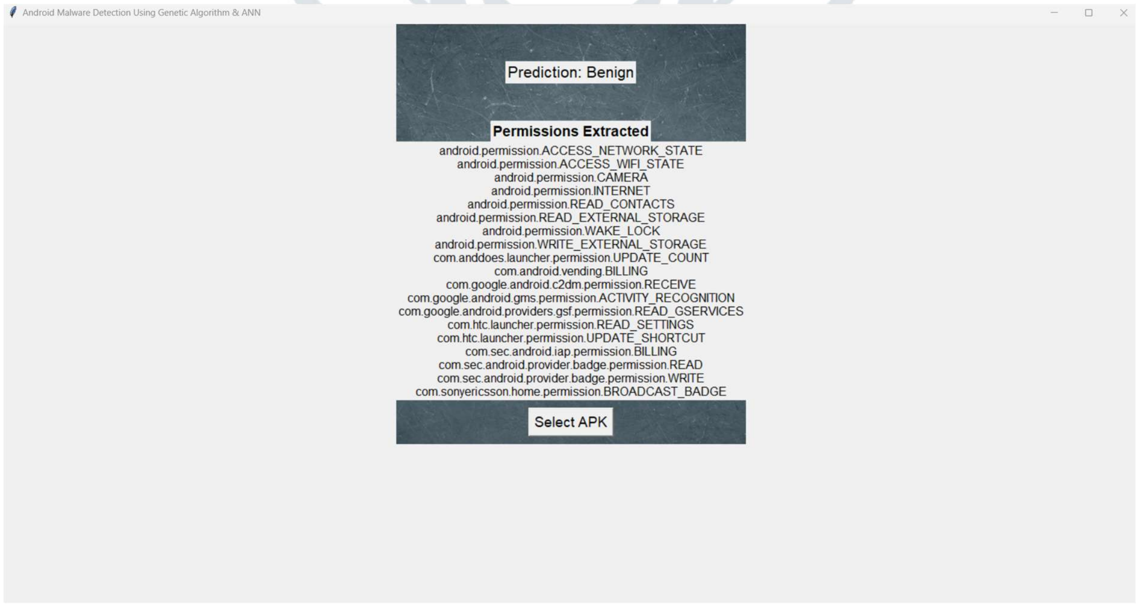

Android Malware Detection Using Genetic Algorithm & ANN
Android Malware Detection Using Genetic Algorithm & ANN
Malware Detection

Figure 7.1 Landing Page

7.2 MALWARE DETECTION PAGE



Prediction: Benign

Permissions Extracted

```
android.permission.ACCESS_NETWORK_STATE
android.permission.ACCESS_WIFI_STATE
android.permission.CAMERA
android.permission.INTERNET
android.permission.READ_CONTACTS
android.permission.READ_EXTERNAL_STORAGE
android.permission.WAKE_LOCK
android.permission.WRITE_EXTERNAL_STORAGE
com.anddoes.launcher.permission.UPDATE_COUNT
com.android.vending.BILLING
com.google.android.c2dm.permission.RECEIVE
com.google.android.gms.permission.ACTIVITY_RECOGNITION
com.google.android.providers.gsf.permission.READ_GSERVICES
com.htc.launcher.permission.READ_SETTINGS
com.htc.launcher.permission.UPDATE_SHORTCUT
com.sec.android.iap.permission.BILLING
com.sec.android.provider.badge.permission.READ
com.sec.android.provider.badge.permission.WRITE
com.sonyericsson.home.permission.BROADCAST_BADGE
```

Figure 7.2 Malware Detection Page

8 PURPOSE

The purpose of this project, "Android Malware Detection Using Genetic Algorithm & ANN," is to develop a system that can effectively detect malware in Android applications. Malware poses a significant threat to the security and privacy of mobile device users, and it is essential to have robust mechanisms in place to identify and mitigate these risks.

The project utilizes a combination of Genetic Algorithm (GA) and Artificial Neural Network (ANN) techniques to enhance the accuracy and efficiency of malware detection. The Genetic Algorithm is employed to select the most relevant features from the dataset, reducing the dimensionality and improving the performance of the system. The Artificial Neural Network is used as the underlying machine learning model to classify applications as either benign or malicious based on their extracted features.

The system's usefulness lies in its ability to provide an automated and reliable approach to identify and flag potentially harmful applications. By accurately distinguishing between benign and malicious apps, it helps users make informed decisions about the safety of the applications they install on their devices. This project contributes to the overall security of Android devices, protecting users' personal data, and mitigating the risks associated with malware infections.

9 CONCLUSION

In conclusion, the project "Android Malware Detection Using Genetic Algorithm & ANN" successfully demonstrates the effectiveness of employing Genetic Algorithm and Artificial Neural Network techniques for detecting malware in Android applications. The system's implementation and integration of these techniques showcase its potential in accurately identifying and flagging potentially harmful applications, contributing to enhanced security and privacy for mobile device users. By automating the malware detection process and providing reliable results, the project offers users a valuable tool to make informed decisions about the safety of the applications they install on their devices, mitigating the risks associated with malware infections.

9.1 SCOPE FOR FUTURE DEVELOPMENT

The project "Android Malware Detection Using Genetic Algorithm & ANN" opens up several avenues for future development and enhancements. Some potential areas for further exploration and improvement include:

1. Feature Engineering: Investigating additional features and data sources that can enhance the accuracy of malware detection. This could involve analysing other characteristics of Android applications, such as network behaviour, code obfuscation techniques, or application metadata.

2. Advanced Machine Learning Techniques: Exploring advanced machine learning algorithms and techniques to further improve the detection accuracy. This could involve utilizing deep learning models, ensemble methods, or hybrid approaches that combine multiple algorithms for more robust results.

3. Real-time Detection: Extending the system to operate in real-time, allowing for on-device or cloud-based scanning of applications during installation or runtime. This would enable immediate detection and prevention of malware, providing a proactive defence mechanism.

4. Malware Family Classification: Enhancing the system to classify detected malware into different families or categories. This would provide more detailed insights into the specific types and behaviours of malware, aiding in targeted mitigation strategies.

5. User Feedback and Community Contributions: Building a feedback mechanism where users can report suspicious applications or provide feedback on detected malware. Additionally, fostering a community-driven approach by allowing users to contribute labelled samples of malware for training and improving the detection models.

6. Cross-Platform Compatibility: Adapting the system to support other mobile platforms, such as iOS, to provide malware detection capabilities for a broader range of devices.

7. Performance Optimization: Optimizing the system's performance to handle large-scale datasets and improve processing speed. This could involve parallel processing, distributed computing, or leveraging hardware accelerators like GPUs to expedite the detection process.

By focusing on these areas, the project can continue to evolve, staying aligned with the ever-evolving landscape of mobile malware and ensuring the development of an effective and comprehensive solution for Android malware detection.

10 REFERENCES

[1] An article Reference on “Android Malware Classification Using Optimized Ensemble Learning Based on Genetic Algorithms” A Taha, O Barukab
https://www.mdpi.com/2071-1050/14/21/14406
[2] An article Reference on “Android Malware Detection Using Hybrid Meta-Heuristic Feature Selection and Ensemble Learning Techniques” by Sakshi Bhagwat & Govind P. Gupta
https://link.springer.com/chapter/10.1007/978-3-031-12638-3_13
[3] An article web reference on “ADAM: Automatic Detection of Android Malware” by Somanath Tripathy, Narendra Singh & Divyanshu N. Singh
https://link.springer.com/chapter/10.1007/978-3-031-17510-7_2
http://205.174.165.80/CICDataset/MalDroid-2020/Dataset/APKs/

[4] An article web reference on “Machine Learning from Theory to Algorithms: An Overview” by J Alzubi, A Nayyar, A Kumar - Journal of physics
https://iopscience.iop.org/article/10.1088/1742-6596/1142/1/012012/meta
https://iopscience.iop.org/article/10.1088/1742-6596/1142/1/012012/pdf
[5] An article book Reference on” Deep neural architectures for large scale android malware analysis” by M Nauman, TA Tanveer, S Khan, TA Syed.
https://link.springer.com/article/10.1007/s10586-017-0944-y
[6] An article web reference of” The Evolution of Android Malware and Android Analysis Techniques” by K Tam, A Feizollah, NB Anuar, R Salleh, Lorenzo Cavallaro.
https://dl.acm.org/doi/abs/10.1145/3017427
[7] An article book Reference on “Dynamic behavior analysis of android applications for malware detection”
By L Singh, M Hofmann
https://ieeexplore.ieee.org/abstract/document/8324010
[8] An article web reference on “Hybrid Feature Selection Method Based on the Genetic Algorithm and Pearson Correlation Coefficient” by R Saidi, W Bouaguel, N Essoussi
https://link.springer.com/chapter/10.1007/978-3-030-02357-7_1
[9] An article Reference on “A new machine learning method consisting of GA-LR and ANN for attack detection”
By S Hosseini - Wireless Networks
https://link.springer.com/article/10.1007/s11276-020-02321-3
[10] An article web reference on “Artificial intelligence, machine learning and deep learning: definitions and differences” by D. Jakhar, I. Kaur
https://academic.oup.com/ced/article-abstract/45/1/131/6597747
[11] An article web reference on “A Hybrid Analysis-Based Approach to Android Malware Family Classification”
by C Ding, N Luktarhan, B Lu, W Zhang
https://www.mdpi.com/1099-4300/23/8/1009
[12] An article book Reference “MalClassifier: Malware family classification using network flow sequence behaviour” by BA AlAhmadi, I Martinovic -
https://ieeexplore.ieee.org/abstract/document/8376209
[13] An article web Reference on “Various Frameworks and Libraries of Machine Learning and Deep Learning: A Survey” by Z Wang, K Liu, J Li, Y Zhu, Y Zhang
https://link.springer.com/article/10.1007/s11831-018-09312-w
[14] An article web Reference on “Malton: Towards On-Device Non-Invasive Mobile Malware Analysis for ART” by L Xue, Y Zhou, T Chen, X Luo, G Gu
https://www.usenix.org/system/files/conference/usenixsecurity17/sec17-xue.pdf
[15] An article book Reference on “AdDroid: Rule-Based Machine Learning Framework for Android Malware Analysis” by A Mehtab, WB Shahid, T Yaqoob
https://link.springer.com/article/10.1007/s11036-019-01248-0
[16] An article web reference on “Deep Learning and Visualization for Identifying Malware Families” by G Sun, Q Qian
https://ieeexplore.ieee.org/abstract/document/8565880
[17] An article book Reference “Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning” by A Fatima, R Maurya, MK Dutta
https://ieeexplore.ieee.org/abstract/document/8769039
[18] An article book Reference on “Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey” by H Liu, B Lang
https://www.mdpi.com/2076-3417/9/20/4396
[19] An article web reference on “Machine Learning and Deep Learning Methods for Cybersecurity” by Y Xin, L Kong, Z Liu, Y Chen, Y Li, H Zhu, M Gao
https://ieeexplore.ieee.org/abstract/document/8359287
[20] An article web reference on “An Exploratory Analysis of Mobile Security Tools” by H Shahriar, MA Talukder, MS Islam
https://digitalcommons.kennesaw.edu/ccerp/2019/research/4/

BIBILIOGRAPHY

Dr.K.N.S Lakshmi Currently working as Professor from Department of Computer Science and Engineering at Sanketika Vidya Parishad Engineering College, affiliated to Andhra University, accredited by NAAC. Madam is currently working as Head of The Department, Published Papers in Various National & International Journals. Her Subjects of interests are Machine Learning, Data Mining & Warehousing.

Amith Prasad persuing his 2nd year, Master of Computer Applications in Sanketika Vidya Parishad Engineering College, affiliated to Andhra University, accredited by NAAC.With his interest in machine learning algorithms and as a part of academic project, he used Android Malware Detection Using Genetic Algorithm and ANN modal. To predict and detect the malware android applications using the extracted features like android premissions. A completely developed project along with code has been submitted for Andhra University as an Academic Project. In completion of his MCA.