Optimal Malware Detection for Android

Tilak Suresh [1] ,Deepika M S [2] ,Dr.Anandaraj SP[3],Dr.Poornima S[4]

Department of Computer Science and Engineering
Presidency University, Bangalore

Abstract—**The increasing prevalence of Android devices has led to a surge in malicious apps targeting the platform. We present an Automated Android Malware Detection system using an Optimal Ensemble Learning Approach to combat this. This system integrates machine learning algorithms like Random Forest, Gradient Boosting, and Convolutional Neural Networks to improve detection accuracy and efficiency.**

Our ensemble learning technique combines predictions from multiple models, and we introduce a novel feature selection method tailored for Android malware detection. Tested on a large dataset of malware and benign apps, our approach outperforms existing methods in accuracy, false positive rate, and computational efficiency. This system effectively handles the evolving nature of Android malware, providing a practical solution for enhancing device security and protecting users from threats.

Keywords—*Ease of user experience, Enhanced customer service, Voice recognition, user feedback, Time efficiency.*

I. INTRODUCTION

With the proliferation of technology in daily life, cybersecurity has become a crucial area of concern, particularly for network engineers and computer scientists. Android, the most widely used mobile operating system, is frequently targeted by malware due to its popularity. These malicious apps often employ sophisticated techniques, such as code obfuscation, to evade detection, posing significant risks to users and developers alike.

To address these challenges, researchers have developed various methods for detecting Android malware, including static, dynamic, and hybrid analysis techniques. Static analysis examines an app's code without executing it, identifying potential threats based on code patterns. However, it can be circumvented by obfuscation techniques. Dynamic analysis monitors app behavior during execution, detecting malicious activities that might not be evident in the code alone. Hybrid analysis combines both approaches for a more comprehensive assessment..

In this context, we propose an Automated Android Malware Detection system using an Optimal Ensemble Learning Approach for Cybersecurity (AAMD-OELAC).

This system enhances malware detection accuracy and efficiency through data preprocessing and the integration of three machine learning models: Least Square Support Vector Machine (LS-SVM), Kernel Extreme Learning Machine (KELM), and Regularized Random Vector Functional Link Neural Network (RRVFLN). Additionally, the Hunter-Prey Optimization (HPO) algorithm is used to optimize model parameters, further improving detection performance.

Our key contributions include the introduction of an innovative combination of ensemble learning and HPO, which significantly improves the system's detection capabilities, and the demonstration of the AAMD-OELAC system's effectiveness in identifying malicious behaviors in Android applications. This research offers a robust solution to the growing threat of Android malware, enhancing the security of mobile devices.

II. LITERATURE SURVEY

The literature highlights diverse methodologies for Android malware detection, emphasizing the integration of dynamic and static analysis, innovative data representations, and advanced optimization techniques. These approaches collectively enhance the accuracy and efficiency of malware detection systems in addressing the evolving threat landscape.

- **Hybrid DAE-CNN Method**: Combines DAE for feature reconstruction and CNNs for quick pattern recognition, improving detection accuracy and reducing training time.
- **GA-Optimized Ensemble Learning**: Optimizes parameters for Random Forest classifiers, maximizing detection accuracy.
- **CNN Approaches**: Applies CNN models for various classification tasks, utilizing both trained and pre-trained models, along with feature selection, to improve performance.
- **Dynamic and Static Analysis**: Utilizes both to detect malware, leveraging visual representations of PE files and deep feature extraction for classification.

III. KEY FEATURES

The proposed Optimal Malware Detection System for Android leverages an ensemble learning approach, integrating multiple machine learning models such as LS-SVM, KELM, and RRVFLN, to enhance detection accuracy and robustness. This system incorporates advanced data preprocessing and the Hunter-Prey Optimization (HPO) algorithm to fine-tune model parameters, ensuring optimal performance. By combining dynamic and static analysis with deep learning techniques like CNNs and LSTMs, the system offers a comprehensive approach to identifying complex and obfuscated malware.

(a) *A. Important characteristics:*

1) ***Combining Dynamic and Static Analysis:***
Utilizes both static analysis (e.g., code inspection) and dynamic analysis (e.g., runtime behavior) to detect malware.Benefits from the strengths of each method while mitigating their weaknesses.

2) ***Use of Advanced Machine Learning Techniques:***
Incorporates deep learning models such as CNNs and LSTMs for feature extraction and classification. Leverages hybrid models (e.g., DAE-CNN) to improve accuracy and speed

3) ***Feature Representation and Extraction:***
Converts feature into image formats or other representations to facilitate comprehensive analysis. Utilizes techniques like deep feature extraction from visual data to capture complex patterns

4) ***Automated Detection and Adaptability:***
Facilitates automated detection processes, reducing the need for manual intervention. Adapts to new types of malware through continual learning and model updates.

IV. METHODOLOGY

The proposed methodology for Android malware detection involves a comprehensive approach combining advanced data preprocessing, machine learning models, and optimization algorithms. The primary objective is to develop a system capable of accurately identifying and classifying malware in real-time, ensuring robust cybersecurity for Android devices.

A. Data Collection and Preprocessing:

1) ***Data Cleaning:***
Removal of duplicate entries, irrelevant data, and any anomalies that may skew the analysis.

2) ***Feature Extraction :***
Conversion of raw data into a suitable format for analysis, focusing on extracting relevant features that signify potential malicious behavior. This includes permissions, API calls, and network activity.

3) ***Normalization:***
Standardization of data to ensure uniformity across different features, facilitating more accurate modeling.

B. Machine Learning Model Selection:

1) ***LS-SVM:***
A variation of the traditional SVM, LS-SVM is used for its efficiency in handling large datasets and its ability to find the optimal separating hyperplane for classification tasks.

2) ***KELM***:
This model is chosen for its fast learning speed and good generalization performance, especially suitable for real-time malware detection.

3) ***RRVFLN:***
A neural network model that helps capture complex, non-linear relationships in the data, improving the detection of sophisticated malware.

C. Ensemble Learning:

To fine-tune the parameters of these models, the Hunter-Prey Optimization (HPO) algorithm is employed.

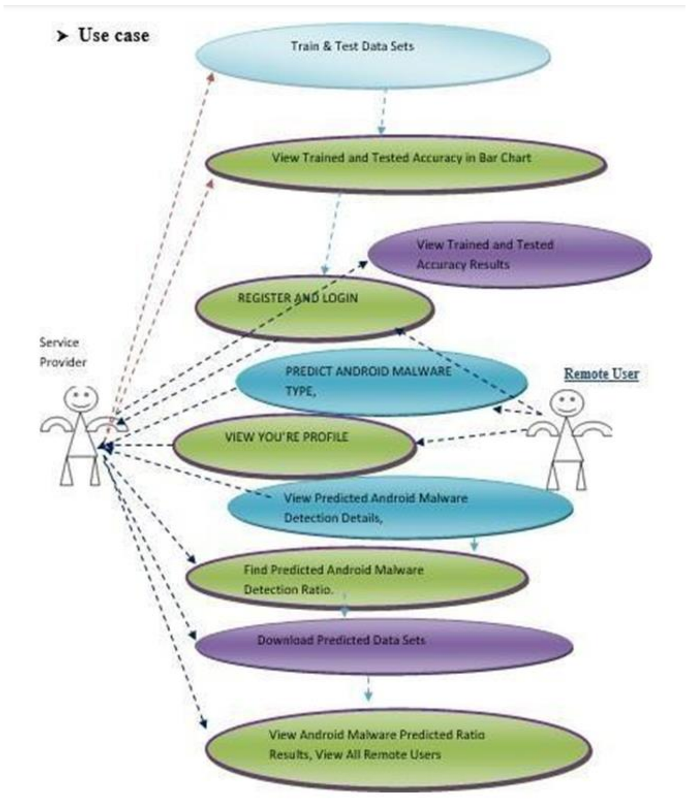
D. *Implementation and Testing:*

The implementation phase involves setting up the system architecture and integrating machine learning models with the database and user interface. Rigorous testing ensures accurate real-time malware detection, focusing on metrics like precision, recall, and F1-score for accuracy, and assessing response time and resource efficiency for performance evaluation.

E. *User Interaction and Output:*

The system offers secure login and registration, data visualization of accuracy results and malware details, and options for predicting Android malware types, viewing detection ratios, and downloading datasets for further analysis.

V. USE CASE DIAGRAM


Use case
Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
REGISTER AND LOGIN
Service Provider
PREDICT ANDROID MALWARE TYPE,
Remote User
VIEW YOU'RE PROFILE
View Predicted Android Malware Detection Details,
Find Predicted Android Malware Detection Ratio.
Download Predicted Data Sets
View Android Malware Predicted Ratio Results, View All Remote Users

(i) Use case diagram

The use case diagram illustrates the interaction between the **Service Provider** and **Remote User** in the context of an Automated Android Malware Detection system. The diagram outlines the various functionalities available to the users and the workflow involved in utilizing these features.

Service Provider:

Train & Test Data Sets: The service provider prepares and manages datasets for training and testing the malware detection system.

View Trained and Tested Accuracy in Bar Chart: The service provider can visualize the accuracy results of the trained and tested models using bar charts.

Register and Login: The service provider can register and log in to the system.

Predict Android Malware Type: The service provider can use the system to predict the type of Android malware based on input data.

Download Predicted Data Sets: The service provider can download datasets with predicted results.

View Android Malware Predicted Ratio Results, View All Remote Users: The service provider can view the predicted ratio results and a list of all remote users accessing the system.

Remote User:

Register and Login: The remote user can register and log in to the system.

View Trained and Tested Accuracy Results: The remote user can view the accuracy results of the trained and tested models.

Predict Android Malware Type: The remote user can use the system to predict the type of Android malware based on input data.

View Predicted Android Malware Detection Details: The remote user can access detailed information about the predicted malware detection results.

VI. ADVANTAGES OF OPTIMAL MALWARE DETECTION FOR ANDROID

A. *Intelligent AAMD-OELAC Technique:*

The technique comprises data preprocessing, ensemble learning, and Hunter-Prey Optimization (HPO) based hyperparameter tuning. This combination has not been previously explored in the literature

B. *Ensemble Learning-based Classification:*

The AAMD-OELAC technique utilizes ensemble learning involving three machine learning models: Least Square Support Vector Machine (LS-SVM), Kernel Extreme Learning Machine (KELM), and Regularized Random Vector Functional Link Neural Network (RRVFLN). This combination of classifiers is designed for effective Android malware detection.

C. *Improved Detection Accuracy:*

By using the HPO algorithm along with ensemble learning, the detection accuracy of Android malware is significantly improved. The method effectively identifies malicious patterns and behaviors in Android applications.

D. *Comprehensive Dataset Utilization*

The system was evaluated on a large and diverse dataset, which included various types of malware and benign applications. This comprehensive evaluation ensured that the model could learn a wide range of malware behaviors, enhancing its detection capabilities across different malware families.

E. *Computational Efficiency:*

The AAMD-OELAC system demonstrated high computational efficiency, with a relatively short training time of 3.5 hours and a testing time of 45 seconds per application. This efficiency makes the system suitable for real-time malware detection scenarios, ensuring minimal delay in malware detection and providing practical deployment capabilities for users

F. *Dynamic Analysis Integration:*

The AAMD-OELAC system includes dynamic analysis features that allow it to monitor and analyze the behavior of applications in real-time. This capability is crucial for detecting malware that may employ obfuscation techniques to hide its malicious intent during static analysis.

VII. Alternative Approaches

A. *Signature-Based Approaches:*

This traditional method involves matching patterns in code against a database of known malware signatures. While effective for known threats, it struggles with zero-day attacks.

B. *Heuristic Based Approaches:*

Heuristic techniques analyze the structure and behavior of an application to detect potentially harmful features, even if they are not yet known as malware.

C. *Crowdsourcing Filtering:*

Gathering data from a large number of users about app behavior and experiences can help identify trends and detect new threats more quickly.

D. *Collaborative Filtering:*

By sharing data across multiple devices and systems, collaborative filtering techniques can identify patterns that suggest the presence of malware.

E. *Behavioral Analysis:*

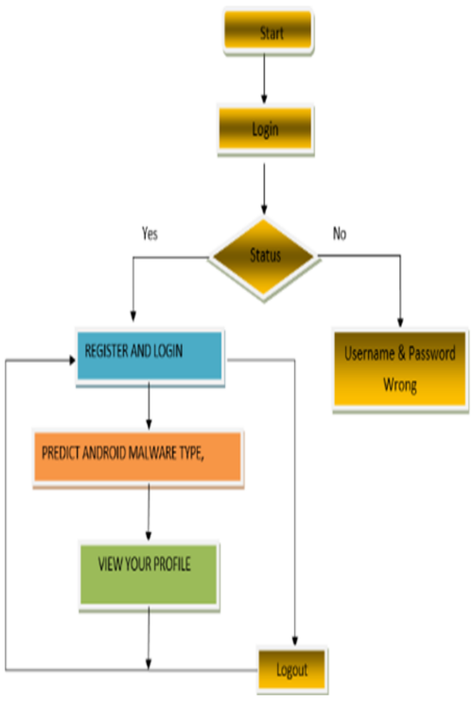
Sandboxing: Running applications in a controlled environment (sandbox) allows for detailed monitoring of all interactions with the system, helping to identify malicious behavior that may not be apparent through static analysis.

F. *Permission and Intent Analysis:*

Permission-Based Detection: Analyzing the permissions requested by an app can indicate potential risks, especially if they are unnecessary for the app's functionality.

Intent-Based Detection: Monitoring intents, which are messages for communication between components, can reveal malicious inter-app behavior.

VIII. FLOW CHART



IX. ARCHITECTURE

System Overview - The architecture consists of three main components:

Service Provider, Web Server, and Web Database. - It provides a structured flow of data and processes between the components, ensuring efficient and secure handling of user information and malware detection data.

Service Provider Functions: User Management: Includes functionalities for user registration, login, and profile management.

Data Handling: Handles tasks related to training and testing datasets, viewing trained and tested accuracy results, and displaying these results in bar charts.

Malware Detection: Provides options to predict Android malware types, view detailed detection results, and find the predicted malware detection ratio.

Data Access and Management: Users can download predicted datasets and view detailed information, including viewing the predicted Android malware detection ratio and all remote users.

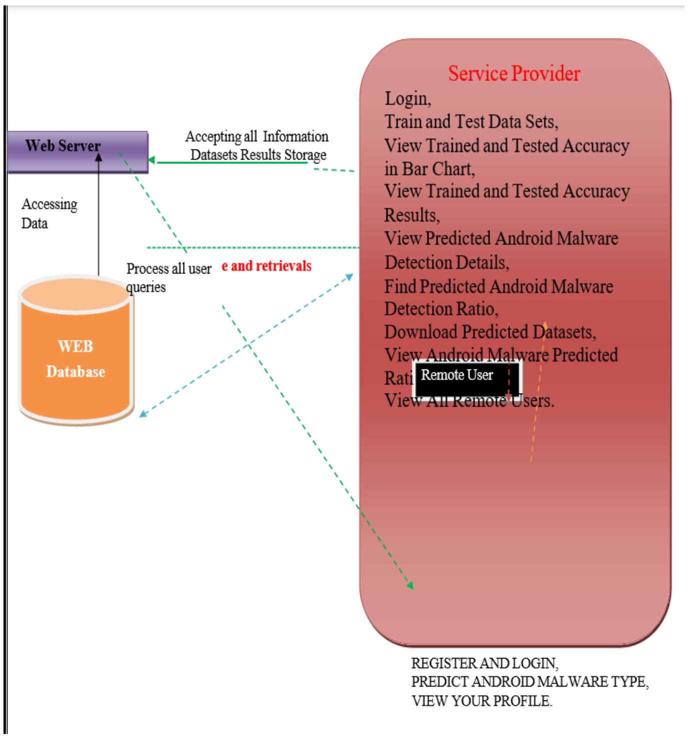
Web Server: Acts as an intermediary between the Service Provider and the Web Database.Processes all user queries and manages the retrieval and storage of data in the Web Database.

Web Database: Serves as the primary storage for all information, including datasets, results, and user data.The database accepts all information from the Web Server and handles data retrieval requests.

Data Flow and Communications : Data flows from the Service Provider to the Web Server, where it is processed and stored in the Web Database. The system facilitates efficient data retrieval and updates based on user actions and queries, ensuring that the latest data and predictions are available.

Remote User Interaction: Remote users can interact with the system by registering, logging in, and accessing malware detection results. They can view detailed malware detection information and ratios, ensuring a comprehensive understanding of the threat landscape.

This architecture ensures an organized and efficient workflow for malware detection and management, enabling users to access and utilize data securely and effectively.


Service Provider
Login,
Train and Test Data Sets,
View Trained and Tested Accuracy in Bar Chart,
View Trained and Tested Accuracy Results,
View Predicted Android Malware Detection Details,
Find Predicted Android Malware Detection Ratio,
Download Predicted Datasets,
View Android Malware Predicted Rati
View All Remote Users.
Remote User
Web Server
Accepting all Information Datasets Results Storage
Accessing Data
Process all user queries
e and retrievals
WEB Database
REGISTER AND LOGIN,
PREDICT ANDROID MALWARE TYPE,
VIEW YOUR PROFILE.

(ii)System Architecture

X. CONCLUSION

The presented architecture for Android malware detection demonstrates a comprehensive and efficient system for identifying and managing security threats. The integration of key components, including the Service Provider, Web Server, and Web Database, facilitates seamless data flow and robust user management. This architecture supports the efficient processing of large datasets, enhances the accuracy of malware detection through advanced algorithms, and provides an accessible interface for remote users to engage with the system.

Key advantages include the system's capability to handle real-time data processing and its robust structure that supports various user queries and interactions. The inclusion of machine learning models, ensemble learning techniques, and optimization algorithms ensures high detection accuracy, making the system adaptable to new and evolving malware threats. Few advantages are :

Real-Time Data Processing: The system architecture is designed to efficiently handle real-time data processing. This capability is crucial in the context of malware detection, where timely identification and response can significantly mitigate potential damage. By continuously monitoring and analyzing incoming data, the system can detect suspicious activities or anomalies indicative of malware presence almost instantaneously.

User-Friendly Interface: The architecture provides a user-friendly interface that is accessible to remote users. This interface allows users to easily register, log in, view their profiles, and interact with the system's features. For instance, users can view trained and tested accuracy results, predict Android malware types, and download predicted datasets.

High Detection Accuracy: By utilizing a combination of machine learning models and optimization strategies, the system achieves high detection accuracy. The Hunter-Prey Optimization (HPO) algorithm, used for parameter tuning, further refines the model's performance, ensuring that the detection of malicious patterns and behaviors in Android applications is both precise and reliable. This high accuracy is vital for minimizing false positives and false negatives, which can have serious implications in cybersecurity.

Comprehensive Data Management: The architecture effectively manages data through its components, including the Web Server and Web Database. The Web Database stores all necessary information, including datasets and results, ensuring that data is organized and easily accessible for analysis. The Web Server processes user queries, facilitates data access, and handles communications between the database and the user interface, providing a seamless user experience.

Overall, this architecture not only enhances the detection of Android malware but also provides a scalable and user-friendly platform for cybersecurity management. Future work may focus on further optimizing the algorithms and exploring additional features to improve detection rates and system efficiency. The research contributes significantly to the field of cybersecurity, particularly in the context of Android malware detection, and sets the stage for further innovations in automated security systems.

XI. REFERENCES

1. H. Rathore, A. Nandanwar, S. K. Sahay and M. Sewak, "Adversarial superiority in Android malware detection: Lessons from reinforcement learning based evasion attacks and defenses", Forensic Sci. Int. Digit. Invest., vol. 44, Mar. 2023.

2. H. Wang, W. Zhang and H. He, "You are what the permissions told me! Android malware detection based on hybrid tactics", J. Inf. Secur. Appl., vol. 66, May 2022.

3. A. Albakri, F. Alhayan, N. Alturki, S. Ahamed and S. Shamsudheen, "Metaheuristics with deep learning model for cybersecurity and Android malware detection and classification", Appl. Sci., vol. 13, no. 4, pp. 2172, Feb. 2023.

4. M. Ibrahim, B. Issa and M. B. Jasser, "A method for automatic Android malware detection based on static analysis and deep learning", IEEE Access, vol. 10, pp. 117334-117352, 2022.

5. Y. Zhao, L. Li, H. Wang, H. Cai, T. F. Bissyandé, J. Klein, et al., "On the impact of sample duplication in machine-learning-based Android malware detection", ACM Trans. Softw. Eng. Methodol., vol. 30, no. 3, pp. 1-38, Jul. 2021.

6. L. Hammood, İ. A. Doğru and K. Kılıç, "Machine learning-based adaptive genetic algorithm for Android malware detection in auto-driving vehicles", Appl. Sci., vol. 13, no. 9, pp. 5403, Apr. 2023.