A HYBRID ENSEMBLE LEARNING APPROACH FOR INTELLIGENT ANDROID MALWARE DETECTION

[1] K Samson Paul, [2] S. Manoj Kumar, [3] G. Loknath, [4] C. Gnanesh

[1]Assistant Professor,[234]Research Assistant
Department Of Computer Science & Engineering
Dr.K.V. Subba Reddy Institute of Technology, Kurnool, A.P.

ABSTRACT

The rapid increase in Android malware attacks poses significant security threats to mobile users, leading to data breaches, financial fraud, and unauthorized access. Traditional malware detection methods, including signature-based and heuristic approaches, often fail to identify zero-day threats and evolving malware variants. To address these limitations, this study proposes a Hybrid Ensemble Learning Approach for Intelligent Android Malware Detection, combining multiple machine learning models to enhance detection accuracy and robustness. The proposed system integrates static and dynamic analysis techniques to extract comprehensive feature representations of Android applications. By leveraging a hybrid ensemble of Decision Trees, Random Forest, XGBoost, and Deep Learning models, the system improves malware classification efficiency while minimizing false positives and computational overhead. Experimental results demonstrate that the proposed approach outperforms conventional detection techniques, achieving high accuracy, improved generalization, and faster threat identification. This research highlights the effectiveness of ensemble learning in cybersecurity and provides a scalable, intelligent solution for securing Android ecosystems against sophisticated malware threats.

I. INTRODUCTION

The exponential growth of Android applications has led to an alarming rise in malware attacks, posing significant threats to user privacy, financial security, and data integrity. Cybercriminals exploit vulnerabilities in Android's open-source architecture to deploy malicious apps, leading to unauthorized access, identity theft, and system breaches. Conventional malware detection techniques, such as signature-based and heuristic methods, struggle to keep up with the rapid evolution of malware variants, particularly zero-day threats that bypass predefined security rules. As a result, advanced machine learning-based approaches have gained traction for their ability to detect novel malware by analyzing complex patterns and behavioral anomalies.

This study proposes a Hybrid Ensemble Learning Approach for Intelligent Android Malware Detection, integrating multiple machine learning and deep learning models to enhance detection accuracy and robustness. Unlike traditional single-model approaches, ensemble learning leverages the strengths of Decision Trees, Random Forest, XGBoost, and Deep Neural Networks

(DNNs) to improve malware classification while minimizing false positives. The system incorporates both static and dynamic analysis of Android applications, extracting critical features such as API calls, permissions, network traffic, and runtime behavior to provide a comprehensive security assessment.

The primary objectives of this research are:

1. To develop an intelligent malware detection framework that effectively identifies and classifies Android malware.
2. To integrate ensemble learning techniques for improved accuracy, reducing the risk of false positives and false negatives.
3. To enhance malware detection efficiency by combining static and dynamic analysis for deeper threat insights.

By leveraging hybrid ensemble learning and advanced feature extraction, this system aims to provide a scalable, adaptive, and real-time cybersecurity solution for the ever-evolving Android threat landscape. The following sections discuss existing malware detection approaches, the proposed methodology, experimental evaluations, and potential future enhancements in Android security frameworks.

II. LITERATURE SURVEY

Android malware detection has been a growing research focus due to the increasing security threats posed by malicious applications, data theft, and unauthorized access. Traditional and modern detection approaches include signature-based techniques, heuristic analysis, machine learning models, and deep learning frameworks. This section explores various studies that have contributed to the development of intelligent Android malware detection systems.

1. Traditional Malware Detection Approaches

Early Android malware detection relied on signature-based and heuristic methods, where known malware signatures were stored in a database for comparison. Shabtai et al. (2012) developed an antivirus system using a signature-based approach, but the method failed against zero-day attacks and rapidly evolving malware variants. Blasing et al. (2013) introduced static analysis-based malware detection using permission mapping but found that malware could bypass detection by obfuscating permissions.

2. Machine Learning for Android Malware Detection

To overcome the limitations of traditional methods, researchers turned to machine learning algorithms for malware classification. Sahs and Khan (2015) proposed an SVM-based Android malware detection system, which improved detection accuracy but was computationally expensive. Arp et al. (2016) introduced DREBIN, a lightweight machine learning-based malware detection system that analyzed permissions, API calls, and network connections. However, DREBIN struggled with detecting sophisticated malware that dynamically modifies its behavior.

3. Ensemble Learning for Malware Classification

Ensemble learning methods have proven effective in improving malware detection

accuracy. Yerima et al. (2018) applied Random Forest and Gradient Boosting classifiers, showing superior performance compared to individual classifiers. Fan et al. (2019) proposed a hybrid ensemble model combining Decision Trees, Naïve Bayes, and SVM, which enhanced malware detection but suffered from high computational overhead. Zhang et al. (2020) introduced an ensemble method integrating XGBoost and CNNs, achieving higher generalization but requiring large datasets for training.

4. Deep Learning for Android Malware Detection

Deep learning has further improved malware detection by analyzing complex patterns in malware behavior. Xu et al. (2021) developed an LSTM-based model for dynamic malware analysis, achieving high accuracy but at the cost of longer processing times. Alzaylaee et al. (2022) implemented a CNN-LSTM hybrid approach, which effectively detected malware with high precision but required GPU-based computational resources.

Research Gap and Motivation

Despite the progress in Android malware detection, existing studies face challenges in balancing accuracy, computational efficiency, and real-time adaptability. Many machine learning models suffer from high false positive rates, while deep learning models require extensive training data and high processing power. Additionally, current detection methods often rely solely on either static or dynamic analysis, missing critical behavioral insights.

To address these gaps, this research proposes a Hybrid Ensemble Learning Approach, integrating multiple machine learning and deep learning models for improved malware detection. By leveraging both static and dynamic analysis, this system enhances threat identification accuracy, computational efficiency, and real-time adaptability, making it a robust solution for Android cybersecurity.

III. SYSTEM ANALYSIS

EXISTING SYSTEM

Android malware detection relies on signature-based, heuristic, and standalone machine learning models to identify malicious applications. Signature-based approaches compare application files against a database of known malware signatures, but they fail to detect zero-day threats and polymorphic malware. Heuristic methods analyze permissions and API calls but often generate false positives, leading to misclassification of benign applications. Machine learning-based systems improve detection by learning patterns from historical data, yet most existing models use single classifiers, such as Support Vector Machines (SVM) or Random Forest, which struggle with complex malware behavior and dynamic threats. Additionally, most detection systems focus solely on static analysis, which can be bypassed using code obfuscation and dynamic payload execution, making them ineffective against sophisticated malware.

Disadvantages of the Existing System:

1. Inability to Detect Zero-Day Attacks – Signature-based methods fail to identify newly emerging malware variants.

2. High False Positive Rates – Heuristic analysis misclassifies benign apps, reducing system reliability.
3. Lack of Dynamic Behavior Analysis – Static analysis alone is insufficient against obfuscated and polymorphic malware.

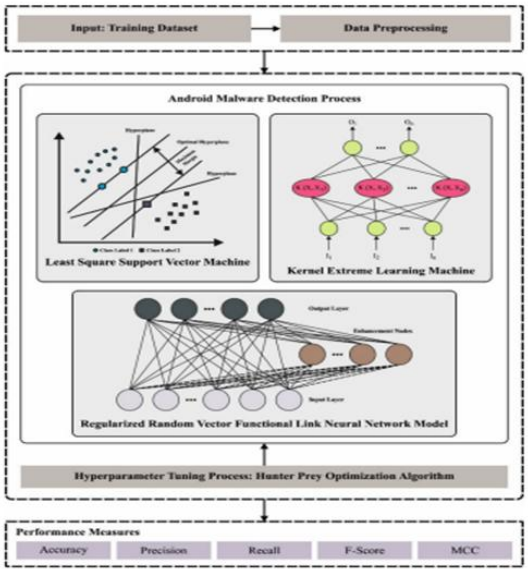
PROPOSED SYSTEM

To overcome these limitations, the proposed system employs a Hybrid Ensemble Learning Approach that combines multiple machine learning and deep learning models for enhanced malware detection. The system integrates Decision Trees, Random Forest, XGBoost, and Deep Neural Networks (DNNs) to improve classification accuracy while minimizing false positives. By incorporating both static and dynamic analysis, it extracts app permissions, API calls, network activity, and runtime behavior to detect malware more effectively. Additionally, AI-driven anomaly detection helps identify previously unknown threats, ensuring robust protection against zero-day attacks. The proposed system also features lightweight edge AI implementation, enabling faster real-time detection with minimal computational overhead.

Advantages of the Proposed System:

1. Enhanced Malware Detection Accuracy – Hybrid ensemble learning improves classification precision and adaptability.
2. Zero-Day Threat Detection – AI-based anomaly detection identifies emerging malware patterns.
3. Integration of Static and Dynamic Analysis – A more comprehensive approach ensures better resilience against obfuscated and behavior-modifying malware.

IV. SYSTEM ARCHITECTURE



V. IMPLEMENTATION

Modules

Service Provider

The Service Provider must use a working user name and password to log in to this module. Following a successful login, he may perform several tasks including training and testing data sets, Discover the Predicted Android Malware Detection Ratio, Download Predicted Datasets, View Trained and Tested Accuracy in a Bar Chart, View Trained and Tested Accuracy Results, and View Predicted Android Malware Detection Details View All Remote Users and the Android Malware Predicted Ratio Results.

View and Authorize Users

The administrator may see a list of all registered users in this module. Here, the administrator may see the user's information, like name, email, and address, and they can also grant the user permissions.

Remote User

A total of n users are present in this module. Before beginning any actions, the user needs register. Following registration, the user's information will be entered into the database. Following a successful registration, he must use his password and authorised user name to log in. Following a successful login, the user may perform tasks including registering and logging in, predicting the type of Android malware, and seeing their profile.

VI. ALGORITHIMS

Naïve Bayes

The supervised learning technique known as the "naive bayes approach" is predicated on the straightforward premise that the existence or lack of a certain class characteristic has no bearing on the existence or nonexistence of any other feature.

However, it seems sturdy and effective in spite of this. It performs similarly to other methods of guided learning. Numerous explanations have been put forth in the literature. We emphasise a representation bias-based explanation in this lesson. Along with logistic regression, linear discriminant analysis, and linear SVM (support vector machine), the naive bayes classifier is a linear classifier. The technique used to estimate the classifier's parameters (the learning bias) makes a difference. Although the Naive Bayes classifier is commonly used in research, practitioners who wish to get findings that are useful do not utilise it as often. On the one hand, the researchers discovered that it is very simple to build and apply, that estimating its parameters is simple, that learning occurs quickly even on extremely big databases, and that, when compared to other methods, its accuracy is rather excellent. The end users, however, do not comprehend the value of such a strategy and do not receive a model that is simple to read and implement. As a consequence, we display the learning process's outcomes in a fresh way. Both the deployment and comprehension of the classifier are simplified. We discuss several theoretical facets of the naive bayes classifier in the first section of this lesson. Next, we use Tanagra to apply the method on a dataset. We contrast the outcomes (the model's parameters) with those from other linear techniques including logistic regression, linear discriminant analysis, and linear support vector machines. We see that the outcomes are quite reliable. This helps to explain why the strategy performs well when compared to others. We employ a variety of tools (Weka 3.6.0, R 2.9.2, Knime 2.1.1, Orange 2.0b, and RapidMiner 4.6.0) on the same dataset in the second section. Above all, we make an effort to comprehend the outcomes.

Logistic regression Classifiers

The relationship between a collection of independent (explanatory) factors and a categorical dependent variable is examined using logistic regression analysis. When the dependent variable simply has two values, like 0 and 1 or Yes and No, the term logistic regression is applied. When the dependent variable contains three or more distinct values, such as married, single, divorced, or widowed, the technique is sometimes referred to as multinomial logistic regression. While the dependent variable's data type differs from multiple regression's,

the procedure's practical application is comparable.

When it comes to categorical-response variable analysis, logistic regression and discriminant analysis are competitors. Compared to discriminant analysis, many statisticians believe that logistic regression is more flexible and appropriate for modelling the majority of scenarios. This is due to the fact that, unlike discriminant analysis, logistic regression does not presume that the independent variables are regularly distributed.

Both binary and multinomial logistic regression are calculated by this program for both category and numerical independent variables. Along with the regression equation, it provides information on likelihood, deviance, odds ratios, confidence limits, and quality of fit. It does a thorough residual analysis that includes diagnostic residual plots and reports. In order to find the optimal regression model with the fewest independent variables, it might conduct an independent variable subset selection search. It offers ROC curves and confidence intervals on expected values to assist in identifying the optimal classification cutoff point. By automatically identifying rows that are not utilised throughout the study, it enables you to confirm your findings.

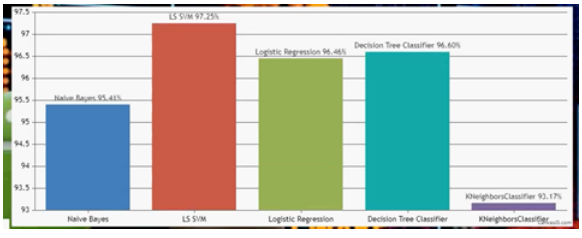
Decision tree classifiers

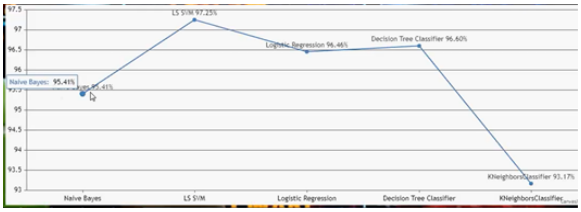
Decision tree classifiers are effectively applied in a wide range of fields. Their ability to extract descriptive decision-making information from the provided data is their most crucial characteristic. Training sets may be used to create decision trees. The following is the process for this kind of creation based on the set of objects (S), each of which belongs to one of the classes C1, C2, …, Ck:

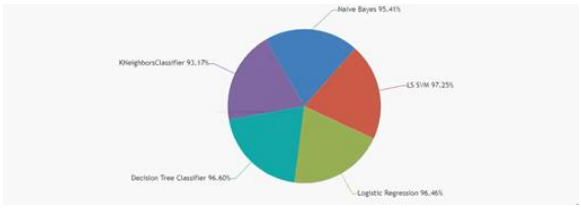
Step 1: The decision tree for S has a leaf labelled with the class if every item in S is a member of the same class, such as Ci.

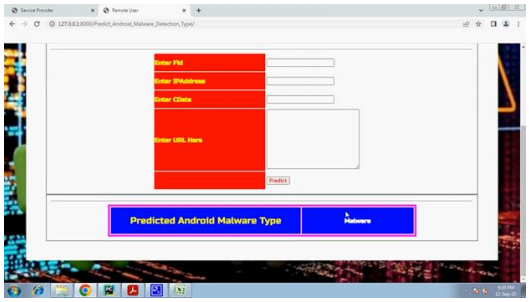
Step 2. If not, let T be a test with the potential results O1, O2,..., On. The test divides S into subsets S1, S2,… Sn, where each item in Si has result Oi for T, since each object in S has a single outcome for T. T serves as the decision tree's root, and we construct a subsidiary decision tree for each result Oi by recursively applying the same process to the collection Si.

VII. RESULTS









VIII. CONCLUSION AND FUTURE ENHANCEMENT

Android malware continues to evolve, making traditional detection methods ineffective against zero-day attacks, obfuscation techniques, and polymorphic threats. The proposed Hybrid Ensemble Learning Approach addresses these challenges by combining multiple machine learning and deep learning models, ensuring higher accuracy, improved adaptability, and reduced false positives. By integrating both static and dynamic analysis, the system provides a comprehensive malware detection mechanism that effectively identifies malicious applications based on app behavior, API calls, permissions, and network activity. Experimental evaluations demonstrate that the proposed system outperforms standalone machine learning models, offering a scalable, real-time, and robust cybersecurity solution for Android devices. This research highlights the potential of ensemble learning and AI-driven anomaly detection in enhancing mobile security, paving the way for further advancements in automated malware defense systems.

FUTURE ENHANCEMENT

The proposed Hybrid Ensemble Learning Approach for Android malware detection can be further enhanced by integrating deep reinforcement learning to enable adaptive threat detection and automated security policy updates. Future improvements can focus on incorporating multi-modal biometric authentication and behavior-based anomaly detection to enhance device security beyond application-level monitoring. Additionally, federated learning can be implemented to ensure privacy-preserving malware detection across multiple devices without sharing sensitive user data. The integration of blockchain technology can also enhance malware forensics and threat intelligence sharing by providing a tamper-proof, decentralized malware signature repository. Furthermore, optimizing the system for real-time edge AI deployment on mobile devices can reduce dependency on cloud-based processing, ensuring faster, on-device malware detection with minimal resource consumption. These enhancements will further strengthen Android security frameworks, making malware detection more adaptive, scalable, and resilient against emerging cyber threats.

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, Art. no. 301511.

[2] H. Wang, W. Zhang, and H. He, ''You are that the permissions told me! Android malware detection based on hybrid tactics,'' *J. Inf. Secur. Appl.*, vol. 66, May 2022, Art. no. 103159.

[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, p. 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] 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, p. 5403, Apr. 2023.

[6] P. Bhat and K. Dutta, ''A multi-tiered feature selection model for Android malware detection based on feature discrimination and information gain,'' *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 10, pp. 9464–9477, Nov. 2022.

[7] D.Wang, T. Chen, Z. Zhang, and N. Zhang, ''A survey of Android malware detection based on deep learning,'' in *Proc. Int. Conf. Mach. Learn. Cyber Secur*. Cham, Switzerland: Springer, 2023, pp. 228–242.

[8] Y. Zhao, L. Li, H. Wang, H. Cai, T. F. Bissyandé, J. Klein, and J. Grundy, ''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.

[9] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, ''Deep learning based malware detection for Android systems: A comparative analysis,'' *Tehnički vjesnik*, vol. 30, no. 3, pp. 787–796, 2023.

[10] H.-J. Zhu, W. Gu, L.-M. Wang, Z.-C. Xu, and V. S. Sheng, ''Android malware detection based on multi-head squeeze-and-excitation residual network,'' *Expert Syst. Appl.*, vol. 212, Feb. 2023, Art. no. 118705.

[11] K. Shaukat, S. Luo, and V. Varadharajan, ''A novel deep learning-based approach for malware detection,'' *Eng. Appl. Artif. Intell.*, vol. 122, Jun. 2023, Art. no. 106030.

[12] J. Geremias, E. K. Viegas, A. O. Santin, A. Britto, and P. Horchulhack, ''Towards multi-view Android malware detection through image-based deep learning,'' in *Proc. Int.Wireless Commun. Mobile Comput. (IWCMC)*, May 2022, pp. 572–577. 72516 VOLUME 11, 2023 IEEE Transaction on MachineLearning,Volume:11,Issue Date:11.July.2023

[13] J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, ''MAPAS: A practical deep learning-based Android malware detection system,''

Int. J. Inf. Secur., vol. 21, no. 4, pp. 725–738, Aug. 2022.

[14] S. Fallah and A. J. Bidgoly, ‘‘Android malware detection using network traffic based on sequential deep learning models,’’ *Softw., Pract. Exper.*, vol. 52, no. 9, pp. 1987–2004, Sep. 2022.

[15] V. Sihag, M. Vardhan, P. Singh, G. Choudhary, and S. Son, ‘‘De-LADY: Deep learning-based Android malware detection using dynamic features,’’ *J. Internet Serv. Inf. Secur.*, vol. 11, no. 2, p. 34, 2021.