Unified Approach for Android Malware Detection: Feature Combination and Ensemble Classifier

Dr.V. Jyothsna[1*], Kavya Priya Dasari[2], Sravani Inuguru[3], Venkat Bharath Reddy Gowni[4], Jaya Teja Reddy Kudumula[5], K Srilakshmi[6]

[1] Associate Prof., Dept of IT, Sree Vidyanikethan Engineering College, Tirupathi, India
*jyothsna1684@gmail.com
[2,3,4,5] UG Scholar, Dept of IT, Sree Vidyanikethan Engineering College, Tirupathi, India
[6]Lecturer, Sri Padmavathi Women's Degree & PG College, Tirupathi, India
dasaripriya653@gmail.com

Abstract. As the smartphone market has expanded enormously, particularly in the Android environment, the necessity for robust anti-malware security has become increasingly apparent. By harnessing the power of machine learning and large datasets, this model demonstrates exceptional capabilities in identifying subtle malicious trends. This study delves into the importance of coexistence in malware detection.This methodology analyzes coexistence patterns crucial for effective malware detection and develops a dataset that integrates these key features. Addressing data imbalance using the SMOTE technique enhances dataset representativeness. Feature selection via Extra Trees Classifier optimizes pattern detection, improving classification precision. This methodology significantly enhances cybersecurity in dynamic digital settings, detecting Android malware with high accuracy. The voting classifier (with MLP, CatBoost, and XGBoost) trained on the above dataset achieved 98% accuracy. This work represents a substantial advancement in efficient and adaptable malware detection techniques tailored for the evolving Android ecosystem.

Keywords: Android, machine learning, malware, anomaly detection, feature enhancement.

1 Introduction

The ever-evolving digital landscape poses a persistent challenge to cybersecurity defenses due to the dynamic nature of malware. This study delves into analyzing evolving malware trends and the need for adaptive defensive strategies.

Malware, ranging from viruses to trojans, continually advances to compromise systems and networks, highlighting the critical need for robust security measures. With DataProt reporting 560,000 new malware types [1] daily and cybercrime costs projected to exceed $10.5 trillion by 2025[2], the urgency for enhanced security is evident. The Android ecosystem, with over 3.43 million apps on the Google Play Store [3], faces security challenges exacerbated by third-party app markets lacking stringent monitoring.

K. R. Madhavi et al. (eds.), *Proceedings of the International Conference on Computational Innovations and Emerging Trends (ICCIET 2024)*, Advances in Computer Science Research 112,
https://doi.org/10.2991/978-94-6463-471-6_47

Statistical insights underscore the vast scope of the Android app ecosystem and the prevalence of malicious applications [4], emphasizing the need for innovative security approaches. This research aims to develop adaptive ML models capable of identifying subtle malicious trends and enhancing detection accuracy to safeguard user data and privacy.

Key enhancements in this paper include dataset refinement, addressing class im-balance using SMOTE, enriching datasets with dynamic features from frequent pat-terns, and employing feature selection with the Extra Trees Classifier. The study also evaluates the performance of diverse ML models individually and as an ensem-ble in a Voting Classifier, achieving impressive accuracy by incorporating binary coexistence features derived from permission attributes.

2 Literature Review

M. E. Z. N. Kambar et al. [6] highlighted the proliferation of mobile applications due to widespread smartphone usage and high-speed Internet access. Despite security enhancements in iOS and Android, there is a persistent rise in incursions targeting mobile applications. Experts employ various techniques for detecting mobile malware, either preemptively or through network traffic analysis, to mitigate associated risks. This document offers insights into different types of mobile malware and their implications.

A. Alzubi et al. [7] introduce a novel approach to Android malware detection by combining the Harris Hawks Optimization (HHO) algorithm with a Support Vector Machine (SVM) classifier. This method optimizes feature weighting and SVM hyperparameters, enhancing detection performance. Through rigorous testing on CIC-Manal2017 datasets, the proposed technique demonstrates effectiveness in evaluating feature importance and exploring correlations with malware attack types.

M. Li, Y. Wu, et al. [8] Because Android is open-source, it is increasingly vulnerable to malware attacks, so efficient detection is essential. Modern advancements heavily rely on machine learning, particularly in the classification stage of Android malware detection. Examining the feature selection mode based on wrappers is vital since specific conventional ranking-based algorithms fail to consider feature relationships. Wrapper-based methods, however, can take a long time to analyze different valid feature subsets when working with a large number of Android features.

Yadav P et al. [9] This study presents a two-step deep learning method that uses picture representations of Android DEX files for Android malware identification and categorization. The system uses EfficientNetB0 to extract information from color photos of malware. As the meta-level classifier, logistic regression, random forest, and linear SVM algorithms serve as base-level classifiers, a stacking classifier can achieve 100% accuracy in binary classification and 92.9% in 5-class classification. The proposed

strategy outperforms 26 state-of-the-art pre-trained CNN models and large-scale learning classifiers on all performance metrics.

N. Sharma and A. L. Sangal et al. [12] This research addresses the surge in smartphone Android malware threats, employing machine learning with the CICInvesAndMal2019 dataset. Using Android permissions and intents as features, Principal Component Analysis aids in feature selection. Among the machine learning models tested, Random Forest proves the most effective, achieving a 99.7% success rate in binary classification and 97.30% for the ransomware category in category classification.

Y. Kanchhal and S. Murugaanandam et al. [13] For more than a decade, Android has remained the dominant mobile operating system worldwide. However, its widespread usage has also attracted the attention of cybercriminals and malware developers, posing significant security threats. Malware presents a universal challenge across all operating systems, including Android. With Android's support for app installations from sources beyond the Google Play Store, there is an increased risk of malware infiltration alongside legitimate apps.

Jyothsna V. et al. [15] Applications for the Internet's technological advancements can be found in many facets of daily life, including banking, public networking, online commerce, and electronic trading. These services' exponential expansion raises network traffic, increasing the possibility of network attacks. Scholars have put up several approaches to deal with problems from decades ago. The research clarifies that machine learning, artificial neural networks, and meta-heuristic approaches have been highly regarded for their ability to handle security assaults. These approaches rely on the characteristics of the requests made to extract knowledge. It has been noted that the network traffic volume is growing exponentially, displaying diverse behavior and feature value deviation. As a result, transaction associability and feature values must be considered.

Jyothsna V. et al. [16] Using neuroimaging data, deep neural networks can accurately estimate the chronological age of healthy persons. Predicted brain age has the potential to be used as a biomarker to detect illnesses associated with aging. Thus, the suggested method (SVM) uses a Convolutional Neural Network (CNN), a deep learning cascade network, and a Support Vector Machine (SVM), a machine learning algorithm. These algorithms have identified three types of patients through brain MRI scan training: Normal (i.e., not impacted by any disease), Alzheimer's disease (AD), and Mild Cognitive Impairment (MCI) from sorted pictures and established ages. The MRI image dataset is trained for age estimation and classification using CNN and SVM methods.

3 Proposed Model

The methodology encompasses data collection from Drebin, Malgenome, and CIC_MALDROID2020 datasets, culminating in the creation of the "lev2" dataset using SMOTE for class imbalance handling. Feature extraction adopts a coexistence-based

strategy, while feature selection optimizes efficiency through the Extra Trees classifier. A novel ensemble technique, the Voting Classifier, integrates MLP, CatBoost, and XGBoost models, utilizing a "soft voting" approach for enhanced resilience. Comprehensive assessment metrics ensure a thorough evaluation of the ensemble model's performance, contributing to a nuanced defense against evolving Android malware threats. Figure 1 illustrates the system architecture overview.

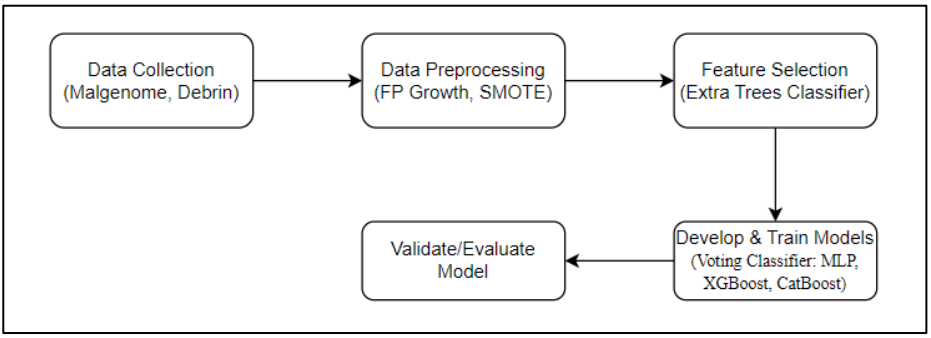

Fig. 1. Methodology Workflow

3.1 Techniques and Algorithms

Smote. SMOTE is a vital technique in addressing class imbalance by generating synthetic instances, ensuring fair representation for the minority class. It achieves a balanced dataset distribution by creating artificial examples interpolated between existing minority instances, reducing bias during model training towards the majority class. Integrating SMOTE in dataset preprocessing enhances subsequent analyses, improving overall model resilience and efficacy.

Pseudocode.

function SMOTE(sample, N1, K):
    $\text{synthetic}_{\text{samples}} = [\square]$
    for i = 1 to N1:
        $\text{random}_{\text{samples}} = \text{randomly}_{\text{select(sample)}}$
        $\text{neighbors} = \text{find}_{\text{K}_{\text{nearest}_{\text{neighbors}}}}$
    $(\text{random}_{\text{sample}}, \text{sample}, \text{K})\text{synthetic}_{\text{sample}} = \text{random}_{\text{sample}} + \text{random}_{\text{uniform}(\square)} *$
$(\text{randomly}_{\text{select(neighbors)}} - \text{random}_{\text{samples}})$
        $\text{synthetic}_{\text{samples}}.\text{append}(\text{synthetic}_{\text{samples}})$
    return $\text{synthetic}_{\text{samples}}$

Frequent Itemset Mining using FP-Growth. The FP-Growth algorithm plays a crucial role in mining frequent itemsets to enrich the dataset and unveil significant patterns. It efficiently identifies recurring item sets from transactional data, aiding in understanding associations among features. The balanced data obtained through SMOTE lays a robust groundwork for FP-Growth to extract frequent item sets. This process entails

identifying frequently occurring feature combinations, offering valuable insights for further analysis and modeling endeavors.

Pseudocode.

```
dataframe_new = empty DataFrame
for each item_set in enumeration of w:
  conditions = None
  for each feature in item_set:
    conditions = (dataframe[feature] == 1) if conditions is None else conditions & (
dataframe[feature] == 1)
  dataframe_new['coexistence_' + str(index)] = 1 if conditions else 0
dataframe_new['class'] = dataframe['class']
```

Extra Trees Classifier. The Extra Trees classifier, a variant of decision tree algorithms like Random Forest, excels in handling high-dimensional data and conducting feature selection. It utilizes a meta-estimator that fits randomized decision trees on dataset subsets, leveraging averaging to enhance accuracy and prevent overfitting. Unlike Random Forest, Extra Trees selects the best split randomly from feature subsets, adding a layer of randomization while optimizing performance.

Voting Classifier. In this study, the Voting Classifier model was employed as an innovative approach to Android malware detection, harnessing the capabilities of Multi-Layer Perceptron (MLP), XG Boost, and CatBoost classifiers.

The concept of "soft voting" in ensemble techniques entails that the final prediction is not solely based on a majority vote but on the weighted average of predicted probabilities from each base model. This means that the Voting Classifier considers the confidence or certainty of predictions from each model and combines them accordingly. This approach is particularly advantageous when working with models that provide probability estimates, such as MLP, CatBoost, and XGBoost. By integrating these models within the Voting Classifier, the ensemble model aims to capitalize on their unique strengths and patterns. MLP, known for its neural network architecture, is adept at capturing intricate data relationships, while CatBoost, a gradient-boosting algorithm, excels in handling categorical features and mitigating overfitting.

3.2 Performance Evaluation Measures

Performance evaluation measures are essential for evaluating intrusion detection models. Metrics like Precision, Recall, and F1 Score offer valuable insights into the model's performance. The confusion matrix provides a comprehensive overview of the model's predictions compared to the actual ground truth**.**

In binary classification, True Positives (TP) are instances correctly identified as positive (e.g., correctly identifying malware), False Positives (FP) are instances incorrectly identified as positive, True Negatives (TN) are instances correctly identified as negative, and False Negatives (FN) are instances incorrectly identified as negative. These metrics provide a comprehensive assessment of the model's performance, highlighting

its ability to distinguish between positive and negative classes accurately.

Accuracy. Used to evaluate the classification model's overall correctness. Although class imbalances in the dataset may impact on this metric's applicability, it offers a broad indication of model performance.

Recall (Sensitivity or True Positive Rate). The proportion of accurate positive predictions among all actual positives, measures the model's ability to detect positive instances.

Precision. It determines the proportion of accurate positive predictions among all positive predictions. Precision quantifies the degree to which the model's positive predictions are accurate.

F1 Score. The harmonic means of precision and recall. It provides a balanced measure of the model's precision and recall.

$$Accuracy = (TruePositives + TrueNegatives)/TotalPredictions$$
$$Recall = TruePositives / (TruePositives + FalseNegatives)$$
$$Precision = TruePositives / (TruePositives + False\ Positives)$$
$$F1\ Score = 2 * (Precision * Recall) / (Precision + Recall)$$

An ideal intrusion detection model should effectively balance recollection and precision., as it minimizes the number of false positives while still being able to detect most instances of unauthorized transactions.

4 Proposed Model

4.1 Dataset

The Malgenome dataset encompasses 3,798 unique programs observed from 2012 to the end of 2015. It comprises 1,260 programs attributed to 49 distinct malware families and 2,538 clean applications. Initially, 181 features were extracted from this dataset, including 109 permissions and 72 APIs. Three distinct subsets were delineated from the Malgenome dataset: “API + Permission combination”, “only Permission Malgenome”, and “only API Malgenome”. This dataset contains a total of 3799 rows and 182 columns.

4.2 Data Preprocessing

Class Imbalance Handling Strategy. In the above Api + Permission combination subset of the Malgenome dataset, to address the class imbalance, synthetic instances of the minority class (class 1) were created using SMOTE to match the number of instances in the majority class (class 0). This balancing strategy ensured that machine learning

models are not biased toward the majority class and could be generalized effectively to both classes.

This approach contributes to enhancing the performance of machine learning models and improving the accuracy of malware detection. After addressing the class imbalance, the number of instances for both class 0 and class 1 is 1260.

Coexistence features. After mitigating class imbalance using the SMOTE technique, the next step involved extracting frequent item sets using the FP-Growth algorithm. This process identified recurring patterns in the dataset, crucial for understanding co-existence relationships among features. Subsequently, a new dataset named "lev2" was created, capturing the bi, tri, and ternary features derived from the extracted frequent pattern item sets.

Feature Selection and Model Training. The top-performing features from the lev2 dataset are selected using the Extra Trees Classifier, resulting in 795 relevant features out of 6487. This strategic integration enhances malware detection models' efficacy by prioritizing key data aspects.

Three individual models are trained on these selected features: MLP with a single hidden layer of 100 neurons and 1000 iterations, CatBoost with 100 iterations, depth of 8, and a learning rate of 0.1 using the MultiClass loss function, and XGBoost with 100 estimators, maximum depth of 8, and a learning rate of 0.1.

Furthermore, a Voting Classifier is developed by combining MLP, CatBoost, and XGBoost using a soft voting strategy based on confidence levels. This ensemble model leverages each model's strengths to improve overall predictive performance.

Performance Comparison. The ensemble method demonstrates positive synergistic effects, improving overall malware detection performance compared to individual algorithms. Figure 2 displays the voting classifier's confusion matrix. Table 2 and Figure 3 shows the classifier overall performance comparison.

Fig. 4 The malware detection methodology begins with a dataset of 2520 entries. Several classifiers, including MLP, CatBoost, and a Voting Classifier, are employed for a thorough analysis. The dataset is partitioned into folds of roughly comparable size, with each fold containing approximately 504 records in 5-fold cross-validation. During each iteration, 2016 records are used for training, while onefold (504 records) is reserved for testing. Figure 4 depicts the average accuracy across all folds and presents accuracy scores for each fold individually. The malware detection system employed a probability threshold of 0.5 to determine the presence or absence of malware in instances.

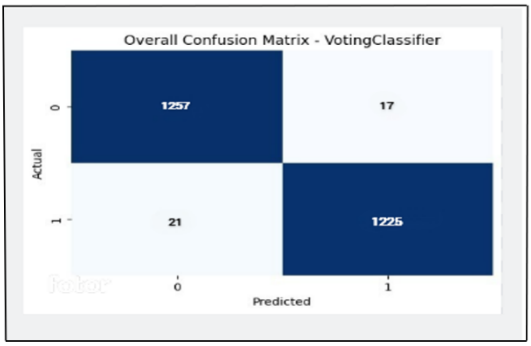

Fig. 2. Overall Confusion Matrix for Voting Classifier

Model	MLP	CatBoost	XG Boost	Voting Classifier
True Positive	1245	1220	1205	1225
False Positive	20	32	50	17
True Negative	1227	1232	1210	1257
False Negative	28	48	55	21
Precision	98.7%	97.4%	98.6%	98.6%
Recall/Sensitivity	97.7%	96.21%	97.7%	98.3%
Accuracy	98%	97.3%	97.8%	98.4%
F1 score	98.04%	96.80%	97%	98.4%

Table 1. Performance Matrix

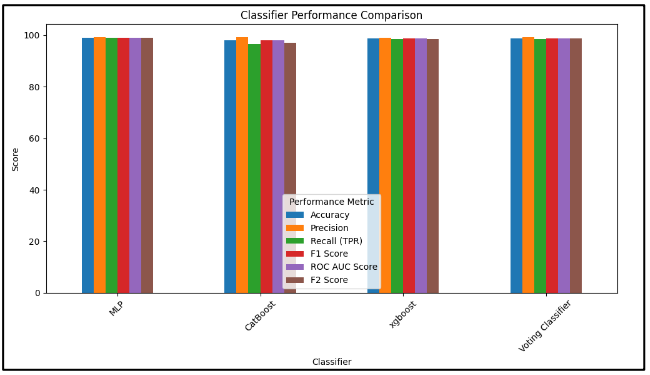

Fig. 3. Overall Performance Comparison of Classifiers

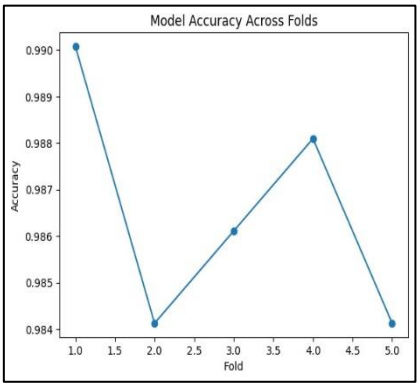

Fig. 4. Model Accuracy Across Folds

5 Conclusion

The proposed model marks a significant leap in Android malware detection, showcasing heightened accuracy and adaptability within the Android ecosystem. Utilizing extensive datasets like Drebin and Malgenome, coupled with advanced algorithms and optimization techniques such as SMOTE for handling imbalanced data, has markedly improved the model's effectiveness.

A standout feature of the model is its incorporation of ensemble techniques, notably the voting classifier, which capitalizes on the strengths of MLP, XGBoost, and CatBoost, resulting in superior accuracy in identifying malware. Moreover, the model's versatility within the dynamic Android environment is a pivotal advantage, providing a sturdy defense against evolving malicious strategies. Through the amalgamation of feature-rich datasets and ensemble strategies, the approach not only delivers heightened accuracy but also lays a robust foundation for ongoing advancements in Android malware detection.

5.1 Future Work

Future work for this paper involves evaluating the coexistence approach with dynamic features and expanding the analysis of API and permission combinations to cover a wider range of malware datasets. Additionally, exploring advanced optimization techniques for machine learning models in Android malware detection is a priority. This dual focus aims to maintain the approach's effectiveness while adapting it to diverse malware scenarios, ensuring robustness and accuracy in identifying malicious trends across various datasets. Furthermore, a shift towards dynamic malware analysis and utilizing versatile datasets beyond API and permission combinations are recommended for a more comprehensive analysis of Android malware. These avenues of future work aim to contribute to the continuous advancement of Android malware detection techniques and enhance cybersecurity measures for mobile users globally.

References

[1] B. Jovanovic, A Not-So-Common Cold: Malware Statistics in 2023, May 2023, [online] Available: https://dataprot.net/statistics/malware-statistics/.

[2] Cyber Security Statistics The Ultimate List Of Stats Data & Trends for 2023, May 2023, [online] Available: https://purplesec.us/resources/cyber-security-statistics/.

[3] M. Iqbal. (2022). App Download Data. Accessed: Oct. 30, 2022.[Online].Available: https://www.businessofapps.com/d
ata/app-statistics/

[4] K. Allix, T. Bissyand, Q. Jarome, J. Klein, R. State, and Y. L. Traon, ''Empirical assessment of machine learning-based malware detectors for android,'' Empirical Softw. Eng., vol. 21, pp. 183–211, Jun. 2016

[5] Esraa Odat; Qussai M. Yaseen, "A Novel MachineLearning Approach for Android Malware Detection Based on the Co-Existence of Features" Feb 2023, doi: 10.1109/ACCESS.2023.3244656

[6] M. E. Z. N. Kambar, A. Esmaeilzadeh, Y. Kim, and K. Taghva, ''A survey on mobile malware detection methods using machine learning,'' in *Proc. IEEE 12th Annu.Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2022, pp. 0215–0221, doi: 10.1109/CCWC54503.2022.9720753.

[7] O. A. Alzubi, J. A. Alzubi, A. M. Al-Zoubi, M. A. Hassonah, and U. Kose, ''An efficient malware detection approach with feature weighting based on Harrishawks optimization,'' *Cluster Comput.*, vol. 25, no. 4, pp. 2369–2387, Aug. 2022, doi: 10.1007/s10586-021-03459-1.

[8] Y. Wu, M. Li, Q. Zeng, T. Yang, J. Wang, Z. Fang, and L. Cheng, ''DroidRL: Feature selection for Android malware detection with reinforcement learning,'' *Comput. Secure.*, vol. 128, May 2023, Art. no. 103126, doi: 10.1016/j.cose.2023.103126.

[9] Avanija, J., K. E. Kumar, Ch Usha Kumari, G. Naga Jyothi, K. Srujan Raju, and K. Reddy Madhavi. "Enhancing Network Forensic and Deep Learning Mechanism for Internet of Things Networks." (2023).

[10] 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, doi: 10.1007/s10207-022-00579-6.

[11] 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, doi: 10.1002/spe.3112.

[12] N. Sharma and A. L. Sangal, ''Machine learning approaches for analysing static features in Android malware detection,'' in *Proc. 3rd Int. Conf. Secure Cyber Comput. Commun. (ICSCCC)*, Jalandhar, India, May 2023, pp. 93–96, doi: 10.1109/ICSCCC58608.2023.10176445.

[13] Kumar, DNS Ravi, N. Praveen, Hari Hara P. Kumar, Ganganagunta Srinivas, and M. V. Raju. "Acoustic Feedback Noise Cancellation in Hearing Aids Using Adaptive Filter." International Journal of Integrated Engineering 14, no. 7 (2022): 45-55.

[14] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, ''Malware detection in Android systems with traditional machine learning models: A survey,'' in Proc. Int. Congr. Human-Comput. Interact., Optim. Robotic Appl. (HORA), Jun. 2020, pp. 1–8, doi: 10.1109/HORA49412.2020.9152840.

[15] Jyothsna, V., Prasad, M. K., GopiChand, G., & Bhavani, D. D. (2022). DLMHS: Flow-based intrusion detection system using deep learning neural network and meta-heuristic scale. International Journal Of Communication Systems, 35(10).

[16] Jyothsna, V., Raja, D. K., Kumar, G. H., & Chnadra, E. D. (2022). A novel manifold approach for intrusion detection system (MHIDS). Gongcheng Kexue Yu Jishu/Advanced Engineering Science, 54(02).