International Journal of Science and Research Archive
eISSN: 2582-8185
Cross Ref DOI: 10.30574/ijsra
Journal homepage: https://ijsra.net/

(RESEARCH ARTICLE)

Check for updates

Android malware detection using machine learning

Sindhu K.P. [1,*], Kumar Siddamallappa U [1] and Anusha Jajur J [2]

[1] *Department of studies in Computer Applications (MCA), Davangere University, Shivagangothri, Davangere-577007, Karnataka, India.*
[2] *Department of studies in Computer Science, Davangere University, Shivagangothri, Davangere-577007, Karnataka, India.*

International Journal of Science and Research Archive, 2025, 16(03), 643–652

Publication history: Received on 02 August 2025; revised on 07 September 2025; accepted on 10 September 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.16.3.2583

Abstract

Because Android is open-source and widely used, it has emerged as the most popular mobile operating system. But because of its widespread use, fraudsters who disseminate malicious software have found it to be a prime target. Although they work well for known threats, traditional signature-based malware detection techniques miss novel or unidentified variations, increasing the danger of zero-day assaults. This paper suggests a machine-learning-based detection framework improved using Genetic Algorithm (GA) for optimal feature selection in order to get over these restrictions. By selecting the most discriminative and pertinent characteristics from big feature sets, the GA lowers dimensionality without sacrificing accuracy. Machine learning classifiers like Support Vector Machines (SVM) and Neural Networks (NN) are then trained using these improved features. According to experimental data, the suggested method reduces computing complexity by almost half while achieving detection accuracy of over 92.56%. This study shows a scalable, effective, and lightweight malware detection system.

Keywords: Malware Detection; GA; SVM; NN

1. Introduction

The Android operating system, which controls the majority of the smartphone market, has become the most widely used mobile platform globally. It's cheap, wide range of applications, and open-source nature are major factors in its success. Users can download millions of apps from the official Google Play Store and many more third-party marketplaces. Although this transparency encourages creativity and accessibility, it also poses serious security risks. Cybercriminals take advantage of this situation by introducing harmful code into programs that appear to be genuine software. After being loaded, these apps have the potential to compromise private information including contacts, call logs, bank account details, and GPS location, or in extreme situations, take over the device entirely.

Despite Google's continuous efforts to enhance its built-in security features, including Google Play Protect and app vetting mechanisms, malicious apps still manage to bypass detection. This persistent threat highlights the limitations of traditional malware detection methods. Signature-based techniques, which rely on comparing suspicious applications with a database of known malware signatures, are particularly ineffective against polymorphic malware and zero-day attacks. As malware authors constantly develop new variants, maintaining and updating signature databases becomes resource-intensive and prone to false negatives.

To address these shortcomings, researchers and practitioners have turned towards machine learning (ML)-based approaches. Unlike signature-based methods, ML techniques can generalize patterns from previously unseen data and are capable of identifying new malware variants. Malware analysis in this context is typically classified into two

* Corresponding author: Sindhu K.P

categories both dynamic and static analysis. Without actually running the application, static analysis entails obtaining and analysing its features, including permissions, APIs, and code structures. In contrast, dynamic analysis uses a controlled environment to watch how a program behaves during runtime in order to identify malicious activity. Although each strategy has advantages, integrating them with machine learning improves the detection systems' overall accuracy and resilience.

However, one of the key challenges in applying ML to malware detection lies in handling large, high-dimensional feature sets. These datasets often contain redundant or irrelevant features, which increase computational complexity, slow down training, and can even degrade detection accuracy. Feature selection therefore becomes a critical step in building efficient ML-based malware detection systems.

Evolutionary algorithms like the Genetic Algorithm (GA) have drawn interest in this setting. GA is useful for finding optimal or nearly optimal feature subsets by searching vast solution spaces, and it is motivated by the principles of natural selection. The dimensionality of the dataset can be greatly decreased while keeping only the most discriminative features when GA is applied to feature selection in Android virus detection. This decrease improves ML classifiers' capacity for generalization while also speeding up training and testing.

A GA-based feature selection framework combined with ML classifiers like Support Vector Machines (SVM) and Neural Networks (NN) is proposed in this work. High malware detection accuracy, low computing costs, and a scalable solution that can successfully manage known and unknown malware threats are the objectives.

2. Literature review

In recent years, the issue of Android malware detection has garnered a lot of research interest, and many machine learning (ML) and optimization-based techniques have been investigated to improve detection efficiency and accuracy.

Mahendru et al. [1] introduced GA Droid, a framework that integrates Genetic Algorithm (GA) with Using machine learning to detect malware on Android. Their results demonstrated that GA effectively reduces feature dimensions while maintaining high accuracy, proving useful for large feature sets.

Mathematics (MDPI) et al. [2] evaluated GA-based feature selection compared with traditional methods. The findings showed that GA enhanced categorization precision in addition to significantly reduced computational complexity, making it practical for real-time detection systems.

Golrang et al. [3] proposed a multi-objective feature selection framework for Android malware detection. By applying GA for optimization, their system achieved improved accuracy while addressing trade-offs between performance and computational efficiency.

Enayat et al. [4] In their subsequent investigation of GA-based feature selection in malware detection,pointed out that optimized subsets improved Support Vector Machine (SVM) and Neural Network (NN) classifiers. Their research reaffirmed GA's contribution to enhancing feature relevancy.

ICACDS et al. [5] coupled ensemble learning classifiers with hybrid meta-heuristic feature selection algorithms. The hybrid strategy was more successful against zero-day threats because it showed improved generalization to unseen malware samples.

Hakim et al. [6] introduced GA-StackingMD, a GA-optimized stacking ensemble method. The study showed that ensembles trained on GA-optimized features significantly outperformed single classifiers regarding precision and robustness.

Wu et al. [7] reinforcement learning (RL) for feature selection in Android malware detection, proposing DroidRL. Despite not being GA-based, their findings demonstrated the promise of intelligent search-based optimization, confirming the usefulness of evolutionary techniques such as GA in this field.

Overall, the literature consistently shows that GA-based feature selection reduces dimensionality, accelerates classifier training, and maintains or improves detection accuracy. These findings strongly support the proposed work, which leverages GA and machine learning to provide an efficient, scalable malware detection framework.

3. Proposed method

3.1. Problem Statement

Android's open-source nature and global popularity make it a top priority for malware attacks. Despite Google Play Store's security measures, malicious applications still bypass detection and compromise user privacy. Traditional signature-based methods are limited, as they cannot detect unknown or zero-day malware variants. This creates a pressing need for intelligent, scalable, and accurate detection techniques. The challenge lies in reducing feature complexity while keeping the precision of detection high for real-world deployment.

3.2. Objective

- To design and implement an effective method for detecting Android malware using Genetic Algorithm (GA) for optimized feature selection.
- To cut down on training time without sacrificing excellent detection accuracy.
- To overcome the drawbacks of conventional signature-based techniques and enhance the identification of known and unknown malware threats.
- To evaluate multiple machine learning classifiers (SVM, Neural Networks) on GA-selected features for better adaptability.
- To help develop a strong and sophisticated framework for detecting Android malware that improves mobile security.

3.3. Existing Method

The current Malware detection for Android systems largely depend on signature-based techniques, where suspicious applications are matched against a database of known malware signatures. This method is straightforward and effective in identifying previously reported threats. However, it struggles with new and evolving malware variants that do not yet exist in the signature database. As malware families often mutate rapidly, these systems become less reliable over time. Moreover, their heavy dependence on continuous database updates makes them inefficient for large-scale or real-time detection scenarios.

Signature-based methods can only identify known threats and fail against zero-day or polymorphic malware. They require frequent updates to remain effective, which increases both time and resource costs. These techniques frequently result in false negatives, allowing harmful applications to bypass detectionFurthermore, they don't adapt well to the constantly changing Android malware landscape Consequently, their accuracy is limited, making them unsuitable for advanced mobile security need

3.4. Implementation

3.4.1. Data Flow Diagram of the system

A data framework's "stream" of information is represented graphically by an information stream outline (DFD). DFDs can also be used to visualize information handling (structured overview). On a DFD, information flows through an internal process from an external information source or an internal information store to an external information sink or an internal information store.

3.4.2. Level 0 Data stream chart

The cooperation between the framework and outside experts, who act as information sources and information sinks, is shown by a connection level or level 0 information stream chart. The connections between the framework and the outside world are shown on the connection chart, also known as the Level 0 DFD, just as far as information streams outside the framework limit. The connection chart shows the entire framework as a single process without providing any details about its internal relationships.

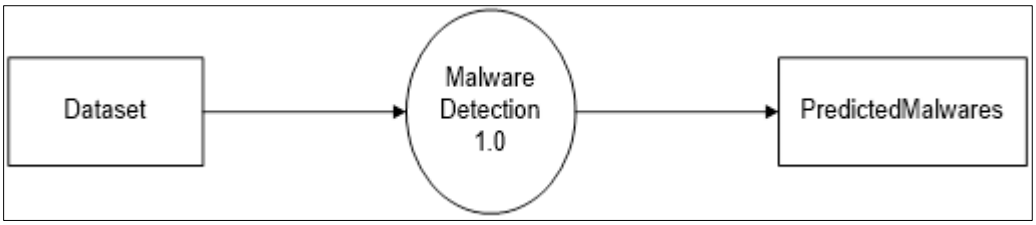

Figure 1 Level 0 Data stream chart

3.4.3. Level 1 Data flow diagram

The Level 1 DFD illustrates the system's division into smaller systems, or processes, each of which handles one or more data flows to or from an external agent and collectively supply the system's overall functionality. Additionally, it illustrates the data flow between the various components of the system and reveals internal data stores that are necessary for the system to function.

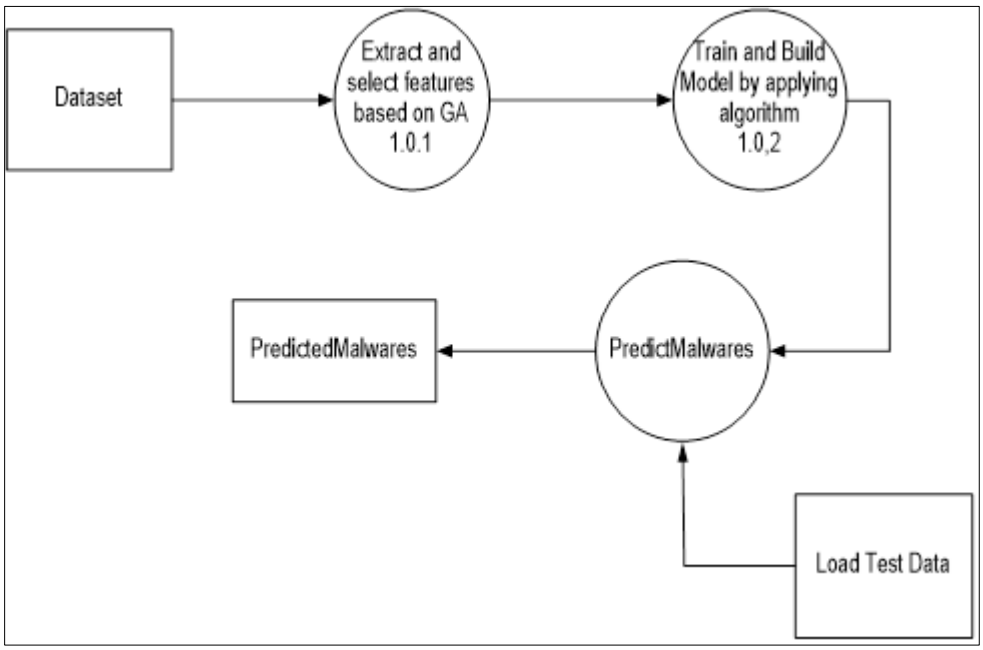

Figure 2 Level 1 Data flow diagram

3.4.4. System Architecture

The conceptual design that establishes a system's behavior and structure is called its architecture. A formal description of a system that is structured to facilitate inference about its structural characteristics is called an architectural description. It outlines the building blocks or components of the system and offers a blueprint for the development of systems and the procurement of goods that will cooperate to carry out the system as a whole.

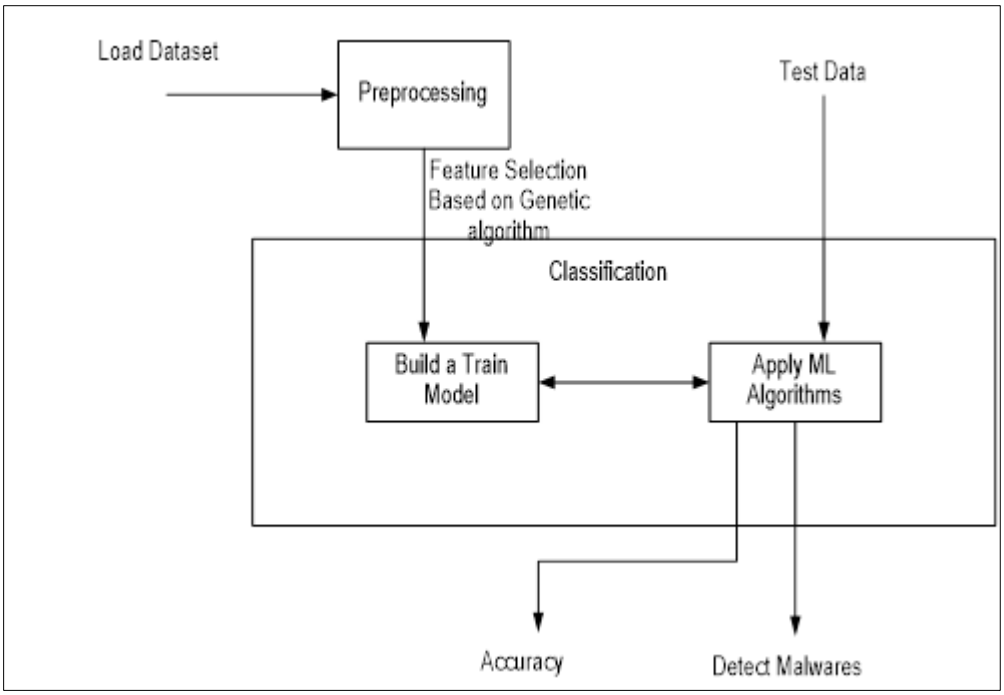

Figure 3 System Architecture

3.4.5. Algorithms

Genetic Algorithm

The following is a summary of the procedures involved in feature selection using genetic algorithms:

- Step 1: Set up the method utilizing feature subsets that are binary encoded so that a feature is represented by 1 in the chromosome if it is included and by 0 if it is excluded.
- Step 2: Launch the algorithm with a randomly generated initial population set.
- Step 3: Assign a fitness score based on the genetic algorithm's established fitness function.
- Step 4: Parent Selection: In order to generate the next generation of offspring, chromosomes with high fitness scores are prioritized over others.
- Step 5: Use the specified probability of crossover and mutation for the creation of offspring to perform crossover and mutation operations on the chosen parents.

Iteratively repeat Steps 3 through 5 until the population's fittest chromosome—that is, the ideal feature subset—is reached.

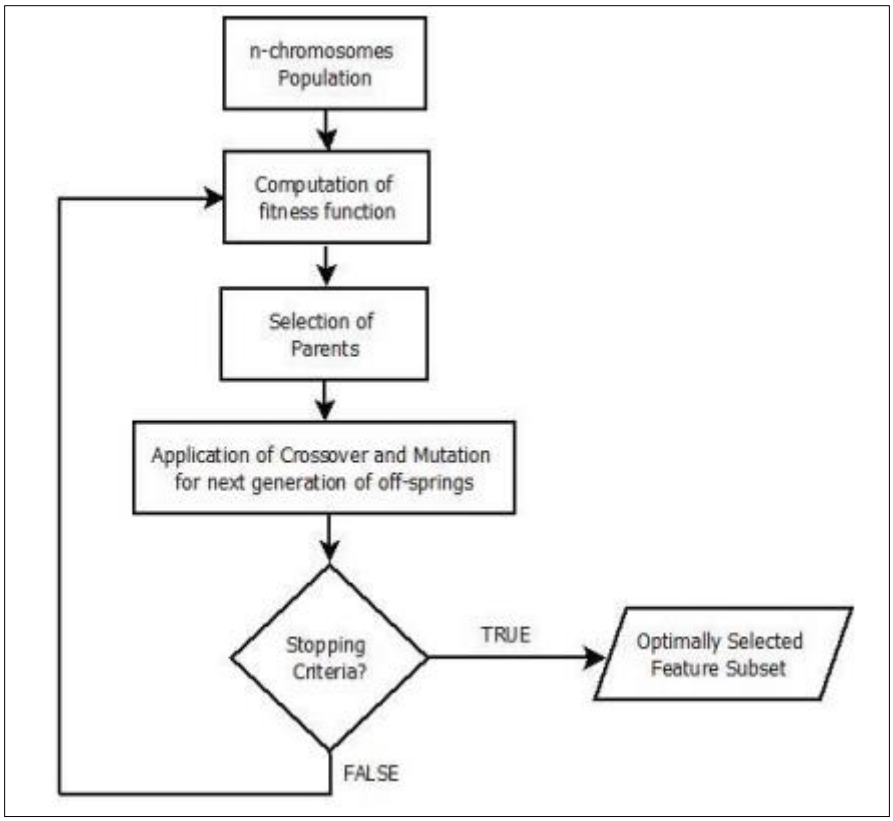

Figure 4 Feature Selection using Genetic Algorithm

3.5. Neural Network Steps

- To conduct a dot product with m features in input X, m weights are required.
- To conduct dot products with n hidden neurons in the hidden layer, you require n sets of weights (W1, W2, … Wn).
- To obtain the hidden output h, you need one hidden layer and n dot products: (h1, h2, …, hn)
- It then functions similarly to a single-layer perceptron, using hidden output h: (h1, h2, …, hn) as input data with n features. The final output, y_hat, is obtained by performing a dot product with one set of n weights (w1, w2, …, wn).

3.6. SVM Classifier Algorithm

SVM is a solution that relies on statistical learning. The mathematics of uncertainty is what statistics is. It seeks to learn and make choices based on information.

There are numerous hyperplanes that can be used to classify data when it is shown along the X and Y axes. However, deciding which is the best or right answer is a very challenging issue. SVM is used to remove this kind of issue.

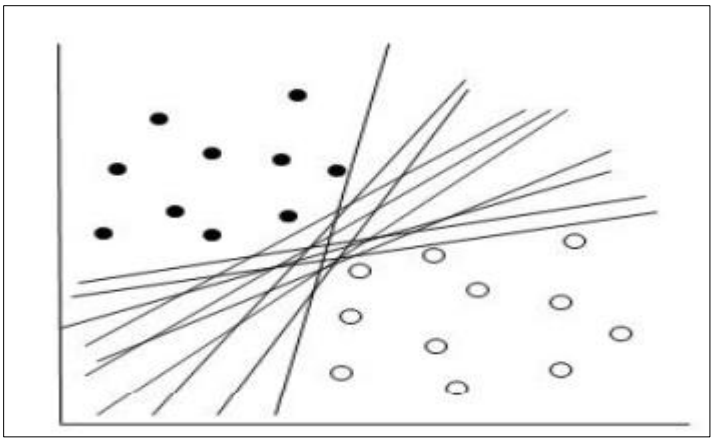

Figure 5 SVM Classifier Algorithm

Only the ones with the largest margin are selected from among the numerous linear classifiers or hyperplanes that divide the data. This is because a hyperplane used for categorization may be more similar to one data set than another. Thus, we employ the idea of hyper planes or maximum margin classifiers.

3.7. Algorithm: GA-based Feature Selection with Classifier Training

Fitness Function

$$\text{Fitness}(C_i) = \text{Accuracy}(C_i, D) \qquad (1)$$

Crossover

$$C_new = C_parent1 \oplus C_parent2 \qquad (2)$$

Feature Selection Objective

$$F* = \text{argmax}_{(F' \subseteq F)} \text{Acc}\big(\text{Classifier}(D, F')\big) \quad (3)$$

SVM Prediction

$$f(x) = \text{sign}(w \cdot x + b) \qquad (4)$$

Neural Network Hidden Layer

$$h(l) = \sigma\big(W(l)h(l-1) + b(l)\big) \qquad (5)$$

Evaluation Metrics

Accuracy

$$\text{Acc} = (TP + TN)/(TP + TN + FP + FN) \quad (6)$$

Precision

$$\text{Prec} = TP/(TP + FP) \qquad (7)$$

Recall

$$\text{Rec} = TP/(TP + FN) \qquad (8)$$

F1 Score

$$F1 = 2 \times (Prec \times Rec)/(Prec + Rec) \quad (9)$$

3.7.1. Output

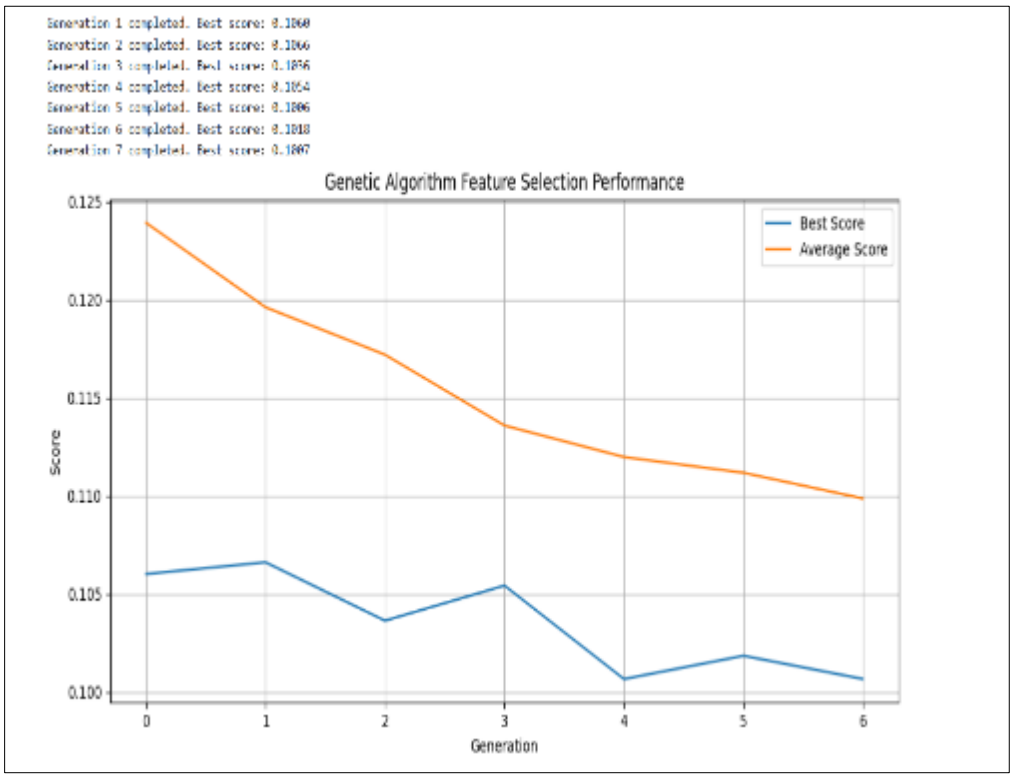

Figure 6 Genetic Algorithm Feature Selection Performance

The graph illustrates the performance of Genetic Algorithm (GA) in feature selection across successive generations. The orange line, representing the average score of the population, shows a steady decline, indicating that the overall quality of solutions improves as generation's progress. The blue line, representing the best score, fluctuates slightly due to crossover and mutation but gradually stabilizes around 0.1007, suggesting convergence towards an optimal feature subset. This trend demonstrates that GA effectively reduces feature dimensionality while maintaining performance, with the population steadily evolving towards better solutions and the best individual achieving consistent accuracy improvements.

Figure 7 Detection of Malware APK file

4. Conclusion

Due to the inefficiency of conventional signature-based detection methods against novel and zero-day variants, the growing incidence of Android malware presents a significant risk to mobile consumers. This experiment showed that the most important properties may be preserved while reducing the dimensionality of huge feature sets through the use of Genetic Algorithms (GA) for feature selection. Through the integration of GA with machine learning classifiers like Neural Networks (NN) and Support Vector Machines (SVM), the suggested method lowered computational complexity while achieving a detection accuracy of over 92.56%. According to the results, the improved feature subsets greatly reduce training time and enhance classifier performance, increasing the system's scalability and efficiency. In general, the suggested framework offers a malware detection system that is clever, dependable, and lightweight, and it can adjust to changing Android threats. Testing with bigger datasets, adding dynamic analysis, and expanding the framework with hybrid deep learning approaches for even more resilience are possible future improvements.

Future Work

Although the proposed framework using Genetic Algorithm (GA) and Machine Learning (ML) classifiers have demonstrated encouraging outcomes in detecting Android malware, there are several avenues for future enhancement. First, the system can be extended by incorporating larger and more diverse datasets to improve generalization across different malware families. Second, hybrid feature extraction techniques that combine static and dynamic analysis could be integrated to capture both code-level and behavioural characteristics of malicious applications. Third, the inclusion of advanced deep learning models Convolutional Neural Networks (CNNs), for example and Recurrent Neural Networks (RNNs), optimized with GA-selected features, may further enhance detection accuracy. Fourth, deploying the framework as a lightweight, real-time mobile application could provide on-device protection without significant performance overhead. Finally, adaptive learning and periodic retraining strategies can be explored to ensure the model remains effective against evolving malware patterns and zero-day attacks. These improvements will contribute to developing a more robust, scalable, and future-ready Android malware detection system.

Compliance with ethical standards

Disclosure of conflict of interest

All declares no conflict of interest.

References

[1] GADroid: A framework for malware detection from Android using Genetic Algorithm as feature selection approach. International Journal of Advanced Science and Technology, 29(5), 5532–5543. Mahindru, A., and Sangal, A. L. (2020).

[2] Machine Learning using Genetic AlgorithmBased Feature Selection for Android Malware Detection.(2021).2813 in Mathematics, 9(21). https://doi.org/10.3390/math9212813

[3] Golrang, A., Yayilgan, S. Y., and Elezaj, O. (2021). The multi-objective feature selection in Android malware detection system. In Intelligent Technologies and Applications (pp. 339–352). Springer. https://doi.org/10.1007/978-3-030-88004-0_26

[4] Enayat, S., Moizuddin, and Ghafir, S. (2022). Detecting Android malware with Genetic Algorithm-based feature selection. In Data Intelligence and Cognitive Informatics (pp. 335–345). Springer. https://doi.org/10.1007/978-3-030-97119-9_28

[5] Hybrid Methods for Android Malware Detection Meta-heuristic Selection of Features and Ensemble Learning Techniques. (2022). In Computing Developments and Data Sciences (pp. 151–163). Springer. https://doi.org/10.1007/978-981-16-3340-2_13

[6] ANNDroid: A framework for detecting Android malware that makes use of machine learning and feature selection algorithms. (2023). In Intelligent Systems Design and Applications (pp. 65–77). Springer. https://doi.org/10.1007/978-3-031-20615-6_7

[7] Hakim, S. B., et al. (2024). GA-StackingMD:A technique for detecting Android malware based on Genetic Algorithm optimized stacking. Applied Sciences, 13(4), 2629. https://doi.org/10.3390/app13042629

[8] Wu, Y., et al. (2022). DroidRL: Reinforcement learning-driven Choosing features for Android malware detection arXiv preprint arXiv:2203.02719. https://arxiv.org/abs/2203.02719

[9] Padmalatha, E., Venkata Krishna Reddy, M., Suvarna Kumari, T., and Kabeeruddin. (2023). using feature selection in conjunction with a hybrid Genetic Algorithm and Simulated Annealing (SVM and DBN) to detect Android malware. International Journal of Current and Emerging Computing Trends and Communication, 11(10), 1481–1487. https://doi.org/10.17762/ijritcc.v11i10.8698 ijritcc.org

[10] Anonymous. (2024). A hybrid approach to machine learning and Genetic Algorithm for malware detection. Journal of Advanced Data Mining and Intelligence — JADM. https://doi.org/10.22044/jadm.2024.13895.2503 jad.shahroodut.ac.ir

[11] Wang, L., Gao, Y., Gao, S., and Yong, X. (2021). Android malware detection using a novel feature selection technique based on a self-variant genetic algorithm. Symmetry, 13(7), Article 1290. https://doi.org/10.3390/sym13071290 DergiParkOUCI

[12] Wang, J., Jing, Q., Gao, J., and Qiu, X. (2020). SEdroid: A powerful malware detector for Android that employs selective ensemble learning. In IEEE Wireless Communications and Networking Conference (WCNC) Proceedings, 2020 (pp. 1–5). https://doi.org/10.1109/WCNC45663.2020.9120537 OUCIarXiv

[13] Wu, Y., Li, M., Wang, J., Fang, Z., Zeng, Q., Yang, T., and Cheng, L. (2022). DroidRL: Reinforcement learning-driven Choosing features for Android malware detection. arXiv preprint arXiv:2203.02719. arXiv

[14] Xu, P. (2021). Android-COCO: Graph Neural Network-based malware detection for native and byte-code Android. ArXiv preprint arXiv:2112.10038. arXiv

[15] Taha, A., and Barukab, O. (2022). Android Malware Classification Using Optimized Ensemble Learning Based on Genetic Algorithms. Sustainability, 14(21), 14406. https://doi.org/10.3390/su142114406.