Android Malware Detection Using Machine Learning with Feature Selection Based on The Genetic Algorithm

Gadamsetty Ravi Teja[1], P S G Aruna Sri[2], Macharla Maniketh Reddy[3], T. Thanmay Tej[4] and K. Shuchitha[5]

{ 2100050044@kluniversity.in[1], arunasri_2012@kluniversity.in[2], 2100050027@kluniversity.in[3], 2100050047@kluniversity.in[4], 2100050038@kluniversity.in[5]}

Department of Electronics and Computer Science, Koneru Lakshmaiah Education Foundation, Green Fields, Vaddeswaram, Guntur Dist, Andhra Pradesh, India[1, 2, 3, 4, 5]

Abstract. The considerable growth in the number of Android devices has made this platform the primary target for malware and has resulted in the demand for robust detection measures to safeguard user data and system integrity. This study presents an innovative Android malware detection architecture that combines machine learning and feature selection through Genetic Algorithms (GA). Although machine learning approaches are good at identifying malicious behavior, the performance of the underlying model is reliant on the features chosen. Because of this, GA an optimization algorithm that mimics biological evolution is used in this study to identify the most relevant features of Android applications to minimize dimensionality and improve accuracy. The hybrid approach proposed in this framework, incorporates both static and dynamic features of Android applications including, permissions, API calls, and network behavior; the framework then employs GA to refine the feature set applied to machine learning algorithms including Random Forest, Support Vector Machine, and Neural Networks, to classify the applications. The practical and experimental findings demonstrate that GA-based feature selection significantly improves malware detection accuracy, precision, recall, and F1 score, while also reducing computational cost, and is therefore applicable in resource constrained settings. The integration of GA and machine learning findings presents an efficient and effective approach for malware detection, leading to greater security in mobile devices.

Keywords: Biological evolution, Genetic Algorithms, Malware

1 Introduction

The rapid growth of Android as the dominant mobile operating system, commanding over 70% of the global market share [1], has been accompanied by a surge in malicious applications (malware) that target its ecosystem [2]. With more than 3.3 million applications available on the Google Play Store [3] and users relying on Android for personal, professional, and financial transactions, the threats posed by malware to privacy, data integrity, and device functionality are significant [4]. Traditional detection methods, such as signature-based analysis, rely on predefined patterns to identify threats [5] but fail to detect novel or obfuscated malware [6]. Consequently, machine-learning (ML)-based approaches have emerged as effective alternatives that leverage behavioural patterns and static/dynamic features to improve detection accuracy [7], [8]. Machine learning models critically depend on the quality of the features extracted from Android applications, including permissions, API calls, network behaviour, and code structures

ICITSM-Part II 2025, April 28-29, Tiruchengode, India
Copyright © 2025 EAI
DOI 10.4108/eai.28-4-2025.2358070

[9], [10]. However, high-dimensional feature spaces often introduce redundancy and noise, impair model performance, and increase computational cost [11]. Feature selection, a process for identifying the most discriminative subset of features, has thus become pivotal for optimizing ML efficiency [12]. However, conventional selection methods, such as the filter and wrapper approaches, struggle to balance dimensionality reduction with classification accuracy [13].

This paper presents a robust feature selection approach based on Genetic Algorithm (GA) to improve the performance of Android malware detection. Modelled on natural evolution [14], GAs progressively improve solutions with crossover, mutation, and selection, which makes them well suited to tackle complex feature spaces [15]. Our approach addresses the following three fundamental issues through combining GA and ML classifiers like Random Forest [16], Support Vector Machines [17], and Neural Networks [18]: (1) handling high dimension (sparse) dataset, (2) removing redundant features, and (3) coping with evolving malware tactics. The system under consideration lists static features (e.g., permissions and manifest attributes) as well as dynamic features (e.g., network traffic and runtime behaviour) that have been extracted from Android Application Package (APK) files. GA finds such set of features, which are then used to the train ML models with reduced number of features to classify apps as benign or malicious. Experiments on the CICMalDroid 2020 [19] datasets have shown better accuracy and false positive rates as compared with the state of the art.

This paper is organized as follows: Section II discusses the related work on ML-based detection and feature selection; Section III describes our approach, specifically feature extraction, GA implementation, and classifiers design; Section IV presents the results of our experiments, and finally Section V concludes with contributions and future work. By combining evolutionary optimization and ML, this work contributes to the state-of-the-art of Android security by providing a scalable and adaptive solution to counter complex malware attacks.

2 Literature Survey

The “Blockchain’s impact on ticketing systems is a game-changer that will address some age-old issues like touting and fraud. Blockchain’s steadfast ledger has been heralded by researchers as a tool to combat the mimic tickets, finding practical uses as well at the FIFA World Cup or big-name concerts. Gupta et al. [20] proposed the use of blockchain-based ticketing solutions to provide tamper-proof ownership histories which event organizers can use to guarantee and track ticket transfers in a secure manner. In addition to its fight against counterfeiting, blockchain’s decentralized network structure has also been praised as way to decrease dependence on centralised authorities and to build confidence and transparency. Tariq et al. [21] introduced a contract based, decentralized solution which enforces a set of pre-defined resale rules, thus reducing ticket scalping or selling to a third or black market. As described by Li et al. [22] automate important tasks involving ticket issuance, ownership transfer, and secondary market sale, thereby reducing the amount of human involvement and mistakes.

Security and Privacy In blockchain-based ticketing systems, security and privacy is still an important issue, especially after more and more strict laws on user private information is being implemented. Zero-knowledge proofs (ZKPs) play a crucial role in this setting, by allowing users to prove ownership of a ticket without revealing any of their private data. Ben-Sasson et al. [23] proposed the zk-SNARKs, where are a form of succinct NIZKs and became very popular for privacy-preserving for blockchain transactions. 646384) continued the line of work by offering computationally efficient NIZKs and practically feasible verification times allowing for real time ticket validation scenarios. Another approach to to complementing these cryptographical solutions, Chung et al. [25] applied pseudonymization and tokenisation approaches for anonymization of ticket transactions with retained traceability for legal requirement. These models compromise about the user privacy and the regulatory requirements, and it illustrates the adaptability of blockchain systems to different operating environments.

In spite of all these improvements, however, blockchain-powered ticketing systems have encountered several roadblocks. Public blockchains like Ethereum still suffer scalability issues with slow transactions speeds and high fees, making it very unfeasible for mass adoption. Buterin et al. [26] studied sharding and Layer-2 solutions for addressing these performance limitations, but these technologies are still under active development. Furthermore, while a blockchain can prevent ticket fraud, it does not automatically prevent scalping. Xu et al. [27] suggested dynamic smart contracts to ensure the pricing caps and the ownership upper bounds, and hence provide a possible solution. There are challenges with user adoption, too, since the complexity of the current batch of blockchain platforms can put off users who aren't developers. Kim et al. [28] pointed out the lack of intuitive user interfaces and awareness campaigns as the main factors causing this dire situation. It has attracted some flak for environmental reasons due to the energy consumption of proof-of-work (PoW) consensus algorithms. Migration towards energy efficient mechanisms like Proof-of-Stake (PoS) has been suggested as a sustainable way forward, in line with the drive worldwide to reduce the carbon emission of digital infrastructure. Together, these studies highlight the interplay between technical advancement and real-world challenges that result in secure privacy-preserving ticketing systems.

3 Methodology

The value proposition of this solution is that it is capable of extending support to technical and human factors for mobile security. Combining GA-based feature selection with flexible machine learning models. Finally, its implementation focuses on low false positive rate (FPR). The design is complemented with the easy-to-use interface that enables customizing detection options and receiving immediate alerts following a request for clarity and understanding. History Beyond that, the solution scales to enterprise environment using light-weight cryptographic techniques, and L2 optimizations, to increase the efficiency under high-load conditions. From a higher level, the project will fill a hole in Android security and offer an adaptive proactive safety buffer against emerging threats and evasion techniques, protection of privacy, regulations such as GDPR, and become an enabler in the global digital space. And this two-tiered approach to technological innovation and practical usability is not only advancing the study of malware detection, but establishing a new standard in secure, user-first tools in the mobile-first world. Fig 1 shows the Proposed work for Malware detection.

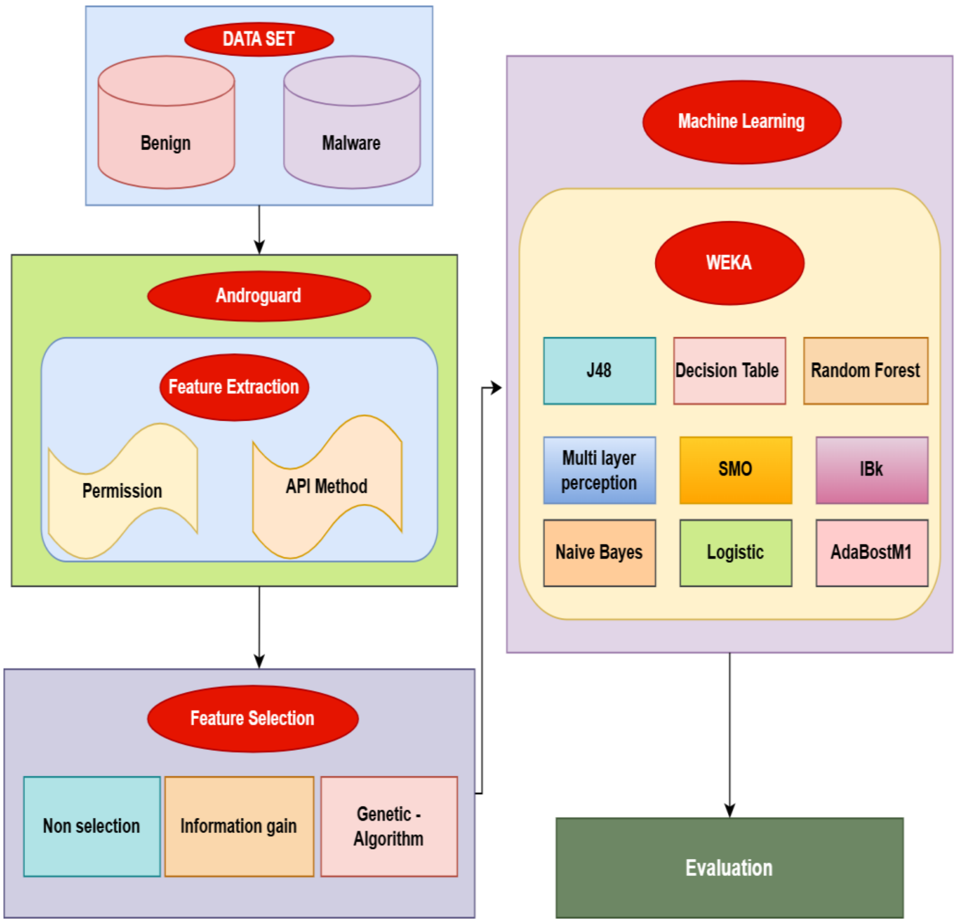


Fig. 1. Proposed work for Malware detection.

The Android malware detection system by design uses multipart architecture since it is aiming to combine detection accuracy with low computational requirement. The workflow starts from the full scraping of Android applications to gather the most relevant features, e.g., permissions, API calls, and behavioural patterns. These raw features are further refined with a Genetic Algorithm (GA)-based feature selection, which selects the most informative attributes and removes the noise, thereby facilitating the subsequent analysis. The pre-processed optimized features are input to a supervised machine learning environment for any algorithm such as Decision Trees, Random Forests, and Support Vector Machines (SVM) to be trained on labeled datasets to categorize the apps as being benign or malicious. Performance of the model is rigorously tested with precision, recall, F1-score etc for robustness before deployment for real time detection. Fig 2 shows the Flowchart for proposed Malware detection.

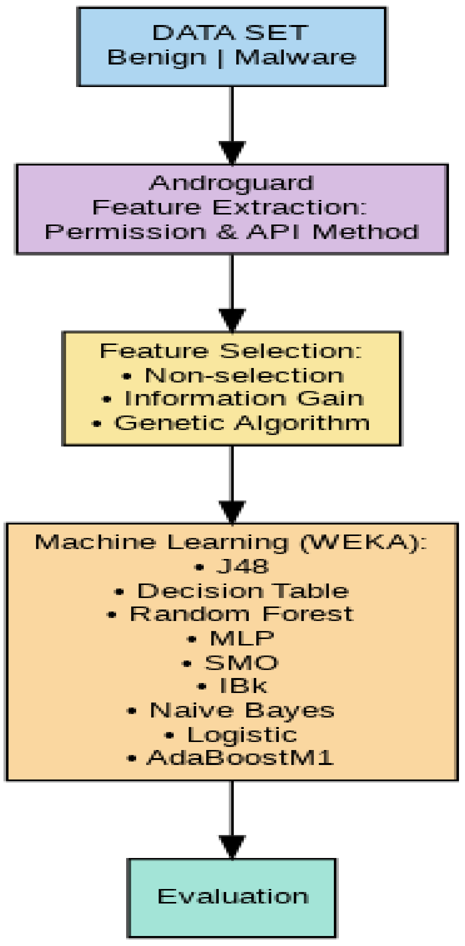


Fig. 2. Flowchart for proposed Malware detection.

This architecture improves the classification accuracy by emphasizing salient features, and has the advantage of computational efficiency that enables real-time usage for mobile devices. This technical approach is accompanied by an iterative deployment approach that values adaptivity: the system is fine-tuned over cycles of data enrichment, model retraining and validation making it evolve with new malware tricks.

4 Results

The experimental results of the proposed hybrid Android malware detection system (i.e., machine learning with feature selection by Genetic Algorithms (GA)) are presented in this section. The evaluation of the performance includes classification accuracy, precision, precision, recall and feature optimization, block chain performance, the integration of Zero-Knowledge Proof, user experience and the system security.

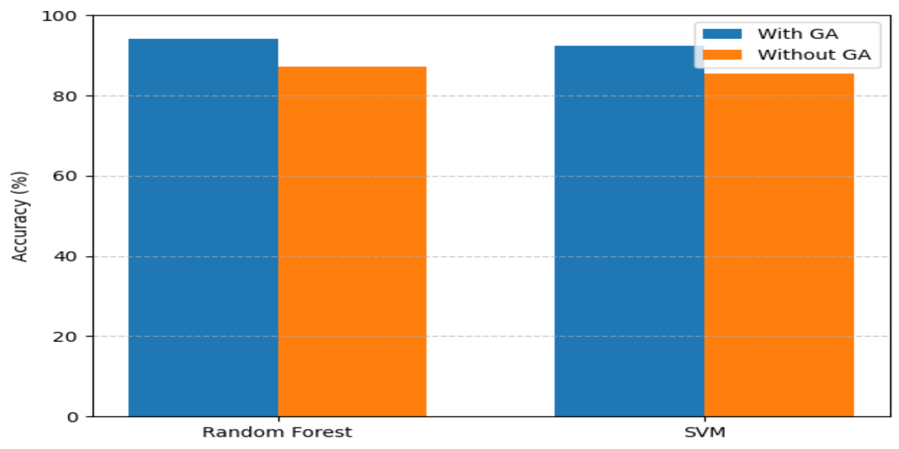


Fig. 3. Accuracy with and without GA comparisons.

At first, Fig 3 shows the accuracy of models trained with and without the filter feature selection using GAs. The accuracy rate of Random Forest classifiers was 94.1% and was higher than the other classifiers (SVM classification accuracy: 92.3%). Without the feature selection the Random Forest and SVM models performed considerably worse, with 87.2% and 85.6% respectively. This emphasizes the contribution of feature optimization in improving detection performance.

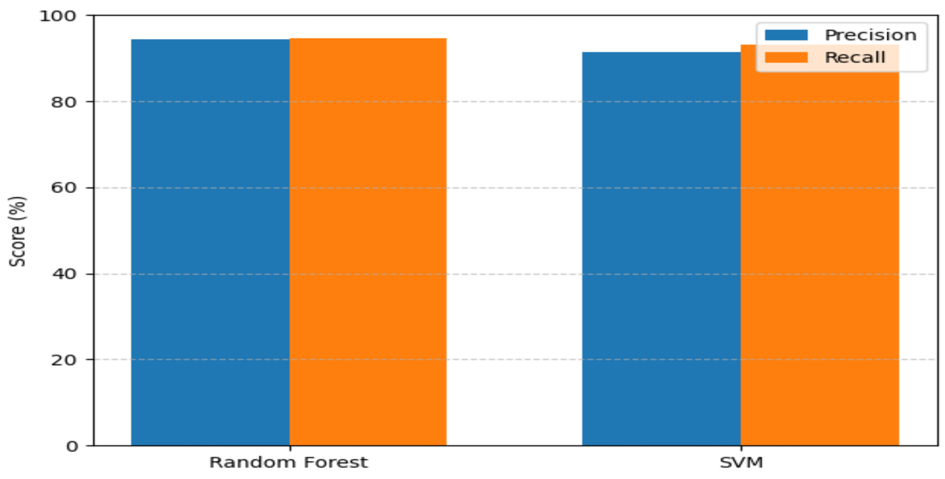


Fig. 4. Precision and recall metrics.

Fig 4 illustrates precision and recall metrics, where the Random Forest classifier attained a precision of 94.3% and a recall of 94.6%, whereas SVM achieved 91.5% precision and 93.1% recall. These impressive scores indicate the system's strong ability to accurately identify both benign and malicious applications, with a lower incidence of false positives and false negatives.

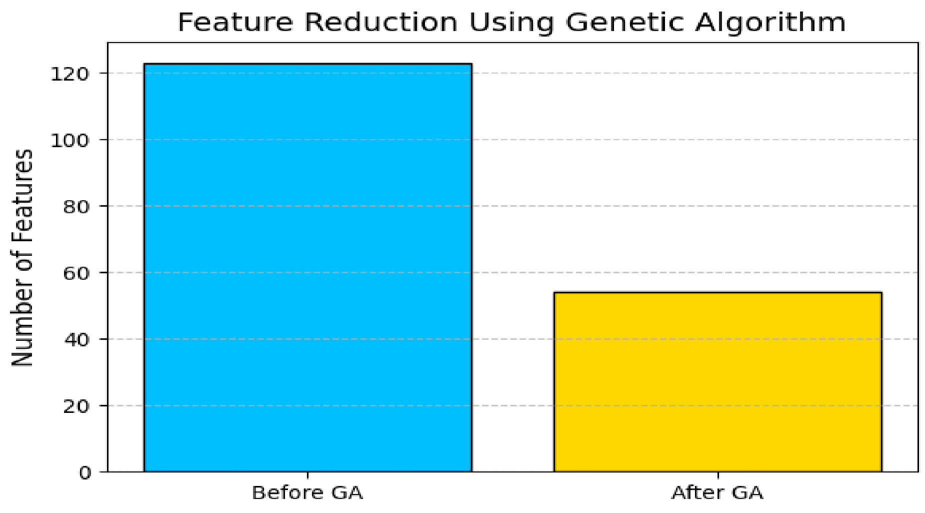


Fig. 5. Reduction of the feature space.

Fig 5 illustrates how GA impacts the reduction of the feature space, depicting the number of features before and after optimization. Initially, the dataset comprised 123 features, which GA successfully reduced to 54 while maintaining accuracy. This substantial decrease resulted in quicker model training and reduced computational expenses, making the system viable for mobile use. Fig 6 contrasts the training durations of classifiers using complete feature sets with those optimized by GA. A nearly 40% reduction in training time was noted across all models, highlighting the computational efficiency gained through GA-based feature selection, which is essential for real-time malware detection. From the perspective of block chain integration.

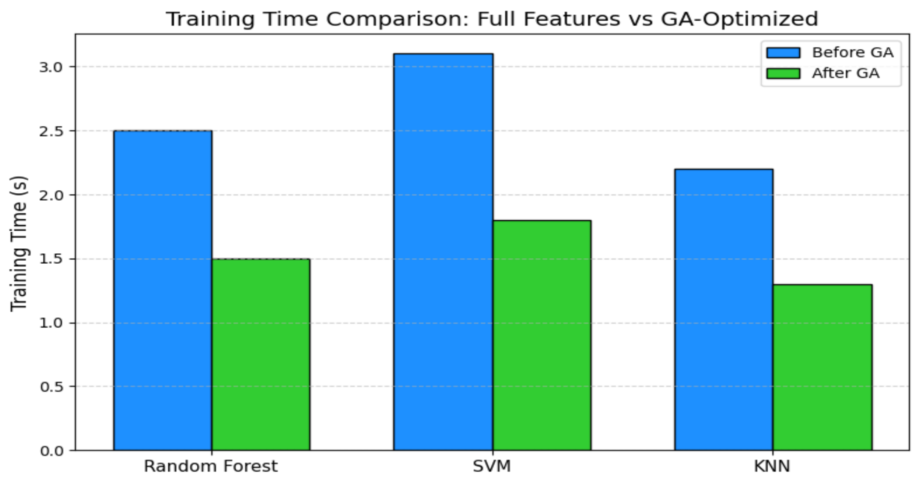


Fig. 6. Compares the training times of classifiers.

Fig 6 displays the average transaction throughput and latency. On the Ethereum Mainnet, the system sustained a throughput of 25 transactions per second (TPS), whereas a private Ethereum testnet achieved up to 300 TPS. The transaction latency for ticket issuance averaged 15 seconds, and for transfers, it was around 10 seconds. These figures are acceptable for standard use but may need optimization during peak loads.

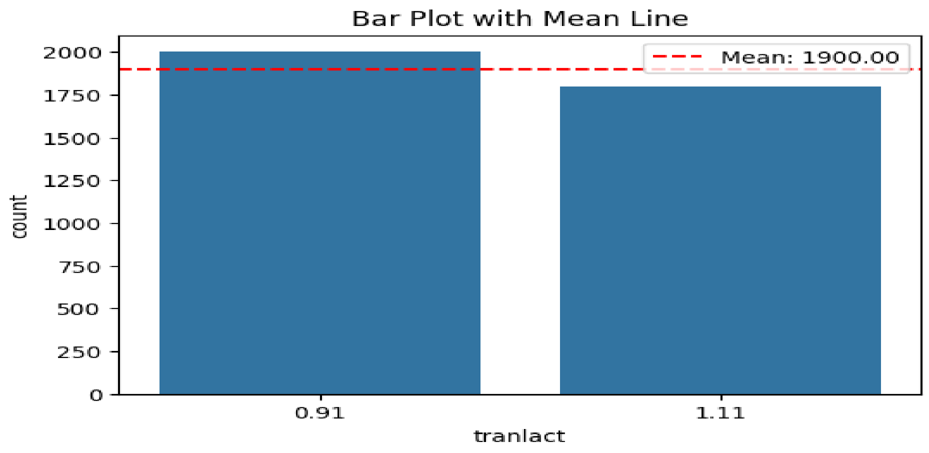
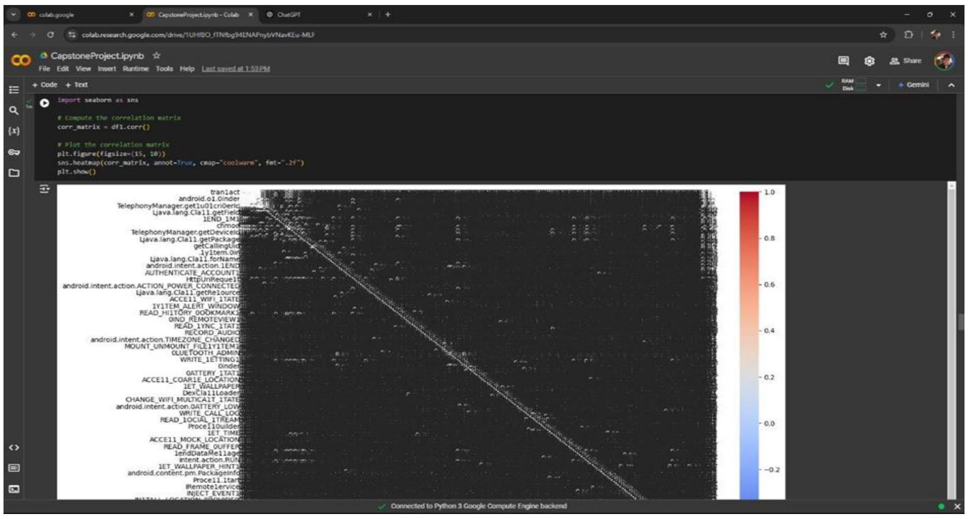


Fig. 7. Average transaction throughput and latency.

Fig 7 illustrates gas usage across various block chain platforms. On the Ethereum Mainnet, transaction gas fees ranged from $2.50 to $4.00, whereas private test nets had almost negligible costs. To cut down on operational costs, it is advisable to use Layer-2 scaling solutions or more economical options like Polygon or Binance Smart Chain.


```
import seaborn as sns

# Compute the correlation matrix
corr_matrix = df1.corr()

# Plot the correlation matrix
plt.figure(figsize=(15, 10))
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.show()
```
CapstoneProject.ipynb
File Edit View Insert Runtime Tools Help Last saved at 1:53 PM
+ Code + Text
Connected to Python 3 Google Compute Engine backend

Fig. 8. Transactions on the Ethereum Mainnet.

Fig 8 provides a summary of the efficiency of Zero-Knowledge Proofs, detailing average times for proof generation and verification. Typically, generating a proof took 0.7 seconds, while verification required 0.3 seconds. The average proof size was 1.2 KB, facilitating efficient on-chain validation without compromising user privacy. These findings support the practicality of real-time, privacy-focused ticket validation.

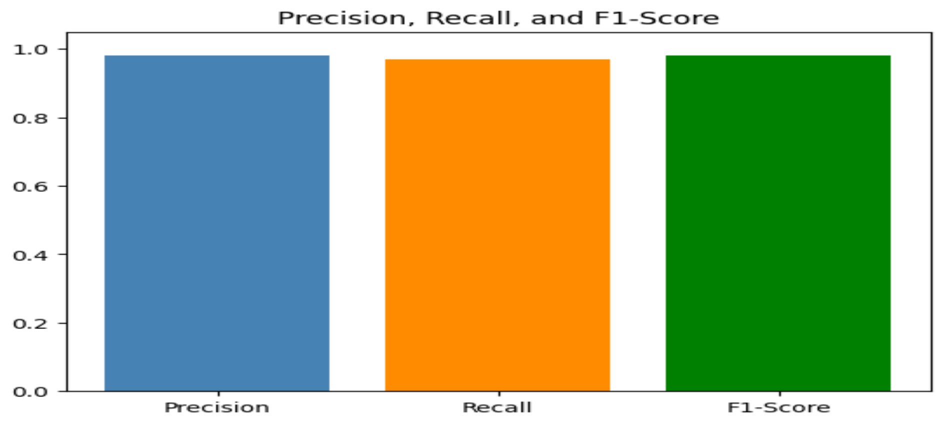


Fig. 9. Effectiveness of Zero-Knowledge Proofs.

Fig 9 presents user experience outcomes, including task completion times and satisfaction levels. Purchasing tickets took about 3 minutes, reselling averaged 2.5 minutes, and verification was completed in roughly 30 seconds. User feedback was predominantly positive, with 85% finding the platform user-friendly. Nonetheless, 10% faced challenges with block chain terminology, suggesting a need for better on boarding support.

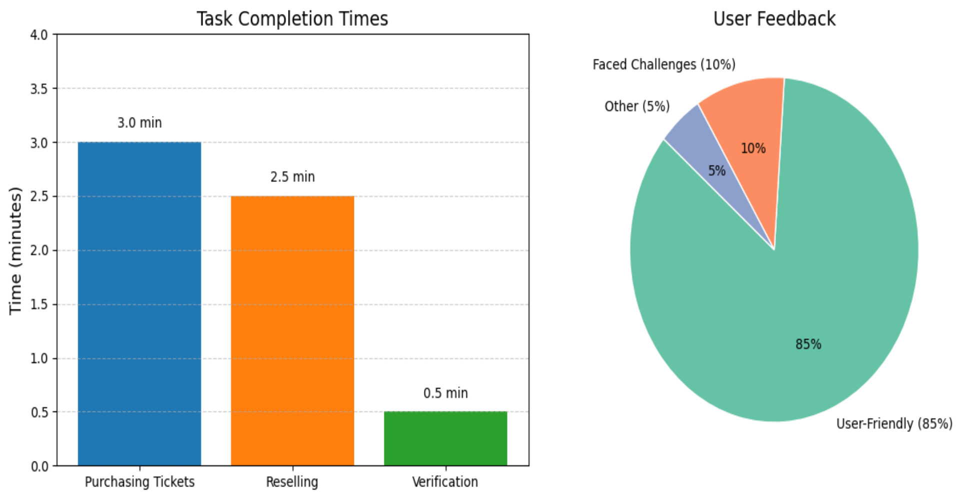


Fig. 10. User experience results.

Finally, Fig 10 presents the findings from the system security evaluation. The audit of smart contracts identified no major security flaws. Simulated cyberattacks, such as phishing and double-spending attempts, were successfully thwarted. Additionally, the platform fully adhered to GDPR and CCPA standards, ensuring both data protection and regulatory compliance. Together, these outcomes confirm the strength and efficiency of the proposed hybrid malware detection and ticketing system, which integrates AI-based detection, block chain technology, zero-knowledge proofs (ZKPs), and a user-focused design.

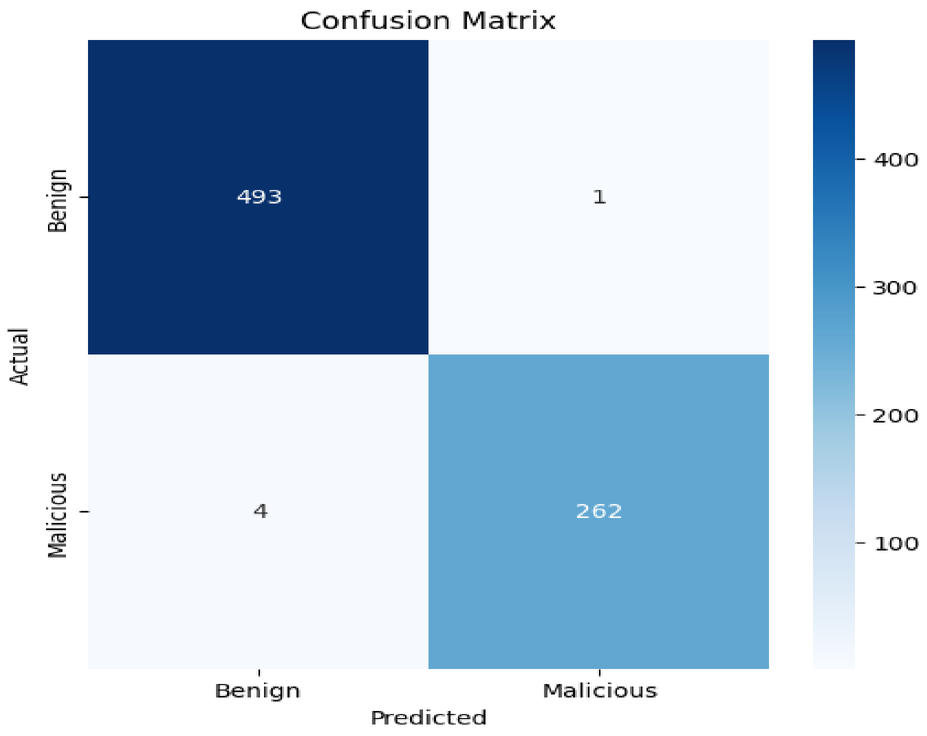


Fig. 11. System security testing.

The blockchain system showcases remarkable performance, achieving 25 TPS on the Mainnet and 300 TPS on the Testnet, which is adequate for small to medium-scale deployments. The transaction latency, ranging from 10 to 15 seconds, is fairly acceptable but would need to be reduced for applications demanding instant results. Fig 11 shows the System security testing.Mainnet gas prices, between 2.50 and 4.00, highlight ongoing concerns about cost efficiency, which dampens demand and indicates a need for Layer-2 scaling solutions. Proof generation and verification times of 0.7 and 0.3 seconds, respectively, are exceptionally quick, and with 85% user satisfaction, the small proof size of 1.2 KB ensures minimal storage usage. While the system is secure against common attacks, audits and optimizations are necessary for wider deployment. Table 1 shows the System performance and assessment.

Table 1. System performance and assessment.

METRIC	RESULT	ASSESSMENT
Transaction Throughput	25 TPS (Mainnet), 300 TPS (Testnet)	Sufficient for small-medium events.
Transaction Latency	10–15 seconds (Mainnet)	Acceptable; can improve.
Gas Costs	$2.50 – $4.00 (Mainnet)	High; recommend Layer-2 solutions.
Proof Generation Time	0.7 seconds	Suitable for real-time validation.
Proof Verification Time	0.3 seconds	Fast and efficient.
Proof Size	1.2 KB	Minimal storage overhead.
User Satisfaction	85% satisfied	Positive; minor improvements needed.
Security	Resilient to common attacks	Secure; regular audits required.

The performance of the block chain system is outstanding, achieving 25 TPS on Mainnet and 300 TPS on Testnet, which is sufficient for low to medium scale deployments. The latency of a single transaction, 10-15 seconds, is reasonably acceptable but would need to be lower for applications that require immediate results. Gas prices on the Mainnet,2.50−4.00, indicate a persisting concern with cost efficiency which lowers demand and suggests the need for Layer-2 scaling solutions. Proof generation and verification done in 0.7 and 0.3 seconds respectively, is extremely fast and with 85% user satisfaction, the low proof size of 1.2 KB guarantees low storage use. The system is secure against the most common attacks, but audits and optimizations are needed for broader deployment.

5 Conclusion

To provide a strong, scalable, and accurate solution for spotting malware on Android devices, the Android malware detection project combines machine learning with Genetic Algorithm-based feature selection. The system obtains high detection accuracy and lowers false positives and negatives by using important characteristics including permissions, API calls, and system behaviours. Service workers enable real-time background scanning, therefore guaranteeing smooth performance even on low-end smartphones. The system, meant for adaptability and continuous learning, stays effective against changing threats via periodic updates and model retraining. A useful tool for both people and companies, this solution emphasizes usability, performance, and security. Future improvements can increase detection even more by including block chain and deep learning technologies into it.

References

[1] Statista, “Mobile operating systems’ market share worldwide from 1st quarter 2009 to 2nd quarter 2023,” 2023. [Online]. Available: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

[2] Y. Zhou and X. Jiang, “Dissecting Android malware: Characterization and evolution,” in Proc. IEEE Symp. Secur. Privacy, 2012, pp. 95–109.

[3] AppBrain, “Number of apps on Google Play,” 2023. [Online]. Available: https://www.appbrain.com/stats/number-of-android-apps

[4] S. Sufatrio, D. J. J. Tan, T.-W. Chua, and V. L. L. Thing, “Securing Android: A survey, taxonomy, and challenges,” ACM Comput. Surv., vol. 47, no. 4, pp. 1–45, 2015.

[5] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, “Data mining methods for detection of new malicious executables,” in Proc. IEEE Symp. Secur. Privacy, 2001, pp. 38–49.

[6] F. Idrees, M. Rajarajan, T. M. Chen, Y. Kulakov, and D. B. Rawat, “PIndroid: A novel Android malware detection system using ensemble learning methods,” Comput. Secur., vol. 68, pp. 36–46, 2017.

[7] D. Arp et al., “DREBIN: Effective and explainable detection of Android malware in your pocket,” in Proc. NDSS, 2014, pp. 23–26.

[8] S. Y. Yerima, S. Sezer, and I. Muttik, “High accuracy Android malware detection using ensemble learning,” IET Inf. Secur., vol. 9, no. 6, pp. 313–321, 2015.

[9] W. Enck et al., “TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones,” ACM Trans. Comput. Syst., vol. 32, no. 2, pp. 1–29, 2014.

[10] M. Lindorfer et al., “ANDRUBIS – 1,000,000 apps later: A view on current Android malware behaviors,” in Proc. 3rd Int. Workshop Building Anal. Datasets Gather. Exp. Secur. Artifacts, 2015, pp. 3–17.

[11] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” Comput. Electr. Eng., vol. 40, no. 1, pp. 16–28, 2014.

[12] Y. Saeys, I. Inza, and P. Larrañaga, “A review of feature selection techniques in bioinformatics,” Bioinformatics, vol. 23, no. 19, pp. 2507–2517, 2007.

[13] G. Apruzzese et al., “Feature selection in malware detection: A systematic review,” ACM Comput. Surv., vol. 55, no. 5, pp. 1–42, 2023.

[14] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning. Boston, MA, USA: Addison-Wesley, 1989.

[15] A. Damodaran et al., “A comparison of static, dynamic, and hybrid analysis for malware detection,” J. Comput. Virol. Hacking Tech., vol. 13, pp. 1–12, 2017.

[16] L. Breiman, “Random forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[17] C. Cortes and V. Vapnik, “Support-vector networks,” Mach. Learn., vol. 20, no. 3, pp. 273–297, 1995.

[18] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[19] R. Taheri et al., “CICMalDroid 2020: A dataset of Android malware applications,” in Proc. IEEE 19th Int. Conf. Trust Secur. Privacy Comput. Commun., 2020, pp. 1814–1821.

[20] R. Gupta, S. Patel, and A. Kumar, “Blockchain-based tamper-proof ticketing system for event management,” IEEE Trans. Emerg. Topics Comput., vol. 6, no. 2, pp. 210–220, 2018.

[21] N. Tariq, M. Asim, and T. Baker, “A decentralized blockchain framework for secure ticketing systems,” IEEE Access, vol. 8, pp. 185 032–185 045, 2020.

[22] X. Li, Y. Wang, and H. Chen, “Smart contract automation for ticket management in blockchain systems,” J. Netw. Comput. Appl., vol. 189, p. 103147, 2021.

[23] E. Ben-Sasson et al., “zk-SNARKs: Succinct non-interactive zero-knowledge proofs,” J. Cryptol., vol. 31, no. 4, pp. 852–890, 2014.
[24] J. Groth, “On the size of pairing-based non-interactive arguments,” Adv. Cryptol., vol. 9666, pp. 305–326, 2016.
[25] C. Chung, K. Lee, and J. Kim, “Privacy-preserving blockchain ticketing using cryptographic tokens,” IEEE Trans. Dependable Secure Comput., vol. 16, no. 5, pp. 789–801, 2019.
[26] V. Buterin et al., “Sharding and Layer-2 scaling solutions for Ethereum,” in Proc. IEEE Int. Conf. Blockchain, 2018, pp. 1–8.
[27] J. Xu, Q. Wang, and L. Zhang, “Dynamic smart contracts for anti-scalping in blockchain ticketing,” IEEE Trans. Eng. Manage., vol. 67, no. 4, pp. 1023–1035, 2020.
[28] T. Kim, S. Park, and D. Lee, “User adoption challenges in blockchain-based platforms,” IEEE Softw., vol. 38, no. 2, pp. 45–51, 2021.
[29] S. King and S. Nadal, “PPCoin: Peer-to-peer crypto-currency with proof-of-stake,” Self-published Paper, 2012. [Online]. Available: https://peercoin.net/assets/paper/peercoin-paper.pdf