ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 5 Issue 2 April 2025 / Page No: 24-34
Paper Id: JETA-V5I2P104 / Doi: 10.56472/25832646/JETA-V5I2P104

Original Article

Securing Android Systems Using Scalable and Lightweight ML for Ransomware Classification and Identifications

Mani Gopalsamy
Independent Researcher

Received Date: 07 January 2025 Revised Date: 12 February 2025 Accepted Date: 06 April 2025

Abstract *: The need to secure mobile devices has never been higher than it is now, given the growing danger of Android ransomware assaults. In order to address this issue, this paper suggests a lightweight machine learning (ML) technique that employs Android malware detection using Decision Tree (DT) and Support Vector Machine (SVM) classifiers. In order to maximize model efficiency, the research was conducted using an Android ransomware dataset and approaches including feature selection, under-sampling, and categorical conversion. At 97.24%, 98.50%, and 98.40%, respectively, ransomware detection accuracy, precision, recall, and F1 scores outperform SVM and traditional machine learning models. It demonstrates how lightweight machine learning models are able to identify ransomware threats while still being computationally efficient to execute on Android devices that are resource-constrained. An improvement in detection accuracy and resilience against evolving ransomware variants can be achieved by future research that combines deep learning with real-time adaptive learning processes.*

Keywords *: Android ransomware, machine learning, ransomware detection, Decision Tree, Support Vector Machine, computational efficiency, ransomware variants.*

I. INTRODUCTION

The vast user base of the Android operating system has made it a desirable target for cybercriminals. Android ransomware has been one of the most common dangers to Android users in recent years' attacks[1]. These malicious attacks encrypt user data without permission, rendering it useless until the attackers get a ransom.. The number and sophistication of Android ransomware attacks have significantly increased, posing serious risks to people, companies, and even government agencies[2][3]. The attackers use a variety of techniques, including malicious applications, using methods such as drive-by downloads and phishing emails to get access to a target's mobile device[4].

Neural networks and SVMs are examples of lightweight ML models, and decision tree utilization balances performance and resource efficiency[5][6]. The use of lightweight models ensures that devices with limited processing power can still participate in securing the network without sacrificing their core functionality. ML models can adapt to new threats, improving their detection capabilities over time [7][8]. Furthermore, these models can be tailored to specific types of devices and use cases, ensuring more accurate and effective protection for each individual device in the network[9].

It is still necessary to carefully assess the accuracy and usefulness of ML-based techniques for Android ransomware detection[10][11]. An ML-based model may be used to defend Android devices against ransomware threats, which are becoming more frequent and effective. Attacks using ransomware on Android smartphones are becoming more frequent, and efficient ML models might help detect and prevent these types of attacks [12][13].

A. Structure of paper

The rest of the article is structured like this. Give an analysis of the research on a Lightweight ML Solution for Android Device Ransomware Detection in Section II. Methods and methodology are presented in Section III, and Section IV deals with the analysis and discussion of the results. The study's findings and future directions are presented in Section V.

II. LITERATURE REVIEW

Provide some earlier research on In this section, will find an Android ransomware detection method that uses a lightweight ML approach. Ransomware detection on Android smartphones has been the subject of speculation by certain authors. Table I discusses the use of in-ransomware detection.

Khaliq et al. (2024) explain the features, capabilities, and variations of ransomware. Lastly, a framework is proposed to detect the ransomware using the “Kaspersky” anti-malware tools and a virtual machine to prevent ransomware attacks. The purpose of ransomware, a type of crypto virus, is to extort money from the victim by exposing or permanently blocking access to their personal data. A strategy used by sophisticated viruses is called crypto viral blackmail, while some ransomware may just lock the machine without erasing any data[14].

Table 1 : Comparative Table for Literature Review Lightweight Machine Learning Solution for Ransomware Detection

Reference	Methodology	Dataset	Performance	Limitations & Future Work
Khaliq et al. (2024)[14]	Framework using "Kaspersky" anti-malware tools and a virtual machine for ransomware detection	EldeRan Dataset	Effective in preventing ransomware attacks	Limited to Kaspersky tools; lacks evaluation with modern ML techniques
Elsadig (2023)[15]	Detecting DoS assaults in WSNs with Gini feature selection in a Decision Tree (DT)	Improved WSN-DS data	99.5% accuracy, lower overhead than RF, XGBoost, and KNN	Limited to DoS attacks in WSN; future work can explore other attack types
Ismail, Dawoud and Reza (2022)[16]	Naive Bayes + LightGBM multi-layer machine learning detection solution for WSN cyberattacks	WSN-DS dataset	Effective in detecting four network-layer DoS attacks	Focuses only on internal WSN attacks; mobile robot integration needs further validation
Guerra-Manzanares (2024)[17]	Review of challenges in Android malware detection	Various datasets	Identified five significant unresolved challenges	Highlights need for better datasets, methodologies, and long-term ML solutions
Hossain et al. (2025)[18]	Detecting Android malware using an ensemble-based machine-learning technique	203,556 network traffic records (10 ransomware types & benign traffic)	Improved accuracy and robustness through multiple classifiers	Requires further testing on real-world applications and dynamic behaviors
Alsoghyer and Almomani (2019)[19]	Static analysis of API requests is used by the ransomware detection system that uses APIs (API-RDS)	Ransomware Dataset	High accuracy in detecting ransomware before execution	Limited to static analysis; future work should include hybrid approaches
Urooj et al. (2022)[20]	An outline of ransomware	Various datasets	Comprehensive review of	Suggests future research

	detection techniques based on ML and DL	from 2019-2021	existing solutions	directions; lacks practical implementation of proposed methods

Elsadig (2023) clarifies WSN limitations, vulnerabilities, and security risks, emphasizing DoS attacks. Recent techniques for identifying denial-of-service attacks have been thoroughly studied, exposing both their strengths and weaknesses. This offers insightful information about the state of recent studies in this area. In order to identify DoS assaults in WSNs, this work suggests a simple ML detection method that uses feature selection using the Gini coefficient and the DT routine. In order to train and test the suggested method, an improved version of the author's WSN-DS dataset was utilised. With a 99.5% accuracy rate and little cost, the suggested method outperforms KNN, RF, and extreme XGBoost classifiers[15].

Ismail, Dawoud and Reza (2022) reduce cyberattacks against WSNs with its lightweight, multi-layer ML detecting system. They want to use a mobile robot to help combat internal WSN assaults. The two ML models that are installed at the BS and monitor nodes are part of the detecting system with many layers. In multi-class classification, second-layer detection was accomplished using the LightGBM approach, and the NB approach was used for first-layer detection in binary classification. Four network-layer internal DoS assaults may be detected by the suggested system based on the WSN-DS dataset[16].

Guerra-Manzanares (2024) Five major outstanding issues need to be addressed by professional study before Android malware detection can be deemed a resolved issue. These issues, which are covered in detail throughout the study, hinder efficient, long-term ML-based Android malware detection. They include limitations on the data set, incorrect postulates, and methodological flaws. The comprehensive review of Android malware detection best practices identifies and promotes potential future research avenues to address the issue[17].

Hossain et al. (2025) propose a new method for Android ransomware detection using ensemble-based ML. This method would enhance detection accuracy and robustness by utilising the capabilities of many classifiers. For optimal model performance, a large dataset consisting of 203,556 records of network traffic derived from 10 different ransomware variants as well as benign traffic was painstakingly preprocessed and designed with features. The approach incorporates many ensemble classifiers and rigorously cross-validates each one. They may improve their models and concentrate on the most predictive characteristics by employing Random Forest feature significance analysis to find important indications of ransomware activity[18].

Alsoghyer and Almomani (2019) In the most recent methods for detecting Android malware were examined. A thorough comparison study was carried out, revealing the main distinctions between the current options. A static analysis paradigm known as API-RDS was developed for Android apps that encrypt files in order to find API-based ransomware detection tools. According to API-RDS, API package requests are the main indicator of ransomware activity, allowing for exact detection prior to device damage. Both ransomware and innocuous apps' API package calls were carefully examined and contrasted. Important API packages were found, along with matching methods[19].

Urooj et al. (2022) give the specifics of the datasets that were collected from their sources and utilized in the investigations on ransomware detection carried out by the various platforms. This paper is distinct in that it summarizes ransomware detection research that employs ML, DL, and a mix of the two methods by utilizing benefits of using dynamic analysis to identify ransomware. The information provided takes into consideration the ransomware detection research that was done between 2019 and 2021. Future research will be facilitated by the comprehensive list of future directions provided by this study[20].

The Table I compares lightweight ML-based ransomware detection methods, datasets, and performance, highlighting their effectiveness, limitations and suggesting future improvements like hybrid approaches, broader attack coverage, and real-world validation.

A. Research Gaps

The Advanced ransomware detection systems still need to resolve several important issues. Current studies utilize ML and DL methods that consume numerous resources, which prevent proper operation on low-power Android devices. Current ransomware detection models have an active learning shortage, which prevents them from tackling new ransomware threats effectively. The detection accuracy of ransomware identification can benefit from expanded feature selection optimization techniques as well as more extensive benchmark dataset analysis. Research on hybrid methods linking ML with DL techniques is scarce, yet there is a need to develop API-based behavioral analysis to enable instant ransomware identification capabilities. Multiple studies analyzing different detection approaches remain scarce, while researchers do not adequately track ransomware mutational patterns. Cloud-based detection systems create security and privacy-related issues that demand

privacy-preserving frameworks to resolve them. The primary drawback of current research lies in its single-layer detection approach since companies need multilayered defensive strategies that unite anomaly detection with behavioral analysis and heuristic learning methods. These areas need improvement to better detect ransomware attacks across Android devices.

III. METHODOLOGY

A thorough explanation of the data flow diagrams in Figure 1 subsequent phases is provided below:

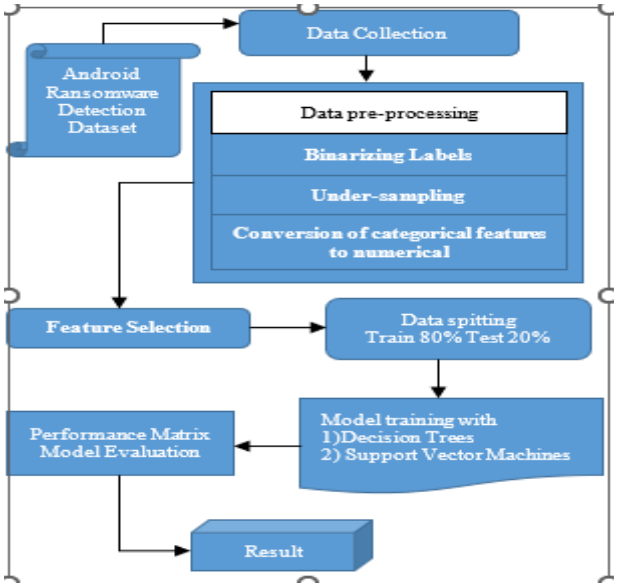

Figure 1 : Structure Of The Data Flow Diagram

The CICIDS 2018 dataset is used in the approach for cyberattack detection, which follows an organized procedure to attain a high accuracy of up to 98%. The data is initially collected from the CICIDS 2018 dataset and put through data preparation in order to enhance the model's performance, which includes feature reduction, zero variance feature removal, and timestamp removal. After that, key characteristics are determined in order to concentrate on those that are most pertinent to the detection procedure. To guarantee data consistency, data normalization is then implemented using the Min-Max and Z-Score normalization approaches scaling. Two models are then trained using the preprocessed and normalized data: These measures are used to evaluate the models' F1-score, recall, accuracy, and precision following training with Multilayer Perceptron with Backpropagation (MLP-BP). Both models demonstrate strong predictive capability, achieving an impressive accuracy of 98.97%, making them highly effective in detecting cyber-attacks. The outcome shows how reliable the method is in differentiating between malicious and benign network data.

A. Data Collection

This dataset contains 392,035 entries which include 85 features and one class label free from any missing value distribution. The dataset contains 11 class names, which are divided between harmless traffic and 10 different Android ransomware categories. An anomaly identification task typically faces a class imbalance problem due to the distribution shown in Figure 2. Approximately 98% of the entries belong to ransomware categories according to the dataset statistics, although it contains only 43,091 records that do not belong to ransomware categories.

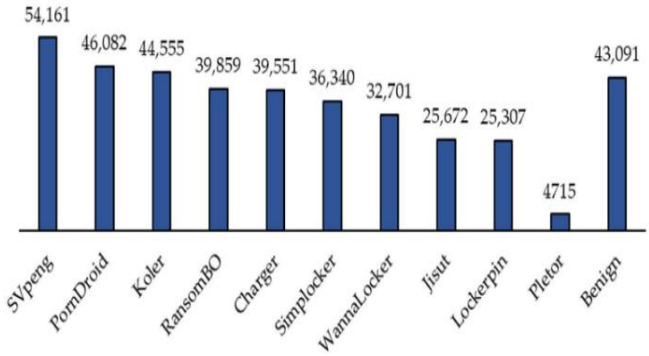

Figure 2 : Distribution of Class Labels

B. Data Preprocessing

The preprocessing phase started with the plot and check of data qualities along with the analysis of class compositions and feature attribute types. Twelve constant features along with "Time Stamp" were removed because they brought no benefit to model training and eliminated 13 features from the dataset. All 11 classes from the dataset underwent a transformation into binary codes where ransomware fell under class 1 and everything else became class 0. Randomized under-sampling was implemented for class balancing by creating equal sample groups (43,091 samples each) based on training data classes. IP source and destination Numerical value translation was applied to IP addresses and flow IDs. In order to choose a final set of 19 crucial features that achieved the best model performance, The feature selection procedure made use of both feature relevance and forward feature selection. The following Figure 3 provides the balancing graph.

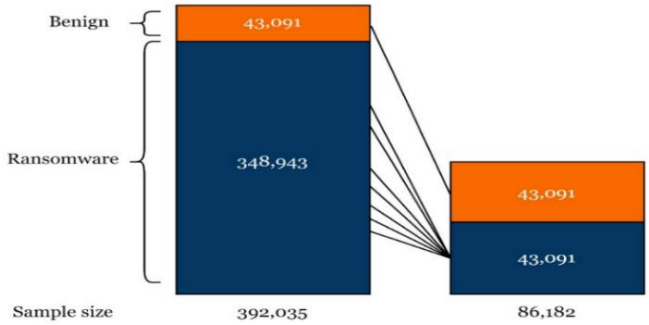

Figure 3 : Under-Sampling Technique Applied to the Dataset

C. Feature Selection

A feature selection process helped improve both the model efficiency and performance by identifying key attributes that received analysis[21]. A set of crucial features described in Table II includes Flow ID combined with Source and Destination IPs and Ports and Protocol together with Flow Duration and packet-related measurements. The chosen features held important value due to their capability to detect fundamental network behavioral patterns. The selection process leads to decreased variables while enhancing model readability together with achieving peak operational efficiency.

Table 2 : The Final Set of Features Selected

No.	Feature	No.	Feature	No.	Feature
1	Flow ID	8	Fwd packet length max	15	Active mean
2	Source IP	9	Fwd packet length min	16	Active std
3	Source port	10	Bwd packet length min	17	Active max
4	Destination IP	11	Init_Win_bytes_forward	18	Idle mean
5	Destination Port	12	Init_Win_bytes_backward	19	Idle std
6	Protocol	13	act_data_pkt_fwd		
7	Flow duration	14	min_seg_size_forward		

D. Dataset Split

During the experiment, the researchers developed two separate groups of ransomware and benign processes, both for training and testing. Model training requires 80% of pre-processed data, whereas validation testing utilizes 20% of data. A model analysis space divided 80% to 20% allows maximum data utilization for creating resilient models and effective validation testing.

E. Classification Phase

The area of ML, a kind of AI, replaces explicit programming by allowing computers to comprehend and learn from data. ML enables self-sufficient solutions for a variety of computational issues.

a) Decision Trees

To determine classes within a dataset, a collection of techniques called decision tree models employ information passed as in the projected class shown in Figure 4, from connected to the nodes at the base. A decision tree is a kind of method for predictive modeling that creates a hierarchical data structure by employing the divide-and-conquer tactic. It works well for

regression and classification using nonparametric techniques. To immediately observe how various tree species relate to one another and how tree features are distributed throughout their distinct groupings [22]. This has the advantage of allowing for less computation because each tree is shallow and may show fewer leaves than a single DT would for the same task time.

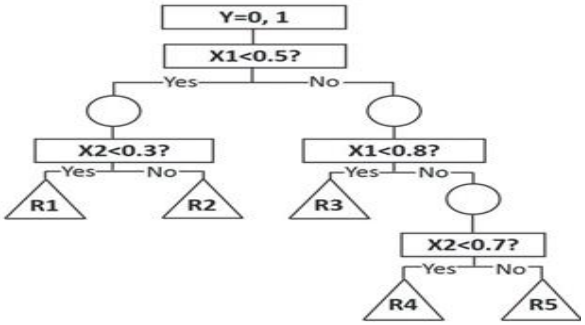

Figure 4 : Decision Tree that is Based on Binary Target Variable Y

These are the main decision tree mathematical Equations (1 to 4):

- Entropy (H)

$$H(S) = -\sum_{i=1}^{c} p_i \log_2 p_i \quad (1)$$

- Information Gain (IG)

$$IG(S,A) = H(s) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} H(S_v) \quad (2)$$

- Gini Impurity (G)

$$G(S) = 1 - \sum_{i=1}^{c} p_i^2 \quad (3)$$

- Gini Gain (GG)

$$GG(S,A) = G(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} G(S_v) \quad (4)$$

F. Support Vector Machines

SVM are widely used in supervised learning to address issues related to classification, regression, and identifying outliers. SVM solves the classification issue by determining the best separating hyperplane between several classes. Using a training dataset, SVM maximizes the distance between classes to identify the optimal hyperplane for successfully separating two classes [23][24]. Through the thoughtful use of feature selection, the SVM algorithm described in this paper illustrates in Figure 5 its distinctiveness, hyperparameter adjustment, and the use of different radial, Kernel functions, include, among other things, polynomial and linear basis functions.

The decision hyperplane is given by Equation (5):

$$w^T x + b = 0 \quad (5)$$

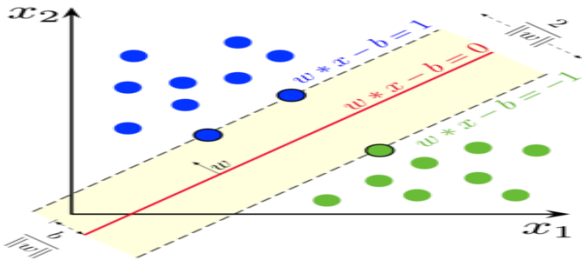

Figure 5 : SVM Model

G. Performance Matrix Model Evaluation

Selecting the evaluation metric requires an understanding of how each metric measures in order to assess the model appropriately. To determine how well ML algorithms work, the goal was to analyze each of these performance metrics, which included F1-Score, Accuracy score, Precision, and Recall.

a) Accuracy

The percentage of instances the model properly classifies and the general error in class prediction is used to evaluate the model's accuracy. This measure summarizes the model's performance across classes. However, performance may be misrepresented by biased data. A classifier may correctly anticipate instances of the majority class while incorrectly categorizing cases of the minority class. Accuracy is formulated in Equation (6):

$$Acuracy = \frac{TP+TN}{TP+TN+FN+FP} \quad (6)$$

b) Precision:

The percentage of instances that are accurately allocated to a class after all the data has been categorized is known as precision. In this instance, it shows the proportion of corona cases that actually are corona cases. The Equation (7) defines the precision:

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

c) Recall

The Recall or sensitivity determines the amount of examples that are successfully classified into a class. This context measures the proportion of properly represented instances by the classifier among all carriers of the illness. It gives as Equation (8):

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

d) F1-score

The F1-score, often known as Recall and accuracy weighted harmonic mean is known as the F-measure. This metric is best suited when the dataset is significantly unbalanced use. The F1-score I formulated in Equation (9)

$$F1 - score = 2 * \frac{precision*recall}{precision+recall} \quad (9)$$

Where,

- TP (True Positive): As anticipated by the model, the actual result was favorable.
- TN (True Negative): The model anticipated a negative result, while the actual value was negative.
- FP (False Positive): The model anticipated a positive result, but the actual value was negative.
- FN (False Negative): The model projected a negative result, while the actual number was positive.

IV. RESULTS AND DISCUSSION

The simulated outcomes of this study are A Lightweight Android Device ML Solution Section for Detecting Ransomware. This section shows the results of the dataset evaluation carried out for this study, along with performance metrics, classifier statistics, and results.

A. Experiment Results

This section shows the outcomes of applying the DT and SVM model to a dataset in a lightweight ML manner to forecast Ransomware detection on Android devices.

Model	Accuracy	Precision	Recall	F1-score
DT	97.24	98.5	98.4	98.45
SVM	89.05	98.05	100	94.21

Table 3 : DT And SVM Model Performance Matrices For Ransomware Detection

Table 3 displays a number of performance characteristics for the DT model, including an accuracy of 97.24%, which indicates overall correctness. Its 98.50% precision indicates that the majority of positive predictions are correct, and its 98.40% recall great shows how well it detects real positives. The SVM model's performance metrics, with an accuracy of 89.05%, and the model's F1-score of 98.45% both show great overall correctness. Its 89.05% accuracy indicates that the majority of positive forecasts are correct, while its recall of 100% demonstrates its effectiveness in identifying true positives. The model's F1 score of 94.21% highlights its excellent balance between accuracy and recall and demonstrates its excellent classification performance.

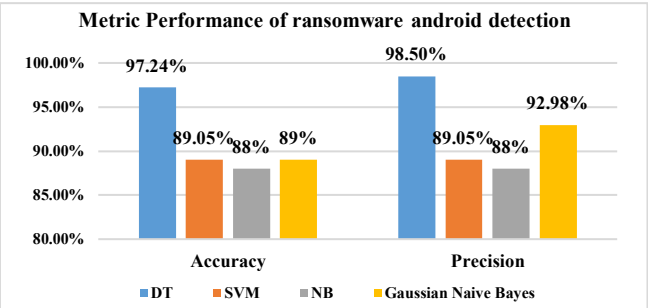

Figure 6 : Metric Performance Of Ransomware Android Detection

Figure 6 displays how DT and SVM classifiers perform regarding metric measurements for ransomware detection on Android devices. The DT model maintains exceptional results throughout all metrics because its accuracy joins with F1-score stays over 95%, as does precision and recall. Although the SVM model retains a perfect recall at 100%, it exhibits lesser precision and accuracy, falling below 90%. SVM demonstrates high capability for detecting ransomware cases but its precision rate sits beneath 90% which indicates more false positives during the detection process. The DT model displays even performance metrics across all assessment points, which establishes its reliability when detecting ransomware on Android operating systems.

The DT classifier received evaluation through confusion matrices appearing as Figures 7 and 8 for Android network traffic classification. The correct outcome consists of TN for benign traffic and TP for ransomware, but misclassification occurs when it identifies FN and FP events. Better model performance occurs when the count of TP and TN remains high. The unoptimized Decision Tree model reached 97.23% accuracy with 98.50% precision and 98.39% recall, which produced an F1-score of 98.44% in its evaluation. The prediction model performed better, with 97.24% accuracy, 98.80% precision, 98.50% recall, and a 98.50% F1-score when 19 characteristics were set to zero during hyperparameter optimization. The DT classifier shows better results because of the tuning process.

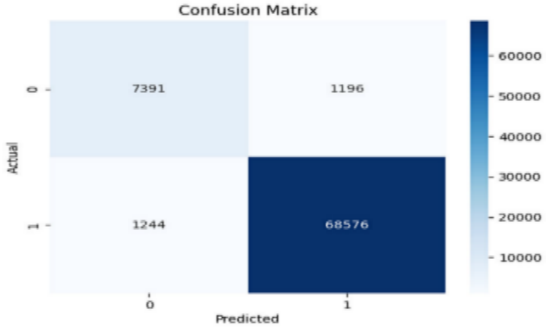

Figure 7 : Experiment 1 – DT Confusion Matrix

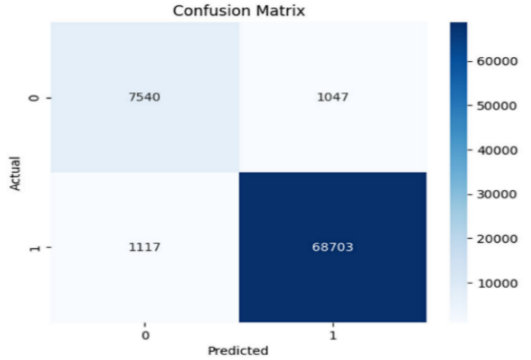

Figure 8 : Experiment 2 – DT Confusion Matrix

B. Comparative Analysis and Discussion

An analysis comparing many models for the detection of ransomware. Table IV below contrasts and analyses several machine learning models used for ransomware Android detection prediction in terms of performance metrics.

Table 4 : Comparison Between Various Models For Ransomware Detection

Model	DT	SVM	NB [25]	Gaussian Naive Bayes[26]
Accuracy	97.24	89.05	88	89
Precision	98.50	89.05	88	92.98
Recall	98.40	100	89	96.16
F1-score	98.45	94.21	88	94.54

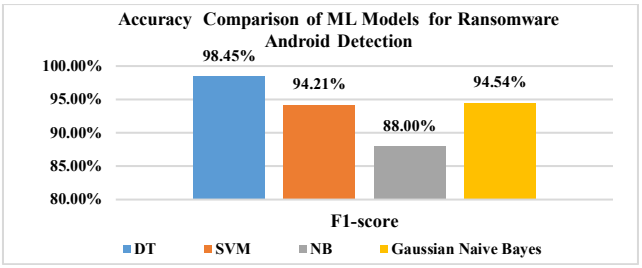

Figure 9 : Accuracy Comparison of ML Models for Ransomware Android Detection

Four ML models' accuracy comparison is displayed in Figure 9 (DT, SVM, NB and Gaussian Naïve Bayes) for ransomware android detection. Gaussian Naïve Bayes and SVM came in second and third, respectively, while NB had the lowest accuracy at 88%. At 97.24%, DT had the highest accuracy.

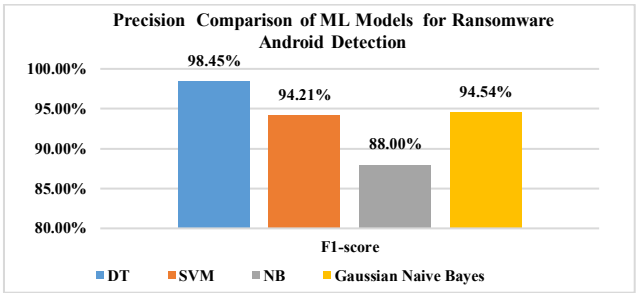

Figure 10 : Precision Comparison of ML Models for Ransomware Android Detection

Figure 10 compares the accuracy of four different ML models (DT, SVM, NB and Gaussian Naïve Bayes) for ransomware Android detection. DT is achieved the highest precision at 92.98%, followed by Gaussian Naïve Bayes and SVM, NB had the lowest accuracy at 88%.

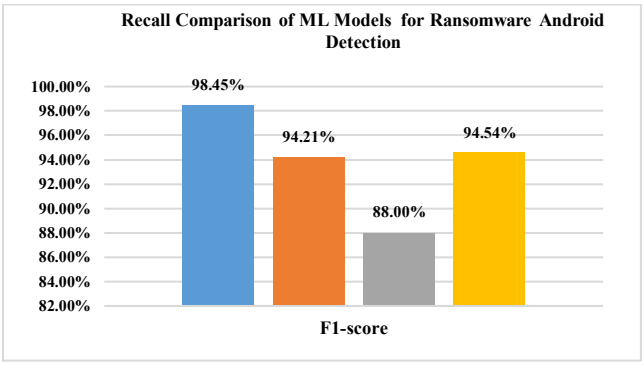

Figure 11 : Recall Comparison Of ML Models For Ransomware Android Detection.

Figure 11 shows the comparison of four ML models' recall (DT, SVM, NB and Gaussian Naïve Bayes) for ransomware Android detection. Gaussian Naïve Bayes and DT came in second and third, respectively, with a maximum accuracy of 100%, while NB had the lowest accuracy of 89%.

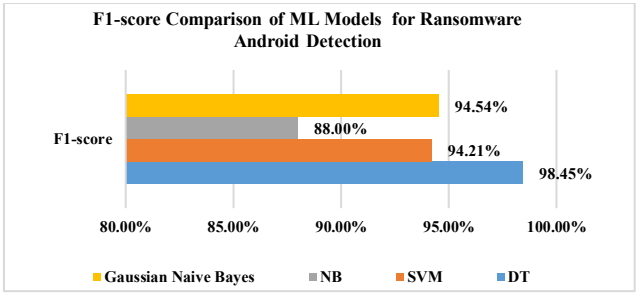

Figure 12 : Comparative Model graph of F1-Score

Figure 12 shows the evaluation of four ML models' recall contrast (DT, SVM, NB and Gaussian Naïve Bayes) for ransomware Android detection. DT is achieved the highest accuracy at 100%, followed Gaussian Naïve Bayes and NB had the lowest accuracy at 88%.

V. CONCLUSION AND FUTURE SCOPE

In this work, DT and SVM classifiers were used; they proposed a low-power ML approach for ransomware detection on Android smartphones. The presented approach effectively addressed the challenges of Android ransomware detection through feature selection, data preprocessing, and machine learning based on classification. The experimental data confirmed that the accuracy of DT model reached the highest at 97.24%, surpassing SVM classifier and more conventional ML techniques. Additionally, the extremely high accuracy, recall, and F1-score values confirm our method's excellent robustness and dependability. They achieved resource-constrained devices-friendly solutions by optimizing model parameters and using efficient data processing techniques such that our solution includes high detection accuracy. Lastly, our results show how lightweight machine learning models may be used to strengthen mobile protection against changing ransomware threats.

Future research will make application of deep learning methods, such as RNNs and CNNs, which will significantly improve ransomware detection accuracy. Moreover, the model can be integrated with real-time threat intelligence as well as with the model can automatically learn and adapt to new ransomware variations thanks to adaptive learning methods. More diverse and recent ransomware samples will help expand the dataset and improve the model generalization. Additionally, they will deploy and test our model in real-world Android environments to find out the practical feasibility and performance in real time. The goal is thus to build up an Android ransomware detection framework that is complete and more adaptable.

VI . REFERENCES

[1] N. P. Hirenkumar Mistry Kumar Shukla, “Transforming Incident Responses , Automating Security Measures , and Revolutionizing Defence Strategies through AI-Powered Cybersecurity,” p. 2024, 2024.

[2] V. Pillai, “Anomaly Detection for Innovators: Transforming Data into Breakthroughs,” Lib. Media Priv. Ltd., 2022.

[3] S. Duary, P. Choudhury, S. Mishra, V. Sharma, D. D. Rao, and A. Paul Aderemi, “Cybersecurity Threats Detection in Intelligent Networks using Predictive Analytics Approaches,” 4th Int. Conf. Innov. Pract. Technol. Manag. 2024, ICIPTM 2024, p. 364, 2024, doi: 10.1109/ICIPTM59628.2024.10563348.

[4] R. Farhan, “An Approach to Android Ransomware Detection Using Deep Learning,” Wasit J. Pure Sci., vol. 3, pp. 90–94, 2024, doi: 10.31185/wjps.325.

[5] S. S. S. Neeli, “Critical Cybersecurity Strategies for Database Protection against Cyber Attacks,” J. Artif. Intell. Mach. Learn. Data Sci., vol. 1, no. 1, p. 5, 2023.

[6] J. Rahul Dattangire, Ruchika Vaidya, Divya Biradar, “Exploring the Tangible Impact of Artificial Intelligence and Machine Learning: Bridging the Gap between Hype and Reality,” 2024 1st Int. Conf. Adv. Comput. Emerg. Technol., pp. 1–6, 2024.

[7] M. K. A Arif, A Khan, “Role of AI in Predicting and Mitigating Threats: A Comprehensive Review,” JURIHUM J. Inov. dan Hum., vol. 2, no. 3, pp. 297–311, 2024.

[8] M. S. Akaash Vishal Hazarika, “Blockchain-based Distributed AI Models: Trust in AI model sharing,” Int. J. Sci. Res. Arch., vol. 13, no. 2, pp. 3493–3498, 2024.

[9] P. M. Rajendra Prasad Sola, Nihar Malali, “Cloud Database Security: Integrating Deep Learning and Machine Learning for Threat Detection and Prevention: 0,” Notion Press, 2025.

[10] P. Piyush, N. S. Gill, P. Gulia, D. D. Rao, Y. Mandiga, and P. K. Pareek, “Systematic Analysis of threats, Machine Learning solutions and Challenges for Securing IoT environment,” J. Cybersecurity Inf. Manag., vol. 14, no. 2, pp. 367–382, 2024, doi: 10.54216/JCIM.140227.

[11] N. Patel, “AI-Enhanced Zero Trust Security Architecture for Hybrid and Multi-Cloud Data Centers: Automating Trust Validation, Threat Detection, and Mitigation,” Int. J. Nov. Trends Innov., vol. 3, no. 1, pp. a13–a18, 2025.

[12] Albin Ahmed, A. Shaahid, F. Alnasser, S. Alfaddagh, S. Binagag, and D. Alqahtani, “Android Ransomware Detection Using Supervised Machine Learning Techniques Based on Traffic Analysis,” Sensors, 2024, doi: 10.3390/s24010189.

[13] M. I. Khan, A. Arif, and A. R. A. Khan, “AI-Driven Threat Detection: A Brief Overview of AI Techniques in Cybersecurity,” BIN Bull. Informatics, vol. 2, no. 2, pp. 248–261, 2024.

[14] K. Khaliq, N. Z. Ab Rahim, K. Hamid, M. Ibrar, U. Ahmad, and M. U. Ullah, “Ransomware Attacks: Tools and Techniques for Detection,” 2nd Int. Conf. Cyber Resilience, ICCR 2024, pp. 2024–2025, 2024, doi: 10.1109/ICCR61006.2024.10532926.

[15] M. A. Elsadig, “Detection of Denial-of-Service Attack in Wireless Sensor Networks: A Lightweight Machine Learning Approach,” IEEE Access, 2023, doi: 10.1109/ACCESS.2023.3303113.

[16] S. Ismail, D. Dawoud, and H. Reza, “A Lightweight Multilayer Machine Learning Detection System for Cyber-attacks in WSN,” in 2022 IEEE 12th Annual Computing and Communication Workshop and Conference, CCWC 2022, 2022. doi: 10.1109/CCWC54503.2022.9720891.

[17] Guerra-Manzanares, “Machine Learning for Android Malware Detection: Mission Accomplished? A Comprehensive Review of Open Challenges and Future Perspectives,” 2024. doi: 10.1016/j.cose.2023.103654.

[18] M. A. Hossain, T. Hasan, F. Ahmed, S. H. Cheragee, M. H. Kanchan, and M. A. Haque, “Towards superior android ransomware detection: An ensemble machine learning perspective,” Cyber Secur. Appl., vol. 3, no. July 2024, p. 100076, 2025, doi: 10.1016/j.csa.2024.100076.

[19] S. Alsoghyer and I. Almomani, “Ransomware detection system for android applications,” Electron., 2019, doi: 10.3390/electronics8080868.
[20] U. Urooj, B. A. S. Al-Rimy, A. Zainal, F. A. Ghaleb, and M. A. Rassam, “Ransomware Detection Using the Dynamic Analysis and Machine Learning: A Survey and Research Directions,” Appl. Sci., 2022, doi: 10.3390/app12010172.
[21] S. Murri, From Raw to Refined: The Art and Science of Data Engineering. Notion Press, 2025.
[22] S. J. Alghamdi, “Classifying High Strength Concrete Mix Design Methods Using Decision Trees,” Materials (Basel)., 2022, doi: 10.3390/ma15051950.
[23] S. Islam et al., “Image Processing and Support Vector Machine (SVM) for Classifying Environmental Stress Symptoms of Pepper Seedlings Grown in a Plant Factory,” Agronomy, vol. 14, no. 9, p. 2043, 2024, doi: 10.3390/agronomy14092043.
[24] Y. H. Rajarshi Tarafdar, “Finding majority for integer elements,” J. Comput. Sci. Coll., vol. 33, no. 5, pp. 187–191, 2018.
[25] Almomani, A. Alkhayer, and W. El-Shafai, “E2E-RDS: Efficient End-to-End Ransomware Detection System Based on Static-Based ML and Vision-Based DL Approaches,” Sensors, 2023, doi: 10.3390/s23094467.
[26] A. Herrera-Silva and M. Hernández-Álvarez, “Dynamic Feature Dataset for Ransomware Detection Using Machine Learning Algorithms,” Sensors, 2023, doi: 10.3390/s23031053.