Natural Resources for Human Health
eISSN: 2583-1194

Original Research
Received 14 March 2026
Revised 18 April 2026
Accepted 22 May 2026

Natural Resources for Human Health 2026, Vol. 6 (2s): 102–116

KEYWORDS:
mHealth, Android devices, Technical communication, Malware detection, Cryptography, Deep learning, Sword Fish Algorithm, Sha1 Hash key, and Adam optimizer

Artificial Intelligence in Technical Communication for Secure Healthcare: An Adaptive Framework for Android Malware Detection

Saptaparni Chatterjee[1], Rabinarayan Panda[2], Sachikanta Dash [3]*, John J P[4], K A Naveen Kumar[5], Bishnu Paramguru Mahapatra[6]

[1,2,4,5]Department of Engineering, Garden City University, Bengaluru, India
[3]Department of CSE(Cyber Security), Madanapalle Institute of Technology and Science, AP, India
[6]Department of English and Foreign Language, Madanapalle Institute of Technology and Science, AP, India
*Corresponding author: Sachikanta Dash, Email Id: dash.sachikanta@gmail.com

ABSTRACT: Objective:The broad uptake of Android based mobile health (mHealth) applications has, in a way, increased the risk of malware attacks, that can endanger private patient data and also disrupt secure technical messaging. This study now puts forward an Artificial Intelligence aided adaptive scaffold, meant for accurate detection that is scalable, plus more secure Android malware defense, specifically inside healthcare settings.

Material and Methods: So there was this Android malware dataset, with both benign and harmful applications, it was collected and then preprocessed in some standard way. After that, a kind of new Dot Hash Swordfish Algorithm was used, mainly to strengthen data confidentiality, with fast enough encryption/decryption. Then the decrypted data set was fed into a Deep Ensure AttackNet approach, meant for spotting malware. At the same time, an Adaptive PropAdam optimization method was added, and it kind of blends Adam and RMSProp together, so the feature learning improves, and it classifies each app as benign or malicious. Finally the whole thing was evaluated via accuracy, precision, recall , F1-score, ROC, computational time, and also scalability, to see how well it behaves under pressure.

Results: The proposed framework showed, in a kind of notable way, better malware detection performance by reaching 96.86% accuracy and an ROC score of 0.99. Its use of encryption together with deep learning and adaptive optimization helped ramp up classification accuracy, speed up convergence a bit, cut down on computational complexity, and also make it more scalable than the usual deep learning–driven malware detection approaches.

Conclusion: The proposed AI assisted framework efficiently strengthens cyber security for Android based healthcare apps, while it also helps with secure technical communication among healthcare professionals, patients, and the digital health platforms in general. Put together, the encryption layer, the smart malware detection, and that adaptive optimization approach makes it a kind of dependable and scalable solution, to shield sensitive healthcare data and also to improve the resilience of these newer digital healthcare systems.

1. INTRODUCTION

Malicious attack detection is one of the essential challenges in cybersecurity, due to the frequent changes in attacker's behaviors overtime. Android malware is intended to control a computer system. Significantly, the malware targets Android OS system to hack the user's confidential information and smashes into the system. Worldwide, android-based devices become more attractive among developers and users. Due to the status of the Android OS device, developers are encouraged to create innovative applications called apps. The users have connected based on connectivity options like Bluetooth, GPS, 4G LTE, Wi-Fi, and NFC. In general, the user's personal information can be accessed anywhere, like messages, browsing history, social network access, banking credentials, and contacts, which have expanded the cybercriminal attacks on Android smartphones. Cybercriminals achieve the benefits of corresponding malware-infected applications, attack users' private information, and accidentally damage the app store [1].

The word "malware" describes to a specific form of intrusive application or software. Some examples of malware are defined as ransomware, adware, trojan horses, spyware, etc…These malware software or applications are introduced to achieve malicious activities such as encouraging access to the devices, stealing personal data and creating modifications in the devices. Till now, there is a variety of malware has been analyzed. In Table 1 below, some of the recently known Android based malware is described with its description [2].

Table 1: Some of the recent Android malware and description [3]

Malware name	Description
Blackrock	It steals the password of applications and essential information from the device.
Agent Smith	It steals the applications and requires them to show more ads to receive profit.
CovidLock	This malware affects the user's device by providing them with COVID-19-related information.
Joker	It steals money from users by accidentally signing them into premium memberships.
Pegasus	This spyware can access all users' information, including SMS, WhatsApp and email conversations.
System Update	It steals photos, videos, and the user's location once it is installed on the mobile device.

Malware-infected applications are determined by using two methods: on-device malware analysis and off-device malware analysis. The on-device malware analysis is developed using a signature-based method. It compares the signature definitions with an existing signature in the datastorage. Although this method cannot able to determine the new or unknown malware attack, it is considered as the disadvantage of this method. Secondly, the off-device malware analysis method requires identifying the functionality of the malware. However, this analysis highly involves human intervention. Here, the malware analyst involves the automatic analysis to determine the malware in a timely manner. To extract relevant features, Android malware detection is primarily performed using two analysis approaches: static analysis and dynamic analysis.

Statical analysis: This type of analysis involves feature extraction without considering Android applications. It analyzes only the source code components to extract features such as permissions, intents, textual information, and other static attributes. It has the disadvantages of incompatibility, code transformations, and Java reflections.

Dynamic analysis: This type of analysis extracts the features of Android malware in a virtual environment. The features involve dynamic code loading that is collected by the emulator. It has the disadvantages of implementation, code average, and complexity [4].

After the analysis of Android malware for feature extraction, machine learning or deep learning methods are introduced to detect the exact Android malware. These artificial intelligence techniques enable computers to learn without being programmed. The learning method determines the unknown behaviour of malware through the existing knowledge. Our research introduced the Adam optimization-based deep learning method for malware detection. The optimization method combines the classifier weights and biases that provide a high accuracy of prediction. It reduces the training error and generalization error in the loss function on the training dataset.

1.1 Problem identification

In recent years, mobile devices have not only been for communication purposes but also professional objectives like document writing, sending emails, generating PDFs, etc.… On mobile devices, various applications are available to perform identical tasks, such as desktops for social networking, mobile banking, games, and online shopping. Also, smartphones have a number of sensors that declare the sensitive information of user location, images, audio, video and other information [5]. The increasing smartphone market has not only attracted users but also malware authors. Malware is developed in mobile applications for threading personal information, and it spreads through various channels like mail attachments, app stores, download links of SMS messages, etc…Therefore, malware detection is more necessary to enhance the security of Android based smartphones. For the identification and classification of malware, previous researchers have suggested various techniques on data. These existing studies had some issues in the data collection, analysis and responses [6]. From the overview, some of the existing novelty-based techniques provide the disadvantages of reduced computational complexity and scaling, minimum accuracy of learning-based mechanism, and high computational cost [7] [8]. The minimum computing and storage resources are not really suitable for large-scale data processing on mobile devices, especially on the go. It seems that the learning-based approach must keep improving towards an optimization-driven detection style, which should boost accuracy, and yes in particular with noisy samples [9]. To fix those shortcomings of the older techniques, this research presents a new technique designed for stronger malware detection on android mobile devices.

1.2 Objectives

The main goal of this study is to enhance the malware detection of Android mobile devices by using encryption and optimization-based classification methods.

- To increase data security, the Novel Dot hash sword fish algorithm performs the encryption and decryption process using a novelty function.
- To enhance secure technical communication by protecting healthcare data and ensuring reliable Android mobile applications against evolving malware threats.
- Identify the malicious network by using the Deep Ensure attack net, which determines the malicious node in the network.
- To provide effective malware detection, the Adaptive prop Adam optimization is for classifying the data as attacked and non-attacked.
- To perform the classification of attacks with respect to metrics of accuracy, precision, recall, f1-score, and ROC curve.

1.3 Paper organization

The manuscript is ordered as follows. Initially, the paper starts with an introduction to the background of the research, problem identification, and research objectives for the proposed method. The second section describes the previously existing literature works and research gaps identified in those existing studies. The third section describes the proposed methodology in a separate subsection. In the fourth section it shows the outcomes of the proposed method based on different performance analyses, more or less. After that the paper sort of wraps up with a conclusion and a future recommendation section, so it feels complete.

2. LITERATURE SURVEY

This section discussed the various malware and anomaly detection methods existing researchers use in multiple environments. The authors in [10] Described the machine learning methods and encryption models to develop the cybersecurity of nonlinear processes. The encryption method consists of a two-tier control architecture, which is associated with ML based cyberattack finder to improve cyber security and operational safety and to enhance the process of nonlinear. In the architecture, the upper layer kind of pulls in encryption via the Lyapunov-based model predictive controller , so it can improve that whole closed loop performance a bit more. The medium level is used to encrypt the group of linear controllers for the stabilization phase. When this strategy improves the closed-loop performance, it discloses the higher level of architecture to potential cyberattacks. To reduce that risk, the ML techniques are applied in FFNN (feed-forward neural network) to detect the attacks. The research provides the complete stability analysis and establishes the error bounds based on the implementation of sample-and-hold controller and quantization. The presented framework would be expanded to any nonlinear process which is measured by the grouping of nonlinear and linear controllers that improves the system cybersecurity. The results estimated the proposed method of robustness and determined the unseen attack patterns. Moreover, in [11] established the defensive ML cyber security applications in the perceptions of decision-makers. For detecting blocking threats and user and software threats, ML provides a new framework in the cybersecurity sector. The increasing amount of cyber-attacks in cyber security applications would make a significant disadvantage. Additionally, the research gives major variations of ML solutions in cybersecurity applications. They evaluated the general observations and perceptions of decision-makers, such as ML capabilities, implementation, and estimation, using machine learning. Similarly, [12] delivered the machine learning techniques for detecting anomalies. The study provided a review of various ML methods for detecting the anomalies in various applications based on four perceptions: anomaly detection applications, machine learning models, performance measures and classification of anomaly detection. The review analyzed the 290 studies from 2000-2020 with various ML methods for anomaly detection[23-26]. These studies contain 22 datasets and other standard datasets applied to ML techniques and then verified the ML classification systems for detecting the anomalies. Finally, they provide some guidelines and recommendations based on these review.

[13] proposed the enhanced Random forest method for detecting the anomalies in cyber security. The intrusion detection system screened the internet users' behavior and categorized it as abnormal or normal. It protects the virtual or physical system from unwanted threats. The study proposed the enhanced model as a random forest method for classification and to achieve a robust IDS model. Random forest classifier is the most effective method compared to other ML classifiers. To enhance data security, the research provides hybrid method classification and feature selection. For classification, the enhanced RF classifier was used and random feature selection method was utilized. In [14] the author presented the federated learning based on a RF classifier detecting the intrusions or abnormalities. Data vulnerability is growing every day with an increasing amount of sophisticated cyber security threats, and it affects business continuity. The research provided the federated learning model that would attain training of data and send the results model to the server. It decreases the requirement for sending all information to the centralized server. Experimental results provide high accuracy compared to other individual RFs in large datasets.

[15] estimated the KNN and SVM models for irregularity detection in cybersecurity. To analyze the cybercriminal, the study provides the KNN and SVM model on the CBS open dataset. The evaluation results provides the proposed mehtod of various performance metrics. Also, I calculated the expectation maximization by using the Gaussian mixture model. Finally, the presented model provides 89% accuracy through the hybrid method. [16] developed the survey of various ML techniques for cyber intrusion detection systems. The described ML techniques are naïve Bayes, random forest, bayesian network, and artificial natural network. Furthermore, they discussed the various cybersecurity datasets and evaluated them with various performance metrics. [17] provided the enhanced IDS by using KNN hyperparameter tuning—the minor attacks from cyberspace that existing machine learning models solve. Based on the disadvantages of previous intrusion detection methods, this research provides an enhanced ID system with a high detection rate. The suggested advanced method combines the k-nearest neighbour hyperparameter tuning and semisupervised learning as cross-validation. In the classifier training , the k-nearest neighbours are looked at primarily, and then the neighbouring data are kind of acquired based on statistical information from the hyperparameter tuning stage. Based on how these neighbouring data points show up, the distance weight and the distance metrics were sorted into a normal class or an abnormal class. The model that is proposed was tested on the NSL-KDD dataset, and it highlights its strengths pretty clearly.

[18]described the random forest and KNN classification for spam detection. [19] proposed the secured control detector architecture to determine the different kinds of cyber attacks. The work enhanced the constancy of the closed-loop system for

detecting cyber attacks in nonlinear systems. A higher level LMPC - Lyapunov-based Model Predictive Controller was utilized to improve closed loop. The Artificial neural network was developed with noisy operating conditions through sensor measurements. The evaluation results show that the presented method is effective for differentiating various classes of intelligent cyber attacks. [20] presented the nonlinear system framework to prevent cyber attacks on chemical process control systems. In industrial control systems, the exposure of cyber attacks is more critical to improve plant safety[27]. The study introduces the nonlinear system framework to understand the control design, process flexibility of cyber attack over the analysis of three control designs. [21] estimated the KNN classification with homomorphic encryption for detecting cyberattacks[28]. The presented hybrid method encrypt the data sample, then estimate how similar the ciphertext data samples are by using pearson similarity, Euclidean distance, and cosine similarity, kind of all together. After that, the data samples used for security classification were obtained through voting rules, or at least that's the idea behind it. The suggested method was applied to the IRIS dataset for classification, and the performance measures were estimated to be 97% accurate. Also, the comparative study was made based on the existing results and proposed results.

3. PROPOSED METHODOLOGY

The proposed research introduces an Artificial Intelligence assisted framework aimed at making Android based healthcare, yes mHealth apps, more safe from malware threats while still allowing secure technical messaging inside digital care environments. At the start, an Android malware dataset is built , it includes both good and harmful application samples, and then it gets preprocessed, just to clean it up a bit. After that, in order to protect the sensitive healthcare information, the dataset goes through an encryption–decryption routine, and this uses the new Dot Hash Swordfish Algorithm. This algorithm boosts data confidentiality, it does it via fast key generation and a safer kind of data transformation. Finally, the encrypted plus the decrypted datasets are saved in CSV form by using public and private key mechanisms, which is basically the way the study describes it. After the dataset gets decrypted, it s processed by the Deep Ensure AttackNet model, in order to spot and detect various categories of Android malware. Then, an Adaptive PropAdam optimization plan is used, where the benefits of Adam and RMSProp optimizers are mixed, it helps with better feature learning, faster convergence, and it enables more accurate decisions about whether apps are benign or malicious. After that, the whole framework is checked using common performance indicators such as accuracy, precision, recall, F1-score, computational time, and scalability . Overall, the methodology is meant to help reinforce cybersecurity for Android based healthcare applications, while still keeping secure technical exchanges between healthcare professionals, patients, and digital healthcare platforms. Figure 1 shows the workflow of this proposed framework is a kind of end to end process.

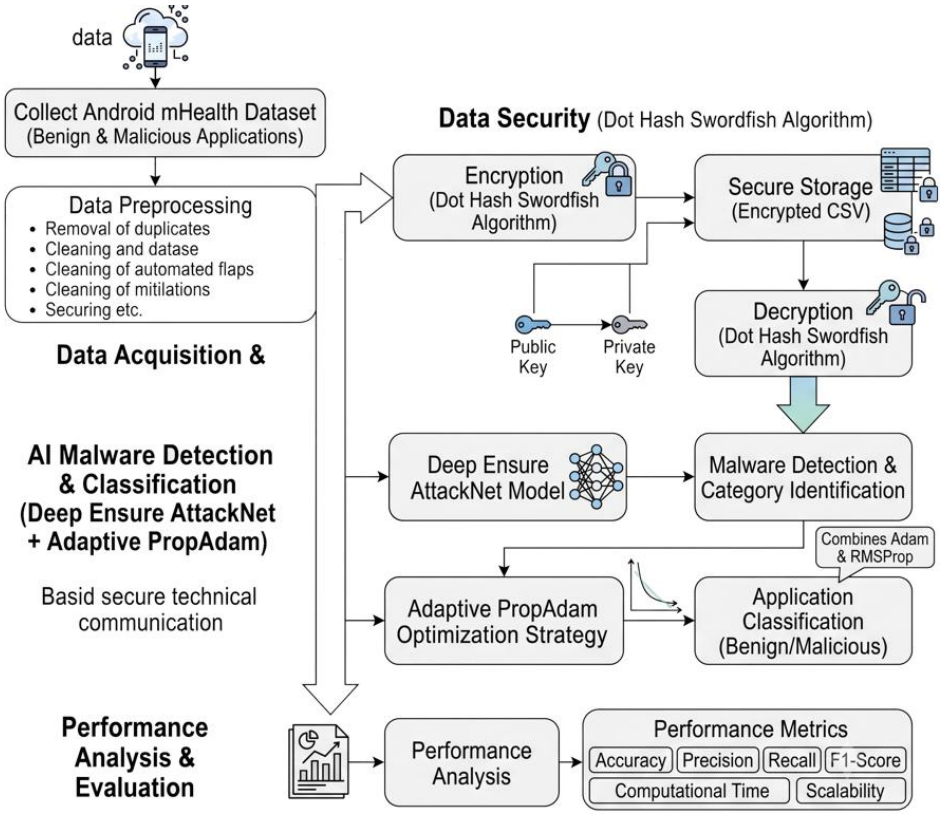

Figure 1: Flow of proposed method

3.1 Data security by using Dot hash swordfish algorithm

The dataset plays an vital role in the prediction of malware in android. The collected dataset will be encrypted to ensure data security. The Swordfish algorithm converts the data into cypher text by using a symmetric encryption key. The converted sequence of characters consists of a fixed length, which is called a "Hash". The hash function is utilized for validating file integrity, to encrypt sensitive inforamtion to generate unique identifiers. The encrypted information is known as ciphertext, which is received at the other end of the medium. This algorithm kinda gives out two different keys, one for the encryption process and one for decryption. Each sender gets a dual key set, the public key plus the private key. When you encrypt, you use the public key, but when you decrypt you rely on the private key. Using the public key, everyone can send information to the user, but only the user can decrypt the message using the private key.

To generate the key, "n" no of bits are used in modules. n_e is the no of bits in the $public\ exponent(e)$, and n_d is the $no\ of\ b$its in the filesystem.

Consider $n_e < \frac{n}{x}\ for\ x\ primes$.

Key genearation:

Step 1: Select a random (n_e)-$bit\ odd\ integer\ e$.

Step 2: Select (n/x $- n_e$)) odd integer $d_{p1a}$and calculate $E_{p1}= ed_{p1a}$.

Step 3: Select $random\ number\ E_{p1}$ with size (n_e+n_d=$\frac{n}{x}$) such that Greatest common devisor (k_{p1}, E_{p1}) = 1.

Step 4: Calculate d_{p1b} and d_{p1a} such that $E_{p1}\ d_{p1b}= k_{p1}\ p_{p1a}$ + 1, here $k_{p1} < d_{p1b} < 2k_{p1}$ and $E_{p1} < d_{p1b} < 2k_{p1}$

Step 5: Calculate $d_{p1}= d_{p1a}\ d_{p1b}$and p1 = $p1_a$ + 1. If $p1\ is\ not\ prime$, then goes to Step 3.

Step 6: Repeat steps 2 to 5 to compute the $primes\ p2,\ldots,px$

Step 7: Calculate modulus (N) as N = p1 ∗ p2 ∗ … ∗ px

$$Public\ key\ is\ (e, N)\ and\ private\ key\ (dp1\ , dp2\ , \ldots , dpx\ , p1, p2, \ldots , px)$$

Once the key is created, the cryptography process attained by using proposed Dot hash sword fish algorithm by using,

Encryption

$$Cipher\ (byte\ Msg, byte\ C1)$$

Begin

{

C1 = MSG Personal recovery key $mode\ \sigma$

End

}

Decryption

$$Decipher\ (byte\ C3', byte\ C2'\ ()\ State\ =\ C3;$$

Begin {

MSG op.sequencing(state) {

$$op.sequencing\ applied\ for\ opposite\ DNA\ sequencing\}\ MSG\_digital.decoding(state);$$

$$convert\ binary\ code(state);$$

C2′ = convert byte(state);

End

}

After the completion of encryption and decryption, the encrypted file and decrypted file are saved as a CSV file for further processing.

3.2 Attack detection by Deep Ensure attack net

Then, the attack is detected by a deep ensure attack net. The malicious nodes are differentiated using the trust threshold value . To remove the contribution of inappropriate nodes from the data network, the true value of the required node is computed with the help of a direct trust evaluation process. The presented detection mechanism message classification unit and decision making unit. In message classification unit, test sets are iterated repeatedly to verify the message quality securely. To reduce the test time on splitting rule, the traffic data is divided into several classes. In this work, the decision-making unit plays an important role. All weights of the aforementioned units are combined in this decision-making unit; this process is done to handle the attack decisions, determining the received messages is attacked or not.

In a deep network, the neuron performs a linear transformation through the weights and biases on its input as,

$$\boldsymbol{X\ =\ (Weight\ *\ input)\ +\ bias} \quad (1)$$

The stimulation function is used to nonlinearly transform the aforementioned result, thus resulting in

$$\boldsymbol{Y\ =\ Activation\ (E\ (weight\ *\ input)\ +\ bias)} \quad (2)$$

3.3 Classification using Adaptive prop Adam optimization

The classification of malware is attained by the adaptive prop Adam optimization technique. Adam stands for Adaptive Moment Estimation, which updates the weights for training to enhance the stochastic gradient descent. The proposed optimiser

improves the learning rate for each network weight and maintains the single learning for overall training. Adam is combined with an adaptive prop to solve the learning rate issues by the adaptive learning rate. This optimization technique involves the training of neural networks by modifying its parameters to reduce the loss of funciton value and it enhances the model. In this optimization, batch weight describes the number of trials used to update the system parameters. Learning rate defines the parameter determining the scale of updated weights model. The weight and bias are learnable parameters of model manage the signal between neurons.

Adam consists of the key component of exponential weighted moving averages to determine the estimation of both momentum of the gradient. For that, it utilizes the state variables,

$$v_t \leftarrow \beta_1 v_{t-1} + (1+\beta_1) g_t \quad (3)$$

$$s_t \leftarrow \beta_2 s_{t-1} + (1+\beta_2) g_t^2 \quad (4)$$

Here B1 and B2 are nonnegative weighting parameters, i.e.) $\beta_1 = 0.9$ and $\beta_2 = 0.999$. It is the variance estimation that moves slowly very much compared to the momentum term. When we initialize $v_0 = s_0 = 0$ for specific amount of bias into smaller values. That smaller value is achieved through $\sum_{i-0}^{t-1} \beta^i = \frac{1-\beta^t}{1-\beta}$ to renormalize terms.

Normalized variables are defined as,

$$\hat{v}_t = \frac{v_t}{1-\beta_1^t} \; and \; \hat{s}_t = \frac{s_t}{1-\beta_2^t} \quad (5)$$

Initially rescale the gradient to equivalent to that RMSProp to achieve,

$$g_t' = \frac{\eta \hat{v}_t}{\sqrt{\hat{s}_t + \varepsilon}} \quad (6)$$

RMSProp utilizes the momentum $\hat{v}_t$ compared than gradient itself. Furthermore, the rescaling attains small difference using $\frac{1}{\sqrt{\hat{s}_t + \varepsilon}}$ instead of $\frac{1}{\sqrt{\hat{s}_t + \varepsilon}}$.

We consider $\varepsilon = 10^{-6}$ for best swap between numerical fidelity and stability.

We obtained the form of,

$$x_t \leftarrow x_{t-1} - g_t' \quad (7)$$

From that equation the review of momentum and scale is clear in Adam optimization. The composition of these both terms provides straightforward RMSProp and the learning rate η permits us to regulate the stage length to report questions of convergence.

Adaptive prop Adam optimization algorithm

Input: θ – the initial parameters, $\beta_1, \beta_2 \epsilon\ [0,1)$ – the delay parameters, ϵ - step size, δ - a small amount constant for numerical stability, m – sample size, C – the cost function.

Output: : θ – the optimal values of the parameters

/* Initialize **momentum variables**

$$s \leftarrow 0$$

$$r \leftarrow 0$$

/* Start the iterations

While stopping criteria are not met, do

/* Sample a mini-batch

$\left\{\left(x_1^{(t)}, y_i^{(t)}\right)\right\}_{i=1}^{m} \leftarrow$ sample m examples from the training data.

/* Compute the gradient $(\hat{y}: prediction,\ y_i: ground\ truth)$

$$g \leftarrow \frac{1}{m}\nabla_\theta \sum_{i=1}^{m} C(\hat{y}_i, y_i)$$

/* Adaptive learning rate

$$s \leftarrow \beta_1 s + (1-\beta_1) g$$

$$\hat{s} \leftarrow \frac{1}{1-\beta_2^t} s$$

/* Adaptive learning rate

$$r \leftarrow \beta_2 r + (1-\beta_2)(g \odot g)$$

$$\hat{r} \leftarrow \frac{1}{1-\beta_2^t} r$$

/* Update the parameter

$$\theta \leftarrow \theta - \epsilon \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$$

end return θ

4. RESULTS AND DISCUSSION

This section discussed the proposed method of results for malware attack detection and estimated the performance measures which is applied by encryption based optimization technique.

4.1 Dataset

Traditional datasets for desktop malware detection are readily accessible. However, the availability of adequate datasets for analyzing Android permissions remains a significant challenge. Gathering and analyzing a large-scale collection of Android apps is difficult due to restrictions imposed by Android app repositories. Consequently, researchers and academics often rely on small datasets or can only gather a limited number of apps, leading to experiment results that are frequently non-reproducible.the proposed method applied on the dataset:2, https://www.kaggle.com/datasets/dannyrevaldo/android-malware-detection_dataset?select=Android_Malware_Benign.csv. That dataset contains the features of permission features, system features, security related features, Device Control Features, Miscellaneous Features etc. The aforementioned dataset assists as a valued resource for researchers to innovate and assess robust techniques for detecting and analyzing malware. Its utilization promises advancements in mobile security specifically tailored for the Android platform.

Table 2: No of columns presented in dataset

	ACCESS_ALL_DOWNLOADS	ACCESS_CACHE_FILESYSTEM	ACCESS_CHECKIN_PROPERTIES	ACCESS_COARSE_LOCATION	ACCESS_COARSE_UPDATES
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

5 rows × 328 columns

Overall, the dataset presents 328 columns. We filtered more required columns, as described in Table 1. Here, the number of columns presented is named as access all in downloads, filesystem, checkin properties, course location, and course updates.

Table 3: Data summary and data types in dataset

Dataset summary		Database types	
Data frame	values	Colum type	Count
No. of rows	4464	int32	327
No. of columns	328	string	1

Table 3 represetns the data summay and detatypes in dataset. Data summary contains data frame and data values. The dataset contains 4464 rows and 328 columns. In columns, which had 327 counts of int32 coulum type and 1 string type. From the dataset, the number of training and testing sets involved is as below,

Dataset Into Training and Testing Sets

X_train(3571, 327)

y_train(3571,)

X_test (893, 327)

y_test (893,)

4.2 Deep ensure Attack Network

Table 4 describes the sequential model of a deep-ensure attack network based on layer type, output shape, and number of parameters. From that table, we analyzed the total no of parameters for trainable and non-trainable. The total number of parameters is 670,881 (2.56 MB). Trainable parameters are 670,497 (2.56 MB), and non-trainable parameters are 384 (1.50 KB).

Table 4: Model: "sequential"

Stage	Layer	Configuration	Output Shape	Trainable Parameters	Purpose
Feature Extraction	Conv1D	64 filters	(None, 325, 64)	256	Extracts local sequential features
Normalization	Batch Normalization	–	(None, 325, 64)	256	Stabilizes learning and accelerates convergence
Regularization	Dropout	–	(None, 325, 64)	0	Reduces overfitting
Feature Extraction	Conv1D	32 filters	(None, 323, 32)	6,176	Learns higher-level feature representations
Normalization	Batch Normalization	–	(None, 323, 32)	128	Normalizes activations
Regularization	Dropout	–	(None, 323, 32)	0	Prevents overfitting
Regularization	Dropout	–	(None, 64)	0	Enhances model generalization
Dense Representation	Dense	32 neurons	(None, 32)	2,080	Generates compact feature embeddings
Reshaping	Flatten	–	(None, 10,336)	0	Converts feature maps into a one-dimensional vector

Normalization	Batch Normalization	–	(None, 32)	128	Improves training stability
Regularization	Dropout	–	(None, 32)	0	Reduces model complexity
Classification	Dense	64 neurons	(None, 64)	661,568	Performs high-level feature learning
Normalization	Batch Normalization	–	(None, 64)	256	Stabilizes dense layer outputs
Output Layer	Dense	1 neuron	(None, 1)	33	Performs binary malware classification

Total number of parameters: 670,881 (2.56 MB)

Training parameters: 670,497 (2.56 MB)

Other parameters: 384 (1.50 KB)

4.3 Performance metrics

The performance measures of proposed method is evaluated based on metrics of accuracy, recall, precision, F1-score and also ROC curve and the mathematical equations described from 8 to 12 respectively. In other words it is assessed through these indicators , to see how it behaves in practice, rather well even though the comparisons can be a bit indirect.

A) Accuracy: The amount of appropriately expected occurrences out of the total cases.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

B) **Precision**: The amount of true optimistic estimation, which is divided by the entire optimistic predictions, made.

$$\text{Precision} = \frac{TP+TN}{TP+FP} \tag{9}$$

C) **Recall:** The no of TP (True Positives) predictions separated into parts by the actual no of effectives.

$$\text{Recall} = \frac{TP+TN}{TP+FN} \tag{10}$$

D) **F1 Score**: A composite metric that balances both precision and recall to provide a complete measure.

$$\text{F1-Score} = \frac{2\ X\ Precision\ X\ Recall}{Precision+Recall} \tag{11}$$

E) **ROC-curve:** Evaluate the difference between the TP rate and the FP rate. AUC close to 1 specifies a moral model.

$$\text{ROC} = \frac{TP}{TP+FN} \tag{12}$$

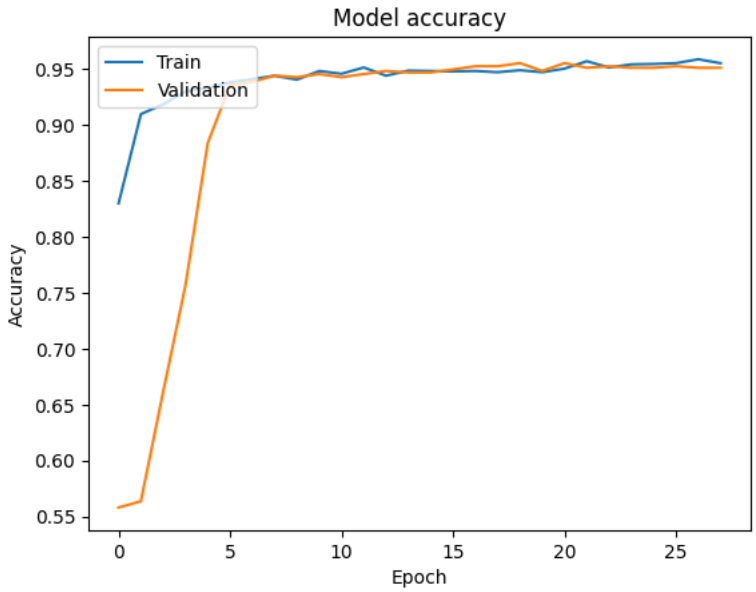

Figure 2: Proposed method of accuracy based on number of epochs

Figure 2 describes the proposed accuracy method based on the number of epochs. From this analysis, the accuracy has literally increased from the initial point of 0.55 with respect to the number of epochs. Finally, we achieved 0.97% accuracy in the proposed encryption-based optimization method for detecting the malware from the android devices.

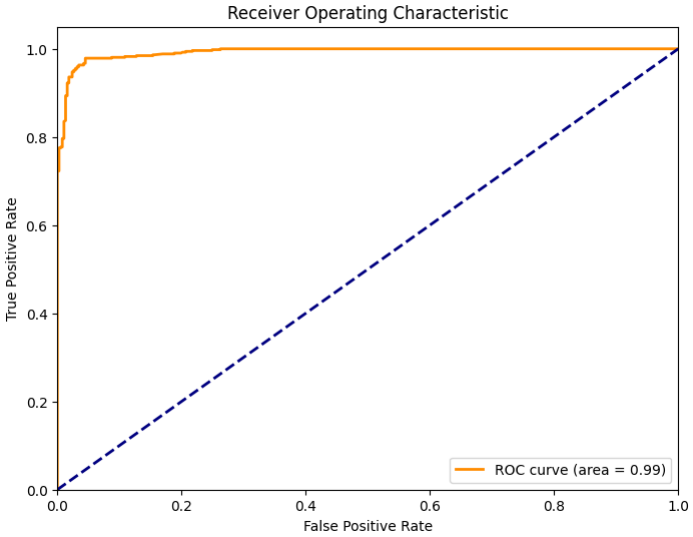

Figure 3: Proposed method of ROC curve

Figure 3 describes the proposed method of the ROC curve based on the TP and FP rates. From these results, the true positive rate literally increased by the false positive rate. Finally, we achieved 0.99% of the ROC curve from the proposed method. This favourable rate provided a high accuracy of malware detection rate on Android OS.

Table 5: Various performance measures of the proposed method

	Accuracy	Precision	Recall	F1-score	Support
Normal	0.97	0.97	0.95	0.96	377
Malware Attack	0.97	0.97	0.98	0.97	516

Table 5 defines the various performance measures of the suggested method based on the metrics of accuracy, precision, recall, f1-score, and ROC curve based on normal and malware attacks. The suggested approach for accuracy gives 0.97% , 0.97% precision, 0.95% recall, 0.96% F1-score, and 0.99% for the Roc curve. When the dataset is malware attacked, the accuracy comes out at 0.97%, 0.97% of precision, 0.98% of recall , 0.97% of f1-score, and 0.99% for the Roc curve.

Table 6: Comparison of proposed method accuracy with existing methods

Method	Accuracy
Proposed	**96.86%**
Recurrent neural network [1]	96.20%
DroidReinforcement Learning [22]	95%
Random forest [29]	96.2%
Random forest[30]	68%

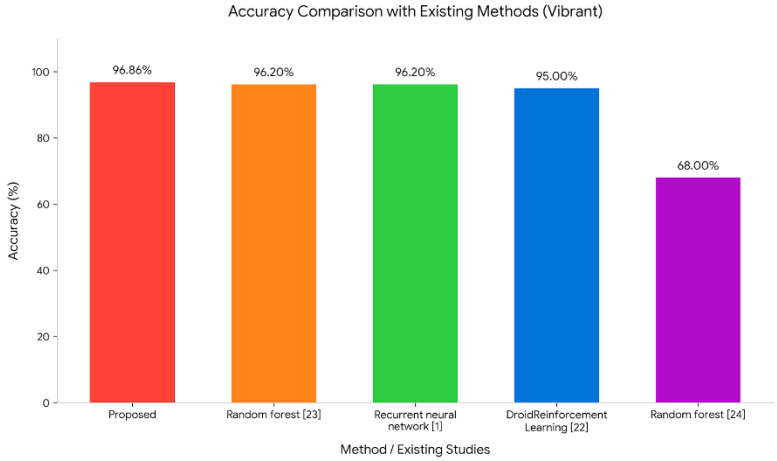

Figure 4. Comparison of accuracy with proposed method

Table 6 and figure 4 provides the comparison of proposed method accuracy with existing methods. Existing method provides the accuracy 96.20% and 84.5%. The proposed method provides the accuracy 96.86%. From this comparative analysis, we analyzed the proposed method had high accuracy compared than other existing methods.

5. CONCLUSION

The growing uptake of Android based mobile health mHealth apps has, in a rather direct way improved health service delivery via remote monitoring, telemedicine, and secure digital chat. Still, the fast increase in Android malware is a real risk to confidentiality , integrity and availability of private medical data. So, it becomes necessary to craft smarter malware detection approaches, so we can keep technical communication safe within digital healthcare ecosystems. This study proposed a sort of Artificial Intelligence assisted, adaptive framework that fuses encryption, deep learning, and adaptive optimization to make Android malware detection better especially for healthcare applications. In the middle of it all, the novel Dot Hash Swordfish Algorithm was used to help with data confidentiality, via an efficient encryption and decryption workflow, but also to keep things more secure in practice. Then, the Deep Ensure AttackNet model was used to effectively recognize malware attack types, like it can tell what kind of attack is happening. Also the Adaptive PropAdam optimization strategy was introduced, it kind of blends the benefits from Adam and RMSProp optimizers together, and as a result it improved learning efficiency, sped up convergence, and boosted the classification performance. It basically outperformed conventional approaches, not just for detection capability but also for computational efficiency, in other words it seemed more responsive and less costly.

Acknowledgement

The authors would like to express deep gratitude to their institution, for the research facilities and computational resources that were necessary . They also thank the reviewers and fellow colleagues for their useful suggestions, which in a way helped uplift the overall quality of this work .

Funding Sources

This research received no specific grant from any funding agency in the public or from any sectors.

Author Contributions

Conceptualization: S.D.; **Methodology:** S.C.; **Formal Analysis:** J.P.; **Investigation:** N.K.; **Data Curation:** R.P.; **Writing – Original Draft Preparation:** S.D; **Writing – Review & Editing:** B.P.M.; **Supervision:** S.D.

Conflict of Interest

The author declares that there are no conflicts of interest regarding the publication of this research work.

References

[1] O. N. Elayan and A. M. Mustafa, "Android malware detection using deep learning," *Procedia Computer Science,* vol. 184, pp. 847-852, 2021.

[2] Z. Ren, H. Wu, Q. Ning, I. Hussain, and B. Chen, "End-to-end malware detection for android IoT devices using deep learning," *Ad Hoc Networks,* vol. 101, p. 102098, 2020.

[3] A. Razgallah, R. Khoury, S. Hallé, and K. Khanmohammadi, "A survey of malware detection in Android apps: Recommendations and perspectives for future research," *Computer Science Review,* vol. 39, p. 100358, 2021.

[4] J. Gajrani *et al.*, "Effectiveness of state-of-the-art dynamic analysis techniques in identifying diverse Android malware and future enhancements," in *Advances in Computers*, vol. 119: Elsevier, 2020, pp. 73-120.

[5] J. Ferdous, R. Islam, M. Bhattacharya, and M. Z. Islam, "Malware resistant data protection in hyper-connected networks: A survey," *arXiv preprint arXiv:2307.13164,* 2023.

[6] H. Alkahtani and T. H. Aldhyani, "Artificial intelligence algorithms for malware detection in android-operated mobile devices," *Sensors,* vol. 22, no. 6, p. 2268, 2022.

[7] V. Syrris and D. Geneiatakis, "On machine learning effectiveness for malware detection in Android OS using static analysis data," *Journal of Information Security and Applications,* vol. 59, p. 102794, 2021.

[8] Y. Pan, X. Ge, C. Fang, and Y. Fan, "A systematic literature review of android malware detection using static analysis," *IEEE Access,* vol. 8, pp. 116363-116379, 2020.

[9] R. Feng, S. Chen, X. Xie, G. Meng, S.-W. Lin, and Y. Liu, "A performance-sensitive malware detection system using deep learning on mobile devices," *IEEE Transactions on Information Forensics and Security,* vol. 16, pp. 1563-1578, 2020.

[10] Y. A. Kadakia, A. Suryavanshi, A. Alnajdi, F. Abdullah, and P. D. Christofides, "Integrating machine learning detection and encrypted control for enhanced cybersecurity of nonlinear processes," *Computers & Chemical Engineering,* vol. 180, p. 108498, 2024.

[11] O. Alshaikh, S. Parkinson, and S. Khan, "Exploring perceptions of decision-makers and specialists in defensive machine learning cybersecurity applications: The need for a standardised approach," *Computers & Security,* vol. 139, p. 103694, 2024.

[12] A. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, "Machine learning for anomaly detection: A systematic review," *Ieee Access,* vol. 9, pp. 78658-78700, 2021.

[13] M. Choubisa, R. Doshi, N. Khatri, and K. K. Hiran, "A simple and robust approach of random forest for intrusion detection system in cyber security," in *2022 International conference on IoT and blockchain technology (ICIBT)*, 2022: IEEE, pp. 1-5.

[14] T. Markovic, M. Leon, D. Buffoni, and S. Punnekkat, "Random forest based on federated learning for intrusion detection," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, 2022: Springer, pp. 132-144.

[15] S. Rani, K. Tripathi, Y. Arora, and A. Kumar, "Analysis of Anomaly detection of Malware using KNN," in *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, 2022, vol. 2: IEEE, pp. 774-779.

[16] H. Alqahtani, I. H. Sarker, A. Kalim, S. M. Minhaz Hossain, S. Ikhlaq, and S. Hossain, "Cyber intrusion detection using machine learning classification techniques," in *Computing Science, Communication and Security: First International Conference, COMS2 2020, Gujarat, India, March 26–27, 2020, Revised Selected Papers 1*, 2020: Springer, pp. 121-131.

[17] R. Wazirali, "An improved intrusion detection system based on KNN hyperparameter tuning and cross-validation," *Arabian Journal for Science and Engineering,* vol. 45, no. 12, pp. 10859-10873, 2020.

[18] S. Joshi, Japanpreet, L. Rani, P. K. Sarangi, and V. P. Dubey, "Strengthening Cybersecurity: A Comparative Study of KNN and Random Forest for Spam Detection," in *International Conference on Recent Developments in Cyber Security*, 2023: Springer, pp. 337-350.

[19] S. Chen, Z. Wu, and P. D. Christofides, "A cyber-secure control-detector architecture for nonlinear processes," *AIChE Journal,* vol. 66, no. 5, p. e16907, 2020.

[20] U. Inayat, M. F. Zia, S. Mahmood, H. M. Khalid, and M. Benbouzid, "Learning-based methods for cyber attacks detection in IoT systems: A survey on methods, analysis, and future prospects," *Electronics,* vol. 11, no. 9, p. 1502, 2022.

[21] J. Liu, C. Wang, Z. Tu, X. A. Wang, C. Lin, and Z. Li, "Secure KNN classification scheme based on homomorphic encryption for cyberspace," *Security and Communication Networks,* vol. 2021, no. 1, p. 8759922, 2021.

[22] Y. Wu *et al.*, "DroidRL: Feature selection for android malware detection with reinforcement learning," *Computers & Security,* vol. 128, p. 103126, 2023.

[23] Dash, S., Padhy, S., Kumar, N. et al. Medisecure: a hybrid approach for enhancing multimedia data protection in healthcare. Cluster Comput 29, 75 (2026). https://doi.org/10.1007/s10586-025-05844-6

[24] Dash, S., Padhy, S., Suman, P., Mal, S., Malviya, L., Suman, A., & Kishore, J. (2025). Privacy-Preserving Diabetes and Heart Disease Prediction via Federated Learning and WCO. International Journal of Computational Intelligence Systems, 18(1), 1-26.

[25] S. Padhy, S. Dash, N. Kumar and G. Kumar, "A Secure COVID Affected CT Scan Image Encryption Scheme Using Hybrid MLSCM for IoMT Environment," in IEEE Access, vol. 13, pp. 82586-82609, 2025, doi: 10.1109/ACCESS.2025.3568448.

[26] Padhy, S., Dash, S., Kumar, N., Singh, S. P., Kumar, G., & Moral, P. (2025). Temporal Integration of ResNet Features with LSTM for Enhanced Skin Lesion Classification. Results in Engineering, 104201.

[27] Dash, S., Padhy, S., Suman, P., & Das, R. K. (2025). Enhancing Lung Cancer Diagnosis through CT Scan Image Analysis using Mask-EffNet. Engineering Access, 11(1), 92-107.

[28] Dash, S., Padhy, S., Devi, S. A., & Patro, K. A. K. (2023). An Efficient Intra-Inter Pixel Encryption Scheme to Secure Healthcare Images for an IoT Environment. Expert Systems with Applications, 120622. https://doi.org/10.1016/j.eswa.2023.120622

[29] M. M. Mijwil, "Malware Detection in Android OS Using Machine Learning Techniques," *International Journal of Data Science and Applications,* vol. 3, no. 2, pp. 5-9, 2020.

[30] Z. Li, H. Zhu, H. Liu, J. Song, and Q. Cheng, "Comprehensive evaluation of Mal-API-2019 dataset by machine learning in malware detection," *arXiv preprint arXiv:2403.02232,* 2024.