Recurrent Neural Network and Multi-Factor Feature Filtering for Ransomware Detection in Android Apps

Inam Ullah Khan[1], Fasih Ud din[2],Fida Muhammad Khan[1],Sohaib Saqib[3],Saadat ullah[3], Zeeshan Ali Haider[1],Shehr Bano[3]

[1]Department of Computer Science, Qurtuba University of Science & Information, Technology, Peshawar, Pakistan.

[2] Department of Computer Science & IT University of Lakki Marwat, Pakistan

[3]Faculty of Computing Gomal University, Dera Ismail Khan, Khyber Pakhtunkhwa(KPK), Pakistan

*** Correspondence:** Inam Ullah Khan: inam1software@gmail.com

Citation | Khan. I. U, Din. F. U, Khan. F. M, Shoaib. M, Ullah. S, Haider. Z. A, Bano. S, "Recurrent Neural Network and Multi-Factor Feature Filtering for Ransomware Detection in Android Apps", IJIST, Vol. 6 Issue. 3 pp 1021-1030, July 2024

Received | June 18, 2024 **Revised |** July 20, 2024 **Accepted |** July 21, 2024 **Published |** July 22, 2024.

The market is flooded with Android Software (apps), and at the same time that number is growing quickly, and so are the many security exploits that take advantage of such apps. The effectiveness of traditional defensive systems is at risk due to the growing diversity of Android malware. This situation has sparked significant interest in improving malware detection accuracy and scalability for smart devices. By examining the Long Short-Term Memory (LSTM) method, we have developed an effective deep learning-based malware detection model for enhanced Android ransomware detection. For feature selection, eight different methods were applied. By comparing the outcomes of all feature selection procedures, we used a simple majority vote process to choose the 19 crucial characteristics. The Android Malware dataset (CI-CAndMal2017) and common performance metrics were used to assess the proposed technique. With a detection accuracy of 97.08%, our model surpasses existing approaches. We advocate our proposed method as effective in malware and forensic analysis based on its remarkable performance.

Keywords: Ransomware Malware, Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), AndriodSecurity.

Introduction:

Applications for every industry are now accessible in the Google Play Store, including those related access utilities, news, periodicals, books, movies, music, entertainment, fitness, health care, productivity, education, business, and finance. This is now achievable because of the abundance of smartphone applications on the market. More than two-thirds of Smartphone users run the Android operating system. While the number of Android users grows quickly and there are a ton of apps available to provide consumers with a variety of features, the Android operating system continues to be a top target for hackers. Mobile gadgets, such as smartphones and tablets, have grown exponentially in popularity in recent years and are now considered basic pieces of personal equipment. Users gradually access smart devices and personalized computing services, which gives Android applications a competitive advantage in their quick growth and popularity. According to Statist, there were 2.1 billion Smartphone users in 2016 and that number increased to 5 billion in 2019 [1].

The appeal of creating Android applications is in offering ones with plenty of features. Android's open-source philosophy and strong tolerance for the verification process have made it a more well-liked platform to work on for developers and users alike. According to the reports, Android's share of the Smartphone market for operating systems recently reached 85.9% in 2018 [2]. As the industry's dominant operating system, Android must contend with possible malware threats that might compromise its security. Malicious software called malware is designed to crash the operating system or steal private data. The authors use updated methods to retrieve private information like as contact lists, IMEIs, and other necessary content-specific data to infect mobile applications and jeopardize smartphone security. Malware of the ransomware variety is specifically developed to prevent users from accessing any computer resources until the attacker receives the demanded payment. Even with a significant focus on protecting users from potential malware, malware detection is a crucial issue that needs to get enough attention. The practical method for protecting users against misleading and repugnant content is to provide defense against malware infection

Behavior serves as a defining aspect of malware feature factors. Very basic structures for malware resistance are believed to be conventional code analysis [3] and signature-based detection [4]. Unfortunately, due to the requirement that threats be properly identified to produce signatures, many approaches fall short of satisfying the security requirements. Several researchers have worked to analyze and find malware using static [5] and dynamic behavior analysis [6] techniques. In contrast to dynamic analysis approaches, which monitor semantic properties while an application is operating in a controlled environment, static analysis refers to the process of monitoring static features of applications without the need for program execution. These methods are effective for identifying a specific threat and malicious activity if it falls under the umbrella of a known class of Android malware. Unfortunately, they struggle to identify unidentified assaults. The effective and safe malware detection system employing deep learning is regarded as the optimum configuration to solve the limitations of prior work [7]. Deep learning is a potential method for the forthcoming era of Android malware detection since it can, as a result, discover connections in the data and extract richer representations from it. Also, it may be used to accurately locate zero-day malware samples. We developed a deep learning-based ransomware detection model to identify Android malware.

The objectives of this research study are as follows:

- To expand and integrate a deep learning-based malware identification system into the Android platform, supplying flexible and conformable protection in opposition to malicious threats.

- To leverage the strong performance of Long Short-Term Memory (LSTM) networks for reinforcing malware detection, while decreasing computational complexity through statistics pretreatment and tremendous feature choice.
- To conduct a substantial sensible examination of the CICAndMal2017 Android malware dataset to evaluate the proposed technique's efficacy.
- To assist the proposed technique as effective for each malware detection and forensic analysis, demonstrating its practical utility and robustness.

The novelty of this study is characterized by the following key aspects:

- **Integration of RNN with Multi-Factor Feature Filtering:** This research extends state-of-art Recurrent Neural Networks (RNNs), and more especially Long Short-Term Memory (LSTM) networks, with multi-factor feature screening techniques. This has improved the detection of ransomware in Android applications with a novel approach.
- **Simple Majority Voting for Feature Selection:** The decision to use a relatively simple majority voting approach to determine the nineteen (19) most important features from eight different feature selection methods enhances the study's robustness.
- **Real-Time Ransomware Detection Framework:** The proposed solution employs a three-step real-time evaluation procedure that encompasses encryption assessment, foreground analysis, and format evaluation which also puts forward a novelty to the real-time malware detection techniques.

The remainder of the article is structured as follows: SECTION II deals with background information on existing studies. SECTION III is the description of the system that entails an elaboration on its design and a description of each part of the system. Implementation analysis is the focus of SECTION IV. The findings from the experiments are enumerated in SECTION V, and the study's implications as well as the key findings of this research are discussed in SECTION VI.

Related Work:

In detecting Android malware, several investigations have been carried out. Malware has been discovered in numerous study papers by utilizing specific traits of Android applications. We only highlight a handful of recent, closely linked projects on adware for Android that are powered by deep learning. To detect malware groups, [8] developed the artificial neural network-based MalDozer technology. The DEX file holds the system's input-only raw API call patterns. We carefully examine a large number of databases. The MalDozer successfully detected patterns, demonstrating its efficacy and feasibility. The author aimed to improve the precision and accuracy of malware detection by using a hybrid system based on deep auto-encoders[9]. The pre-training step and various CNN structures for malicious behavior employ the autoencoder. A precise test conducted on a dataset with a total of 23000 records, both legitimate and malicious, showed that it achieves a high level of accuracy of 99.80%. To train the model to identify the fine-grained malware families a deep neural network-based method was created by D. Li and colleagues and applied to static properties. An extensive feature set that was manually extracted was utilized to evaluate the model's performance, which was 97% accurate in its identification.

Deep Refiner [4] suggested an automated feature extraction system for malware classification that made use of deep neural networks with several hidden layers. In the first detection phase, the algorithm extracted XML values from XML files. Additionally, applications those deep neural networks utilized as inputs are imported by Deep Refiner as vectors[7]. In addition to 62,915 malicious apps and 47,525 good apps, Deep Refiner was evaluated on a sizable sample. Results showed that 97.74% accuracy was achieved by Deep

Refiner in detecting adware. In [5], the author provided a structure for examining Android API calls made by a program. The authors created a classification scheme for the (CNN) sequence that directed convolution assignments along the sequence while learning larger forms for each zone as the convolution frame descended the sequence. Several CNN layers were also used in the CNN architecture to gradually elevate and enlarge features from localized tiny characteristics. The model, which was evaluated using a dataset with 1016 APK entries, achieved 99.4% accuracy. The detection method proposed by Liang et al. [6] is solely based on rule sets specified through rights stated in manifest files based on requests from both malicious and innocuous apps. With a success rate of 96% for malicious and 88% for innocuous identification, the technique beats. [10] proposed extending the work by combining user rights from manifest files with system clearances in a two-layered permission-based detection approach. With 98% of findings correctly identifying malware, it has greatly improved. [11] introduced the App context framework, which analyzes static programs and pulls helpful security-related sensitive behaviors to guide applications in differentiating between legitimate and fraudulent behavior. With an accuracy score of 87.7%, the framework accurately detects 192 malicious apps.

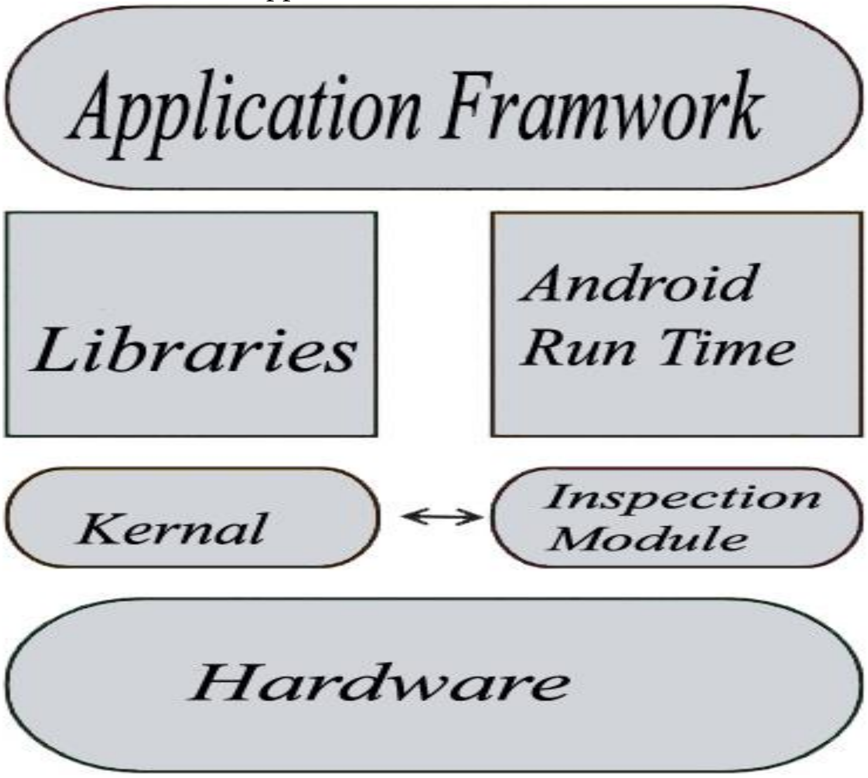


Figure 1. Shows the examination module operating in kernel mode.

Methodology:

The proposed method, which relies on extended short-term memory, is effective, scalable, and capable of detecting malware on Android devices. (LSTM). Since Android OS is entirely built on Linux, the proper placement of the proposed model can be seen there. This makes it very simple for us to integrate the security module shown in "Figure 1". The method that has been demonstrated distinguishes between malicious and helpful behavior using an LSTM architecture that is based on deep learning. The preprocessing and recognition phases made up the system. Using 8 distinct machine learning filtering methods, the preprocessing step includes the extraction of key features. Additionally, the discovery step employs the DL algorithm to find malicious activity in Android applications.

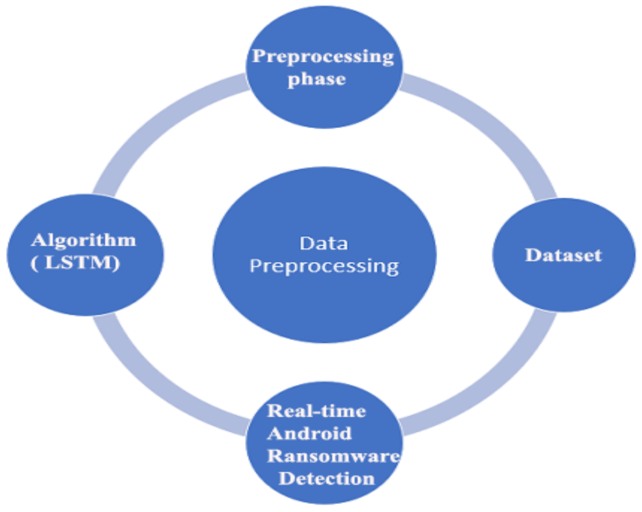


Figure 2. Proposed Architecture

Preprocessing phase:

The high dimensional characteristics are included in Android apps, and pretreatment of the data is necessary to determine the most significant and pertinent information. Enhancing malware detection accuracy and lowering the computational cost for deep learning model training are the goals of feature filtration approaches that should be used before classification. For our research, we have selected the traits that have the highest rankings in ChiSquare, CVAttribute Eval, Gain Ratio Eval, Knowledge Evaluations of Gain, Ease, Importance, and Symmetrical UncertAttributeand One R Attribute Eval,. Weka, a program created specifically for data mining activities, is used to experiment with feature filtration. There were 40000 samples altogether, with a total of 20,000 benign and 20,000 ransomware signatures. After analyzing the outcomes of 8 different feature filtration procedures, we selected the 19 crucial characteristics that received the most support from the algorithms. Our classification of ransomware and the 20th characteristic is benign.

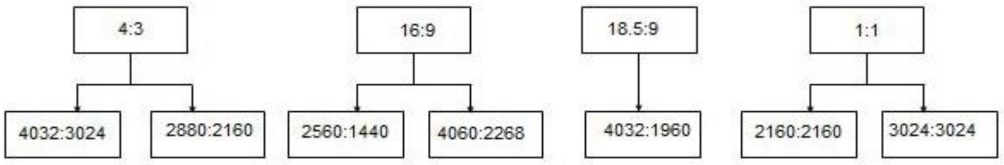
Algorithm (LSTM):

Hoch Reiter and Schmidt Huber have suggested the architecture known as (LSTM).The LSTM component of a recurrent neural network (RNN). An LSTM unit consists of one cell, input, output, and ignore stages. Three gates control the flow of information into and out of a cell, and the cell stores the value for various amounts of time. LSTM networks are excellent for categorizing, processing, and making predictions while keeping track of time data. Moreover, RNN neurons are built with layer communication in mind to improve network performance as a whole.

Real-time Android Ransomware Detection:

A three-step analysis procedure is proposed to identify whether or not a program is ransomware. A key consideration in this strategy is whether the user intended for encryption to be enabled on their device. Depicts the method's total procedure. Important files that have been predefined by the user and were encrypted using the Shannon entropy value in real time are the first files to be watched and examined for encryption analysis. The procedure advances to the following phase if it is discovered that a crucial file is encrypted; otherwise, it keeps watching. Second, foreground analysis looks for Activity Manager Activity, which is the activity of the encrypting program, at the top of the activity stack. This step's objective is to guarantee that the foreground initiates the encryption process. In other words, it verifies that the device's screen displays the encryption process. The procedure advances to the next phase if the program's activity is at the top of the stack; otherwise, it pauses the application and notifies the user. The information from the manifest, layout, and XML files is extracted in the layout analysis phase. One presumes that common encryption programs display a list of files, texts, and buttons. Static analysis can achieve this by generating hint strings that appear on the user's screen. When these elements are present, the step is completed and the user is alerted; if not, the application is stopped and the process resumes at the beginning.

Dataset:



The CICAndMal2017 Android malware dataset is the source of our system's design [17]. The information is acquired by running both safe and malicious apps on a real smartphone rather than through an emulator. The four main types of malware samples included in the CICAndMal2017 are adware, ransomware, scareware, and SMS malware. There are 1509550 records in the entire dataset. 460976 of these include malicious ransomware intentions, 1048574 of which are benign. The dataset sample comprises 83 attributes overall, including a Tag (Attack, Normal). There are 84 numbers in all for the features. As far as we are aware, our proposed malware detection technique is the first to test its efficacy on the CI-CAndMal2017 dataset.

Analysis and Implementation:

Evaluation Matrices:

To decide on a model on several common parameters, such as testing recall, confusion matrix, precision, Training score, and accuracy F1-Score, we analyzed our proposed performance measures. False Negative is also identified using the False Positive, False Negative, True Positive, and Confusion matrix, which is then used to find more common components. To assess the models' efficacy, recall, and the right metrics were applied. Precision, which is measured in this study, (1) was the optimum measure to take into account. Here is because the dataset we created has an equitable distribution of classes. Remember that the True Positive Ratio (TPR) measures the proportion of positive samples (Ransomware programmers) that are accurately predicted compared to all samples, as demonstrated in (2). According to calculations made in, precision is the ratio of correctly predicted events (by ransomware programmers) to all previously predicted events, whether correctly or incorrectly.

Accuracy= TP+TN/TP+FP+TN+FN **Eq.1**

Recall= TP/TP+FN **Eq.2**

Precision= True Positive (TP)/True Positive+False Positive **Eq.3**

The size of the generated predictive model was also examined as an additional indicator (in KB). The size of the model may cause the intrusion detection system to have concerns, as the majority of Android devices are portable and have limited resources (IDS). In some circumstances, especially when it comes to Internet of Things (IoT) applications, this predictive model has to be installed on the Android Smartphone. Consequently, prioritizing the model's accuracy, complexity, or compromise while selecting a suitable predictive model.

Experiments Setup:

Python has served as a development tool in our experiments. Environment. Data has been divided into an 80:20 distribution for testing and training objectives. The division of the dataset's overall 80% is equivalent to 1207640 records, where LSTM is given for training. While 301910 documents make up about 20% of the total, testing objectives. To implement the LSTM network, the Scilearn library and Keras were employed. The training community comprises six LSTM layers with the following values: 100, 100, 50, 20, 10, 1. Neuronal activation and function exist in reality. Our sigmoid was used as the activation function on the last layer. The settings are 1024 epochs and 50 for the batch size. We have selected the number of epochs and batch size for LSTM by our Repeat trials several times to achieve the best results.

Experimental Results:

This section presents the outcomes of our experiment. The percent of results obtained has been calculated to make the graphs look nicer. Accurate detection is the main evaluation factor for the model created by artificial intelligence. The detection precision of the LSTM is shown in "Figure 3" using test data. A 97.08% accuracy rate for LSTM has been demonstrated. When the dataset size was large, it provided exceptional results.

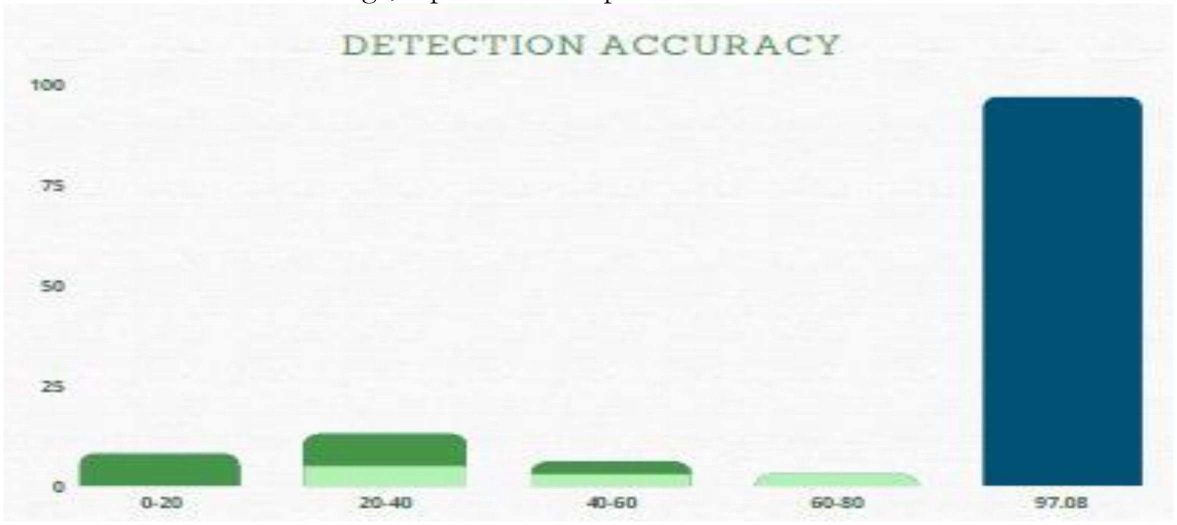


Figure 3. Proposed Model Accuracy.

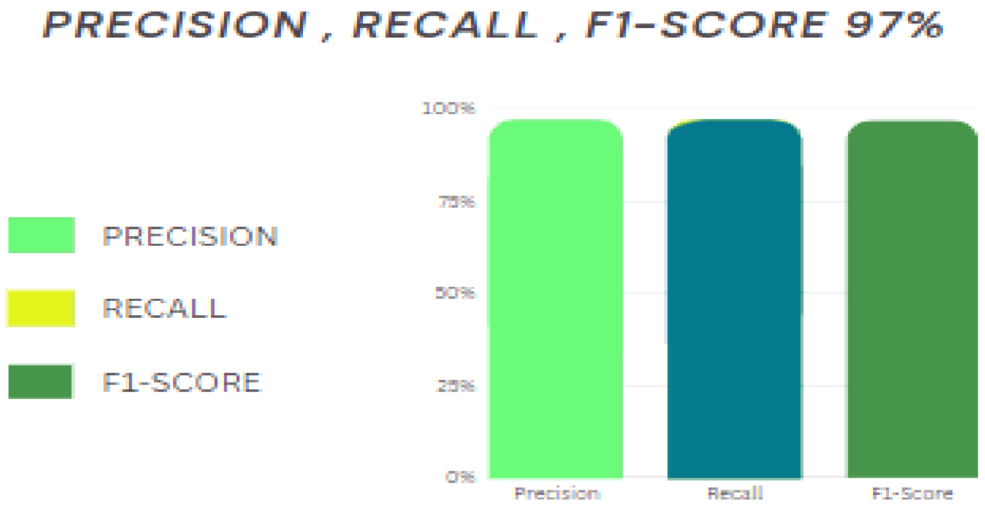


Figure 4. Classification Report of LSTM model

We have also observed the LSTM described in "Figure 3"'s precision, recall, and F score. Ransomware is classified 97% properly, according to precision. Although Recall claims that their methodology classifies 97% of malware as harmful. F1 Measure demonstrates a balance between accuracy and recall. For a better evaluation of the model, we have additionally observed the approach for Matthews Correlation Coefficient (MCC), Negative Predictive Value (TPR), and True Negative Rate (TNR), which is shown in "Figure 3". The better the value between 0 and 1 the more accurate the forecast as a balanced Measure TNR, NPV, and MCC have been estimated to have respective values of 0.94, 0.95, and 0.93.

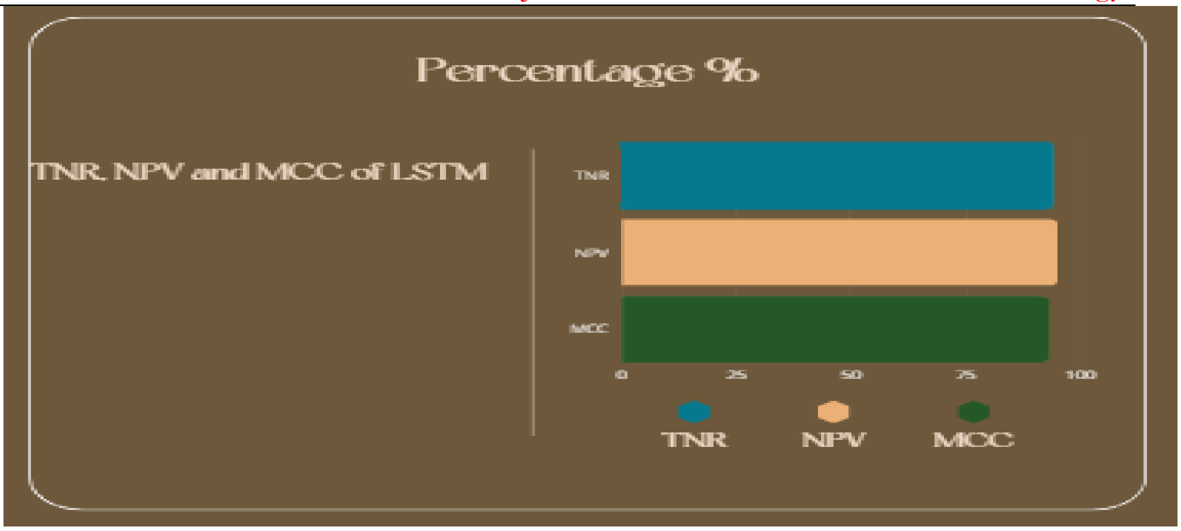


Figure 5. The LSTM's TNR, NPV, and MCC

According to "Figure 5", real figures for the False Positive Rate (FPR), False Discovery Rate (FDR), False Omission Rate (FOR), False Negative Rate (FNR), and False Positive Rate (FOR) for the detection of ransomware in the Android operating system are 0.018, 0.05, 0.02, and 0.04 respectively.

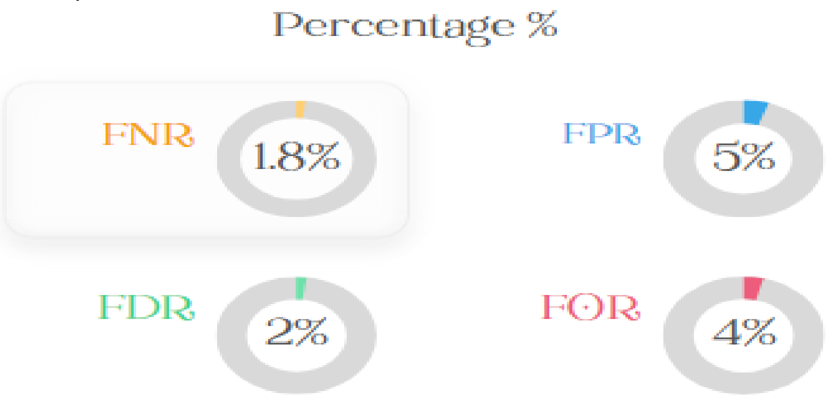


Figure 6. The FNR, FPR, FDR, and FOR of the LSTM

ROC is a graph that compares the true positive rate and false positive rate; it shows that the more AUC (Area Under Curve) there is, the better the system performs.
In "Fig. 7," the ROC curve for the LSTM is described.

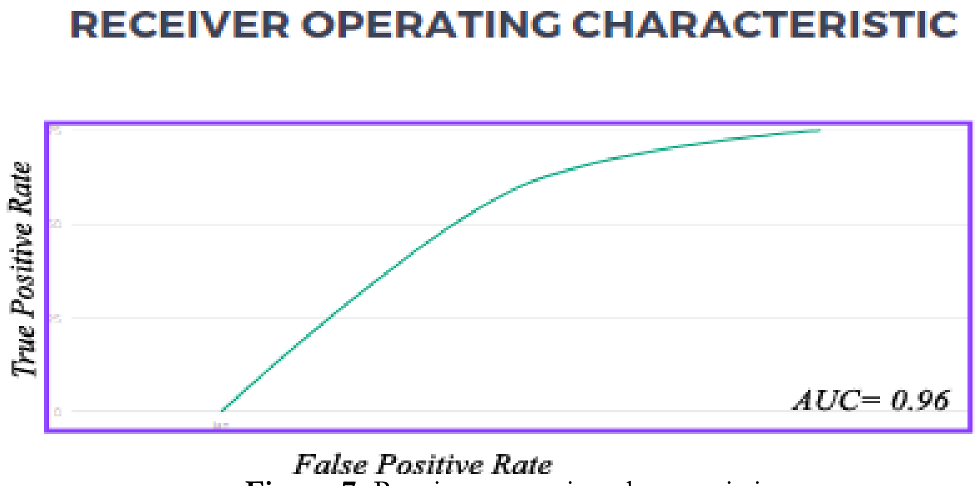


Figure 7. Receiver operating characteristic

Discussion

The results show that the proposed LSTM-based fully malware detection model is exceptionally capable of detecting ransomware on Android packages, driving a brilliant detection accuracy of 97.08% in the CICAndMal2017 dataset by integrating multi-description attribute filtering with the LSTM version, when using a simple majority opinion technique for feature selection guarantees that the best applicable features are used Compared to state-of-the-art methods, the proposed model is not good enough to reveal high accuracy however it additionally exhibits strong performance on various assessment criteria. Ratings, in conjunction with MCC, verified its real usefulness and reliability for real-time malware detection and forensic evaluation in Android environments. Test's approach gives outstanding advances in malware detection generation, delivering a robust machine of defenses in the route of evolving ransomware threats

Conclusion:

In this research study, we examined the shortcomings of two earlier investigations into the real-time detection of new/variant/unknown ransomware on Android. We then put out a real-time ransomware detection system and the specifications needed to use it. This technique can stop new/variant/unknown ransomware in an Android environment with a reduced performance impact. The deep learning architecture based on LSTM exhibits a strong ability to recognize ransomware on the Android platform with greater accuracy and specificity. The proposed model's detection accuracy is 97.08%, showing the wide capacity of LSTM to anticipate complex malware. In our upcoming study, we intend to improve the performance of the LSTM by using different additional deep learning models and comparing them for effective malware detection protection.

Acknowledgment: This work is supported by the IT Lab, Department of Computer Systems Qurtuba University of Science & Information, Technology, Peshawar.
Funding: This research received no external funding.
Conflict of Interest: The authors declare no conflict of interest.

References:

[1] M. Alazab, R. A. Khurma, D. Camacho, and A. Martín, “Enhanced Android Ransomware Detection Through Hybrid Simultaneous Swarm-Based Optimization,” Cognit. Comput., pp. 1–15, Jun. 2024, doi: 10.1007/S12559-024-10301-4/METRICS.

[2] C. B N and B. S H, “Revolutionizing ransomware detection and criticality assessment: Multiclass hybrid machine learning and semantic similarity-based end2end solution,” Multimed. Tools Appl., vol. 83, no. 13, pp. 39135–39168, Apr. 2024, doi: 10.1007/S11042-023-16946-X/METRICS.

[3] A. K. Al Hwaitat et al., “Overview of Mobile Attack Detection and Prevention Techniques Using Machine Learning,” Int. J. Interact. Mob. Technol., vol. 18, no. 10, pp. 125–157, May 2024, doi: 10.3991/IJIM.V18I10.46485.

[4] D. Soi, A. Sanna, D. Maiorca, and G. Giacinto, “Enhancing android malware detection explainability through function call graph APIs,” J. Inf. Secure. Appl., vol. 80, p. 103691, Feb. 2024, doi: 10.1016/J.JISA.2023.103691.

[5] E. Calik Bayazit, K. Sahingoz, and B. Dogan, “Tehnički vjesnik,” vol. 30, pp. 787–796, 2023, doi: 10.17559/TV-20220907113227.

[6] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, “Protecting Android Devices From Malware Attacks: A State-of-the-Art Report of Concepts, Modern Learning Models and Challenges,” IEEE Access, vol. 11, pp. 123314–123334, 2023, doi: 10.1109/ACCESS.2023.3323396.

[7] A. 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, Vol. 24, Page 189, vol. 24, no. 1, p. 189, Dec. 2023, doi: 10.3390/S24010189.

[8] A. R. Zaidi, T. Abbas, H. Zahid, and S. A. Ramay, "Effectiveness Of Detecting Android Malware Using Deep Learning Techniques," J. NANOSCOPE, vol. 4, no. 2, pp. 1–21, Nov. 2023, doi: 10.52700/JN.V4I2.90.

[9] Q. M. Yaseen, "The Effect of the Ransomware Dataset Age on the Detection Accuracy of Machine Learning Models," Inf. 2023, Vol. 14, Page 193, vol. 14, no. 3, p. 193, Mar. 2023, doi: 10.3390/INFO14030193.

[10] A. Mahindru and A. L. Sangal, "FSDroid:- A feature selection technique to detect malware from Android using Machine Learning Techniques: FSDroid," Multimed. Tools Appl., vol. 80, no. 9, pp. 13271–13323, Apr. 2021, doi: 10.1007/S11042-020-10367-W/TABLES/21.

[11] I. Almomani et al., "Android Ransomware Detection Based on a Hybrid Evolutionary Approach in the Context of Highly Imbalanced Data," IEEE Access, vol. 9, pp. 57674–57691, 2021, doi: 10.1109/ACCESS.2021.3071450.