Iraqi Journal of Information and Communications Technology(IJICT)
Vol. 9, Issue 1, April 2026
www.ijict.edu.iq
ISSN:2222-758X
e-ISSN:2789-7362

PERFORMANCE ANALYSIS OF FEATURE SELECTION AND CLASSIFICATION FOR ANDROID MALWARE USING MACHINE LEARNING AND DEEP LEARNING MODELS

Tabarek K. Al-Bayati [1], Lahieb M. Jawad [2], Ghazali Bin Sulong [3]
[1] Department of Computer Networks Engineering, College of Information Engineering, Al-Nahrain University, Jadriya, Baghdad, Iraq
[2] Department of Cyber Security Engineering, College of Information Engineering, Al-Nahrain University, Jadriya, Baghdad, Iraq
[3] University Technology Malaysia, Johor, Malaysia
tabarek.mit23@ced.nahrainuniv.edu.iq [1], lahieb1978@gmail.com [2], ghazali558@gmail.com [3]
Corresponding Author: **Ghazali Bin Sulong**
Received:07/03/2025; Revised:29/04/2025; Accepted:10/07/2025
DOI:10.31987/ijict.9.1.323

***Abstract*- Android malware creates a growing security risk due to the increasing number of applications that work on Android platform nowadays. The need for effective detection methods has made the use of machine learning and deep learning a viable solution. This study presents a comparison between different Machine Learning (ML) and Deep Learning (DL) models including Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), Gated Recurrent Unit (GRU), Support Vector Machine (SVM), K Nearest Neighbor (KNN), and Logistic Regression (LR) on two datasets Android Malware Detection (AMD) and CICMalDroid2020 to evaluate their performance using a unified architecture with the same parameters. Random Forest method is used to select features from the AMD dataset, by calculating the most significant features that have a direct impact on the process of detecting Android malware samples, thereby increasing models' accuracy. Result for both datasets shows that DL models, especially BiLSTM, outperform other models with accuracy metric that reach 99.77% with full features, 99.88% with selected features for AMD dataset and 89.70% for CICMalDroid2020 dataset.**

keywords: **Security, Android platform, Deep learning, Machine learning, Random Forest.**

I. INTRODUCTION

In recent years, Android platform has become the primary goal for malware attacks because of its open-source nature, and the exponential growth of Android devices. Malwares can introduce serious threats on users' security and integrity by taking different forms including Adware, Ransomware, Trojan, Spyware, Virous, Worm, Botnet, and Scareware, each with its different mechanism and effect. Many studies have been carried out for Android malware detection through learning and classification process to protect Android devices, where the existing form of malware can easily evade detection using different hiding techniques [1].

Traditional detection techniques for Android malware can be classified into static, dynamic and hybrid features analysis methods. Where static analysis depends on features extracted statically from the APK file to detect malware, without running the application. While dynamic analysis involves running the application in a sandbox or simulator to detect malware, based on the application behavior at run time. Finally, hybrid analysis combines both techniques to overcome their problems and maximize their advantages. For example, static analysis cannot catch obfuscation code while it easily detected with dynamic analysis. Alternatively, static analysis is safer than dynamic analysis since it does not affect the mobile device and less in both time and cost consumption [2].

ISSN:2222-758X
e-ISSN:2789-7362

However, the rapid growth of methods that attackers usually use to evade detection such as encryption, packaging, and code hiding, increased the need for advanced detection techniques. Machine Learning (ML) and Deep Learning (DL) models have been broadly used in detecting Android malware in many recent studies. In ML model the detection of Android malware can be done automatically by building models that identify patterns from large amount of data without being specifically programmed [3]. While in DL, which is subset of ML with many layers, the learning model can automatically extract features from raw data, where it shows very good results when massive amount of data is involved, thereby improve the detection accuracy of complex malware [4], [5].

However, both ML-DL approaches have shown a promising result in the field of Android malware detection by many recent studies [2]. In this paper, a comparative analysis between six models is introduced, including: Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), Gated Recurrent Unit (GRU), Support Vector Machine (SVM), K Nearest Neighbor (KNN), and Logistic Regression (LR). All models were evaluated and tested on two datasets, Android Malware Dataset (AMD) and CICMalDroid2020 dataset. This study highlights the potential of ML-DL models in advancing Android malware detection in the field of cybersecurity.

The rest of this paper is arranged as follows. Section II shows a literature review on Android malware detection. Section III presents an overview of the used datasets and the proposed methodology. Section IV presents the experimental result for all 6 models. Section V ends with the conclusion of this work.

II. RELATED WORKS

Authors in [6] introduced Android malware classification and recognition approach called AAMD-OELAC using ensemble learning process with LS-SVM, KELM, and RRVFLN ML models. Result of malware detection was improve using HPO algorithm that works on tuning the optimal parameters for the three models. The simulation results show the AAMD-OELAC method supremacy over other existing approach.

In [7], Android malware detection approach is introduced using different sets of ML models based on permission-based features for static analysis. Different datasets were used to collect samples, with total of 5000 samples from Drebin dataset, and 5000 samples from Androzoo dataset. The number of extracted features were minimized using feature selection method to determine the most important features for the detection process. The results showed that Random Forest model outperform other models with accuracy of 91.59%.

Authors in [8] developed different ML-DL models based on both static and dynamic analysis methods, with a detailed comparation between all models. Results showed that LSTM model with static analysis achieved accuracy of 98.8% and CNN-LSTM model for the dynamic analysis achieved accuracy of 95.3%. These results demonstrate that the models achieve higher accuracy than the models mentioned in the literature.

Authors in [9] introduced an Android malware detection system with the use of DL models. CICInvesAndMal2019 dataset is used with 8115 features to evaluate LSTM, BiLSTM, and GRU models for malware detection system. Results shows that BiLSTM model outperform other models with accuracy of 98.85%.

Authors in [10] introduced Recurrent Neural Network (RNN) model for malware detection based GRU. Application

ISSN:2222-758X
e-ISSN:2789-7362

Programming Interface (APK) and Permissions features were extracted from CICAndMal2020 dataset, with static analysis for the Android application. Results showed that the achieved accuracy of 98.2% for GRU model outperform many other methods.

III. METHODOLOGY

In this section a comprehensive approach is introduced to compare the performance of different ML-DL models for Android malware classification. As shown in Fig. 1, a detailed information are included of collecting data from multiple dataset, extracting and selecting features from row dataset, applying ML and DL models on full and selected features, and finally evaluation matrixes used to evaluate these models' performance.

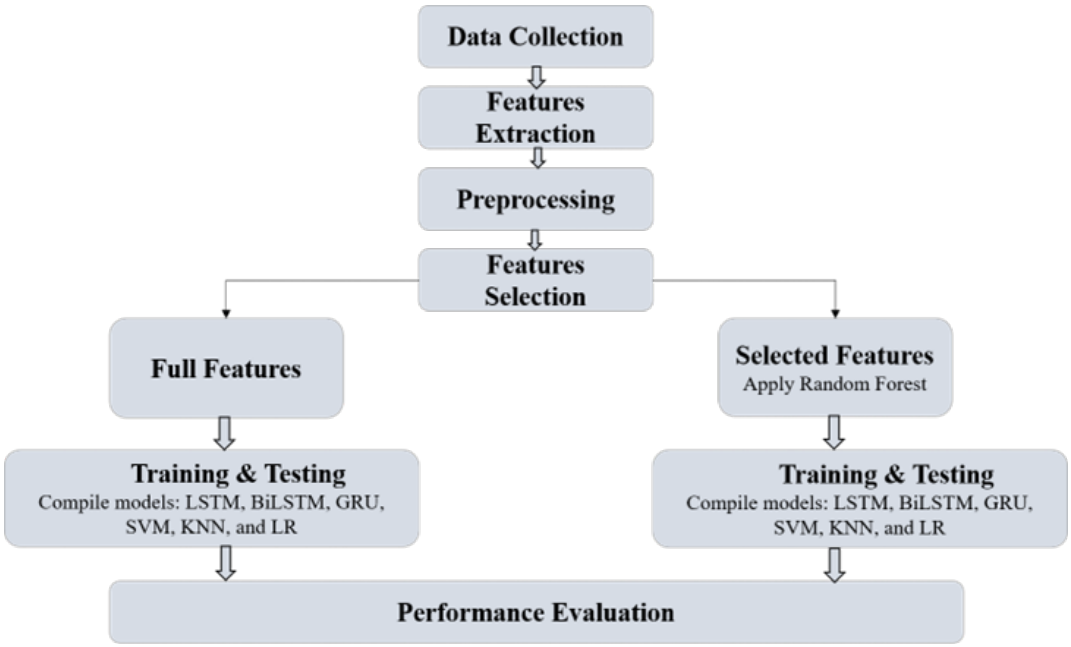


Figure 1: Android malware detection workflow.

A. Data Collection

The choice of proper dataset presents an important role in developing the performance of malware detection models. This study uses two datasets, which are described below.

1) Android Malware Detection (AMD) dataset: AMD is one of the most recent standard repositories that is publicly accessible for authors in the field of Android security [11]. This dataset provides more than 80 features that describe network traffic flows, such as IP addresses, ports, packet counts, flow duration, inter-arrival times, TCP flag counts, header lengths, and activity/idle times. These features capture both forward and backward traffic behaviors, providing

ISSN:2222-758X
e-ISSN:2789-7362

a detailed view of each flow for accurate classification. All these features are extracted statically and scrapped from CIC repository [12].

2) CICMalDroid2020 dataset: CICMalDroid2020 is a traffic analysis dataset that contain 17,341 sample for five different malware families: Adware, Banking, SMS, Riskware, and Benign [13]. It provides statically and dynamically extracted features, such as: intents, permissions and services, frequency counts for different file types, incidents of obfuscation, sensitive API invocations for statically extracted information, system calls, binder calls, composite behaviors for dynamically observed behaviors, adding to that a PCAP of all the network traffic captured during the analysis [14].

B. Machine Learning Models

ML is a subset of Artificial Intelligence (AI) that allows computers, thought training algorithms on large amount of data, to recognize patterns and make decisions without being specifically programmed [15]. However, in this study the following ML models are used to assess and compare their efficiency in the process of malware detection.

1) K-Nearest Neighbor (KNN): is one of the simplest ML models used for classification and regression. KNN is non parametric model where it can make prediction without being specifically programmed. KNN depends on picking the value of K that gives the optimal result to categorize each object belong to which class [16].
2) Support Vector Machine (SVM): is a classification ML model that intent to separate two classes with a line perfectly by finding the optimal points for that line. SVM can be applied to complex classification applications due to its ability in estimating values that are not included in the training set. SVM model is successfully applied for Android malware classification due to their effectiveness in high dimensional space [17].
3) Logistic Regression (LR): is a supervised ML model for binary classification. Probabilities between 0 and 1 will be mapped from predicted values using logistic function. These probabilities help in mapping each input value with its class. LR plays a crucial rule in detecting an attack in the field of cybersecurity [16].

C. Deep Learning Models

DL is a subset of ML with several hidden layers inspired by the structure of human brain. DL models can analyze and learn patterns by extracting large amount of data from large scale datasets [18]. Three DL models are used in this study to capture malware applications.

1) Long Short-Term Memory (LSTM): is a type of Recurrent Neural Network (RNN) with input, output, and forget gates. LSTM model works will with sequential data where it can maintain the information memory of the previous unit [3], [19].
2) Bidirectional Long Short-Term Memory (BiLSTM): is a developed version of LSTM model with two separate layers where input data moves forward and backward between these two layers. The model output can be evaluated by resulting the information in both past and future information [20].
3) Gate Recurrent Unit (GRU): GRU model is the most effective model for malware detection where it can handle large amount of data and classify sequence efficiently [21]. GRU is a type of RNN designed to deal with sequential data, making them perfect for dynamic behavior training for Android applications [22].

ISSN:2222-758X
e-ISSN:2789-7362

D. Features Extraction and preprocessing

The AMD dataset includes 355630 samples with 86 static features distributed as following: Android_Adware: 147443, Android_Scareware: 117082, Android_SMS_Malware: 67397, Benign: 23708[11]. The dataset contains missing values and string entries that need to be handled before initiating the training process on ML-DL models so that it does not affect their performances. The missing values are handled using mean imputation method, where each missing value in a numerical feature was substituted with the mean value of that feature calculated from the existing data. While the string entries are converted using label encoder method, which allocates a unique integer to each distinct category. This transformation allows ML-DL models to process categorical data efficiently without altering the dataset structure. In CICMalDroid2020 dataset, the selected CSV file consist of 470 extracted features for 11,598 APK files comprising frequencies of system calls, binders, and composite behaviors for five categories Adware: 1253, Banking: 2100, SMS malware: 3904, Riskware: 2546, Benign: 1795. Unlike AMD dataset, CICMalDroid2020 does not contain any missing values or string entries, eliminating the need for imputation or label encoder methods.

E. Performance Evaluation

Different metrics are used to measure the performance of ML-DL models, as follow [16], [23]:

1) Accuracy: this matrix can be calculated by dividing the True Positive (TP) and True Negative (TN) values by the sum of all true and negative values. Where:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{1}$$

2) Recall: in this matrix the TP will be divided by the sum of TP and False Negative (FN) values. Where:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

3) Precision: is the ratio of TP values to the sum of variety of all positive results TP and False Positive (FP), Where:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

4) F-Measure: this matrix shows the power of classification models for detecting Android malware. Which can be obtained by:

$$\text{F-Measure} = \frac{(\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \times 2 \tag{4}$$

IV. Experimental Results

In this study six models including LSTM, BiLSTM, GRU, SVM, KNN, and LR are evaluated and tested on two datasets: AMD and CICMalDroid2020.

A. Model Architecture

The experiments are performed on Google Colab with consistent parameters across all models including 80:20 train-test split, two hidden layers with 64 units each, using the Adam optimizer to train the models and categorical cross-entropy for loss. Training is done over 10 epochs with a batch size of 64 and a dropout rate of 0.3 to avoid the overfitting problem.

B. Performance on the AMD Dataset

To compare the performance of ML-DL models using the AMD dataset all the 86 features are extracted and selected to evaluate the detection accuracy in classifying malware samples for each model. The result from applying all the six models on AMD dataset shows that DL models outperformed ML models in terms of accuracy with 99.77% for BiLSTM, 99.75% for GRU, and 99.62% for LSTM. The high accuracy achieved by BiLSTM model required 14.27 minutes for training making it the slowest among deep learning models, where GRU and LSTM required 7.18 minutes and 6.4 minutes respectively.

On the other hand, in SVM model the required time to achieve an accuracy of 98.75% was significantly high with total of 238.9 minutes. While KNN and LR shows significantly faster execution time with 3.52 minutes for KNN and 0.41 minutes for LR, but with relatively lower accuracy at 97.08% and 98.73% for KNN and LR respectively.

C. Performance on the CICMalDroid2020 Dataset

The result indicates that the accuracy for all models was lower than those achieved in the AMD dataset, due to the increased diversity of static and dynamic features where all the 470 features were extracted and selected at this point. For DL models, BiLSTM model achieved the highest accuracy of 89.70% with training time of 0.99 minutes, followed by GRU at 87.94% with 0.6 minutes and LSTM at 88.87% with 0.52 minutes. While ML models showed slightly lower accuracy, where the achieving accuracy for SVM is 86.72% with training time of 0.94 minutes, KNN 86.98% with 0.05 minutes, and LR 85.47% with 0.14 minutes. All results are shown below in Table I. As shown in both charts of Fig. 2, the ML-DL models

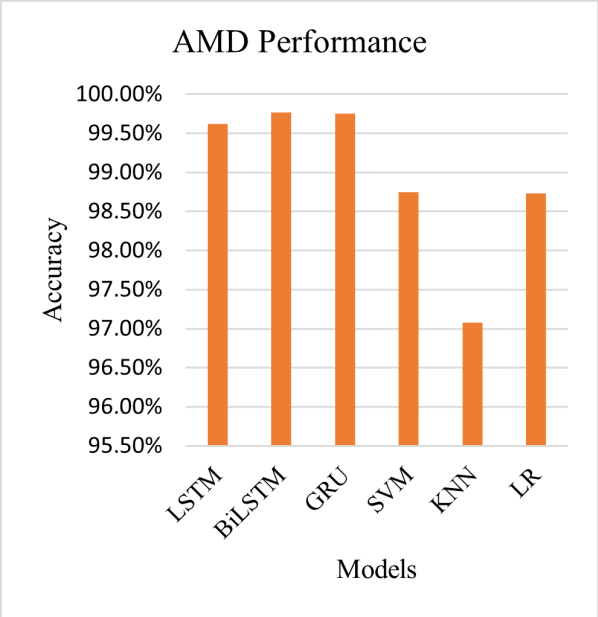
TABLE I
A Comparation of Achieved Accuracy and Total Execution Time for Applying Different ML-DL Models.

Dataset	Features Type	Size (Rows*Cols)	Target Categorical	Method	Total Execution Time	Accuracy
AMD	Static features	3556331*86	-Android_Adware -Android_Scareware -Android_SMS_Malware -Benign	LSTM	6.4 min	99.62%
				BiLSTM	14.27 min	99.77%
				GRU	7.18 min	99.75%
				SVM	238.9 min	98.75%
				KNN	3.52 min	97.08%
				LR	0.41 min	98.73%
CICMalDroid2020	Static and dynamic features	11599*471	-Adware -Banking -SMS malware -Riskware -Benign	LSTM	0.52 min	88.87%
				BiLSTM	0.99 min	89.70%
				GRU	0.6 min	87.94%
				SVM	0.94 min	86.72%
				KNN	0.05 min	86.98%
				LR	0.14 min	85.47%

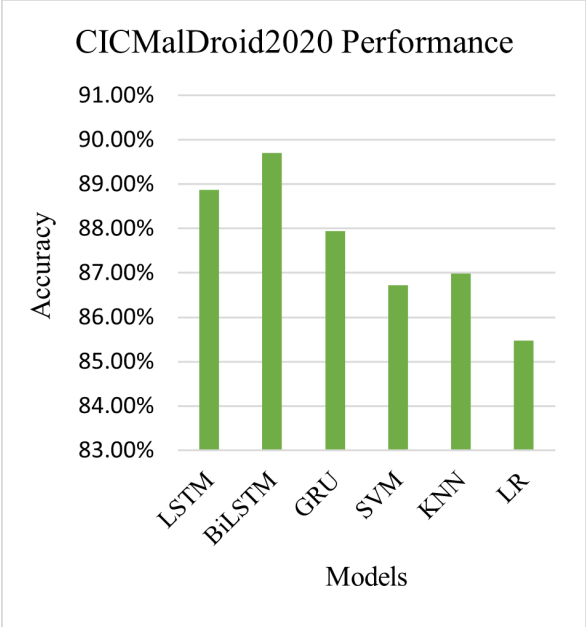
achieved higher accuracy when evaluated on the AMD dataset compared to the CICMalDroid2020 dataset, likely due to the AMD dataset's larger size, more balanced label distribution, and cleaner structure. With over 355,000 samples across four well-defined categories, the AMD dataset provides sufficient examples for each class, enabling better generalization [12]. In contrast, CICMalDroid2020 contains significantly fewer samples 11,598. Adding to that, the CICMalDroid2020 dataset mostly uses dynamic features like system calls and app behaviors during execution. However, many of these features are

ISSN:2222-758X
e-ISSN:2789-7362

either missing or have very low values, which means there's not much useful information in each record. This makes it harder for ML-DL models to find clear patterns which may lead to poor performance [13]. On the other hand, the AMD dataset has more complete and consistent features, making it easier for the models to learn and make accurate predictions. However, Random Forest algorithm are used as a feature selection method to improve model performance and reduce



(a) performance on AMD dataset



(b) performance on CICMalDroid2020 dataset.

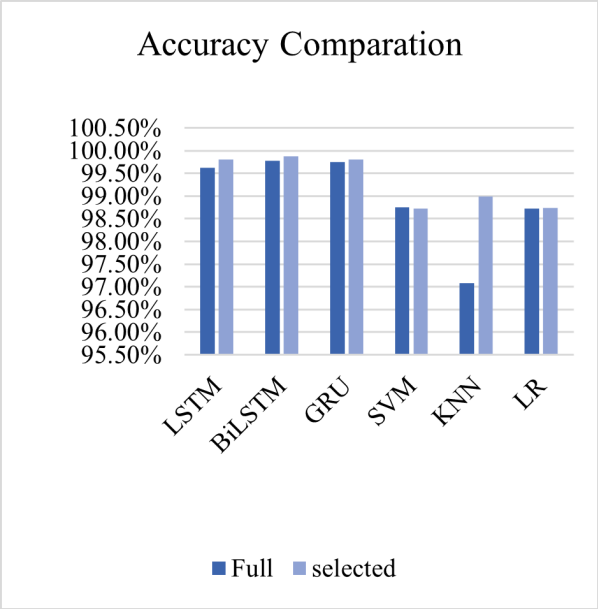
Figure 2: Comparation between different ML and DL models.

complexity. This method selects features based on ranking the importance of the most relevant ones and neglect the less significant features [24]. Since all ML-DL models showed better performance on AMD dataset, a random forest algorithm is applied to enhance the detection accuracy even more by selecting the features that give the most informative data for the classification process. Out of 86 only 20 features of AMD dataset are selected as a top priority for malware classification by all the six models. This selection helped in optimizing the training time for models without reducing the accuracy, by applying the dimensionality reduction for the dataset. As shown in Table II and Fig 3. Where there is a noticeable difference in accuracy and execution time using the full and selected features method.

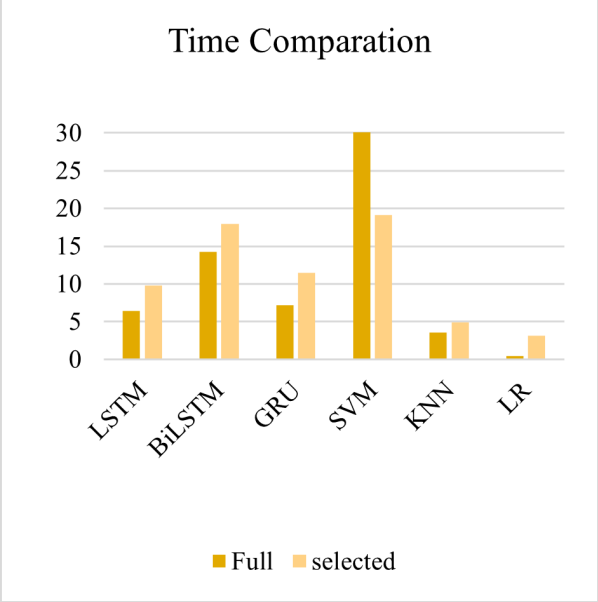
ISSN:2222-758X
e-ISSN:2789-7362

TABLE II
Applying Random Forest Method for Feature Selection

Model	Method	Accuracy	Time
LSTM	Full	99.62%	6.40 min
	Random Forest	99.80%	9.82 min
BiLSTM	Full	99.77%	14.27 min
	Random Forest	99.88%	17.94 min
GRU	Full	99.75%	7.18 min
	Random Forest	99.80%	11.46 min
SVM	Full	98.75%	238.9 min
	Random Forest	98.73%	19.13 min
KNN	Full	97.08%	3.52 min
	Random Forest	98.99%	4.93 min
LR	Full	98.73%	0.41 min
	Random Forest	98.74%	3.17 min



(a) Full and selected features accuracy



(b) Full and selected features time

Figure 3: Comparation between full and selected features performance on AMD dataset.

V. CONCLUSIONS

Android platform as considered as the most used operating system for mobile devices. Whereas securing these devices is becoming the primary goal to protect users financial or personal information from attackers. This study demonstrated the difference between ML-DL models including: LSTM, BiLSTM, GRU, SVM, KNN, and Logistic Regression. Through

applying these models on two datasets AMD and CICMallDroid2020, with unified architecture parameters, a comparison is produced in Table I. A Random Forest method is used to increase the accuracy of ML-DL models on AMD dataset by mathematically minimizing the number of features required for the classification process. The result showed that DL models consistently outperformed traditional ML models across both datasets. BiLSTM showed the highest accuracy but with the cost of longer execution times. LSTM and GRU offered a good balance between accuracy and computational efficiency, making it a practical alternative. However, SVM, KNN, and LR presented lower accuracy. Therefore, it is imperative to emphasize on improving models performance in future work by appending additional datasets with dynamic features analysis.

FUNDING

None.

ACKNOWLEDGEMENT

The author would like to thank the reviewers for their valuable contribution in the publication of this paper.

CONFLICTS OF INTEREST

The author declares no conflict of interest.

REFERENCES

[1] S. Xiong, X. Chen, H. Zhang, and M. Wang, "Domain Adaptation-Based Deep Learning Framework for Android Malware Detection Across Diverse Distributions," Artificial Intelligence Advances, vol. 6, no. 1, pp. 13–24, Jun. 2024, doi: 10.30564/aia.v6i1.6718.

[2] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, "A Deep Learning Based Android Malware Detection System with Static Analysis," in 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), IEEE, Jun. 2022, pp. 1–6. doi: 10.1109/HORA55278.2022.9800057.

[3] Z. Ma, H. Ge, Z. Wang, Y. Liu, and X. Liu, "Droidetec: Android Malware Detection and Malicious Code Localization through Deep Learning," ArXiv, vol. 1, pp. 1–13, Feb. 2020, [Online]. Available: http://arxiv.org/abs/2002.03594

[4] S. Li, Q. Zhou, R. Zhou, and Q. Lv, "Intelligent malware detection based on graph convolutional network," J Supercomput, vol. 78, no. 3, pp. 4182–4198, Feb. 2022, doi: 10.1007/s11227-021-04020-y.

[5] N. M. Dahham and L. M. Jawad, "State-Of-The-Art for Email Phishing Detection Technique based on Deep Learning," in The International Middle Eastern Simulation and Modelling Conference, MESM 2023, 2024, pp. 18–21, baghdad, Nov. 2023, pp. 1–4.

[6] H. Alamro, W. Mtouaa, S. Aljameel, A. S. Salama, M. A. Hamza, and A. Y. Othman, "Automated Android Malware Detection Using Optimal Ensemble Learning Approach for Cybersecurity," IEEE Access, vol. 11, pp. 72509–72517, 2023, doi: 10.1109/ACCESS.2023.3294263.

[7] J. Mohamad Arif, M. F. Ab Razak, S. Awang, S. R. Tuan Mat, N. S. N. Ismail, and A. Firdaus, "A static analysis approach for Android permission-based malware detection systems," PLoS One, vol. 16, no. 9, pp. 1–23, Sep. 2021, doi: 10.1371/journal.pone.0257968.

[8] Bayazit E Calik, Sahingoz O Koray, and Dogan B, "Deep Learning based Malware Detection for Android Systems: A Comparative Analysis," Tehnicki vjesnik - Technical Gazette, vol. 30, no. 3, pp. 787–796, Apr. 2023, doi: 10.17559/TV-20220907113227.

[9] Bayazit E Calik, Sahingoz O Koray, and Dogan B, "A Deep Learning Based Android Malware Detection System with Static Analysis," in 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), IEEE, Jun. 2022, pp. 1–6. doi: 10.1109/HORA55278.2022.9800057.

[10] O. N. Elayan and A. M. Mustafa, "Android Malware Detection Using Deep Learning," Procedia Comput Sci, vol. 184, pp. 847–852, 2021, doi: 10.1016/j.procs.2021.03.106.

[11] T. Gera, J. Singh, A. Mehbodniya, J. L. Webber, M. Shabaz, and D. Thakur, "Dominant Feature Selection and Machine Learning-Based Hybrid Approach to Analyze Android Ransomware," Security and Communication Networks, pp. 1–22, Nov. 2021, doi: 10.1155/2021/7035233.

[12] F. Wei, Y. Li, S. Roy, X. Ou, and W. Zhou, "Deep Ground Truth Analysis of Current Android Malware," in Computer Science, vol. 10327, Springer, Cham, 2017, pp. 252–276. doi: 10.1007/978-3-319-60876-1_12.

[13] S. Mahdavifar, A. F. Abdul Kadir, R. Fatemi, D. Alhadidi, and A. A. Ghorbani, "Dynamic Android Malware Category Classification using Semi-Supervised Deep Learning," in 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Canada: IEEE, Aug. 2020, pp. 515–522. doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00094.

[14] S. Mahdavifar, D. Alhadidi, and Ali. A. Ghorbani, "Effective and Efficient Hybrid Android Malware Classification Using Pseudo-Label Stacked Auto-Encoder," Journal of Network and Systems Management, vol. 30, no. 1, p. 22, Jan. 2022, doi: 10.1007/s10922-021-09634-4.

[15] I. Yazici, I. Shayea, and J. Din, "A survey of applications of artificial intelligence and machine learning in future mobile networks-enabled systems," Engineering Science and Technology, an International Journal, vol. 44, pp. 1–40, Jun. 2023, doi: 10.1016/j.jestch.2023.101455.

[16] Aqsa Ijaz et al., "Innovative Machine Learning Techniques for Malware Detection ," Journal of Computing Biomedical Informatics , vol. 7, no. 1, pp. 1–24, Jun. 2024.

ISSN:2222-758X
e-ISSN:2789-7362

[17] A. Batuhan Yilmaz, Y. Selim Taspinar, and M. Koklu, "Classification of Malicious Android Applications Using Naive Bayes and Support Vector Machine Algorithms," International Journal of Intelligent Systems and Applications in Engineering, vol. 10, no. 2, pp. 269–274, May 2022, doi: 10.1039/b000000x.
[18] M. Aamir et al., "AMDDLmodel: Android smartphones malware detection using deep learning model," PLoS One, vol. 7, no. 1, pp. 403–424, Jun. 2024, doi: 10.1371/journal.pone.0296722.
[19] A. T. Salim and Ban Mohammed Khammas, "Performance Evaluation of Deep Learning Techniques in The Detection of IOT Malware," Iraqi Journal of Information and Communication Technology, vol. 6, no. 3, pp. 12–25, Dec. 2024, doi: 10.31987/ijict.6.3.233.
[20] A. R. Nasser, A. M. Hasan, and A. J. Humaidi, "DL-AMDet: Deep learning-based malware detector for android," Intelligent Systems with Applications, vol. 21, pp. 1–10, Mar. 2024, doi: 10.1016/j.iswa.2023.200318.
[21] I. Ullah and Q. H. Mahmoud, "Design and Development of RNN Anomaly Detection Model for IoT Networks," IEEE Access, vol. 10, pp. 62722–62750, 2022, doi: 10.1109/ACCESS.2022.3176317.
[22] T. Lu, Y. Du, L. Ouyang, Q. Chen, and X. Wang, "Android Malware Detection Based on a Hybrid Deep Learning Model," Security and Communication Networks, pp. 1–11, Aug. 2020, doi: 10.1155/2020/8863617.
[23] M. Abd Al Abbas and B. M. Khammas, "Efficient IoT Malware Detection Technique Using Recurrent Neural Network," Iraqi Journal of Information and Communication Technology, vol. 7, no. 3, pp. 29–42, Dec. 2024, doi: 10.31987/ijict.7.3.249.
[24] Y. Akhiat, Y. Manzali, M. Chahhou, and A. Zinedine, "A New Noisy Random Forest Based Method for Feature Selection," Cybernetics and Information Technologies, vol. 21, no. 2, pp. 10–28, Jun. 2021, doi: 10.2478/cait-2021-0016.