Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

Exploring the Effectiveness of Machine and Deep Learning Techniques for Android Malware Detection

Khalid Murad Abdullah[1*], Ahmed Adnan Hadi[2]

[1,2]Open Educational College, Al-Qadisiyah Center, Iraq.*

Email: [2]ah2036@gmail.com
Corresponding Email: [1]Khalid.mu.abdullah@gmail.com*

Received: 29 September 2023 **Accepted:** 17 December 2023 **Published:** 01 February 2024

Abstract: ***The increasing occurrence of Android devices, coupled with their get entry to to touchy and personal information, has made them a high goal for malware developers. The open-supply nature of the Android platform has contributed to the developing vulnerability of malware assaults. presently, Android malware (AM) analysis strategies may be labeled into foremost categories: static evaluation and dynamic evaluation. These techniques are employed to analyze and understand the behavior of AM to mitigate its impact. This research explores the performance of DL model architectures, such as CNN-GRU, as well as traditional ML algorithms including SVM, Random Forest (RF), and decision tree (DT). The DT model achieves the highest accuracy (ACC) of 0.93, followed by RF (0.89), CNN-GRU (0.91), and SVM (0.90). These findings contribute valuable insights for the development of effective malware detection systems, emphasizing the suitability and effectiveness of the examined models in identifying AM.***

Keywords: Malware Detection, Deep Learning, Machine Learning, Random Forest, CNN-GRU, SVM.

1. INTRODUCTION

As our society becomes more interconnected, the number and variety of cell devices are swiftly increasing. by 2020, it is projected that there could be about 6.1 billion mobile device customers [1]. these devices store and offer access to a considerable amount of private information, making them an appealing target for cybercriminals [2]. fairly, many users do not set up antivirus or anti-malware applications on their mobile devices, and the effectiveness of such apps is a topic of debate [3]. therefore, mobile devices are often taken into consideration the weakest link in business enterprise protection by security professionals. Android, being the dominant open-source phone operating system with a marketplace share of over eighty% [4], gives users a large selection of programs (apps) available for down load and use from diverse 0.33-birthday party app stores [5], [6]. however, because of its giant

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

adoption and open nature, Android will become a prime target for telephone attacks, accounting for almost 99% of such attacks [4]. Malware serves because the primary vehicle for these malicious activities.

While all mobile operating systems are susceptible to malware attacks, the focus is generally on those with a larger market share. Android, in particular, has become a prime target due to its substantial market presence [7]. Additionally, the flexible publishing policy of Google Play, the official app market for Android, has contributed to its popularity among malware developers. The Android permission-based security model offers limited protection, as users commonly grant re-quested permissions without hesitation [8]. Furthermore, there have been reported incidents of malicious apps successfully infiltrating Google Play [9]. These circumstances highlight the need for more effective tools and techniques for analyzing Android malware.

Various strategies have been proposed to enhance the security of the Android ecosystem, including measures such as app reinforcement, vulnerability detection, developer reviews, and malware detection (MD) [10]. Android MD, in particular, is a widely used security measure aimed at preventing the distribution and installation of harmful software through the Android app market. There is also, previous research that has categorized Android MD techniques into static detection, dynamic detection, and hybrid detection approaches [11]–[13]. Previous research has categorized the techniques for detecting AM into three main categories: static detection, dynamic detection, and hybrid detection [13][6]. Static detection involves examining potentially malicious code without executing the Android app. Whilst this technique offers comprehensive code coverage, it is prone to countermeasures which include code obfuscation and dynamic code loading. then again, dynamic detection involves studying the conduct of the Android app at the same time as it is running, uncovering security vulnerabilities that static analysis can also forget about. however, dynamic detection requires significant computational resources and time. To address these limitations, hybrid detection combines elements of both static and dynamic detection, striking a balance between effectiveness and efficiency [14].

In latest years, machine mastering (ML) has received traction in the area of AM detection. ML-based detection methods can perceive malware that has not been formerly visible, enhancing the effectiveness and efficiency of detection [15], [16]. That is in contrast to conventional techniques like signature-based detection, which depend on recognizing particular patterns of recognized malware. The utilization of ML techniques in AM detection has been explored in preceding studies, highlighting its capacity for enhancing detection abilties. This article presents a comprehensive research into the classification of AM using diverse ML and deep learning (DL) models.

2. RELATED WORK

Many studies have been carried out in the last few years to use ML approaches for the detection of dangerous Android applications. Nevertheless, there aren't many articles that go

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

into great detail about the particular ML methods used. As far as we are aware, no thorough classification of mobile malware detection (MMD) systems exists at this time, based on the metrics and machine learning techniques applied. This section compares our suggested method with a chronological analysis of pertinent literature contributions from 2020 to 2023. The researchers unveiled DL-Droid [17], a DL system that uses stateful input generation and dynamic analysis approaches to detect malicious Android applications. DL-Droid runs on an automated platform that can perform analysis, both dynamic and static. Over 31,000 Android applications, including over 11,000 malware samples, were gathered for the study's dataset. On a real Android device, each application was evaluated for about 190 seconds. With only dynamic analytic characteristics, DL-Droid was able to attain a high detection rate of 97.8%. The detection rate rose to 99.6% when static analysis data was included. The RF classifier was used by the researchers in the detection procedure.

In a study [18], researchers described four malware detection (MD) techniques based on the Hamming distance methodology. These techniques, namely FNN (first closest neighbor), ANN (all nearest neighbors), WANN (weighted all nearest neighbors), and KMNN (k-medoid-based nearest neighbors), were developed to identify commonalities among malware samples. To select 300 significant features, the researchers employed the RF Regressor feature selection method. The evaluation of these techniques using various classifiers such as SVM, DT, RF, and MLP yielded promising results, with ACC ranging from approximately 90% to 99%.

In a study [19], researchers proposed the DANdroid MMD system, which utilizes DL for application classification. The system incorporates three features: opcodes, permissions, and API calls. To evaluate the system’s effectiveness, the researchers utilized the Drebin dataset, which consisted of approximately 70,000 applications obfuscated using five different strategies. The results showed that when employing the CNN algorithm, the system achieved an impressive F-score of 97.3%. In [20], the authors investigated the impact of incorporating an ensemble model in the area of MMD (Malware Detection and classification). The ensemble model become constructed with the aid of combining the results of three essential models: Logistic Regression, MLP (Multi-Layer Perceptron), and SGD (Stochastic Gradient Descent). The study tested that making use of a larger and more homogeneous collection of outside instances inside the ensemble model led to progressed performance compared to the use of a smaller and more heterogeneous set of examples. additionally, the researchers determined that the extrinsic ensemble model performed better results while a random subset of features, rather than the complete feature set, became employed. appreciably, remarkable AUC values of 99.4%, 99.three%, and 99.7%, along with ACC scores of 98.3%, 98.7%, and 99.1%, have been suggested on the AndroZoo, VirusShare, and Drebin datasets, respectively, highlighting the effectiveness of the ensemble method in improving MMD performance.

3. METHODOLOGY

Our research focuses on the classification of malware and the application of various ML and DL models. To evaluate the performance of these models, we divide the dataset into separate

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

training and test sets. The training set is utilized to train the models, while the test set is used as an independent dataset to assess their ability to generalize. We assess how several deep learning model architectures, such as CNN-GRU, RF, SVM, and DT, affect malware categorization. Because each model has distinct qualities and skills, we can examine each model's performance in this particular task in great detail.

During the training process, we improve our models' parameters using gradient descent and backpropagation. These models get better at identifying significant dataset features during multiple training epochs, which leads to more precise predictions. After training, we assess these models' performance on a different dataset. Here, the models forecast malware categories by using the test samples. We evaluate the accuracy of these predictions by comparing them to the actual labels and calculating measures like accuracy (ACC) and overall performance. We evaluate the models' performance using multiple metrics, such as ACC, precision (PRE), recall (REC), and F1-score (F1-S), to guarantee a comprehensive assessment. These metrics reveal how well the models are able to identify the existence of malware.

Our goal in doing this research is to gain a deeper knowledge of the possibilities and difficulties related to various deep learning (DL) and machine learning (ML) models for malware classification. In addition to gaining knowledge, our goal is to aid in the creation of malware detection technologies that are more reliable and effective. This entails determining which machine-learning algorithms are best suited for categorizing malware. Figure 1 shows the Malware Classification approach we employ.

Dataset Description

In this research study, the effectiveness of a malware classification approach is evaluated using the ”Dataset mal-ware/benign permissions Android.” The dataset contains information about the permissions utilized by Android applications, which are then classified as either malicious or benign. The information is represented in binary form, where the presence of a privilege is denoted by way of 1, and its absence is denoted by 0. each sample within the dataset is labeled as both type 1 for malware or type 0 for non-malware. Such datasets are commonly used in ML to train models that can classify objects based on specific traits or attributes.

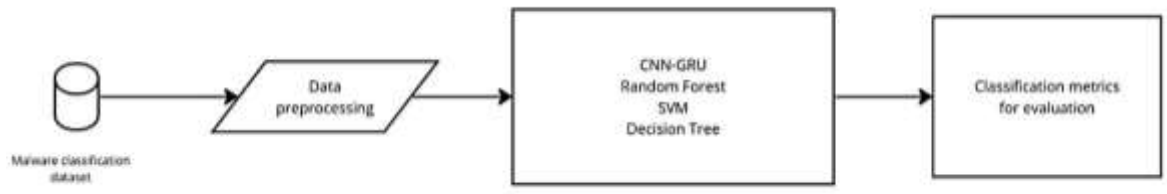

Figure.1 Malware Classification Methodology

Performance metrics including ACC, PREC, REC, and F1 score will be utilized to gauge the model's efficacy after it has been trained and evaluated using the dataset. The goal is to create

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

a dependable and effective approach for classifying malware that can precisely detect and classify malware on Android devices.

Data Preprocessing

In our research, data preprocessing is an important stage where we run multiple operations on the dataset prior to model training. Dividing the dataset into distinct training and test sets is a crucial step.

Having a separate collection of data to assess the models' performance after training is the primary goal of partitioning the dataset. 70% of the dataset is divided between the training and test sets, with the remaining portion going to the training set. This segment aids in evaluating the models' ability to generalize to previously encountered data. We are able to evaluate the models' malware classification accuracy by separating the dataset into train and test sets. This step allows us to gauge their performance on unseen data and make informed decisions about their effectiveness in real-world scenarios.

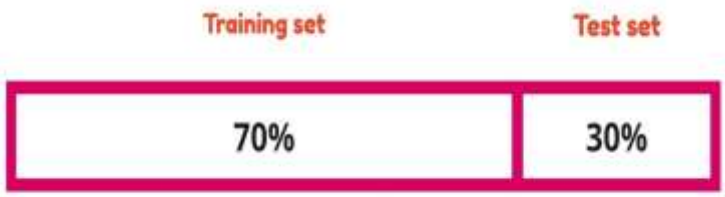

Figure 2 Dataset Split Ratio.

Classification Ml Models

In this research, many algorithms are applied to classify Android malware.

A. CNN-GRU model

The CNN-GRU model is a mixed-reality architecture that combines CNNs and GRUs to manage sequential and geographical data processing tasks. Given its ability to extract hierarchical features from input data and capture spatial dependencies, CNNs are a good fit for certain tasks. Because of their capacity to manage the particularities of Android applications, CNN-GRU models have drawn interest in the field of AM categorization. Traditional detection approaches frequently fail to correctly identify malicious activities due to the growing sophistication and complexity of malware. In order to capture both spatial and sequential patterns in Android malware, the CNN-GRU architecture combines the advantages of CNNs and GRUs to offer a potent solution. This model is implemented with an 8-kernel 1D convolutional layer with 32 filters and a ReLU activation function. The dimensions of the training data define the shape of the input. In order to avoid overfitting, three 10-unit GRU layers are added, with a 0.2 dropout imposed after each GRU layer. Next, the model is flattened and joined to a dense layer using a sigmoid activation function and a single unit. It is constructed with the Adam optimizer and binary cross-entropy loss, with ACC acting as the assessment measure.

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

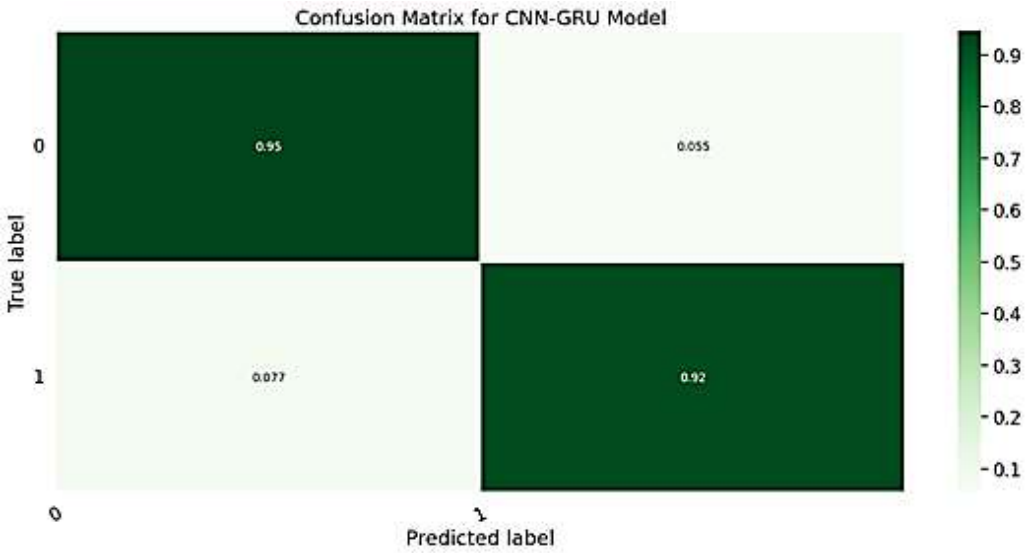

Figure 3 Confusion matrix for CNN-GRU

B. Random Forest model

Regression and classification tasks are major uses of RF in machine learning. It is an ensemble method that combines different DTs to provide predictions. Subsets of the training data are used to train a group of DTs in RF, and after that, each tree independently predicts the future. Strong machine learning algorithm RF has shown successful in AM classification. It predicts if an application is malicious or benign by utilizing an ensemble of DTs. To train numerous DTs in the context of AM classification, RF uses several features that are taken from the dataset, like opcodes, permissions, and API calls. Every tree assesses various feature subsets on its own and generates its own forecasts. The final classification decision is made by combining the predictions from all the individual trees in the RF model.

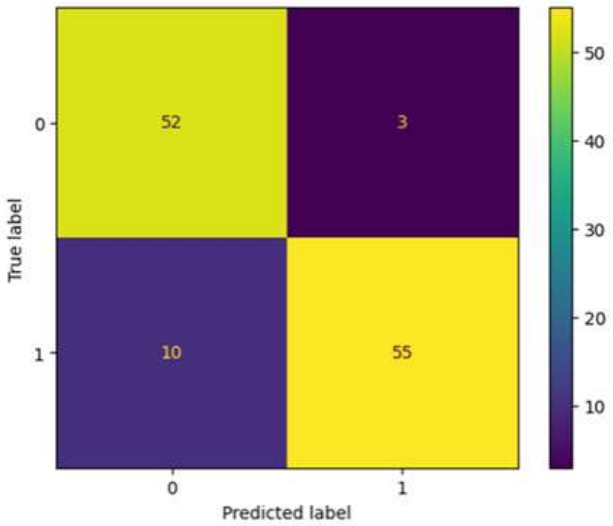

Figure 4 Confusion matrix for RF

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

C. SVM Model

For the categorization of Android malware, SVM has become a well-liked and efficient machine-learning algorithm. The goal of the supervised learning model SVM is to find the best hyperplane for efficiently dividing different classes of data points. SVM seeks to distinguish between benign and dangerous programs in the context of malware classification by finding the best decision boundary inside a high-dimensional feature space. This is achieved by employing a kernel function to transfer the input data into a higher-dimensional space, making it easier to discern a distinct division between the classes.

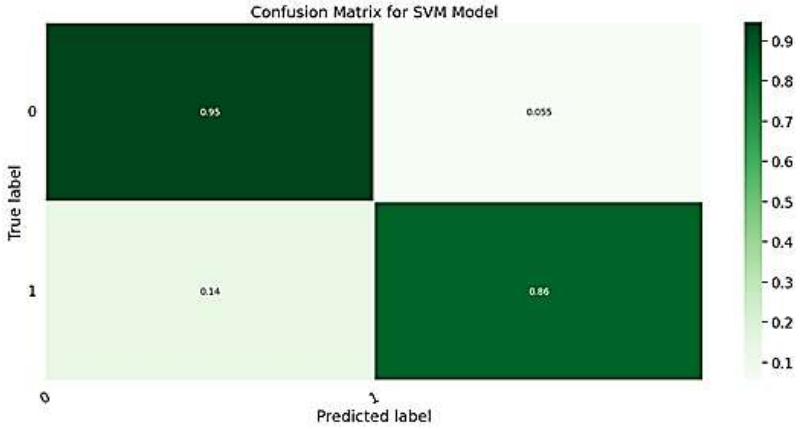

Figure 5 Confusion matrix for SVM

D. Decision Tree Model

A well-liked machine learning approach for AM classification is called DT. Based on the characteristics of the input, this supervised learning model learns a hierarchical structure of conditions and decisions. With each core node representing a feature or characteristic and each leaf node indicating a class label or outcome, the DT model builds a structure like a tree. Recursively segmenting the data based on the values of different attributes allows for the construction of the tree, maximizing the homogeneity or purity of the resultant subsets. Because people can picture and grasp the taught principles, DTs are easily interpretable and understandable.

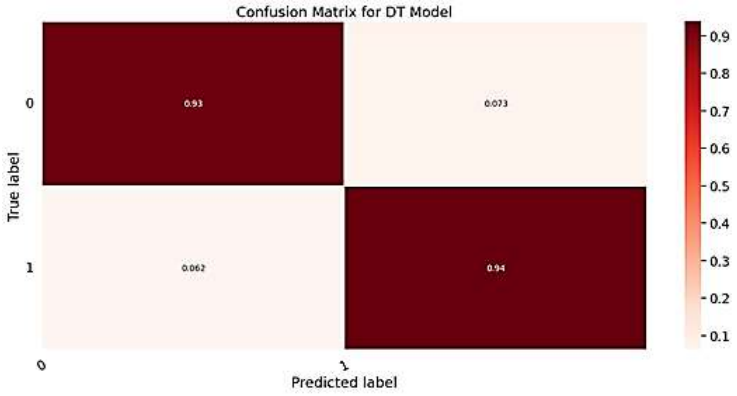

Figure 6 Confusion matrix for DT

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

Comparison Between Models

The following table presents a comparison between the different models applied in this research.

Table 1 Comparison Of Classification Results

Model	PREC	REC	F1-S	ACC
RF	0.84	0.95	0.89	0.89
SVM	0.85	0.95	0.90	0.90
DT	0.91	0.93	0.92	0.93
CNN-GRU	0.85	0.96	0.91	0.91

All models fared well in differentiating between malicious and benign apps, according to the AM classification findings utilizing various models. With high PREC and REC for both classes, the DT model had the greatest ACC of 0.93. With ACCs of 0.89 and 0.91, respectively, and balanced PREC, REC, and F1-Ss, the RF and CNN-GRU models also demonstrated good performance. The SVM model showed strong PREC and REC for both classes and an ACC of 0.90. Taken together, these results indicate that the CNN-GRU, DT, RF, and SVM models are good at detecting malware on Android devices, which offers important information for creating reliable malware detection systems.

4. CONCLUSION

In conclusion, the classification of AM using different ML and ML models was the main emphasis of our research. To assess the effectiveness of these models, we separated the dataset into training and test sets. We investigated various architectures, such as CNN-GRU, RF, SVM, and DT, and evaluated their effects on malware classification. We enhanced the model's parameters throughout the training phase by applying techniques like gradient descent and backpropagation. The models acquired the ability to extract pertinent information and generate precise predictions through iterative training epochs. By evaluating the test set, we were able to measure metrics like ACC, PREC, and REC and compare the projected labels with the ground truth. By utilizing these techniques, we sought to comprehend the capabilities and constraints of ML and DL models for malware categorization. According to our research, the DT model had the best ACC of 0.93 and the best PREC and REC for both classes. Strong performance was also shown by the RF and CNN-GRU models, which balanced PREC, REC, and F1-Ss and had ACCs of 0.89 and 0.90, respectively. With an ACC of 0.90, the SVM model showed strong PREC and REC. These findings show that AM can be successfully identified by the DT, RF, SVM, and CNN-GRU models, offering insightful information for the creation of reliable malware detection systems.

To guarantee that the models are still capable of identifying the most recent threats, it will be essential to keep adding fresh samples to the dataset on a regular basis and to keep an eye on newly discovered malware families. Exploring the interpretability of the models provides deeper insights into their decision-making process and enables better understanding and trust. Optimizing the models for resource-constrained mobile devices can enable real-time and on-

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

device malware detection, enhancing user security and privacy. By pursuing these future directions, we can further advance the field of AM classification, develop more resilient solutions, and stay ahead of evolving threats in the mobile ecosystem.

5. REFERENCES

1. Andy Boxall. The number of smartphone users in the world is expected to reach a giant 6.1 billion by 2020, 2015.
2. Ali Dehghantanha and Katrin Franke. Privacy-respecting digital investigation. In 2014 Twelfth Annual International Conference on Privacy, Security, and Trust, pages 129–138. IEEE, 2014.
3. Jason Walls and Kim-Kwang Raymond Choo. A review of free cloud-based anti-malware apps for Android. In 2015 IEEE Trust-com/BigDataSE/ISPA, volume 1, pages 1053–1058, 2015.
4. Andrea Saracino, Daniele Sgandurra, Gianluca Dini, and Fabio Mar-tinelli. Madam: Effective and efficient behavior-based Android malware detection and prevention. IEEE Transactions on Dependable and Secure Computing, 15(1):83–97, 2016.
5. Feng Shen, Justin Del Vecchio, Aziz Mohaisen, Steven Y Ko, and Lukasz Ziarek. Android malware detection using complex flows. IEEE Transactions on Mobile Computing, 18(6):1231–1245, 2018.
6. Suleiman Y Yerima and Sakir Sezer. Diffusion: A novel multilevel classifier fusion approach for Android malware detection. IEEE trans-actions on Cybernetics, 49(2):453–466, 2018.
7. M Kitagawa, A Gupta, R Cozza, I Durand, D Glenn, K Maita, L Tay, T Tsai, R Atwal, M Escherich, et al. Market share: final pcs, ultra mobiles, and mobile phones, all countries. Technical report, 2q15 update, Tech. 385 rep, 2015.
8. C Chia, K-KR Choo, and Dennis Fehrenbacher. How cyber-savvy are older mobile device users? In Mobile security and privacy, pages 67–83. Elsevier, 2017.
9. Nicolas Viennot, Edward Garcia, and Jason Nieh. A measurement study of google play. In The 2014 ACM international conference on Measurement and Modeling of computer systems, pages 221–233, 2014.
10. Sufatrio, Darell JJ Tan, Tong-Wei Chua, and Vrizlynn LL Thing. Securing Android: a survey, taxonomy, and challenges. ACM Computing Surveys (CSUR), 47(4):1–45, 2015.
11. SH Qing. Research progress on Android security. Journal of Software, 27(1):45–71, 2016.
12. Joao Lopes, Carlos Serrao, Luıs Nunes, Ana Almeida, and Joaõ Oliveira. Overview of machine learning methods for Android malware identification. In 2019 7th International Symposium on Digital Forensics and Security (ISDFS), pages 1–6. IEEE, 2019.
13. Mahima Choudhary and Brij Kishore. Haamd: Hybrid analysis for Android malware detection. In 2018 International Conference on Computer Communication and Informatics (ICCCI), pages 1–4. IEEE, 2018.

Journal of Image Processing and Intelligent Remote Sensing
ISSN 2815-0953
Vol: 04, No.02, Feb-Mar 2024
http://journal.hmjournals.com/index.php/JIPIRS
DOI: https://doi.org/10.55529/jipirs.42.1.10

14. Kaijun Liu, Shengwei Xu, Guoai Xu, Miao Zhang, Dawei Sun, and Haifeng Liu. A review of Android malware detection approaches based on machine learning. IEEE Access, 8:124579–124607, 2020.
15. Milad Taleby Ahvanooey, Qianmu Li, Mahdi Rabbani, and Ahmed Raza Rajput. A survey on smartphone security: software vulnerabilities, malware, and attacks. arXiv preprint arXiv:2001.09406, 2020.
16. Alireza Souri and Rahil Hosseini. A state-of-the-art survey of malware detection approaches using data mining techniques. Human-centric Computing and Information Sciences, 8(1):1–22, 2018.
17. Mohammed K Alzaylaee, Suleiman Y Yerima, and Sakir Sezer. Dl-droid: Deep learning-based Android malware detection using real devices. Computers & Security, 89:101663, 2020.
18. Rahim Taheri, Meysam Ghahramani, Reza Javidan, Mohammad Sho-jafar, Zahra Pooranian, and Mauro Conti. Similarity-based Android malware detection using the hamming distance of static binary features. Future Generation Computer Systems, 105:230–247, 2020.
19. Stuart Millar, Niall McLaughlin, Jesus Martinez del Rincon, Paul Miller, and Ziming Zhao. Dendroid: A multi-view discriminative adversarial network for obfuscated Android malware detection. In Proceedings of the tenth ACM Conference on Data and application security and Privacy, pages 353–364, 2020.
20. Nektaria Potha, Vasileios Kouliaridis, and Georgios Kambourakis. An extrinsic random-based ensemble approach for Android malware detection. Connection Science, 33(4):1077–1093, 2021.