Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Mobile based Malware Detection using Artificial Intelligence Techniques a review

Mawj faez mahdi[a], Sarah Saadoon Jasim [b]*

*[a,b] **Department of Information Management Technologies, Technical College of Management-Baghdad, Middle Technical University, Baghdad, Iraq**. Email: (dac2013@mtu.edu.iq), (sara-sm@mtu.edu.iq)*

ARTICLEINFO

Article history:
Received: 22 /2/2024
Rrevised form: 10 /3/2024
Accepted : 24 /3/2024
Available online: 30 /3/2024

Keywords:

mobile;
Android;
malware;
detection;
artificial intelligence techniques.

ABSTRACT

Malware attacks on mobile devices are becoming more common and more complicated every year. Malware writers see the open-source Android app as their main target because so many people use it. Artificial intelligence is used by most of the literature's mobile malware detection methods to find ransomware. Our research, on the other hand, makes it clear that most of the earlier studies used different metrics and models, as well as different datasets and classification features that came from static, dynamic, or hybrid analysis strategies. This makes comparing the different suggested detection methods more difficult and may also make the results less certain. The goal of this work is to solve the problem of AI-powered malware detection by sorting current methods and approaches into three groups: the type of dataset, the type of detection method used (machine learning models, deep learning models, and Behavioral Analysis model), and how well the method works. In this way, we suggest a convergent plan that can be used as a basis for future methods of finding malware on Android and as a solid standard for artificial intelligence work in this area.

MSC...

https://doi.org/10.29304/jqcsm.2024.16.11439

1. Introduction

Since numerous services are now offered to us via mobile applications, mobile apps have become an essential part of our daily lives. Since the latter are typically put on smart gadgets, they alter how we communicate. Smart gadgets, as opposed to personal computers, have more advanced sensors, such as GPS, gyroscopes, and cameras, in addition to microphones (Delmastro et al.). These numerous sensors produce enormous volumes of data, some of which contain extremely sensitive information, and bring up a whole new universe of applications for end users (Delmastro et al.). This increases the need for security solutions to shield users from malicious apps that exploit smart devices' complexity and sensitive data.[1],[2] Which draws cybercriminals' attention to them as targets. Android is the most widely used and powerful

∗Corresponding author Mawj faez mahdi

Email addresses:

Communicated by 'sub etitor'

operating system for mobile devices (the Android operating system, 2019). Considering that Android is used on various devices, including tablets and smartphones, it held 76% of the global market share in September 2019 (Mobile OS market share, 2019). Additionally, it was successful in making its way to additional smart devices, including watches (Android Wear Operating System, 2016), TVs (Android TV, 2016), and automobiles (Android Auto, 2016). Furthermore, Android OS is being incorporated into Internet of Things systems more and more, particularly after Google released Android Things (Android Things operating system, 2020), an embedded operating system made for IoT devices with little power and resources [3]. Android operating system (OS) occupies most of the market share. According to the global smartphone operating system market report released by International Data Corporation (IDC), Android leads the way with a market share of 86.7% [4],[5] Its open-source design and ease of customization at many levels drew in users. They encouraged manufacturers to focus on creating affordable smart devices. Because of its SDK, which may assist developers in developing, Android is also well-liked by the developer community. Android apps that require little work to reach such a big goal. Owing to the greater community of individuals and its renowned attackers using malware programs to target Android devices, they have become more active, notably during the past few years. AV-Test data indicates that the total amount of malware has increased over the previous ten years, from 182 million to 1342 million.[6] In its mobile terminal products, 3,503,952 malicious installation packages were discovered. From 40,386 attacks on mobile devices in 2018 to 67,500 attacks in 2019, there was a 50% increase in these attacks. Stallerware is becoming increasingly common on mobile devices, in addition to Trojan horses and spyware in conventional network security. The abundance of Android malware, the speed at which updates are released, and the frequent appearance of new malware variants make it difficult to research efficient malware detection techniques that also shorten detection times and increase detection efficiencies [7],[52]. The study's objective is to compare cutting-edge artificial intelligence methods for identifying mobile malware in Android systems, addressing the lack of research on effective malware detection in the face of the increasing complexity and prevalence of malware on Android devices. This paper is organized into six main sections. The second part details the process of gathering existing studies. The following techniques for detecting mobile-based malware are presented in detail. Below is a comparative study of Mobile-based Malware Detection techniques using Advanced Artificial Intelligence Techniques. The final section examines the comparative analysis of Advanced Artificial Intelligence Techniques for Malware Detection methods.

2. Research method

The objective of this systematic review is to detect malware. The supervised machine learning approaches that have been effectively used in the actual world for malware detection are highlighted in the machine learning workflow for malware detection, besides outlining the benefits and drawbacks of every method and recommending how to improve those detection methods. Consequently, the current research articles pertinent to detection methods have been gathered from Google Scholar, IEEE Explore, and well-known journals. Considering the search term Table 1 shows that around 2,500 papers that were already published were found during the initial search. it also witnesses a field of Android malware detection that is seeing a rise in the use of machine learning approaches, as traditional signature-based detection techniques frequently cannot keep up with the volume and unpredictability of harmful software. Given the ubiquity of the Android operating system and the frequency of malware attacks, it is imperative to provide efficient detection techniques to guard against these dangers.

Table 1. Description of search words

Search of word	Set of keywords
Malware	badware, malicious software, malignant, virulent, Malicious application, Malcode
Mobile	mobile devices, android, Model for Android, Smartphone, PDA (Personal Digital Assistant)
Advanced	advance, applying, evolvable, Modern, Developed
Artificial Intelligence techniques	deep learning, machine learning, behavioural analysis, AI approaches, Neural Network methods, Intelligent Systems methods

General workflow for malware detection using supervised machine learning techniques is a common machine learning type used for predicting, forecasting or classification tasks. They are named supervised learning because the learning algorithm uses label data. Figure 1 demonstrates the general steps for malware detection using supervised machine-learning techniques.

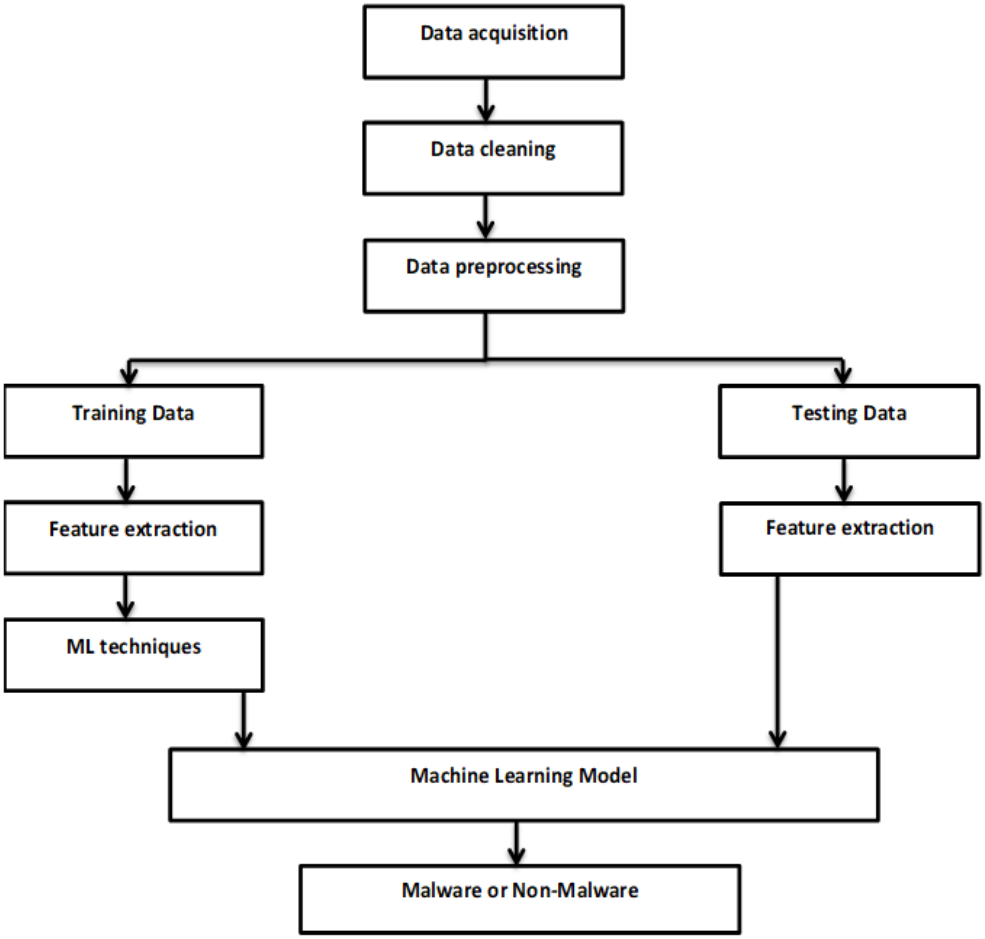

Fig. 1 Supervised detection techniques malware

Data Collection

Gathering or collecting data is the first stage in any supervised machine-learning technique. The quantity and quality of the data collected are essential to create an accurate model. Malware detection, for example, Android malware detection with the MH-100K dataset [8], CICAndMal2017[9],[6] CICInvesAndMal2019 dataset[10], CCCS-CIC-AndMal-2020[11], The study utilizes the Andro-AutoPsy [12], Android Malware, McAfee Labs[13], Android malware AndroZoo[14], While some produce their datasets and employ various machine learning methodologies to validate them[15],[50] Table 2 lists the different dataset types that were used in malware detection. Because of its extensive representation of real-world network traffic, a wide range of cyberattacks, large volume, and established reputation within the cybersecurity research community, the Canadian Institute for Cybersecurity (CIC) dataset is the most commonly utilized. This allows for a thorough evaluation of intrusion detection methods and promotes advancements in the field. It can also be obtained from reliable and accessible sources. You can access the dataset below directly from the source.

Table2. Types of datasets in malware detection

Dataset Names	Behavioral Features	Machine Learning	Deep Learning
MH-100K dataset [6]		✖	
CICAndMal2017 [7], [4]		✖	✖
CICInvesAndMal 2019 [8]		✖	✖
CCCS-CIC-AndMal-2020[9]	✖		
Andro-AutoPsy [10]		✖	
Android Malware, McAfee Labs [11]		✖	✖
AndroZoo [12]		✖	

Data cleaning and pre-processing

In its current format, acquired data cannot be directly put into a machine-learning pipeline. As a result, raw data needs to be cleaned up and converted into a machine-readable format. Since real-world data is frequently missing, noisy, inconsistent, and incomplete, data pretreatment is crucial for data warehousing and mining. Data minimization, integration, transformation, and purification are all involved. Data screening techniques are used to eliminate redundant data, fill in gaps, and eliminate noisy data to maintain the robustness and dependability of the detection model, missing values in the dataset for malware detection on Android smartphones can be handled using statistical imputation techniques and feature selection methods. Statistical imputation techniques such as mean and mode imputation for categorical features to maintain the integrity and efficacy of the detection model features with a high percentage of missing values can also be excluded using feature selection approaches. [16-18]

3. Feature Extraction (Machine Learning Models)

(Binary classification models in machine learning) If more grouping strategies are needed to choose useful API calls, more investigation can be done to increase the precision of mobile malware detection. Advanced persistent threats (APTs) and zero-day attacks are two more categories of mobile malware attacks that can be detected and classified using the suggested classification approach. To ensure the model is reliable and broadly applicable, its performance can be assessed on bigger and more varied datasets. To offer real-time protection against

mobile malware, the model can be included in currently installed antivirus or mobile security software.[19],[20] Investigate using more sensitive API calls to boost the precision and accuracy of the Android malware detection system. Examine whether adding other machine learning algorithms or ensemble techniques to the integrated learning model can improve its detection performance even more. Carry out tests using a more extensive and varied dataset to verify the effectiveness of the suggested plan in practical situations. By regularly updating the list of sensitive API calls and adjusting for changing malware behaviours, assess the scheme's capacity to identify novel and developing forms of Android malware.[21] According to the article, future research should enhance risk identification accuracy by adding elements like dynamic permissions and behavioural analysis. To improve the system's detection capabilities, it also suggests investigating the application of additional machine learning methods and methodologies.

To further confirm the efficacy of the suggested strategy, the authors advise running trials on a bigger dataset. To give users access to real-time risk assessment, future research can also investigate integrating the risk detection system into app stores or security tools.[22] According to the authors, future studies should investigate different metaheuristic algorithms for feature selection in Android malware detection. More research can be done to compare the suggested method with other feature selection strategies and assess its efficacy on larger datasets. Subsequent research endeavours may investigate the amalgamation of the dragonfly algorithm with alternative machine-learning algorithms to augment the precision of Android malware identification. To evaluate the dragonfly algorithm's efficacy in various cybersecurity scenarios, researchers can also look into how it is applied in other fields.[23] To address sophisticated malware, the study proposes including code2vec and other program language embedding technologies into the system. To increase the scalability of the GNN-based malware detection method in the future, the authors intend to investigate more invariant graph properties. To manage sophisticated malware, they also discuss the potential integration of additional program language embedding technologies, such as code2vec, into their system. The authors suggest using Word2Vec and GNN algorithms to extract ultrafunction semantic and interfunction structure information automatically for efficient malware identification.[24] The research recommends combining FT and SigPID into DREBIN to enhance the efficiency of running time and the accuracy of malware detection. Future research will examine this integration.[25] we addressed three major groups of methodologies: hybrid, dynamic, and static analysis. Static analysis uses the least time and processing resources of all of these. It is also the simplest strategy to put into practice. For this reason, static analysis has been used in most studies published in the literature. We find that significantly fewer academics are concentrating on dynamic analysis because it is more difficult to set up and operate than static analysis. However, the outcomes demonstrate that dynamic analysis can perform as well as static analysis and, in many instances, even better. In theory, hybrid analysis combines the advantages of static and dynamic features to provide the best of both worlds. All of the studies that used hybrid analysis revealed high accuracy rates; however, they were similar to dynamic analysis.[26] Lately, the mobile ecosystem has had a significant security risk from mobile malware. To tackle security concerns, machine learning algorithms have to achieve higher accuracy levels. These concerns are typically related to the effective feature selection and the effective operation of the chosen classifiers. To demonstrate that a higher degree of accuracy may be attained while retaining high efficiency and effectiveness, we have statically examined the Android environment in this study. We've chosen a foundational strategy from Zhu et al. in which we trained the Rotation Forest classifier to identify characteristics chosen from permissions, system API calls and events, and additional categories. We looked closely at Google's and Zhu et al.'s risky permission sets using a variety of tests to determine the greatest number of permission sets that will increase efficiency without sacrificing detection accuracy. Using the permissions dataset, we assessed the various classifiers—Random Forest, SVM, Naïve Bayes, and Rotation Forest.[27],[28] Although the field of research on using intelligent algorithms to detect ransomware is still in its early stages, it is expanding, suggesting that there may be opportunities for future advancements in this area. There are still untapped possibilities for future advancements in machine learning algorithms for ransomware detection.[29]

Table 3. Machine learning models

Ref	Year	Dataset	Features	Classification	Accuracy
[25]	2018	Benign apps	Analyze permissions	SigPID	93.62%
[19]	2020	Real malware	Permission requests and API calls	F-measure	94.30%
[21]	2021	malware detection scheme	Static analysis	DT+SVM+KNN	94%
[24]	2021	AndroZoo, Drebin	Static analysis via GNN	CGDroid	99.22%
[22]	2022	Various sources	Static permissions achieved	ANN	96.70%
[26]	2022	Drebin, Android Gnome Project, AMD, VirusShare	static, Dynamic, hybrid analysis	SVM and logistic regression models	98%
[27]	2022	PerDRaML scheme	Permissions	Random Forest	89.96%
[29]	2023	Supplementary Materials	Dynamic	RandomForest	99.00%
[23]	2023	DREBIN Android Malware	Static and hybrid	Random Forest (RF)	96.52%

4. Deep Learning Models

study the application of other deep learning models for Android malware detection, such as Convolutional Neural Networks (CNNs). Examine whether adding elements to the detection model, including permissions and API requests, improves its efficacy. Assess the suggested approach using a more extensive dataset to confirm its efficacy and applicability. To determine which proposed method is better than others, compare it with other cutting-edge malware detection techniques in trials.[30],[31] As malware obfuscation and detection avoidance techniques get more sophisticated, further study may be done to examine how well DL-Droid detects new and developing forms of Android malware. The authors propose examining DL-Droid's performance on a more extensive dataset of Android applications to verify its efficacy and scalability in practical situations. Given the differences in security features and vulnerabilities throughout Android OS versions, it would be advantageous to investigate DL-Droid's capability for malware detection in various versions. Future research can also enhance DL-Droid's functionality by cutting down on processing overhead and boosting the effectiveness of the deep learning system for mobile malware detection in real time. [32] Chimera-S, which presently employs a rather shallow Deep Neural Network (DNN) architecture, will be the subject of further investigation to investigate deeper designs.[33] Deep learning models may perform better if more characteristics and data types are explored for Android virus detection. Examining the combination of static analysis data with other forms of data, including network traffic data or dynamic analysis data, may help to enhance detection accuracy and offer a more thorough knowledge of Android malware. Finding the best setups for multimodal learning in Android malware detection may require experimenting with various deep network designs and input conditions. Researching whether the suggested multimodal learning strategy can be applied to other mobile platforms, such as iOS, could increase the detection techniques' scope of application.

[34] It is necessary to conduct more research on the effectiveness of different deep learning models for Android botnet detection. Investigating any new features or variables that can improve Android botnet detection is advisable. A more thorough knowledge of deep learning models' performance would be possible through their evaluation of bigger and more varied datasets than the ISCX botnet dataset. It is necessary to analyze the drawbacks and potential difficulties of using deep learning models for Android botnet detection. Deep learning approaches and other cutting-edge detection methods could be compared to determine the advantages and disadvantages of each strategy.[35] JSON log files for every APK file are parsed to extract the features. This results in constructing a feature vector that represents all low-level behavioural information. Yeo-Johnson power transformation is used to normalize a dataset of feature vectors. Rescaling data to have a mean of zero and a standard deviation of one (unit variance) is commonly associated with standardization. Data becomes more Gaussian-like through power transformation.

[36] We suggested MAPAS, a successful and practical method for detecting malware. MAPAS uses a deep learning technique to identify common aspects of API call graphs that are retrieved from malicious applications. The malware is then identified using an efficient lightweight classifier, depending on the features. According to the findings of our test, MAPAS performs better than MaMaDroid, a cutting-edge method, in terms of both accuracy and computer resource utilization when it comes to identifying unknown malware. Furthermore, MAPAS has a high degree of accuracy while detecting any malware.[37] Selecting attributes for classification is crucial because they help identify the class to which the new record will belong. From this vantage point, all Android applications' permissions and API calls are taken out, and both were included in the dataset as features. Androguard is a complete package utility limited to Python environments that is made to work with Android files. One usage for it is as a tool for Android application reverse engineering. By extracting the DEX file permissions for every APK file individually, the Androguard program is used to analyze APK files. As a result, we created a data frame with features (columns) and applications (rows). The columns in the data frame represent different permissions or API calls with binary values, while the rows comprise both benign and malicious APK files.[38] According to the article, future research should enhance the DeepAMD method's effectiveness in locating and identifying Android malware It also recommends investigating the application of other deep learning architectures or approaches to improve malware detection performance and accuracy even more. The report also notes that as attackers are always improving their methods, they must create tools for detecting and identifying new and emerging varieties of Android malware The authors suggest integrating DeepAMD with current security measures to offer a thorough defence against malware for Android devices. Moreover, the study recommends carrying out more thorough tests and analyses on bigger datasets to confirm DeepAMD's efficacy in practical situations.[39],[40] .

Table 4. Deep learning models

Ref	Year	Dataset	Features	Classification	Accuracy
[30]	2019	DREBIN	System call sequences, LSTM	LSTM	96.60%
[34]	2019	benign and malicious Android Packages (APKs)	permissions, denoted as hardware features	Multimodal deep neural network (DNN)	94.50%
[32]	2020	Intel Security (McAfee Labs)	Dynamic and static features	DL-Droid	99.60%
[35]	2021	the ISCX botnet	NN, DNN, LSTM, GRU, CNN-LSTM, CNN-GRU, permissions, API calls	CNN-GRU, DNN	99.10%
[36]	2021	mobile malware Mini dump	Dynamic Analysis-Based Behavior Monitoring	De-LADY	98.08%
[38]	2021	CICAndMal2017	Permissions and API calls, static analysis	Recall (GRU)	99.20%
[39]	2021	CICInvesAndMal2019, CICAndMal2017	Artificial Neural Network (ANN)Static and Dynamic layers	DeepAMD	93.40%
[37]	2022	API call graphs of applications	utilizes CNN, Features of API call graphs of malware	MAPAS	91.27%
[33]	2022	Omnidroid	CNN, DNN, TN, DEX, static, Dynamic analysis, Permission	Voting classifier (Chimera)	89.70%

5. Behavioural Analysis model

According to the article, future research should focus on a more thorough feature selection investigation incorporating different algorithm combinations to enhance the identification of malicious Android applications. As a guide for future study, the evaluation offers insights into the composition of application executions, the effects of various machine learning strategies, and the kind and volume of inputs to dynamic analyses. Since the system call encoding demonstrated superior quality than the frequency representation, suggesting its potential for useful application in malware detection, the researchers suggest further examination of the system call encoding into a feature vector representation. Further investigations could investigate new methods and strategies for dynamic analysis-based malware detection in Android. [41] Using various dynamic behaviour features, the article suggests a dynamic analysis framework dubbed EnDroid for extremely accurate Android malware detection. It covers common application-level malicious activities like stealing personal information, subscribing to premium services, and communicating with malicious services, as well as system-level behaviour traces. EnDroid combines an ensemble learning technique for classification and a feature selection approach to extract important behaviour features. Experiments on two datasets demonstrate the efficacy of EnDroid; Stacking achieves the best classification performance and exhibits potential in identifying Android malware.[42] The authors intend to look into how resistant MAMADROID is to potential evasion methods, like malicious apps that have been repackaged and API calls that have been injected to modify Markov models deliberately. Additionally, they want to investigate the usage of finer-grained abstractions and the potential for dynamic analysis to be used as a seed for behavioural modelling rather than static analysis. Future studies will rigorously investigate the possibility of generating sequences that result in Markov chains that resemble benign programs while doing malicious activities. The authors note that although they have not made the datasets and features [43], they Examine the resistance to evasion tactics, such as malicious apps that have been repackaged and API calls that have been injected to modify Markov models maliciously. Examine the feasibility of using dynamic analysis rather than static data to seed MaMaDroid's behavioural modelling. To overcome the drawbacks of static analysis techniques for malware detection on Android, incorporate dynamic analysis into the models that MaMaDroid uses to construct its models.[44] The sophistication of Android malware is rising steadily. Current signature-based antimalware mechanisms cannot detect zero-day assaults, and even small code changes can hide an infection. Malware authors typically combine disparate portions of malware code or add functionality to already existing malware; this is why Android malware is categorized into families, each of which shares dangerous behaviour. The method we describe in this research for identifying Android malware families is based on model-checking ways to examine and validate the Java Byte code generated by the source code compilation.[45] The study recommends that to enhance further the Android malware classification system's effectiveness, future research should investigate various weighting algorithms for the dynamic weighted federated averaging (DW-FedAvg) methodology. The authors also suggest looking into how other hyperparameters, such as batch size and learning rate,

affect the effectiveness of the DW-FedAvg approach. Further investigation may examine the DW-FedAvg approach's application to domains [18] other than Android malware classification to assess its efficacy in various contexts [46].

Table 5. Behavioural Analysis model

Ref	Year	Dataset	Features	Classification	Accuracy
[41]	2016	Google Play, the Drebin frequency, random LASSO and ridge	sequences of system calls	MALINE tool	96%
[43]	2016	Benign Applications, Malicious Applications	Markov Chain Abstraction, Feature Vector, Mode-dependent Vector Size	MAMADROID	99%
[45]	2016	Drebin project	Analyzing the Java Bytecode	DroidKungFuMorph	94%
[42]	2018	Benign Application, Malicious Application, AndroZoo, Drebin	Dynamic behavior	EnDroid	96.82% on dataset M1, 97.02% on dataset M2
[44]	2019	Benign, malicious apps	Static-analysis, sequences, frequency	MAMADROID	99%
[46]	2022	Melgenome, Drebin, Kronodroid, and Tuandromd	Classification system, dynamic, weighted, behavioral patterns	DW-FedAvg	99.40%
[51]	2022	malicious Android apps	dynamic analysis and behaviour-based analysis, static analysis.	MalApp (uses static analysis)	98.33%

The process of learning

In machine learning, the classifier requires input from extracted features, which are then validated to obtain results. Selecting the appropriate classifier is an essential step as it plays a vital role in determining the accuracy of the prediction [47]. Thus, understanding the advantages and disadvantages of each classifier can assist in accurately choosing the appropriate classifier. Machine learning classification approaches like SVM, Artificial neural network (ANN), Naïve Bayes (NB), and K-NN are considered highly effective and suitable techniques for classifiers [48]– [49]. Also, in deep learning, Data processing is the conversion of data from one form to another, intending to make it more relevant and informative. By leveraging Machine Learning algorithms, mathematical modelling, and statistical expertise, it is possible to automate this entire process fully. Deep learning classification approaches like convolutional neural network (CNN), recurrent neural network (RNN), long short-term memory networks (LSMN), etc. Table 8 reviews machine learning classifiers used for detecting Mobile Malware, highlighting their advantages and disadvantages.

Table 6. Pros and downsides for (machine and deep) learning algorithms [48-49]

classification	Pros	Downsides
SVM	- Functions effectively when the margin is unambiguous and, the number of surpasses the samples. - Ensure optimization operator is and the dimensions number of mal memory in high-dimensional spaces.	- The performance decreases while dealing with a large dataset due to the increased time required for training. - The performance of the system will decrease when the dataset contains a significant amount of noise.
KNN	- Simple to execute, particularly for multi-class categorization. - The training phase is unnecessary. - The addition of new data does not impact the accuracy.	- The performance decreases when the dataset is huge and contains noise. - The input features must be consistent.
ANN	- Very good predictive capability.	- The processing time is indeterminate.

	- Well-suited for models that do not involve mathematical calculations. - The ability to derive relevance from intricate or ambiguous information	- Quality predictions necessitate a substantial amount of data.
CNN	- They can classify and identify crucial visual aspects without human supervision. - CNNs efficiently handle high-dimensional data and communicate information between layers.	- Using a lot of computer resources and time. - May lose information, however, residual structures can help.
RNN	- Effectively manage text, speech, and time series. - Unlike feedforward NN, this model is capable of processing inputs of any length. - Increase training efficiency by sharing weights between time steps.	- Vanishing or ballooning gradients might make RNN training challenging. This occurs when the loss function's gradients concerning the parameters grow exceedingly small or huge over time.
LSTM	- Effectively capturing and managing long-term dependencies in sequential data.	- Requiring a substantial quantity of training data can be computationally costly.

Comparative analysis and discussion of mobile malware detection techniques

Based on the reviewed research paper, we analyze the techniques from two perspectives: i) the feature-based techniques and ii) the type of classifier employed by each method. Regarding the point on the featured-based approach, it was shown that Mobile-based Malware characteristics can yield favourable outcomes if they are not influenced by noise. From a classifier perspective in machine learning, the SVM classifier is widely utilized and yields favourable outcomes, whether by utilizing individual features or hybrid features, compared to the ANN classifier, which produces superior results. In deep learning, it is observed that employing grids (CNN-LSTM) yields superior outcomes compared to the remaining algorithms. Furthermore, utilizing a substantial dataset with an effective feature selection technique might enhance the classifier's precision.

6. Conclusion

This paper provides a thorough examination of the most recent techniques employed in the detection of Android malware through the utilisation of artificial intelligence. To achieve this, we categorise and thoroughly analyse the latest and cutting-edge research in the field from 2015 to 2023, spanning eight years. This study is predicated on the specific sort of analysis performed, the methodology implemented for extracting features, the dataset utilised, and the artificial intelligence classification algorithms employed. In addition, we offer additional information and examine the current trends in our study. The findings indicate that the majority of the methods employ different essential parameters, such as the dataset, the analysis (feature collection), and the assessment criteria for detection. To address this issue, we developed a four-step convergence process that can act as a foundation and catalyst for future mobile machine learning-based Android malware detection techniques. Future research should incorporate an initial evaluation of the proposed convergence strategy by comparing the outcomes derived from its several pathways concerning the age of the dataset, the analytical technique employed, and the machine learning methodologies selected. We have discussed a systematic study that relies on the latest developments in identifying malicious programmes in both official and unofficial Android Markets. This research showcases many detection methods and systems that utilise static, dynamic, and hybrid approaches. In addition, we discuss various tactics that might be employed to counteract the update assault, such as attack trees, permissions, network traffic monitoring, mitigation measures, and permission patterns.

Reference

[1] E. B. Karbab, M. Debbabi, A. Derhab, and D. Mouheb, "MalDozer: Automatic framework for android malware detection using deep learning," Digit. Investig., vol. 24, pp. S48–S59, 2018.

[2] Abbas Aqeel Kareem, Dalal Abdulmohsin Hammood, Ruaa Ali Khamees, and Nurulisma Binti Hj. Ismail, "Object Tracking with the Drone: Systems Analysis," Journal of Techniques, vol. 5, no. 2 SEEngineering, pp. 89–94, Jun. 2023, DOI: https://doi.org/10.51173/jt.v5i2.755.

[3] E. B. Karbab, M. Debbabi, A. Derhab, and D. Mouheb, "Scalable and robust unsupervised android malware fingerprinting using community-based network partitioning," Comput. Secur., vol. 96, p. 101932, 2020.

[4] K. Liu, G. Zhang, X. Chen, Q. Liu, L. Peng, and L. Yurui, "Android malware detection based on sensitive patterns," Telecommun. Syst., vol. 82, no. 4, pp. 435–449, 2023.

[5] Asaad Yaseen Ghareeb, S. K. Gharghan, and Rosdiadee Nordin, "Wireless Sensor Network-Based

Artificial Intelligent Irrigation System: Challenges and Limitations," Journal of Techniques, vol. 5, no. 3 SEEngineering, pp. 26–41, Sep. 2023, DOI: https://doi.org/10.51173/jt.v5i3.1420

[6] S. Shakya and M. Dave, "Analysis, detection, and classification of android malware using system calls," arXiv Prepr. arXiv2208.06130, 2022.

[7] H. Bai, N. Xie, X. Di, and Q. Ye, "Famd: A fast multifeatured android malware detection framework, design, and implementation," IEEE Access, vol. 8, pp. 194729–194740, 2020.

[8] H. Bragança, V. Rocha, L. Barcellos, E. Souto, D. Kreutz, and E. Feitosa, "Android malware detection with MH-100K: An innovative dataset for advanced research," Data Br., vol. 51, p. 109750, 2023.

[9] F. Taher, O. AlFandi, M. Al-kfairy, H. Al Hamadi, and S. Alrabaee, "DroidDetectMW: A Hybrid Intelligent Model for Android Malware Detection," Appl. Sci., vol. 13, no. 13, p. 7720, 2023.

[10] S. Liaqat, K. Dashtipour, K. Arshad, K. Assaleh, and N. Ramzan, "A hybrid posture detection framework: Integrating machine learning and deep neural networks," IEEE Sens. J., vol. 21, no. 7, pp. 9515–9522, 2021.

[11] R. Islam, M. I. Sayed, S. Saha, M. J. Hossain, and M. A. Masud, "Android malware classification using optimum feature selection and ensemble machine learning," Internet Things CyberPhysical Syst., vol. 3, pp. 100–111, 2023.

[12] J. Lee, H. Jang, S. Ha, and Y. Yoon, "Android malware detection using machine learning with feature selection based on the genetic algorithm," Mathematics, vol. 9, no. 21, p. 2813, 2021.

[13] N. McLaughlin et al., "Deep android malware detection," in Proceedings of the seventh ACM on conference on data and application security and privacy, 2017, pp. 301–308.

[14] K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun, and H. Liu, "A review of Android malware detection approaches based on machine learning," IEEE Access, vol. 8, pp. 124579–124607, 2020.

[15] T. Kim, B. Kang, and E. G. Im, "Runtime detection framework for android malware," Mob. Inf. Syst., vol. 2018, 2018.

[16] S. A. Alasadi and W. S. Bhaya, "Review of data pre-processing techniques in data mining," J. Eng. Appl. Sci., vol. 12, no. 16, pp. 4102–4107, 2017.

[17] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," Glob. Transitions Proc., vol. 3, no. 1, pp. 91–99, 2022.

[18] S. S. Jasim and A. K. A. Hassan, "Modern drowsiness detection techniques: A review," Int. J. Electr. Comput. Eng., vol. 12, no. 3, p. 2986, 2022.

[19] M. Alazab, M. Alazab, A. Shalaginov, A. Mesleh, and A. Awajan, "Intelligent mobile malware detection using permission requests and API calls," Futur. Gener. Comput. Syst., vol. 107, pp. 509–521, 2020.

[20] A. Aldelemy and Raed A. Abd-Alhameed, "Binary Classification of Customer's Online Purchasing Behavior Using Machine Learning," Journal of Techniques, vol. 5, no. 2 SE-Management, pp. 163–186, Jun. 2023, DOI: https://doi.org/10.51173/jt.v5i2.1226

[21] J. Yu, C. Zhao, W. Zheng, Y. Li, C. Zhang, and C. Chen, "Android Malware Detection Using Ensemble Learning on Sensitive APIs," in Edge Computing and IoT: Systems, Management and Security: First EAI International Conference, ICECI 2020, Virtual Event, November 6, 2020, Proceedings 1, Springer, 2021, pp. 126–140.

[22] M. Dhalaria and E. Gandotra, "Risk Detection of Android Applications Using Static Permissions," in Advances in Data Computing, Communication and Security: Proceedings of I3CS2021, Springer, 2022, pp. 591–600.

[23] M. Guendouz and A. Amine, "A New Feature Selection Method Based on Dragonfly Algorithm for Android Malware Detection Using Machine Learning Techniques," Int. J. Inf. Secure. Priv., vol. 17, no. 1, pp. 1–18, 2023.

[24] P. Feng, J. Ma, T. Li, X. Ma, N. Xi, and D. Lu, “Android malware detection via graph representation learning,” Mob. Inf. Syst., vol. 2021, pp. 1–14, 2021.

[25] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye, “Significant permission identification for machine-learning-based android malware detection,” IEEE Trans. Ind. Informatics, vol. 14, no. 7, pp. 3216–3225, 2018.

[26] A. Muzaffar, H. Ragab Hassen, M. A. Lones, and H. Zantout, “An in-depth review of machine learning based Android malware detection,” Comput. Secur., vol. 121, p. 102833, 2022, Doi: https://doi.org/10.1016/j.cose.2022.102833.

[27] F. Akbar, M. Hussain, R. Mumtaz, Q. Riaz, A. W. A. Wahab, and K.-H. Jung, “Permissions-based detection of android malware using machine learning,” Symmetry (Basel)., vol. 14, no. 4, p. 718, 2022.

[28] S. S. Jasim and A. A. M. Al-Taei, “A Comparison Between SVM and K-NN for classification of Plant Diseases,” Diyala J. Pure Sci., vol. 14, no. 2, pp. 94–105, 2018.

[29] J. A. H.-S. and M. Hernández-Álvarez, “Dynamic feature dataset for ransomware detection using machine learning algorithms,” Sensors, vol. 23, no. 3, p. 1053, 2023.

[30] X. Xiao, S. Zhang, F. Mercaldo, G. Hu, and A. K. Sangaiah, “Android malware detection based on system call sequences and LSTM,” Multimed. Tools Appl., vol. 78, pp. 3979–3999, 2019.

[31] SamaAndroid malware detection based on system call sequences and LSTM Hayder Abdulhussein AlHakeem, Nashaat Jasim Al-Anber, Hayfaa Abdulzahra Atee, and Dr. Mahmod Muhamad Amrir, “Iraqi Stock Market Prediction Using Artificial Neural Network and Long ShortTerm Memory,” Journal of Techniques, vol. 5, no. 1 SE-Management, pp. 156–163, Apr. 2023, Doi:

https://doi.org/10.51173/jt.v5i1.846

[32] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, “DL-Droid: Deep learning based android malware detection using real devices,” Comput. Secur., vol. 89, p. 101663, 2020.

[33] A. S. de Oliveira and R. J. Sassi, “Chimera: an android malware detection method based on multimodal deep learning and hybrid analysis,” Authorea Prepr., 2023.

[34] J. McGiff, W. G. Hatcher, J. Nguyen, W. Yu, E. Blasch, and C. Lu, “Towards multimodal learning for android malware detection,” in 2019 International Conference on Computing, networking and Communications (ICNC), IEEE, 2019, pp. 432–436.

[35] S. Y. Yerima, M. K. Alzaylaee, A. Shajan, and V. P, “Deep learning techniques for android botnet detection,” Electronics, vol. 10, no. 4, p. 519, 2021.

[36] V. Sihag, M. Vardhan, P. Singh, G. Choudhary, and S. Son, “De-LADY: Deep learning-based Android malware detection using Dynamic features.,” J. Internet Serv. Inf. Secur., vol. 11, no. 2, pp. 34–45, 2021.

[37] J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, “MAPAS: a practical deep learning-based android malware detection system,” Int. J. Inf. Secur., vol. 21, no. 4, pp. 725–738, 2022.

[38] O. N. Elayan and A. M. Mustafa, “Android malware detection using deep learning,” Procedia Comput. Sci., vol. 184, pp. 847–852, 2021.

[39] S. I. Imtiaz, S. ur Rehman, A. R. Javed, Z. Jalil, X. Liu, and W. S. Alnumay, “DeepAMD: Detection and identification of Android malware using high-efficient Deep Artificial Neural Network,” Futur. Gener. Comput. Syst., vol. 115, pp. 844–856, 2021.

[40] نادية، Sadik Kamel Gharghan, Ammar Hussein Mutlag, and M. G. M. Abdolrasol, “Children Tracking System Based on ZigBee Wireless Network and Neural Network,” Journal of Techniques, vol. 5, no. 1 SE-Engineering, pp. 103–113, Apr. 2023, DOI: https://doi.org/10.51173/jt.v5i1.838

[41] M. Dimjaševid, S. Atzeni, I. Ugrina, and Z. Rakamaric, “Evaluation of Android malware detection based on system calls,” in Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics, 2016, pp. 1–8.

[42] P. Feng, J. Ma, C. Sun, X. Xu, and Y. Ma, “A novel dynamic android malware detection system with ensemble learning,” IEEE Access, vol. 6, pp. 30996–31011, 2018.

[43] E. Mariconti, L. Onwuzurike, P. Andriotis, E. De Cristofaro, G. Ross, and G. Stringhini, “Mamadroid: Detecting android malware by building Markov chains of behavioural models,” arXiv Prepr. arXiv1612.04433, 2016.

[44] L. Onwuzurike, E. Marconi, P. Andriotis, E. De Cristofaro, G. Ross, and G. Stringhini, “Mamadroid: Detecting android malware by building Markov chains of behavioural models (extended version),” ACM Trans. Priv. Secur., vol. 22, no. 2, pp. 1–34, 2019.

[45] P. Battista, F. Mercaldo, V. Nardone, A. Santone, and C. A. Visaggio, “Identification of Android Malware Families with Model Checking.,” in ICISSP, 2016, pp. 542–547.

[46] A. Chaudhuri, A. Nandi, and B. Pradhan, “A Dynamic Weighted Federated Learning for Android Malware Classification,” in Soft Computing: Theories and Applications: Proceedings of SoCTA 2022, Springer, 2023, pp. 147–159.

[47] S. S. Jasim and A. K. A. Hassan, “Driving sleepiness detection using electrooculogram analysis and grey wolf optimizer,” Int. J. Electr. Comput. Eng., vol. 12, no. 6, p. 6034, 2022.

[48] S. S. Jasim, A. K. A. Hassan, and S. Turner, “Driver drowsiness detection using gray wolf optimizer based on face and eye tracking,” Aro-The Sci. J. Koya Univ., vol. 10, no. 1, pp. 49–56, 2022.

[49] S. Turner, S. S. Jassin, and A. K. A. Hassan, “Optimizing artificial neural networks using LevyChaotic mapping on Wolf Pack optimization algorithm for detect driving sleepiness,” Iraqi J. Comput. Commun. Control Syst. Eng., vol. 22, no. 3, pp. 128–136, 2022.

[50] S. Seraj, E. Pimenidis, M. Pavlidis, S. Kapetanakis, M. Trovati, and N. Polatidis, “BotDroid: Permission-Based Android Botnet Detection Using Neural Networks,” in International Conference on Engineering Applications of Neural Networks, Springer, 2023, pp. 71–84.

[51] N. Paul, A. J. Bhatt, and S. Rizvi, “Malware Detection in Android Apps Using Static Analysis,” J. Cases Inf. Technol., vol. 24, no. 3, pp. 1–25, 2022.

[52] G. Canfora, E. Medvet, F. Mercaldo, and C. A. Visaggio, “Detecting android malware using sequences of system calls,” in Proceedings of the 3rd International Workshop on Software Development Lifecycle for Mobile, 2015, pp. 13–20.