Telematics and Informatics Reports 14 (2024) 100130

Contents lists available at ScienceDirect

Telematics and Informatics Reports

journal homepage: www.elsevier.com/locate/teler

Android malware detection and identification frameworks by leveraging the machine and deep learning techniques: A comprehensive review

Santosh K. Smmarwar *, Govind P. Gupta, Sanjay Kumar

Department of Information Technology, National Institute of Technology, Raipur 492010, India

ARTICLE INFO

Keywords:
Machine learning
Deep learning
Android malware detection
Malware detection

ABSTRACT

The ever-increasing growth of online services and smart connectivity of devices have posed the threat of malware to computer system, android-based smart phones, Internet of Things (*IoT*)-based systems. The anti-malware software plays an important role in order to safeguard the system resources, data and information against these malware attacks. Nowadays, malware writers used advanced techniques like obfuscation, packing, encoding and encryption to hide the malicious activities. Because of these advanced techniques of malware evasion, traditional malware detection system unable to detect new variants of malware. Cyber security has attracted many researchers in the past for designing of Machine Learning (*ML*) or Deep Learning (*DL*) based malware detection models. In this study, we present a comprehensive review of the literature on malware detection approaches. The overall literature of the malware detection is grouped into three categories such as review of feature selection (*FS*) techniques proposed for malware detection, review of *ML*-based techniques proposed for malware detection and review of *DL*-based techniques proposed for malware detection. Based on literature review, we have identified the shortcoming and research gaps along with some future directives to design of an efficient malware detection and identification framework.

Introduction

Malware (malicious software) is a malicious code that is harmful and destructive to computer system, digital system, and mobile and electronic devices. The intension of malware is to damage the digital system and steal the valuable information [1,2]. Malware is major threat to system's confidentiality, integrity, and privacy [3]. There are various types of malwares exist such as Viruses, Worms, Trojans, Backdoor [4], Ransomware [5], Adware, Spyware [6]. The tremendous increase in the year wise number of malware attacks worldwide [7] as shown in Fig. 1 has become one of the biggest threats to software applications and various system such as personal computers, portable laptops, android operating system-based smart devices, Internet of Thing [8].

The journey of malware began in the year of 1986 when first malware named as brain was developed. It was distributed very fast into the computer systems and infected millions of computer. This was the growing phase of malware development and computerization system. Therefore, at that time very limited malware was available [9]. Nowadays, with growth of digital devices or smart devices number of new malware are created to compromise the security of devices and steal valuable information. In the new era of digital technology malware writers upgraded themselves with new malware evasion techniques that make the malware detection hard for antivirus software and detection tools, these malware evasion techniques [10] such as obfuscation, polymorphisms, metamorphism, packers and crypters, which make detection process difficult and generates complex malicious hidden programs [11].

The expansion in utility of android based devices, smart systems and online services in various fields such healthcare, banking, education, manufacturing, agriculture etc. has attracted the attention of malware attackers to compromises weak security of these devices for financial gain or infected the systems files [12]. The sources of malware access to the devices are as through downloading software's from untrusted websites, by clicking the malicious *URLs* and phishing emails. Traditional techniques of malware detection work on the basis of signature patterns matching. The signatures can be cryptographic hashes and byte patterns, opcode analysis, these signature based analysis is known as static analysis, and it was performed without running the codes using reverse engineering of extracted features. The problem with static analysis is that it cannot detect the new signature or new malware

* Corresponding author.
E-mail addresses: sksmmarwar.phd2019.it@nitrr.ac.in (S.K. Smmarwar), gpgupta.it@nitrr.ac.in (G.P. Gupta), skumar.it@nitrr.ac.in (S. Kumar).

https://doi.org/10.1016/j.teler.2024.100130
Received 27 November 2023; Received in revised form 28 February 2024; Accepted 12 March 2024
Available online 12 March 2024

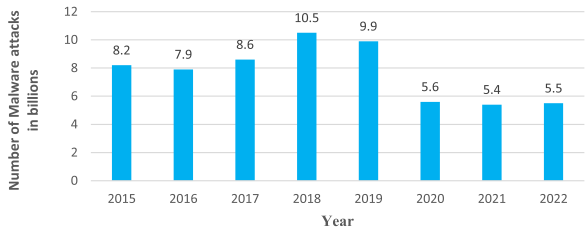

Fig. 1. Year wise depiction in Number of Malware Attacks.

because of not updating of signatures database. Therefore, to overcome these problems the dynamic analysis methods was developed to monitor the run time behaviors of software. Dynamic analysis is performed in virtual environment using cuckoo sandbox tool to unpack the hidden patterns of programs. Decodes the obfuscated and polymorphism activities that is activates while executing programs with malicious files [13]. However, sometime dynamic analysis is also inefficient to find out the zero-day malware detection due to overlapping nature of features. In order to solve these challenges researches has evolved the hybrid analysis approach for malware detection, which is the combination of static and dynamic analysis. Although, hybrid analysis has to be effective in finding the malicious activity into computers and smart systems [14].

Over the period of time, the growth in malware creation is increasing regularly. Thus, identification of malware through conventional methods like signature based, rule-based, graph based, entropy-based is difficult. Recently, the trends has been shifted towards artificial intelligence based malware detection system. Thus, to overcome these issues and challenges of new malware detection, the potential of *ML* and *DL* techniques have explored for designing malware detection framework as shown in Fig. 2 to detect new malware variants. In literature, various *ML* and *DL*-based frameworks have been mentioned [15]. The *ML* and *DL* models work on the basis training and testing dataset, the classification algorithm need to be learned using train dataset after proper training of classifier. Then, tested using new dataset for correct prediction of target class. Overall, these two factors have an effects on prediction capability of classifiers [16].

The *ML* classifiers are employed extensively for malware detection. Each classifiers has its own merits and limitations. The results of *ML* classifier also depends on quality features or optimal feature [17,18]. Whereas *DL* models is known for handling huge amount of dataset due to its automatic feature extraction capability. Many work has employed *DL*-based models for malware detection. Still the hunting of *DL* capability remain to be explored to detect advance malware for cross-platform [19]. To handle the advance and conventional malware we need to explore effective, efficient and reliable automated malware detection system. Although, number of author have designed automatic malware analysis systems to defeat the advanced obfuscation method, packing method and polymorphism method. In order to make effective these automated malware detection system the regular analysis of malware is needed to update the detection system with the new pattern and variants of existing malware [20]. Overall, malware detection is a proactive defence measure that plays a critical role in safeguarding digital assets, maintaining operational continuity, and preserving the privacy and security of users and organizations.

The primary contribution of this paper is as follows:

i. Present a detailed background of malware, effect of malware detection, importance of malware detection, type of malware, various malware evasion techniques and taxonomy of malware analysis.
ii. Present a detailed review of feature selection techniques used in the design of malware detection system and its comparative analysis with state-of-the-art *ML* and *DL* models.
iii. Present a comprehensive review of *ML*-based techniques proposed for malware detection and, its results analysis with different types feature extraction, feature selection methods and malware datasets.
iv. Present a review of *DL*-based techniques proposed for malware detection and its summary of different feature extraction method and different *DL* models.
v. Present the main shortcomings and research gaps that are observed in the review of the literature related to malware detection and analysis.
vi. Present the sources of different malware dataset used in the literature survey and provides future directives for emerging malware detection.

This survey study is structured in the following sections: Section 2 provides a motivation and problem statements. Section 3 is background

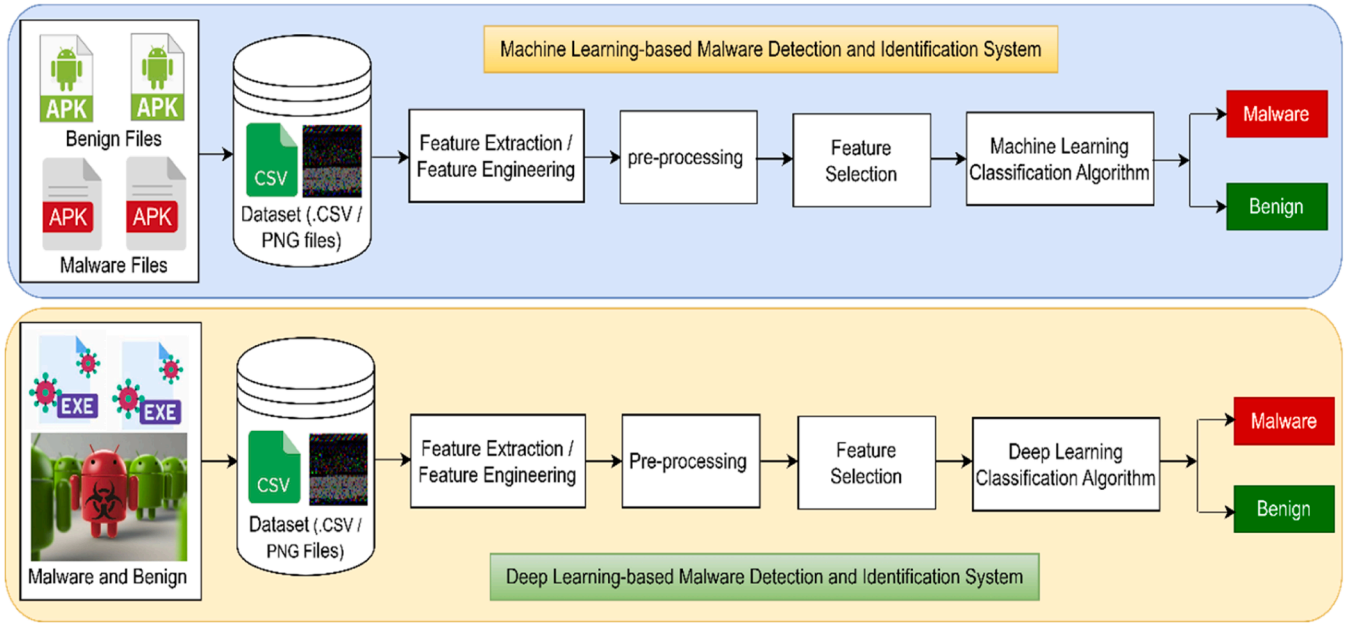

Fig. 2. General diagram of Malware Detection using *ML* and *DL*.

of malware. Section 4 mentioned about literature of different malware detection techniques. Section 5 presents shortcomings and research gap of study. Section 6 defines machine learning models used for malware detection. Section 7 show deep learning approaches. Section 8 is sources of malware dataset. Section 9 is recommendation for malware Detection system. Finally, Section 10 is conclusion of this study.

Motivation and problem statement

The cybersecurity landscape is constantly evolving, with new and sophisticated malware variants emerging regularly. Ongoing malware detection research ensures that security solutions stay up-to-date and effective against the latest threats. Malware poses a significant threat to computer systems, networks, and sensitive data. Detecting and removing malware is crucial to prevent unauthorized access, data breaches, and potential damage to hardware and software components [21]. Malware can cause financial losses by stealing sensitive information, such as credit card numbers or banking credentials, leading to fraudulent transactions or identity theft. Detecting and neutralizing malware helps protect users from financial harm. Malware can invade user's privacy by capturing personal information, monitoring online activities, and spying on user's activity. Malware detection helps safeguard user privacy and prevents unauthorized data collection. Some types of malware, such as ransomware can encrypt or delete files, leading to permanent data loss if the user doesn't pay the ransom. Early detection of ransomware can prevent such data loss. Malware can consume system resources, slow down computer performance, and cause crashes. Detecting and removing malware ensures that systems run efficiently and without disruptions.

The motivation of this research study is to develop effective malware detection system on the basis of shortcomings and research gap from literature review that can accurately and efficiently identify malicious software, protect users and systems from cyber threats, and contribute to improving overall cybersecurity. In addition, it can accurately distinguish between malicious and benign files, and minimize false positives (flagging legitimate files as malware) and false negatives (failing to detect actual malware) [22].

Background of malware

Effects of malware

Malware can have significant and far-reaching effects on individuals, organizations, businesses, governments and society as a whole. The effects of malware can vary depending on the type of malware, the target, and the intentions of the attackers. Here are some common effects of malware:

- **Data Theft and Breaches:** Malware can be used to steal sensitive information such as personal identification, financial data, login credentials, and intellectual property. This stolen data can be used for identity theft, fraud, or sold on the dark web [23].
- **Financial Loss:** Malware can lead to financial losses through various means, including unauthorized access to online banking accounts, credit card fraud, and ransom demands [24].
- **System Disruption:** Malware can disrupt the normal functioning of computer systems, leading to crashes, freezes, and slowdowns. This can result in loss of productivity and downtime for individuals and businesses [25].
- **Data Destruction:** Some malware is designed to destroy data on infected systems, making it irrecoverable. This can be particularly damaging for businesses that rely on critical data for operations.
- **Privacy Violations:** Malware can compromise the privacy of individuals by monitoring their online activities, capturing keystrokes, recording webcams, and intercepting communications [26].
- **Botnet Formation:** Malware can create botnets, which are networks of infected devices that can be controlled by a malicious actor. Botnets are often used to launch coordinated attacks, such as Distributed Denial of Service (*DDoS*) attacks [27].
- **Propagation:** Malware can spread rapidly across networks and devices, infecting more systems and expanding its reach. This can lead to a wider outbreak and damage [28].
- **Reputation Damage:** If malware leads to data breaches or other cyber incidents, it can damage the reputation of individuals, businesses, and organizations. Customers and stakeholders may lose trust in the affected entities [29].
- **Legal and Regulatory Consequences:** Depending on the nature of the malware attack, legal actions and regulatory fines might be imposed on the responsible parties. Organizations that fail to adequately protect their systems may face legal liabilities [30].
- **Resource Consumption:** Some malware consumes system resources, such as processing power and bandwidth, to carry out its malicious activities. This can result in decreased system performance and increased operating costs [31].
- **Supply Chain Attacks:** Malware can be introduced into a software supply chain, infecting legitimate software updates or products. This can lead to the distribution of malware to a large number of users who trust the compromised source [32].

Importance of malware detection

Malware detection is crucial for several reasons, primarily focused on protecting computer systems, networks, and users from the harmful effects of malicious software. Here are some key reasons why malware detection is important:

- **Security:** Malware can compromise the security of systems, leading to data breaches, theft of sensitive information, unauthorized access, and other security threats. Detecting malware helps identify and neutralize these threats before they can cause significant damage [33].
- **Data Protection:** Malware can steal, corrupt, or destroy data. Detection helps prevent data loss and safeguards the integrity and confidentiality of valuable information, including personal, financial, and business-critical data [33].
- **System Integrity:** Malware can disrupt the normal operation of computer systems, causing crashes, slowdowns, and system instability. Detection ensures that systems remain stable, reliable, and available for users [34].
- **Financial Impact:** Malware can lead to financial losses through activities like fraud, unauthorized transactions, and ransom demands. Detecting malware helps mitigate these financial risks.
- **Privacy:** Malware, especially spyware and keyloggers, can invade users' privacy by monitoring their activities, capturing sensitive information, and even spying on them through webcams and microphones. Detection helps protect users' privacy [34].
- **Preventing Propagation:** Malware can spread rapidly, infecting multiple systems and devices within a network [29]. Effective detection helps halt the propagation of malware and prevents it from spreading to other parts of the infrastructure.
- **Mitigating Reputation Damage:** Malware incidents can damage the reputation of individuals, businesses, and organizations. Detecting and mitigating malware helps prevent these reputation-damaging events [35].
- **Regulatory Compliance:** Many industries and regions have data protection regulations that require organizations to take appropriate measures to secure their systems and data. Malware detection is a fundamental component of such compliance efforts [30].
- **Preventing Secondary Infections:** Malware can create vulnerabilities that other malware exploits. Detecting and removing initial infections can prevent subsequent attacks [36].

- **Protection against Evolving Threats:** The landscape of malware is constantly evolving, with new variants and techniques emerging regularly. Robust malware detection mechanisms, often powered by threat intelligence, help stay ahead of these evolving threats [37].

Types of malwares

There are various types of malware, each designed to exploit or compromise computer systems and networks in different ways. Here are some common types of malwares whose description is given as follows:

- **Virus**: A computer is a type of malicious software (malware) that is designed to replicate itself and spread from one computer virus to another. It can attach itself to legitimate programs or files and can execute malicious actions without the user's knowledge or consent. Computer viruses are a form of cyber threat and are considered harmful to both individual users and organizations [38,39].
- **Worms**: Computer worms are a type of malicious software (malware) that spread across computer networks and systems, often exploiting vulnerabilities to replicate themselves. Unlike viruses, worms do not need to attach themselves to existing files or programs to spread. They can independently execute and propagate, making them particularly effective at rapid proliferation. Worms are a significant cybersecurity threat due to their potential for causing widespread and quick damage [38,39].
- **Ransomware**: Ransomware is a specific type of malicious software that encrypts the victim's files or locks them out of their system, rendering their data inaccessible [39]. The attacker then demands a ransom payment, usually in cryptocurrency, in exchange for providing the decryption key or restoring access to the compromised data. Ransomware attacks have become increasingly prevalent and disruptive in recent years, affecting individuals, businesses, and even critical infrastructure [40].
- **Spyware**: Spyware is a type of malicious software that secretly gathers information from a user's computer or device without their knowledge or consent. It is designed to monitor and collect data about the user's activities, including their online behavior, personal information, browsing habits, and more. This collected information is then typically sent to a remote server controlled by the attacker or the entity behind the spyware [39].
- **Adware**: Adware, short for "advertising-supported software," is a type of software that displays unwanted advertisements on a user's device. Unlike malware that aims to steal data or damage systems, adware's primary purpose is to generate revenue for its creators by delivering ads to users' computers, often in an aggressive or intrusive manner. Adware can come in the form of browser extensions, software applications, or even pre-installed on certain devices [40].
- **Trojans**: A Trojan, also known as a Trojan horse, is a type of malicious software that disguises itself as legitimate software or files to deceive users into executing or opening them. Unlike viruses or worms, Trojans do not replicate themselves [39]. Instead, they often rely on social engineering tactics to trick users into willingly installing or executing them, leading to unauthorized access, data theft, or other malicious activities [40].
- **Key loggers**: A keylogger, also known as keyboard capture, is a type of malicious software designed to record and capture the keystrokes typed on a computer or other input devices like keyboards. The primary purpose of a keylogger is to secretly monitor and collect the information entered by a user, including passwords, usernames, credit card numbers, messages, and other sensitive data [41].
- **Fileless Malware**: Fileless malware, also known as non-malware or memory-based malware, is a type of malicious software that operates without writing files to the targeted system's disk. Unlike traditional malware that relies on executable files stored on the hard drive, fileless malware resides in the system's memory or uses existing legitimate processes and tools to carry out its malicious activities. This makes fileless malware particularly difficult to detect using traditional antivirus and security tools that focus on file-based threats [42].
- **Rootkits**: A rootkit is a type of malicious software that is designed to provide unauthorized and often hidden access to a computer system, while actively concealing its presence and activities from users and security software. Rootkits are particularly stealthy and intrusive, as they operate at a deep level within the operating system, allowing attackers to maintain control over compromised systems while evading detection [38,39].
- **WannaCry:** WannaCry" is a type of ransomware that gained widespread attention in May 2017 due to its rapid and highly disruptive global spread. It targeted computers running Microsoft Windows operating systems, exploiting a vulnerability in the Windows Server Message Block (*SMB*) protocol [43].

Malware evasion techniques

Nowadays, malware writers employing Obfuscation method that is intentionally designed to be difficult to detect and analyze [44]. It employs various obfuscation techniques to hide its true nature and evade traditional security measures. The primary goal of obfuscation malware is to bypass antivirus and other security software, making it challenging for security analysts to identify and mitigate the threat [45]. There are some common malware evasion techniques that are described as follows:

a) **Polymorphism:** It was generated in 1900 by Mark Washburn. This is the techniques to generate unlimited new variants of malware from existing class to make analysis and detection harden. The polymorphism code change itself at every infection to avoid detection. Polymorphic malware uses the code obfuscation techniques to make analysis undetectable by using packing, encryption, and junk code insertion [46,47].
b) **Metamorphism:** Metamorphic malware is similar to polymorphic malware, but it goes a step further by completely rewriting its code each time it infects a new host. This makes it extremely challenging to detect, as the malware's code appears entirely different with each infection [47,48].
c) **Data Obfuscation:** It is data masking or data anonymization, is the process of transforming data in such a way that it becomes more difficult to understand or identify, while still retaining its overall structure and usability for certain purposes. The primary goal of data obfuscation is to protect sensitive or private information, particularly in scenarios where sharing or analyzing the original data could pose risks to individuals' privacy, security.
d) **Packers and Crypters:** Malware authors use packers and crypters to compress and encrypt malicious code. This changes the file's signature, making it difficult for signature-based antivirus solutions to recognize the malware [49,20].

Process of malware investigation

Malware analysis involves the process of dissecting and understanding malicious software to determine its functionality, behavior, and potential impact. Malware analysis provides the clear view of malware in the code and meaning of the byte string used to steal the information or change the original code by the attacker. In analysis process, different key feature of code is extracted that provide the information and functionality of malware's activities into the system or network system [50]. There are three most commonly used techniques for malware analysis [41] as shown in Fig. 3.

a) **Static Analysis**: Static analysis of malware involves examining the malicious code or file without executing it. This analysis technique allows security researchers and analysts to understand the structure,

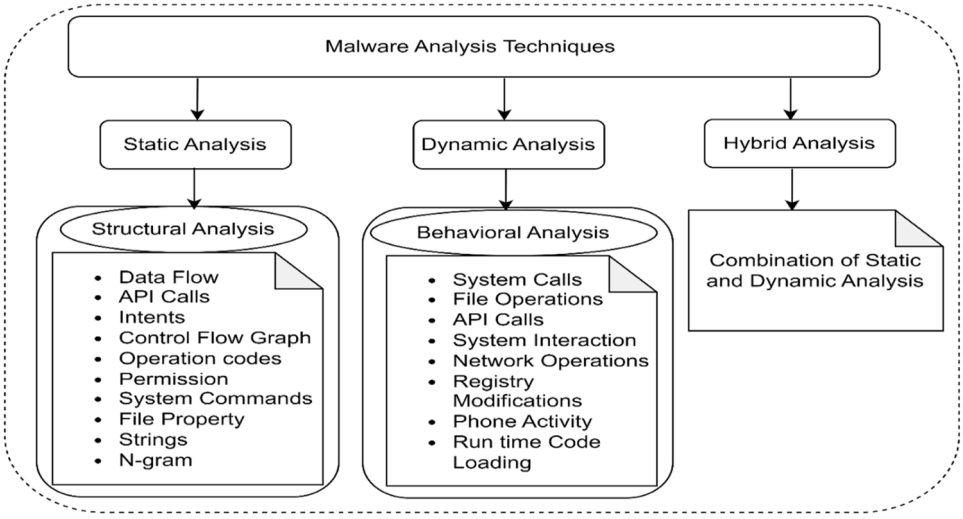

Fig. 3. Taxonomy of Malware Analysis Techniques.

behavior, and potential threats posed by the malware without running the risk of infecting a system. Static analysis can provide valuable information about the malware's behavior, capabilities, and potential impact on a system. Static analysis is particularly useful for quickly identifying known malware signatures and extracting important information from the code or files [48,49]. Functionality of malware code could be analyzed by inspecting internal code of malware. It provides the information about the malware's identity, libraries, *URL*, programming language [51] as shown in Fig. 4. The static analysis process is faster and provides the deeper knowledge about the execution path of malware. However, it has disadvantage of not detecting new variants of malware families or polymorphic malware that is specifically designed to evade static analysis [52].

b) **Dynamic analysis:** In this execute the malicious code in a controlled environment to observe its behavior and understand its capabilities. Unlike static analysis, which examines the code without running it [49]. In Dynamic analysis provides real time observation of how malware adapts and behaves in response to specific environmental conditions. By observing the malware in action, analysts can gain insights into its intended purpose, propagation methods, and potential damage it could cause [53]. However, conducting dynamic analysis carries some risk since the malware is actively running, and there is a possibility of unintended consequences. Therefore, it is essential to conduct dynamic analysis in a controlled and isolated environment to minimize the potential impact on the host system and the broader network [54]. The process of dynamic analysis is depicted in Fig. 5.

c) **Hybrid Malware Analysis:** Hybrid malware analysis, also known as combined analysis or integrated analysis, is an approach that combines multiple techniques, such as static analysis, dynamic analysis, and behaviour analysis, to gain a comprehensive understanding of malware. By leveraging the strengths of different analysis methods, hybrid analysis aims to overcome the limitations of individual techniques and provide a more robust and accurate assessment of the malware's behaviour, capabilities, and potential impact [55].

Literature review

In this review, overall literature of the malware detection is grouped into three categories such as review of feature selection (*FS*) techniques proposed for malware detection, review of *ML*-based techniques proposed for malware detection and review of *DL*-based techniques proposed for malware detection.

Review of feature selection techniques used in the design of malware detection system

This section present review of the literature on FS techniques used in the design of malware detection system. Cai et al. [56] have presented an information gain (*IG*) based android malware detection (*AMD*) framework to selects optimal features from extracted features of eight categories. This framework calculates the feature weight using Joint Optimization Weight Mapping (*JOWM*) function and three *ML* classifier, and utilized the differential evolution algorithm (*DEA*) to jointly optimize parameters of weight map function and classifiers.

Singh et al. [57] have presented a lightweight malware detection framework by using the Latent Semantic Indexing (*LSI*) approach to reduced dimensionality of dataset and improve the detection rate. This lightweight detection system is evaluated on *CICInvesAndMal2019* dataset using a Random Forest (*RF)* classifier that show 93.92 % accuracy.

Wang et al. [58] introduced the improved new self-variant genetic algorithm to selects the appropriate features for improving the *AMD*. The proposed method demonstrated the binary classification using the *ML* models such as *RF*, Logistic Regression (*LR*), K-nearest neighbour (*KNN*), Gaussian Naive Bayes (*GNB*). The best accuracy obtained by the *GNB* classifier is 95.5 % and 93.3 % by *KNN*.

Sahin et al. [59] have proposed the *ML*-based *AMD* framework, which is enabled with linear regression-based *FS* method to eliminates the irrelevant features and selects the most appropriate permission features that can increase the accuracy of the model. The best accuracy of the proposed model is achieved by *RF* that is 96.45 %. Alzubi et al. [60] proposed a hybrid *SVM* and metaheuristic based Harris Hawks feature optimization (*HHO*) algorithm to improve results of malware detection system (*MDS*) using *CICMalAnal2017* dataset. This model has achieved 93.12 % accuracy. The proposed model has the major limitation of high time and poor accuracy for new kinds of malware.

Bhat and Dutta [61] presented a multi-tier feature selection framework using the filter-based Information Gain (*IG*) technique for *AMD*. Three types of important features such as permission, *API* calls and intents were selected for static analysis. The selected malware features classified using the five *ML* models such as *RF, DT, LR, NB* and support vector machine *(SVM)*. The highest accuracy achieved by *RF* is 96.28 %. The proposed work is dedicated for binary classification and did not performed the experiment for multiclass malware which is a limitation.

Sharma and Agrawal [62] used the Binary Particle Swarm Optimization (*BPSO*) algorithm to optimize the android malware features. The optimal features obtained by this method are 72 features out of 215. The

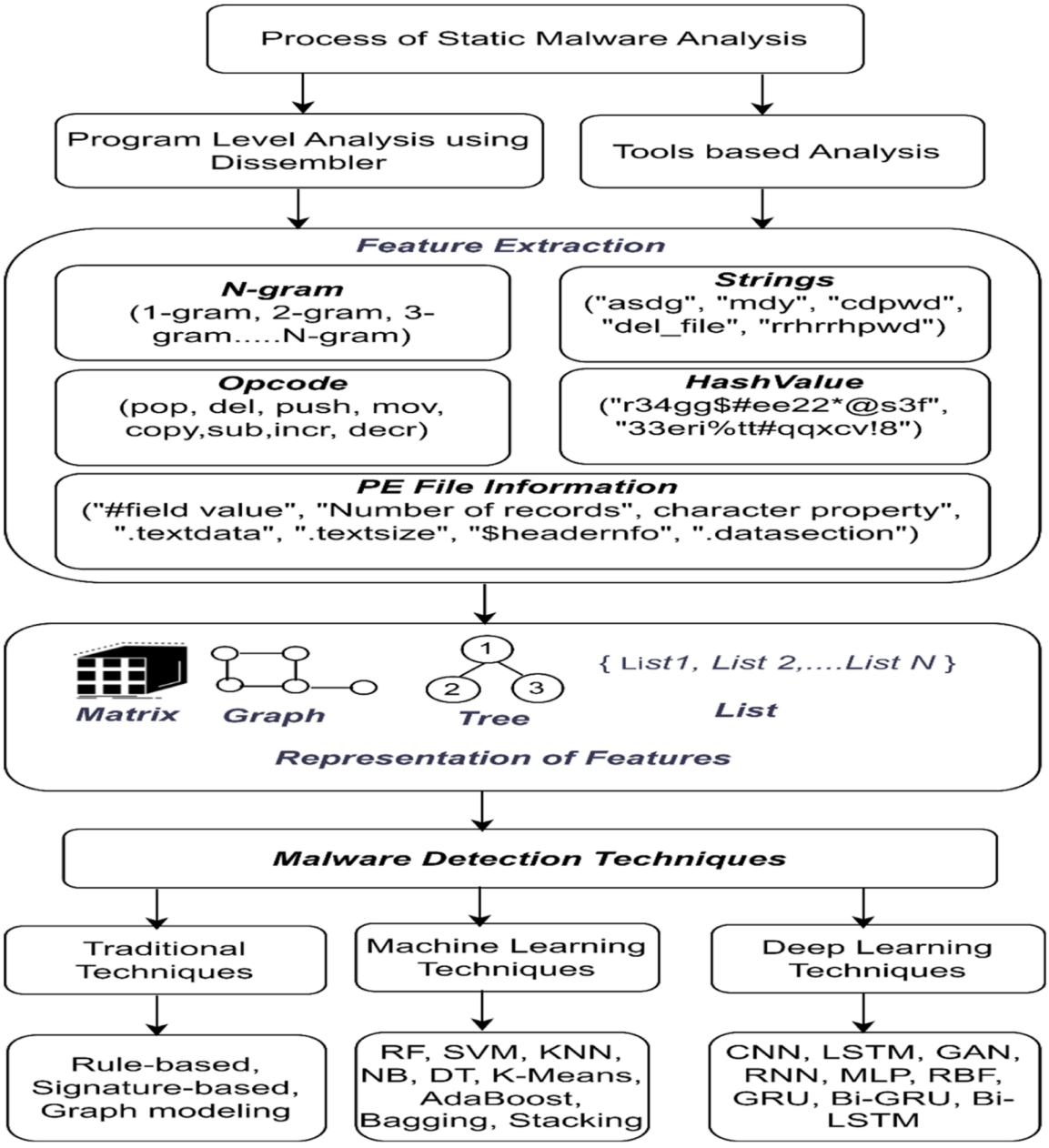

Fig. 4. Schematic diagram of static analysis for malware detection [9].

accuracy by the proposed Deep Neural Network (*DNN*) model is 94.92 % on the Drebin dataset.

Shatnawi et al. [63] introduced an effective *ML*-based *AMD* by using the Recursive Feature Elimination (*RFE*) feature reduction approach. The *LR* model was used for classification of selected feature into a malware and benign class. The malware classification was performed on two types of android features like *API* call and permission. The best performing classifier on this dataset is *SVM* that can detect the malware with 94.36 % accuracy with permission features and 83% accuracy with *API* calls. However, the presented model performed poor in terms of Recall and *F1*-score.

Alazzam et al. [64] proposed a wrapper method for android malware classification using the improved binary Owl optimizer and *RF* classifier. The accuracy of the method is 98.84 % on Drebin dataset using *ML* model. However, the *F1*-score of the model is very poor. Hossain et al. [65] proposed a particle swarm optimization (*PSO*) based feature optimization method to detect the android ransomware attack. Then, *RF* and *SVM* classifiers are applied for classification in which the *RF* achieved the better classification accuracy on malware datasets. However, the model is not evaluated on latest datasets that contains the various kinds of new malwares. The accuracy achieved in detecting the ransomware attack is 81.58 %.

Chemmakha et al. [66] proposed a *ML*-based malware detection model which applied the embedded methods (*RF* and *LightGBM*) to reduce the dimensionality and selects informative features. The proposed model provides the better results with *RF* and *XGBoost* that achieved more than 99 % accuracy. The proposed model did not mentioned the results of other evaluation parameter such precision, recall and *F1*-score for effective analysis of the model.

Grace and Sughasiny [67] have designed the android malware detection system using the Aquila optimizer to optimize the feature sets into optimal solutions. Then, used the hybrid *LSTM-SVM* model for analysing malware on the basis of permission features. It achieved 97 % accuracy on *CIC-AndMal-2017* dataset. However, the proposed

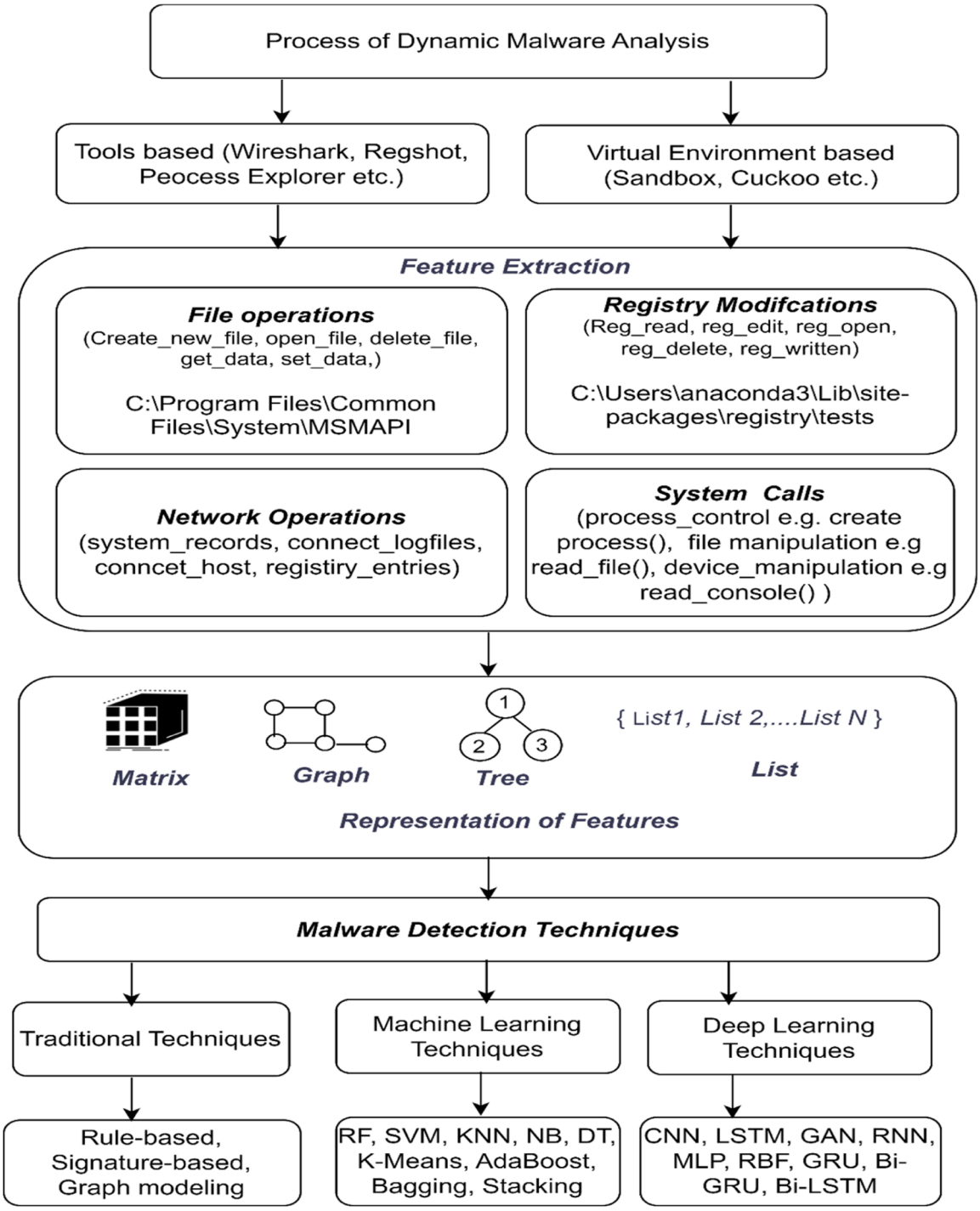

Fig. 5. Schematic diagram of dynamic analysis for malware detection [9].

techniques suffer from low recall and *F1*-score to assess the overall performance of the model. Sharma and Agrawal [68] presented an android malware detection system using modified Intelligent Water Drop Algorithm (*IWD*) as feature optimization to improve accuracy. Then, applied hybrid *DNN* for malware detection that achieved the 99.12 % accuracy on Malgenome dataset.

Soundrapandian and Subbiah [69] presented the a framework for malware detection using light weight *ML*. It used the evolutionary method for *FS* to learn multivariate behaviour of features and mahalanobis distance metric to classify as benign or malware. The malware detection accuracy of this model is 95.69 % on *CICMalDroid-2020* dataset. However, this framework is unable to detect the obfuscated malware on real time basis and required the multiclass malware detection.

Ghazi and Raghava [70] designed the wrapper-based metaheuristic Mayfly Algorithm (*MA*) for malware detection. The presented model used the *ML* classifiers such as *RF, SVM* and *KNN* to evaluate the model using *CIC-MalMem-2022* dataset. It obtained the best detection accuracy 99.99 %. However, authors consider this work for binary classification and not performed the experiment for multiclass malware detection.

Al-Andoli et al. [71] presented the hybrid feature optimization techniques of backpropagation (*BP*) and *PSO* algorithm for malware detection using optimal features. This work is evaluated on four types of malware dataset to assess the in-depth performance. The proposed

framework is efficient and scalable with the parallel computing setup to improve the detection accuracy. However, this work did not indicated effectiveness of the model for multiclass malware or unknown malware.

Abbasi et al. [72] proposed a new approach for ransomware detection which is based on the automatic feature selection by *PSO*. The important features are selected from various group of ransomware features data using *PSO* on the basis of feature importance. Then, optimal features are given to the five *ML* models for classification of ransomware attack. This work shows the binary and multiclass classification of ransomware variants on Resilient Information Systems Security (*RISS*) dataset. However, the accuracy of binary and multiclass is poor or not satisfactory to early works.

Islam et al. [73] designed a *ML* empowered android malware classification (*AMC*) framework to optimal feature selection based on weighted voting ensemble learning (*EL*). *RFE* and Principal Component Analysis (*PCA*) are employed for informative feature selection (*IFS*). This work experimented for multiclass malware detection along with dynamic features. The ensemble model consisted of *RF, DT, KNN, SVM* and Multilayer Perceptron *(MLP)*. The accuracy obtained by weighted voting model is 95 %.

Mahesh and Hemlatha [74] have introduced a hybrid Adaptive Red Fox Optimization (*ARFO*)-based *CNN* architecture for *AMD*. The proposed model was evaluated on collected malware and benign application that can classify the malware and benign application with 97.29 % accuracy. However, this work focused on binary classification of malware and not shown malware family classification.

Alomari et al. [75] proposed the Correlation-based *FS* (CFS) for malware detection using *LSTM* model. The proposed *FS* method achieved the reduction in dimensionality of features to obtain optimal features. However, the performance of proposed method is not effective on selected features.

Albakri et al. [76] designed a Metaheuristic FS (MFS)-based *AMD* model for binary classification. This work applied the Rock Hyrax Swarm Optimization (*RHSO*) algorithm to reduce the complexity of dataset, minimize the computational cost and improve the performance of the model. Although, detection accuracy is 99.05 % by *DL*-Attention Recurrent Auto-Encoder (*DL-ARAE*) model which indicated that model is better than previous works. The weakness of this work is that, authors did not performed the multiclass malware detection to detect the new malware.

Daniel et al. [77] introduced the malware detection framework for cyber physical system (*CPS*). This work applied the Snake optimizer to reduce the dimensionality and select optimal features for Graph convolutional network (*GCN*) model. The *GCN* is used for malware classification along with the Flower Pollination Algorithm (*FPA*) which tuned the optimal parameter. The proposed work obtained better results with 98 % accuracy for binary classification.

Mahindru and Sangal [78] have proposed a *LSSVM* (Least Square Support Vector Machine) based malware detection model. This work is dedicated to classifying applications as malware or benign. Authors applied the ranking and subset feature selection approaches and achieved the detection accuracy of 98.8 %. The obtained accuracy indicates that the model is robust and effective. However, the presented model is dedicated for binary classification.

Sahin et al. [79] presented the work dedicated to filter based feature selection for *ML* based *AMD*. The presented model is static analysis based model that used eight feature selection techniques to improve the detection results. However, in this work only permission based features considered for malware detection.

Another work presented by Sahin et al. [80] is permission based *AMD* by employing dimension reduction methods like *PCA* and Linear discriminant analysis (*LDA*). The permission features extracted from APK files and on extracted features dimension reduction methods was applied. The results obtained on reduced features set by *LDA* is better than *PCA* on permission features. The limitation of the proposed model is that it has considered only permission features for android malware detection.

Chimeleze et al. [81] presented the feature selection based *AMD* using *ML* and the detection rate is 99 % with employed feature selection like backward, forward and exhaustive subset selection method. Presented method consumes less memory and required minimum running time.

Wu et al. [82] presented a new FS technique for *AMD* using reinforcement learning (*RL*). In this work, presented a new wrapper based Double Deep Q Network (*DDQN*) feature selection method that reduces the computational time of the model and used word embedding technique to increase the feature importance. The obtained accuracy is 95.6 % by *RF* on 24 optimal static features that indicates the effectiveness of proposed framework.

Review of ML-based malware detection system

ML is a subset of Artificial Intelligence (*AI*) technology used for malware detection and threat prediction. *ML* algorithms works on basis of input data to perform the task of detection, classification and pattern matching by deep analysis of data. A detail literature review of related works on the malware detection using *ML* techniques are discussed in this section.

Garg and Baliyan [83] presented a parallel classifier-based zero-day malware detection using *ML*. It is based on supervise learning. The parallel *ML* classifiers like Pruning Rule-based Classification Tree (*PART*), Ripple Down Rule Learner (*RIDOR*), *SVM* and *MLP* were used on 10-fold cross validation to improve the malware detection accuracy. The proposed work has a good accuracy of 98.27 % with ensemble of parallel classifiers, which demonstrates the robust performance of the applied methods.

Wang et al. [84] presented the Mobile Malware Detection (*MMD*) framework for network traffic features analysis. The proposed framework collected the network traffic data from mobile apps and extracts the hypertext transfer protocol (*HTTP)* request and transmission control protocol *(TCP)* flow features for learning the malicious behaviours. The categorization of the network traffic features into malware or benign is done by *C4.5* classifier that obtained the 97.89 % detection rate.

Bahtiyar et al. [85] presented an approach of multidimensional *ML* for advance malware detection like Stuxnet. The detection process is carried out to find the correlation between conventional malware and advances malware using five features of windows *API* calls. The regression models were applied to predict the advanced malware on defined features. The proposed model was not evaluated using standard metrics such as accuracy, precision and recall of any *AI* models.

Xiaofeng et al. [86] proposed *MDS* that combines the *API* sequences and statistics features. In this study, *ArguMent-Hashing*-based *API* correlation fast Analysis algorithm was used. This study has used hybrid of *RF* and *BiLSTM* models for classification that achieved 96.7 % accuracy. This work did not performed the multiclass classification and evaluated with small amount of dataset.

Karbab and Debbabi [87] proposed a *MalDy* framework based on supervise learning for portable malware detection. This model generates the behaviour reports of features using Bags of Words (*BoW*) which is a natural language processing (*NLP*) technique. Then, these features are used by ensemble of *ML* models for threat classification as malware or benign. The *F1*-score of the proposed model is 94.86 %.

Han et al. [88] presented the profiling-based malware detection framework named as *MalInsight*. The profiling of malware consisting in three categories: basic level, low level and high level behaviour. The *MalInsight* can detect the obfuscated malware with accuracy of 99.7 %. In addition, the accuracy of family classification is 94.2 % that shows effectiveness of *MalInsight*.

Roy et al. [89] presented the *AMD* model using *ML* approach to sort out the temporal bias found in previous work. This work extracts the vulnerable features of the application, then applied the aggregation method to count the occurrence of the features. After that, to find and

reduce the optimal features, a *ML*-based techniques called Non-negative Matrix Factorization (*NMF*) was applied. The prediction of malware applications was done by *SVM* classifier using reduced feature that achieved the 88.72 % accuracy and without feature reduction *SVM* attained the 93.35 % accuracy. However, this work lacks class-wise classification of malware family to detect the new obfuscated malware.

Gupta and Rani [90] have addressed the issues of malware detection in big data environment. They have proposed a weighted voting to calculate feature weight and stacking approaches for ranking of features using ensemble learning. The Cuckoo Sandbox tool was used to conduct the static and dynamic malware analysis. For scalability, Apache spark framework used to process huge amounts of malicious big data. The outcome of the proposed approach is 99.5 % accuracy by weighted voting method.

Amer and Zelinka [91] presented a embedding technique-based malware detection to understand the contextual relationship between *API* call sequences, along with this, it also incorporates segregating method to make the cluster of similar contextual relationship of *API* calls. Then, these distinct features are classified using Markov chain model. The proposed model can detect and predict malware or goodware with an average accuracy of 99.7 %. However, this work was evaluated on single dataset and performed binary classification.

Surendran et al. [92] introduced a Tree Augmented Naive Bayes (*TAN*)-based malware detection model. This model addresses the issues of multicollinearity problems that can degrade the performance of the classifiers. This work dedicated for malware detection by hybridizing the static and dynamic feature for efficient detection. Then, three ridge regularized logistic regression classifiers applied to classify threat as malware or benign application. The proposed model detects the malware with accuracy of 97 %.

In [93], the authors have presented a static analysis feature based *MDS* to improve *TPR* and minimize *FPR*. This framework applied the improved filter-based feature selection techniques such as *KNN*-based Relief (*KNN-R*) and Chi-square. To enhance the detection rate, this work upgraded the kernel for *SVM* termed as optimized *SVM*.

Singh and Singh [94] presented a behaviour-based malware detection model using cuckoo sandbox tool to conduct dynamic analysis. Different features were extracted by this tool such as Printable String Information (*PSI*), *API* calls, registry files, file system and network operation. Three types of operation was performed on the printable strings features, such as generation of high dimensional matrix using text mining, used Singular Value Decomposition (*SVD*) technique to decrease dimension and calculated Shannon entropy. At last, all these fine-tuned features were incorporated for training and testing to the *ML* classifiers to detect malware. These proposed work applied large number of classifiers. The highest accuracy of 99.54 % obtained with *ADA* ensemble learning model.

Surendran et al. [95] presented an *AMD* system using the mechanism of graph signal for compact feature representation, which is a low dimensional feature representation and extraction method. This work achieved 99 % accuracy with *RF* classifier. However, the proposed model is unable to detect the emulator-based malware, this makes model ineffective for obfuscated malware detection.

Shhadat et al. [96] presented the performance analysis of *ML* classifiers to unknown malware detection. For feature selection, *RF* was utilized on benchmark malware dataset. The highest accuracy 98.2 % is obtained by *DT* for binary class and accuracy of 95.8 % by *RF* for multiclass classification.

D'Angelo et al. [97] presented an association rule-based malware detection framework. The proposed framework performed run time malware analysis using cuckoo sandbox tool on the basis of *API* call sequences. To trace the behaviour of *API* calls recurring sub sequences alignment-based algorithm was used in association rule for malware classification. The proposed work obtained 99.03 % accuracy for malware detection.

Sun et al. [32] presented a classified behaviour graph (*CBG*) based malware detection framework for *IIoT*. The *API* features was extracted using cuckoo sandbox, normalize features and applied *N*-gram algorithm for finding different length of *API* call sequence. After that, utilized the graph optimization on the *CBGs* for mapping behaviour by using two approaches Common Subgraph Matching and Common String Matching to detect malware. This study has obtained 99.9 % accuracy.

Usman et al. [98] presented an intelligent malware detection system for analysis of internet protocol address for forensic data analytics. The proposed model follows hybrid approach based on dynamic analysis using cuckoo sandbox. The *DT* is used for detection purpose that achieved 93.5 % accuracy.

Panker and Nissim [99] proposed a malware detection model from volatile memory dumps for Linux Cloud environments. The memory dumps were collected using virtual machine (*VM*), then from collected memory data, 171 critical features were extracted using *ML*. These features were classified with proposed trusted framework in malware or benign. This has major limitation of experiment, it can be performed when *VM* is in frozen state which may cause in delay of client service.

Syrris and Geneiatakis [100] have proposed a *ML*-based malware detection framework using static data. The presented work assesses the performance of the *ML* classifiers like *NB, RF, RidgeReg, LassReg,* artificial neural network *(ANN)* and *SVM*. Best accuracy among all these classifiers was 99 % by *SVM*. However, the proposed model was trained on high dimensional dataset that makes model more complex in regards to training time. Moreover, in this work, partial dataset was utilized that may impact the detection of new variants of malware.

Sihag et al. [5] designed an opcode segment based *BLADE* system to identify obfuscation method used in evasion of malware. In this work, authors have applied the semantic approach to simply the dalvik opcodes features. The proposed work outperform on three benchmark datasets, which indicates that BLADE performed better against obfuscated techniques and achieved the 92.47 % accuracy. However, this work uses huge amount of malware dataset and we observed that BLADE is less effective against class encryption techniques.

Sasidharan and Thomas [101] proposed a behaviour based malware detection system named *ProDroid* to identify the malicious *API* call features and their classes. The suspicious *API* calls are identified in twenty different categories. Then, created the Multiple Sequence Alignment (*MSA*) of application in the family. Furthermore, the *MSA* is used to produces the Profile Hidden Markov Model (*PHMM*) with utilization of *ML* techniques. The *PHMM* model classify the application as benign and malware on the basis of generated feature score. The *ProDroid* achieved the 94.5 % accuracy.

Imtiaz et al. [40] proposed a *DeepAMD* model for multiclass and binary classification that classify the different categories of malware attacks using both static and dynamic layer data. The author has applied the *DeepAMD* on the *CICInvesAndMal2019* dataset. The proposed work used both *ML* and *DL* model for malware classification. On static layer the obtained accuracy of 93.4 % for binary class and on dynamic layer the accuracy around 80 % for malware category classification, 59 % accuracy for family classification.

Wu et al. [102] presented a hybrid *BiLSTM-GNN* model for *AMD* named as *DeepCatra*. The presented model is based on call traces by generating critical *API* call features. It incorporate the graph modelling concepts of text mining to find the critical features. The proposed work detects the malicious application with 95.94 % accuracy. However, the recall value of the model is poor which shows the lower detection rate against obfuscated techniques.

Mat et al. [103] introduced a *AMD* system using Bayesian probability method. The proposed system was tested using Androzoo and Drebin datasets. To selects optimal feature subset, *IG* and chi-square were applied. The obtained accuracy on reduces features is 91.1 %. The overall accuracy of the proposed model is lower that can be improved using latest metaheuristic optimization techniques.

Alani and Awad [104] designed a adware malware detection framework named as *AdStop* using *ML*. The motive of the framework was

to improve accuracy and reduce time overhead that could handle by reduction of dimensionality of features. *RFE* was used to optimize feature and remove meaningless features. The optimal features obtained by *RFE* is 13 out of 79 features. Then, this selected features used for malware detection with different ML classifiers. The highest accuracy of 98.02 % produced by *RF*. However, obtained results is better, it takes more training time and used partial dataset.

Urmila [105] proposed a behaviour-based malware detection system using *ML* techniques. This framework applied the Ensemble EfficientNet and Xeception with ResNet (*EEXR*), EfficientNet and LightGBM for malware detection. The highest accuracy obtain by the *EEXR* classifier is 96.75 %.

Gracia et al. [106] addressed the issues of concept drift problems in *ML* models used for malware detection, which impacted in the new threat detection. This issue was overcome with Transfer Learning (*TL*) techniques for malware detection with hyper parameter setting. The *TL* was provided with unbalance dataset to detect new variants with good detection rate. The cuckoo sandbox tool was used for dynamic analysis that extracted the total 1135 features of such APIs, signature and networks features. The classification models was *KNN, MLP, RF* and Extreme Gradient Boosting (*XGB*). The performance of the model measured in terms of average *MCC* which is 97.75 % by *XGB*.

Ahmed et al. [107] designed an image malware detection model using *ML* and *DL* models. For data normalization, a min-max and scaling method is used. The models used for malware detection are *ANN, LR, CNN, LSTM* and I*nceptionV3*. The highest accuracy is 98.76 % obtained by *InceptionV3* on Microsoft BIG 2015 dataset.

Kamboj et al. [24] presented a malware detection in the downloaded files through different types of feature analysis. Exploratory data analysis was performed on the dataset to remove unwanted features. The proposed model is based on supervise learning to detect multi-class malware. *RF* achieved 99.9 % accuracy.

Naeem et al. [108] designed a volatile memory forensics based malware detection system using deep stacked ensemble that fuse the weak learner of *CNN* and meta learners models. It consists of three modules as memory sample collection, feature extraction and fusion, and deep stacked ensemble module for malware detection. The designed system is platform independent that detect the run time behaviour of active process to identify obfuscated malware. The hybrid deep model extracts local and global features to reduce the dimension for the models. The executables files of windows and android systems were converted to grayscale and RGB images. The accuracy of the proposed deep stacked ensemble model is 99.9 %. However, this model unable to identify the advance android based polymorphic malware with applied models.

Tsafrir et al. [109] proposed a unknown *MP4* malware detection system enabled with *ML* techniques. This work has proposed two-way efficient feature extraction method such as file structure based and knowledge-based feature extraction. Then, these extracted features were passed through the conventional features selection methods like *IG, Chi-square, Fisher-score* and *LR* to gain best features. The results shows the average *AUC* is 0.9951 %.

Ceschin et al. [36] evaluated the impact of concept drift on the malware classifiers using two dataset set such as *AndroZoo* and *Drebin*. *Word2vec* and *TF-IDF* algorithm applied to extracts the features. This work assessed both feature extractors and four concept drift detector using Adaptive Random Forest (*ARF*) and Stochastic Gradient Descent (*SGD*) classifiers. Then, to mitigate concept drift and find best method for real environment conducted the experiments. The best method was observed in this work is SGD that achieved accuracy of 99 % on Drebin dataset. However, the detection rate is good on Androzoo dataset that need to be improved.

Rustam et al. [110] presented an image based malware detection system using Transfer Learning and *ML* algorithms. A hybrid of the *VVG-16* and *ResNet-50* models are utilized to extract hybrid attributes sets from the input sample data. The proposed model termed as bi-model that added sequentially using stacking approach to increase accuracy. This work obtained the 100 % accuracy of 25 classes of malware on malimg dataset. However, this model is computation time is high and evaluated using single dataset and unable to identify real time malware identification.

Dabas et al. [111] designed the *ML* based windows malware detection using hybrid feature selection techniques. The presented model considered only three types of *API* calls (usage, frequency and sequence) features. This model is evaluated on individual *API* call set and integrated *API* calls feature set. The obtained results by integrated feature set achieved around 99 % accuracy which is better than individual feature set. However, the proposed model employed the traditional feature reduction techniques on limited API calls features that reduces generalization capability and scalability of the model.

Sahin et al. [112] proposed multiple linear regression based *AMD* model on permission features. To boost the performance of the proposed model, bagging (majority voting) ensemble learning technique employed. The presented work used the four different android malware dataset samples. However, very limited sample of malware and benign files have been includes in this dataset that may reduce the robustness and scalability of new malware detection.

AlOmari et al. [113] shown a comparative analysis of *ML* algorithms for *AMD* using dynamic features. The performance of the different models have been evaluated on *CICMalDroid2020* dataset. The best performing model is Light Gradient Boosting (*LGB*) model that achieve 92.72 % accuracy on reduced feature set.

Zhu et al. [115] designed a ensemble learning based multi model to improve malware detection using hybrid feature extraction on imbalanced dataset. In this framework *SVM* used as a base classifier and majority voting approach is used for final prediction. Furthermore, in order to get good feature characteristics this framework adopted feature fusion method to enhance accuracy of ensemble framework.

Seraj et al. [116] designed a *MVDroid* framework to identify the malware based virtual private networks(*VPN*) using the optimized deep learning. This work the used permission based malicious android *VPN* dataset for performance testing of the proposed framework. The average accuracy of proposed model is 92.81 %. However, the performance of the proposed *MVDroid* can be improved by using feature selection techniques and other advanced *DL* models.

Review of DL-based malware detection system

DL has its capability to automatically extract features. This capability has increased the popularity of using *DL*-based models in the malware detection [22]. Different *DL*-based models are employed in malware detection problem in the literature [117]. This section briefly described literature review of the *DL*-based malware detection system.

Venkatraman et al. [118] introduced a *DL* and *ML*-based hybrid malware detection model using image visualization to detect new malware variants. The presented hybrid model consists of the *CNN-BiLSTM* and *CNN-BiGRU* with hyper parameter tuning to extracts spatial and sequential features for improving the detection rate. This model has obtained 99 % accuracy on the malimg dataset. However, the lower detection accuracy of family class on *MMB-15* dataset is observed. This indicates that the proposed model is not efficient in detection of all kinds of malware variants.

Zhong and Gu [119] designed a Multi-Level Deep Learning System (*MLDLS*) that integrated several *DL* models by applying tree structure for malware detection. *MLDLS* functions in five steps such as extraction of static and dynamic features, partitioned a dataset into multiple clusters using an improved K-means technique, created parallel subtree cluster, and constructed the *DL*-model for each clustered dataset, and finally fusion of decision value of *DL*-based models in tree form for malware detection. Kang et al. [120] presented a static analysis based malware detection framework using *LSTM* model. The proposed model analyses opcode and *API* calls features for malware detection. The word2vec

techniques was applied for reduction of feature dimensionality. The proposed method observed an accuracy of 97.59 %.

D'Angelo et al. [121] presented an *API* sequences-based *AMD* in the mobile environment. The proposed system provides the systematic solution summary of *API* calls execution history. Then, auto encoder was applied to extracts useful features from the solution vector of features. The *ANN* model used for malware detection on reduced features sets. This study has obtained 95 % accuracy. Gao et al. [122] proposed cloud-based malware detection using semi-supervised transfer learning (*SSTL*) for different feature extraction and designed the *asm* classifier to improve accuracy. This study has achieved 96.9 % accuracy.

Shaukat et al. [123] presented a hybrid of *CNN* and *SVM* based malware detection using colour image. It has three phase such as *PE* files that were converted to images. Extraction of deep features from images using fine-tuned *DL* model and then, identification of malware with *SVM*. Comprehensive evaluation of the proposed work was done using 15 *DL* and 12 *ML* models on malimg dataset. The accuracy of the proposed hybrid *CNN-SVM* was 99.06 %. However, this work has taken the malware samples of windows *PE* files which may be not effective on the cross-platform of android devices and IoT malware dataset.

Zhu et al. [124] presented an end-to-end image-based *AMD* using *CNN* model. This work has proposed a novel pre-processing method for *Dex* files to make model less expensive, efficient and improve the accuracy. The proposed model applied *CNN* variants diverse receptive fields using max pooling and average pooling named as *MADRF-CNN*. It was evaluated with *virushare* and google play dataset of malware and benign samples. The obtained accuracy is 96.9 % of the proposed model. However, this work did not mention class-wise malware detection to detect the new variants of malware.

Fasci et al. [125] presented a *GAN*-based malware detection model for identifying the variants of malware. Then, detection the malware was performed using visualization pattern of the images generated. The proposed model attained the 100 % detection rate which indicates the high detection and robustness. The limitation of the proposed work is that it takes more training time and has less training stability.

Alzaylaee et al. [126] proposed dynamic features based android malware detection model. In this work total 426 features of permission and events are extracted for experiments using dynalog automatic feature extraction framework. The *DL-Droid* framework was evaluated on11505 malware samples and 19,620 benign samples. The performance of this model using dynamic features is 97.8 % detection rate and on combination of both static and dynamic features the detection rate is 99.6 %. However, the scalability of model is limited and it is evaluated for binary classification that reduces the generalization capability of multiclass malware.

Sahin et al. [127] presented a comparative analysis of *DL* models (*DNN, 1D-CNN* and *2D-CNN*) for android malware detection. In this static features is being converted to images for malware classification. The proposed model was tested on two android datasets such malgenome-215 and drebin-215. The obtained accuracy is 99.47 % and 97.83 % respectively.

Waqar et al. [128] also presents a study of *IoT* based *AMD* using hybrid *DL* models (*GRU-BiLSTM*). Although, the obtained detection accuracy of the presented model is better than other techniques, but it suffers from higher computation time, complexity, lower scalability and limited to binary classification.

Shortcomings and research gaps

In literature, there has been numerous malware detection approaches are available using *ML* and *DL*-based models. Tables 1-3 focuses on the key facts of the various study mentioned in the literature. The main research gaps that are observed in the review of the literature related to malware detection and analysis, are mentioned below:

- The literature show that most of the existing feature selection approaches used for malware detection and analysis, suffer from low detection accuracy and higher false alarm rate. Thus, design of an optimal feature selection scheme that may improve the detection rate, classification accuracy of new variants of the malware and reduction of unnecessary features from huge amount of dataset is still an open research issue and critical challenge.
- Feature selection methods often rely on static features extracted from malware samples. However, dynamic and polymorphic malware can change their behaviour and characteristics over time. Thus, design of

Table 1
Summary of different Feature Selection and Extraction techniques.

Authors	Dataset	Feature Selection Method	Detection Technique	Accuracy (%)	Recall (%)
Cai et al. [56]	Drebin, AMD, Google play store	IG, JOWM, DEA	LR	96.67	96.15
Singh et al. [57]	CICInvesAndMal2019	LSI	RF	93.92	88.64
Wang et al. [58]	UCI repository, AMD	Asexual GA,	GNB	95.5	95.8
Sahin et al. [59]	Android malware samples, APKpure	Linear regression	MLP	×	×
Alzubi et al. [60]	CICMalAnal2017	HHO	SVM	93.12	93.12
Bhat and dutta [61]	Drebin,VirusTotal and VirusShare	Information Gain	RF	96.28	97.92
Sharma and Agrawal [62]	Drebin dataset	BPSO algorithm	DNN	94.92	96.35
Shatnawi et al. [63]	CICInvesAndMal2019	RFE	SVM	94.36	82.6
Alazzam et al. [64]	Debin dataset	Improved binary Owl optimizer	RF-OWL	98.84	99.56
Hossain et al. [65]	CICAndMal2017	PSO	SVM and RF	81.58	68.98
Chemmakha et al. [66]	Windows Portable Executable (PE) files	Embedded method	RF and XGBoost	99.47	×
Grace and sughasiny [67]	CIC-AndMal-2017	Aquila optimizer	Hybrid LSTM-SVM	97	90
Sharma and Agrawal [68]	Drebin, Msghic, malgenome	Modified IWD	DNN	99.12	96.68
Soundrapandian and Subbiah [69]	CICMalDroid 2020	Evolutionary feature selection	Mahalanobis Algorithm	95	95
Ghazi and Raghava [70]	CIC-MalMem-2022	Wrapper-based MA	RF	99.99	99.99
Al-Andoli et al. [71]	VirusShare	Hybrid BP-PSO	Parallel DL	97.7 %,	98.7
Abbasi et al. [72]	RISS	Wrapper based PSO	Regularized LR	97.33	×
Islam et al. [73]	CCCS-CIC-AndMal-2020	Wrapper based RFE and PCA	Ensemble ML	95	×
Mahesh and Hemlatha [74]	Android malware dataset	ARFO	CNN-ARFO	97.29	93.21
Alomari et al. [75]	CICAndMal2019	Filter based CFS	DL-LSTM	94.59	×
Albakri et al. [76]	Andro-AutoPsy	RHSO	DL-ARAE	99.05	99.05
Daniel et al. [77]	CICIDS-2017, NSL-KDD-2015	Snake optimizer, FPA	GCN	98.28	98.53
Mahindru and Sangal [78]	VirusTotal, Drebin	LSSVM	ML, RBF	98.8	98.75
Sahin et al. [79]	VirusShare (APK files)	Filter method	ML	×	×
Sahin et al. [80]	Malgenome-215 and VirusShare	PCA and LDA	ML	×	×
Chimeleze et al. [81]	Mendeley repository	Embedded BFE algorithm	ML	99	×
Wu et al. [82]	AndroZoo and Drebin	RL(DDQN)	RF	95.6	×

Table 2
Summary of *ML*-based techniques applied in malware detection.

Authors	Dataset	Feature Selection	Detection Techniques	Accuracy (%)	Recall (%)
Garg and Baliyan [83]	Androzoo, AMD	×	Ensemble of MLP, SVM, PART	98.27	98.79
Wang et al. [84]	Drebin,	×	C4.5	97.89	×
Bahtiyar et al. [85]	CSDMC 2010	×	Multi-dimensional ML	×	×
Xiaofeng et al. [86]	API call sequences	×	Hybrid of RF and BiLSTM	96.7	×
Karbab and Debbabi [87]	Drebin, AndroZoo	×	Ensemble of ML classifiers	×	×
Han et al. [88]	VirusShare	×	RF	99.76	99.30
Roy et al. [89]	Drebin, CICAndMal2019	NMF	SVM	88.72	81.94
Gupta and Rani [90]	Windows files, VirusShare, VX heaven	Stacking (RF, DT, SVM) approach for feature ranking	Ensemble learning(weighted voting)	99.5	99.6
Amer and Zelinka [91]	CSDMC 2010	×	Markov chain model	99.0	99.0
Surendran et al. [92]	Drebin	×	TAN	99	1
D and P [93]	AAGM, AMD	KNN based Relief algorithm	Optimized SVM	×	76
Singh and Singh [94]	VirusShare, VirusTotal	SVD	Ensemble ML	99.54	99.82
Surendran et al. [95]	Drebin, AMD	Graph signal for low dimensional feature	RF	99	98
Shhadat et al. [96]	VirusShare	RF	DT	97.8	87.3
D'Angelo et al. [97]	Malware and Benign files	×	ML	99.03	96.30
Sun et al. [32]	Malware and Benign files	Graph optimization	AdaBoost	99.9	×
Usman et al. [98]	Network traffic files	×	DT	93.5	98
Panker and Nissim [99]	Volatile Memory	IG and fisher score	KNN	98.9	97.6
Syrris and Geneiatakis [100]	Drebin	Chi-square	SVM	99	95.9
Sihag et al. [5]	AndroAutopsy, AndroTracker,	IG	RF	98.18	98.2
Sasidharan and Thomas [101]	Drebin and genome	×	ML	94.5	×
Imtiaz et al. [40]	CIC-InvesAndMal-2019	×	DeepAMD	93.4	93.4
Wu et al. [102]	Google play store and AndroZoo	×	BiLSTM and GNN	95.94	92.58
Mat et al. [103]	AndroZoo and Drebin	IG, chi-square	Bayseain classification	91.1	91.1
Alani and Awad [104]	CIC-AAGM2017	RFE	RF	98.02	98.02
Urmila [105]	APK files	×	EEXR	96.75	97.15
Gracia et al. [106]	VirusShare	×	KNN	97.80	×
Ahmed et al. [107]	Microsoft BIG 2015	×	InceptionV3	98.76	94.8
Kamboj et al. [24]	VirusShare	×	RF	99.99	×
Naeem et al. [108]	AndroZoo, CICMalMem-2022	×	Hybrid deep stacked ensemble of CNN and MLP	99.8	99.0
Tsafrir et al. [109]	Malware(1163), Benign(5066), VirusTotal	IG, Fisher score Chi-square, Lasso Regression	Transfer learning using CNN	N-gram, MinHash	97.6
Ceschin et al. [36]	Drebin	TF-IDF	Adaptive RF	98.71	74.17
Rustam et al. [110]	Malimg	×	Bi-RF	1	1
Kumar et al. [114]	Malimg	Bag of Visual Words (BoVW)	SVM	98.23	98.23
Dabas et al. [111]	VirusShare	PCA and RFE	ML	99.6	98.84
Sahin et al. [112]	AMD	×	EL(Bagging)	96.9	×
AlOmari et al. [113]	CICMalDroid2020	PCA	Light Gradient Boosting (LGB) model	92.72	91.41
Zhu et al. [115]	Drebin-215	×	EL (majority voting based SVM)	95.14	97.06
Seraj et al. [116]	Malware VPN	×	DL(CNN)	92.81	93.91

Table 3
Summary of *DL*-based techniques applied in malware detection.

Authors	Dataset	Feature Selection	Detection Techniques	Accuracy (%)	Recall (%)
Venkatraman et al. [118]	Malimg	×	CNN-BiGRU	96.3	91.5
Zhong and Gu [119]	VirusShare, VX heaven, VirusSign	IG	MLDLS	×	92
Kang et al. [120]	MMC BIG-15	Word2vec	LSTM	97.59	×
D'Angelo et al. [121]	Malgenome, VirusShare,	Sparse Auto Encoder(SAE)	MLP	96	96
Gao et al. [122]	MMB-15, VirusShare	×	SSTL(RNN)	96.90 %	96.90
Shaukat et al. [123]	Malimg	×	Hybrid CNN-SVM	99.06	98.52
Zhu et al. [124]	VirusShare and google play store	×	MADRF-CNN	96.9 %	98.9
Fasci et al. [125]	Malimg	×	GAN, CNN	100	×
Alzaylaee et al. [126]	Virusshare	IG	DL	95.21	97.76
Sahin et al. [127]	Malgenome-215 and Drebin-215	×	DL(DNN, CNN)	99.47 and 97.83	×
Waqar et al. [128]	CICAndMal2017	×	Hybrid DNN-GRU-BiLSTM	99.87	99.9

feature selection scheme for malware data analysis to identify the relevant and stable features is a challenging research issue.

- Malware datasets are often highly imbalanced, meaning that the number of samples from different malware families is not evenly distributed. This can lead to biased models and impact the detection performance on underrepresented malware types. Most of the existing malware detection schemes have not considered data imbalance issue. Thus, addressing the issue of class imbalance is crucial to ensure accurate detection across all malware categories.
- Many feature selection methods struggle with high-dimensional data, which is common in the context of malware detection due to the large number of features used to represent executable files. As the number of features increases, traditional feature selection algorithms might become computationally expensive and less effective. Thus,

design of an effective and light weight feature selection scheme would be a research gap.

- The malware attacker used advanced techniques to hide malicious code by the detection system. Therefore, it is difficult to design a detection model that can efficiently identify the different attributes of malicious codes is still an ongoing research challenge.
- Malware can target various operating systems and devices, and *ML* models trained on one platform may not generalize well to others. Research is needed to develop cross-platform malware detection techniques that can effectively protect different environments.
- Zero-day malware refers to previously unknown and unseen malware strains that have no known signatures or patterns. Detecting zero-day malware is a significant challenge since conventional *ML* models rely on historical data and known patterns. Developing techniques that can detect previously unseen malware without labelled training data is a critical research gap.
- As the volume of malware samples continues to grow exponentially, it becomes essential to develop *ML* models that can scale efficiently and handle large-scale datasets without compromising detection accuracy.
- Most of the existing solutions for designing an intelligent malware detection system are ineffective for detecting new variants, obfuscated, polymorphic and advanced real time malware detection. Thus, designing of an efficient *MDS* with high generalization capability is a challenging research issue.
- Most of the existing *DL* models trained on specific malware families may struggle to generalize to new or unseen malware families. These models are susceptible to overfitting, meaning they might perform well on the training data but fail to generalize to new and previously unseen malware variants. Thus, there is a necessity to design an accurate and effective *DL*-based model which can be able to detect unknown malwares.

Machine learning for malware detection

ML is a subset of *AI* used in detection and classification task in multiple areas. *ML* algorithms, such as *RF, NB, KNN, SVM,* and *DT* have been widely utilized for malware detection and classification.

a) **Naive Bayes (*NB*):** *NB* is supervised learning based classification algorithms. It works on the basis of the probabilities function in which each attributes belongs to particular class. It takes a strong assumption that the attributes are conditionally independent. The assumption used in the Naive Bayes algorithm makes computing probability easier. It calculate whether a data point may fall into a particular class or not. It is able to predict class accurately of the test dataset for binary and multiclass dataset. It is efficient and highly scalable that can perform well for irrelevant and small size of dataset. It employed the Bayes theorem for decomposition of a conditional probability. However, *NB* is gives poor performance with highly correlated attributes to each other and considered the attributes must be independent [129].

b) **K-Nearest Neighbours (*KNN*):** *KNN* is non-parametric supervised learning algorithm means which means it does not make any assumptions about the underlying distribution of the data. It work on the basis proximity of feature similarity or find the nearest neighbours on a given dataset and used the majority voting to classify new data point. *KNN* determines feature similarity of new data points based on previously stored data points using Euclidean distance between two data points. This algorithm also known as lazy learning algorithm because it does not require to tuning the parameters and work without trained the model. Instead of trained the model it takes all the data points at the time of prediction. However, it has some disadvantage like high cost, slow speed, less scalability for high dimensional dataset [129].

c) **Decision Tree (*DT*):** *DT* is a supervised learning classifiers used for classification and regression problems. It can work on numerical and categorical data also. It follows the approach of tree model in which data points are divided into two branches and make conclusion of significant features at each node of the tree [129]. Decision Trees is to generate a training flowchart structure that can be employed to classify a class or value of a target variable based on decision rules learned from past data. It is capable to work well with huge dataset and noisy dataset as compared to *KNN* and *SVM* models [39].

d) **Support Vector Machine (*SVM*):** *SVM* is a powerful supervised learning model used for both classification and regression problems. The purpose of *SVM* is to create best hyperplane that separate the data point of one class to other class accurately in training and testing dataset. In *SVM* dataset indicated as a collection of data points separated by a line called as hyperplane, which is used to classify the class of dataset. It is effective classifier to handle the large dataset efficiently. It is used to overcome the problem of linear function in high dimensional feature spaces. However, it does not work well in case of large dataset with noisy dataset and takes more training time [39].

e) **Random Forest (*RF*):** *RF* is an ensemble learning based classifier consist of number of decision tree that run in parallel, the forest it builds is termed as ensemble of decision tree, each tree in ensemble model has data samples taken from training data with replacement, known as bootstrap sample which is collection of learning models to improves overall results of the models [129]. In *RF* decision is taken by majority voting approach for classification of the target class. It is easy, flexible and simple algorithm that produce better result even without hyper parameters tuning most of the time. It reduces the overfitting which resulted into improve accuracy in decision tree [130]. It work well for both categorical and continuous data. Although, the speed of *RF* is slow because it requires more time in training the set of decision trees to build strong classifier [39].

Deep learning for malware detection

DL is a subset of *ML*, which is most popular technique used for automatic feature extraction and to processed high dimensional data. Recently, it has wide applications in various domain like fraud detection, security, malware detection and intrusion detection. *DL* models consisted of number hierarchical layers which is combination of input layers, hidden layers and output layers [131]. The backbone of deep neural network is comprises of artificial neurons, each neuron works on basis of its weight and activation function which may be different from other neurons. As the weights of each neuron may vary at different layers, the initial weight can be set randomly or similar during initialization. Later on, the model will adjusted its weights while compilation as per the error value. Numerous *DL* models have been applied in malware detection. The most popular models are like *CNN, GRU, LSTM, DNN* and *GAN* [12,132].

a) **Convolutional Neural Network (*CNN*):** *CNN* is a *DL* model consist of mainly three layers like convolutional layers, pooling layers and fully connected layers [133]. The convolutional layer is main building component of *CNN* used to feature mapping. It has filters or kernel through which parameters are to be learned during training. The size of the filter is smaller than the input image. The output of the first layer is passed to pooling layer. The function of pooling layer is to decrease the size of dimension of feature map that is width and height while preserving the depth or number of channels and reduce the computational cost [130]. The activation function used by *CNN* like Relu, tanh and sigmoid to bring the nonlinearity need to sort out nonlinear features. The last layer that is fully connected layer builds the mapping among features and target variables or class by integrating all connected neurons with previous layers to output layers [13].

b) **Gated Recurrent Unit (*GRU*):** *GRU* is a Recurrent Neural Network (*RNN*). It is used for detection and classification problems. *GRU* is employed to overcome the vanishing gradient problem that is related to RNN. The *GRU* consist of two gates such as update and reset gates [130]. Each gates having its own weights and biases to control the flow of information. Update gate decide how much information will pass to next cell ahead and reset gate separate the unnecessary information to pass in *GRU* network. *GRU* has advantages over long short term memory (*LSTM*). *GRU* requires less memory and is faster than *LSTM* [132]. However, *LSTM* is more accurate when using datasets with longer sequences [37].

c) **Long Short-Term Memory (*LSTM*):** *LSTM* is an enhanced version of *GRU* architecture utilized in Deep Learning. It is used to learn, process, and classify sequential data because these networks can learn long-term dependencies between time steps of data [37]. *LSTM* has higher memory power to remember the outputs of each node for a more extended period to produce the outcome for the next node efficiently. *LSTM* use a series of gates which control how the information in a sequence of data comes, stored and leaves the network. There are three gates in a typical *LSTM*: forget gate, input gate and output gate [13]. *LSTM* has a large number of parameters and computations which makes it difficult to train and optimize. *LSTM* also suffers from the vanishing and exploding gradient problems, which hamper the learning of long-term dependencies and cause instability and divergence [117,134].

d) **Deep Neural Network (*DNN*):** *DNN* has multiple hidden layers between input and output layer. It is a feed forward network used to solve the nonlinear problems. The input layer processed the input data and hidden layers are used to perform feature extraction task and computational operation with its weight and biases [135]. The output of hidden layers is send to the output layer that predict the target value. To minimize the error and loss of the model, number of activation functions are iteratively fine-tuned with stochastic gradient descent, ReLu and sigmoid function [136].

e) **Generative Adversarial Network (*GAN*):** *GAN* is a kind *DL* model which consist of two model such generator and discriminator. The generator takes the fake sample with some random noise and passes the fake data to the discriminator network [137]. The objective of discriminator is to differentiate between real and fake data [12]. The *GAN* model is based on min-max theory, it means generator try to minimize the fake results and discriminator try to maximize the fake results with its power to make prediction in real or fake samples [138].

Sources of malware dataset

This study shows the *ML* and *DL* based malware detection system that uses different publicly available android, windows and *IoT* based cross platform datasets, and also utilized online malware repositories like virusShare and VirusTotal. The detection accuracy depends on the size and proper ratio of malware and benign samples for training and testing the models. Most of the existing models shows better detection accuracy and some of the models suffers from poor accuracy due to lack of access to real time malware data. Here are some of the most usable source of dataset mentioned below:

- ***CICInvesAndMal2019:*** It is a publicly available *CICInvesAndMal2019* android malware dataset [139]. It is created by installing 15,037 instances that it is collection of 9477 benign and 5560 malware samples on real android devices.
- ***CICMalMem2022:*** *CICMalMem2022* [140] dataset is a obfuscated malware for memory analysis. It is created to test obfuscated malware detection methods through memory dumps. It is consists of 58, 596 instances, in which 29,298 instances are benign class and 29,298 are malware class. The malware memory dump was generated by running application software instances gathered from VirusTotal on a Virtual Machine [141].
- **Malimg**: Malimg is a windows malware dataset that contains 9339 grayscale malware images of windows executable files that belong to 25 families of malware such as worms, Trojans, *PWS*, dialer, Downloader, rogue, Backdoor, and Worm:autoIT. The malware binaries of 8-bit are transformed into grayscale images.
- ***IoT*-malware**: It is collection malicious, benign, and hackware images which includes executable and linkable format *(ELF)* files, portable executable *(PE)* files, and software binaries collected from Kaggle [142]. The total malware samples are 3813 in which 2999 samples are benign, 103 samples are hackware, and 711 samples are malware.
- **Microsoft Malware classification Challenge BIG 2015 (*MMB-15*):** *MMB-15* dataset [143] is more than half a terabyte in size. It comprises of byte code files and asm (assembly) code files of twenty thousand malware samples. It is a collection of 9 types of malware families. Each malware file is distinguished by an identifier, a twenty-character unique hash value and a class label which is separating each of the 9 malware family names. A total of 10,349 malware samples are collected in this work of worms, adware, backdoor, Trojan and obfuscated malware attacks.
- **Drebin**: It is a collection of 5560 samples of malware and 123,453 benign samples that belongs to 179 malware families. It was collected between august 2010 to october 2012. This is android based malware dataset includes the features like *API* calls, permission and URLs [131,36].
- **VirusShare:** It is an online repository of malware samples that gives access to different types of malicious and benign samples to evaluate the performance detection system [144].
- **VirusTotal:** It is publicly available website for collecting malware samples and benign samples [131,145].

Recommendation for emerging malware detection and identification

This study provides some significant achievements in the areas of Android malware detection and Cross-Platform malware detection and identification. Here, the following are some possible recommendation for future research work:

- We need to explore advanced feature optimization methods to detect advanced *AI*-based malicious code, to predict emerging malware types and patterns and, make machine learning models more interpretable and understandable for large dataset.
- It is necessary to design and develop an effective hybrid meta-heuristic based feature selection techniques that can identify optimal features and reduces computational cost of features optimizing process.
- We need to enhance the proposed work by integrating the fog computing and federated learning technique for design of scalable and efficient malware detection model, because limited work related to malware detection using this environment is available to the best of my knowledge.
- Currently, detection of obfuscated and zero-day malware detection is still remain a critical challenge for researchers. So, it would be interesting to incorporate advance ensemble of deep learning models and reinforcement learning techniques to detect obfuscated and zero-day malware.

Conclusion

Undoubtedly, Cyber security has attracted many researchers in the past for designing of *ML* and *DL* based malware detection models. In this study, we present a comprehensive review of the malware detection approaches. Overall, the literature review of malware detection is

grouped into three categories such as review of *FS* techniques proposed for malware detection, review of *ML*-based techniques proposed for malware detection and review of *DL*-based techniques proposed for malware detection. Based on the literature review, different shortcomings and research gaps, and some future directives have been suggested in order to design a sophisticated malware detection framework. Currently, detection of obfuscated and zero-day malware detection is still remain a critical challenge for researchers. So, it would be interesting to incorporate advance ensemble of deep learning models and reinforcement learning techniques to detect obfuscated and zero-day malware. Furthermore, it is lack of access to real-world malware samples and labelled datasets is often limited due to privacy and security concerns. Consequently, feature selection methods might not be fully validated in real-world scenarios, and their performance might differ when tested on more comprehensive and diverse datasets. So, there is need to includes more samples of recent malware families and evaluate the performance of the designed method in order to enhance the detection accuracy for new sophisticated malwares with different *ML* and *DL* models.

CRediT authorship contribution statement

Santosh K. Smmarwar: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Formal analysis, Data curation, Conceptualization. **Govind P. Gupta:** Writing – review & editing, Visualization, Supervision, Investigation, Formal analysis, Data curation. **Sanjay Kumar:** Visualization, Validation, Supervision, Investigation, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] P. Kumar, R. Tripathi, and G.P. Gupta, "A Review on Intrusion Detection System and Cyber Threat Intelligence for Secure IoT-enabled Network: Challenges and Directions." [Online]. Available: https://www.researchgate.net/publication/359370843.

[2] R. Kumar, P. Kumar, R. Tripathi, G.P. Gupta, S. Garg, M.M. Hassan, A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network, J. Parallel Distrib. Comput. 164 (2022) 55–68, https://doi.org/10.1016/j.jpdc.2022.01.030. Jun.

[3] A.K. Dey, G.P. Gupta, S.P. Sahu, A metaheuristic-based ensemble feature selection framework for cyber threat detection in IoT-enabled networks, Decis. Anal. J. 7 (January) (2023) 100206, https://doi.org/10.1016/j.dajour.2023.100206.

[4] P. Kumar, R. Kumar, G.P. Gupta, R. Tripathi, A Distributed framework for detecting DDoS attacks in smart contract-based Blockchain-IoT Systems by leveraging Fog computing, Trans. Emerg. Telecommun. Technol. 32 (6) (2021), https://doi.org/10.1002/ett.4112. Jun.

[5] V. Sihag, M. Vardhan, P. Singh, BLADE: Robust malware detection against obfuscation in android, Forensic Sci. Int. Digit. Investig. 38 (2021) 301176, https://doi.org/10.1016/j.fsidi.2021.301176.

[6] V. Sihag, M. Vardhan, P. Singh, A survey of android application and malware hardening, Comput. Sci. Rev. 39 (2021) 100365, https://doi.org/10.1016/j.cosrev.2021.100365.

[7] Statista Research Department, Annual Number of Malware Attacks Worldwide from 2015 to First Half 2022, Statista Research Department, 2023. Accessed: Aug. 08[Online]. Available: https://www.statista.com/statistics/873097/malware-attacks-per-year-worldwide/.

[8] P. Kumar, G.P. Gupta, R. Tripathi, Toward Design of an Intelligent Cyber Attack Detection System using Hybrid Feature Reduced Approach for IoT Networks, Arab. J. Sci. Eng. 46 (4) (2021) 3749–3778, https://doi.org/10.1007/s13369-020-05181-3. Apr.

[9] J. Singh, J. Singh, A survey on machine learning-based malware detection in executable files, J. Syst. Archit. 112 (July 2020) (2021) 101861, https://doi.org/10.1016/j.sysarc.2020.101861.

[10] M. Conti, P. Vinod, A. Vitella, Obfuscation detection in Android applications using deep learning, J. Inf. Secur. Appl. 70 (September) (2022) 103311, https://doi.org/10.1016/j.jisa.2022.103311.

[11] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei, Q. Zheng, IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture, Comput. Networks 171 (January) (2020) 107138, https://doi.org/10.1016/j.comnet.2020.107138.

[12] S.K. Smmarwar, G.P. Gupta, S. Kumar, Deep malware detection framework for IoT- based smart agriculture, Comput. Electr. Eng. 104 (PA) (2022) 108410, https://doi.org/10.1016/j.compeleceng.2022.108410.

[13] S.K. Smmarwar, G.P. Gupta, S. Kumar, AI-empowered malware detection system for industrial internet of things, Comput. Electr. Eng. 108 (April) (2023) 108731, https://doi.org/10.1016/j.compeleceng.2023.108731.

[14] T. Sharma, D. Rattan, Malicious application detection in android - A systematic literature review, Comput. Sci. Rev. 40 (2021) 100373, https://doi.org/10.1016/j.cosrev.2021.100373.

[15] D. Gibert, C. Mateu, J. Planes, The rise of machine learning for detection and classification of malware: Research developments, trends and challenges, J. Netw. Comput. Appl. 153 (July 2019) (2020) 102526, https://doi.org/10.1016/j.jnca.2019.102526.

[16] S. Abijah Roseline, S. Geetha, A comprehensive survey of tools and techniques mitigating computer and mobile malware attacks, Comput. Electr. Eng. 92 (March) (2021) 107143, https://doi.org/10.1016/j.compeleceng.2021.107143.

[17] S. Madan, S. Sofat, D. Bansal, Tools and Techniques for Collection and Analysis of Internet-of-Things malware: A systematic state-of-art review, J. King Saud Univ. Comput. Inf. Sci. 34 (10) (2022) 9867–9888, https://doi.org/10.1016/j.jksuci.2021.12.016.

[18] D. Ucci, L. Aniello, R. Baldoni, Survey of machine learning techniques for malware analysis, Comput. Secur. 81 (2019) 123–147, https://doi.org/10.1016/j.cose.2018.11.001.

[19] A. Muzaffar, H. Ragab Hassen, M.A. Lones, H. Zantout, An in-depth review of machine learning based Android malware detection, Comput. Secur. 121 (2022) 102833, https://doi.org/10.1016/j.cose.2022.102833.

[20] M. Gopinath, S.C. Sethuraman, A comprehensive survey on deep learning based malware detection techniques, Comput. Sci. Rev. 47 (2023) 100529, https://doi.org/10.1016/j.cosrev.2022.100529.

[21] T. Yi, X. Chen, Y. Zhu, W. Ge, Z. Han, Review on the application of deep learning in network attack detection, J. Netw. Comput. Appl. 212 (December 2022) (2023) 103580, https://doi.org/10.1016/j.jnca.2022.103580.

[22] E. Mbunge, B. Muchemwa, J. Batani, N. Mbuyisa, A review of deep learning models to detect malware in Android applications, Cyber Secur. Appl. 1 (April 2022) (2023) 100014, https://doi.org/10.1016/j.csa.2023.100014.

[23] P. Mishra, A. Gupta, P. Aggarwal, E.S. Pilli, vServiceInspector: Introspection-assisted evolutionary bag-of-ngram approach to detect malware in cloud servers, Ad Hoc Networks 131 (January) (2022) 102836, https://doi.org/10.1016/j.adhoc.2022.102836.

[24] A. Kamboj, P. Kumar, A.K. Bairwa, S. Joshi, Detection of malware in downloaded files using various machine learning models, Egypt. Informatics J. 24 (1) (2023) 81–94, https://doi.org/10.1016/j.eij.2022.12.002.

[25] M. Wadkar, F. Di Troia, M. Stamp, Detecting malware evolution using support vector machines, Expert Syst. Appl. 143 (2020) 113022, https://doi.org/10.1016/j.eswa.2019.113022.

[26] S. Wu, P. Wang, X. Li, Y. Zhang, Effective detection of android malware based on the usage of data flow APIs and machine learning, Inf. Softw. Technol. 75 (2016) 17–25, https://doi.org/10.1016/j.infsof.2016.03.004.

[27] S. Srinivasan, D. P, Enhancing the security in cyber-world by detecting the botnets using ensemble classification based machine learning, Meas. Sensors 25 (November 2022) (2023) 100624, https://doi.org/10.1016/j.measen.2022.100624.

[28] S. Yoo, S. Kim, S. Kim, B.B. Kang, AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification, Inf. Sci. (Ny). 546 (2021) 420–435, https://doi.org/10.1016/j.ins.2020.08.082.

[29] T. Muralidharan, N. Nissim, Improving malicious email detection through novel designated deep-learning architectures utilizing entire email, Neural Networks 157 (2023) 257–279, https://doi.org/10.1016/j.neunet.2022.09.002.

[30] A.A. Solanke, Explainable digital forensics AI: Towards mitigating distrust in AI-based digital forensics analysis using interpretable models, Forensic Sci. Int. Digit. Investig. 42 (2022) 301403, https://doi.org/10.1016/j.fsidi.2022.301403.

[31] T. Rezaei, F. Manavi, A. Hamzeh, A PE header-based method for malware detection using clustering and deep embedding techniques, J. Inf. Secur. Appl. 60 (June) (2021) 102876, https://doi.org/10.1016/j.jisa.2021.102876.

[32] Y. Sun, A.K. Bashir, U. Tariq, F. Xiao, Effective malware detection scheme based on classified behavior graph in IIoT, Ad Hoc Networks 120 (March) (2021) 102558, https://doi.org/10.1016/j.adhoc.2021.102558.

[33] H.J. Kim, Image-based malware classification using convolutional neural network. Lecture Notes in Electrical Engineering, Springer Verlag, 2018, pp. 1352–1357, https://doi.org/10.1007/978-981-10-7605-3_215.

[34] R. Kumar, P. Kumar, R. Tripathi, G.P. Gupta, N. Kumar, M.M. Hassan, A Privacy-Preserving-Based Secure Framework Using Blockchain-Enabled Deep-Learning in Cooperative Intelligent Transport System, IEEE Trans. Intell. Transp. Syst. (2021), https://doi.org/10.1109/TITS.2021.3098636.

[35] U. Ahmed, J.C. Lin, G. Srivastava, Mitigating adversarial evasion attacks of ransomware using ensemble learning, Comput. Electr. Eng. 100 (April) (2022).

[36] F. Ceschin, M. Botacin, H.M. Gomes, F. Pinagé, L.S. Oliveira, A. Grégio, Fast & Furious: On the modelling of malware detection as an evolving data stream, Expert Syst. Appl. 212 (February 2021) (2023) 118590, https://doi.org/10.1016/j.eswa.2022.118590.

[37] P. Kumar, G.P. Gupta, R. Tripathi, S. Garg, M.M. Hassan, DLTIF: Deep Learning-Driven Cyber Threat Intelligence Modeling and Identification Framework in IoT-Enabled Maritime Transportation Systems, IEEE Trans. Intell. Transp. Syst. (2021) 1–10, https://doi.org/10.1109/tits.2021.3122368. Nov.

[38] E.D.O. Andrade, J. Viterbo, C.N. Vasconcelos, J. Guérin, F.C. Bernardini, A model based on LSTM neural networks to identify five different types of malware, Procedia Comput. Sci. 159 (2019) 182–191, https://doi.org/10.1016/j.procs.2019.09.173.

[39] J. Singh, J. Singh, A survey on machine learning-based malware detection in executable files, J. Syst. Archit. 112 (March) (2021) 101861, https://doi.org/10.1016/j.sysarc.2020.101861.

[40] S.I. Imtiaz, S. ur Rehman, A.R. Javed, Z. Jalil, X. Liu, W.S. Alnumay, DeepAMD: Detection and identification of Android malware using high-efficient Deep Artificial Neural Network, Futur. Gener. Comput. Syst. 115 (2021) 844–856, https://doi.org/10.1016/j.future.2020.10.008.

[41] M. Akour, I. Alsmadi, M. Alazab, The malware detection challenge of accuracy, in: 2016 2nd Int. Conf. Open Source Softw. Comput, 2017, pp. 1–6, https://doi.org/10.1109/osscom.2016.7863676.

[42] H. Naeem, S. Dong, O.J. Falana, F. Ullah, Development of a deep stacked ensemble with process based volatile memory forensics for platform independent malware detection and classification, Expert Syst. Appl. 223 (March) (2023) 119952, https://doi.org/10.1016/j.eswa.2023.119952.

[43] A.O. Almashhadani, D. Carlin, M. Kaiiali, S. Sezer, MFMCNS: a multi-feature and multi-classifier network-based system for ransomworm detection, Comput. Secur. 121 (May 2017) (2022) 102860, https://doi.org/10.1016/j.cose.2022.102860.

[44] A. Damodaran, F. Di Troia, C.A. Visaggio, T.H. Austin, M. Stamp, A comparison of static, dynamic, and hybrid analysis for malware detection, J. Comput. Virol. Hacking Tech. 13 (1) (2017) 1–12, https://doi.org/10.1007/s11416-015-0261-z.

[45] L. Martignoni, M. Christodorescu, S. Jha, OmniUnpack: Fast, generic, and safe unpacking of malware, in: Proc. - Annu. Comput. Secur. Appl. Conf. ACSAC, 2007, pp. 431–440, https://doi.org/10.1109/ACSAC.2007.15.

[46] I.A. Saeed, A. Selamat, A.M.A. Abuagoub, A Survey on Malware and Malware Detection Systems, Int. J. Comput. Appl. 67 (16) (2013) 25–31, https://doi.org/10.5120/11480-7108.

[47] M. Khan, D. Baig, U.S. Khan, A. Karim, Malware Classification Framework using Convolutional Neural Network, in: 1st Annual International Conference on Cyber Warfare and Security, ICCWS 2020 - Proceedings, Institute of Electrical and Electronics Engineers Inc., 2020, https://doi.org/10.1109/ICCWS48432.2020.9292384. Oct.

[48] B. Bashari Rad, M. Masrom, S. Ibrahim, Camouflage In Malware: From Encryption To Metamorphism, Int. J. Comput. Sci. Netw. Secur. 12 (8) (2012) 74–83.

[49] M. Alazab, S. Huda, J. Abawajy, R. Islam, J. Yearwood, S. Venkatraman, R. Broadhurst, A Hybrid Wrapper-Filter Approach for Malware Detection, J. Networks 9 (11) (1969) 2878–2891, https://doi.org/10.4304/jnw.9.11.2878-2891.

[50] M. Ahmadi, A. Sami, H. Rahimi, B. Yadegari, Malware detection by behavioural sequential patterns, Comput. Fraud Secur. 2013 (8) (2013) 11–19, https://doi.org/10.1016/S1361-3723(13)70072-1.

[51] L. Wang, Z. Li, Y. Chen, Z.J. Fu, X. Li, Thwarting zero-day polymorphic worms with network-level length-based signature generation, IEEE/ACM Trans. Netw. 18 (1) (2010) 53–66, https://doi.org/10.1109/TNET.2009.2020431.

[52] M. Vasilescu, L. Gheorghe, N. Tapus, Practical malware analysis based on sandboxing, in: Proc. - RoEduNet IEEE Int. Conf., 2014, https://doi.org/10.1109/RoEduNet-RENAM.2014.6955304.

[53] S. Kumar, C. Rama Krishna, N. Aggarwal, R. Sehgal, S. Chamotra, Malicious data classification using structural information and behavioral specifications in executables, in: 2014 Recent Adv. Eng. Comput. Sci. RAECS 2014, 2014, pp. 6–8, https://doi.org/10.1109/RAECS.2014.6799525.

[54] M.A.M. Ali, M.A. Maarof, Dynamic innate immune system model for malware detection, in: 2013 Int. Conf. IT Converg. Secur. ICITCS 2013, 2013, pp. 3–6, https://doi.org/10.1109/ICITCS.2013.6717828.

[55] I. You, K. Yim, Malware obfuscation techniques: A brief survey, in: Proc. - 2010 Int. Conf. Broadband, Wirel. Comput. Commun. Appl. BWCCA 2010, 2010, pp. 297–300, https://doi.org/10.1109/BWCCA.2010.85.

[56] L. Cai, Y. Li, Z. Xiong, JOWMDroid : Android Malware Detection Based on Feature Weighting with Joint Optimization of Weight-Mapping and Classifier Parameters, Comput. Secur. (2020) 102086, https://doi.org/10.1016/j.cose.2020.102086.

[57] A.K. Singh, G. Wadhwa, M. Ahuja, K. Soni, K. Sharma, Android Malware Detection using LSI-based Reduced Opcode Feature Vector, Procedia Comput. Sci. 173 (2019) (2020) 291–298, https://doi.org/10.1016/j.procs.2020.06.034.

[58] L. Wang, Y. Gao, S. Gao, X. Yong, A new feature selection method based on a self-variant genetic algorithm applied to android malware detection, Symmetry 13 (7) (2021) 1–21, https://doi.org/10.3390/sym13071290.

[59] D.Ö. Şahin, O.E. Kural, S. Akleylek, E. Kılıç, A novel permission-based Android malware detection system using feature selection based on linear regression, Neural Comput. Appl. 1 (2021), https://doi.org/10.1007/s00521-021-05875-1.

[60] O.A. Alzubi, J.A. Alzubi, A.M. Al-Zoubi, M.A. Hassonah, U. Kose, An efficient malware detection approach with feature weighting based on Harris Hawks optimization, Cluster Comput. 25 (4) (2022) 2369–2387, https://doi.org/10.1007/s10586-021-03459-1.

[61] P. Bhat, K. Dutta, A multi-tiered feature selection model for android malware detection based on Feature discrimination and Information Gain, J. King Saud Univ. Comput. Inf. Sci. 34 (10) (2022) 9464–9477, https://doi.org/10.1016/j.jksuci.2021.11.004.

[62] R.M. Sharma, C.P. Agrawal, A BPSO and Deep Learning Based Hybrid Approach for Android Feature Selection and Malware Detection, in: Proc. - 2022 IEEE 11th Int. Conf. Commun. Syst. Netw. Technol. CSNT 2022, 2022, pp. 628–634, https://doi.org/10.1109/CSNT54456.2022.9787671.

[63] A.S. Shatnawi, Q. Yassen, A. Yateem, An Android Malware Detection Approach Based on Static Feature Analysis Using Machine Learning Algorithms, Procedia Comput. Sci. 201 (C) (2022) 653–658, https://doi.org/10.1016/j.procs.2022.03.086.

[64] H. Alazzam, A. Al-Adwan, O. Abualghanam, E. Alhenawi, A. Alsmady, An Improved Binary Owl Feature Selection in the Context of Android Malware Detection, Computers 11 (12) (2022) 1–19, https://doi.org/10.3390/computers11120173.

[65] M.S. Hossain, N. Hasan, M.A. Samad, H.M. Shakhawat, J. Karmoker, F. Ahmed, K. F.M.N. Fuad, K. Choi, Android Ransomware Detection from Traffic Analysis Using Metaheuristic Feature Selection, IEEE Access 10 (December) (2022) 128754–128763, https://doi.org/10.1109/ACCESS.2022.3227579.

[66] M. CHEMMAKHA, O. HABIBI, M. LAZAAR, Improving Machine Learning Models for Malware Detection Using Embedded Feature Selection Method, IFAC-PapersOnLine 55 (12) (2022) 771–776, https://doi.org/10.1016/j.ifacol.2022.07.406.

[67] M. Grace, M. Sughasiny, Malware detection for Android application using Aquila optimizer and Hybrid LSTM-SVM classifier, ICST Trans. Scalable Inf. Syst. 10 (1) (2022) e1, https://doi.org/10.4108/eetsis.v9i4.2565.

[68] R.M. Sharma, C.P. Agrawal, MH-DLdroid: A Meta-Heuristic and Deep Learning-Based Hybrid Approach for Android Malware Detection, Int. J. Intell. Eng. Syst. 15 (4) (2022) 425–435, https://doi.org/10.22266/ijies2022.0831.38.

[69] P. Duraisamy Soundrapandian, G. Subbiah, MULBER: Effective Android Malware Clustering Using Evolutionary Feature Selection and Mahalanobis Distance Metric, Symmetry 14 (10) (2022), https://doi.org/10.3390/sym14102221.

[70] M.R. Ghazi, N.S. Raghava, Machine Learning Based Obfuscated Malware Detection in the Cloud Environment with Nature-Inspired Feature Selection, in: 2022 5th Int. Conf. Multimedia, Signal Process. Commun. Technol. IMPACT 2022, 2022, pp. 8–12, https://doi.org/10.1109/IMPACT55510.2022.10029271.

[71] M.N. Al-Andoli, S.C. Tan, K.S. Sim, C.P. Lim, P.Y. Goh, Parallel Deep Learning with a hybrid BP-PSO framework for feature extraction and malware classification, Appl. Soft Comput. 131 (2022) 109756, https://doi.org/10.1016/j.asoc.2022.109756.

[72] M.S. Abbasi, H. Al-Sahaf, M. Mansoori, I. Welch, Behavior-based ransomware classification: A particle swarm optimization wrapper-based approach for feature selection, Appl. Soft Comput. 121 (2022) 108744, https://doi.org/10.1016/j.asoc.2022.108744.

[73] R. Islam, M. Islam, S. Saha, M. Jamal, A. Masud, Android malware classification using optimum feature selection and ensemble machine learning, Internet Things Cyber-Physical Syst. 3 (October 2022) (2023) 100–111, https://doi.org/10.1016/j.iotcps.2023.03.001.

[74] P.C.S. Mahesh, S. Hemalatha, An Efficient Android Malware Detection Using Adaptive Red Fox Optimization Based CNN, Wirel. Pers. Commun. 126 (1) (2022) 679–700, https://doi.org/10.1007/s11277-022-09765-0.

[75] E.S. Alomari, R.R. Nuiaa, Z.A.A. Alyasseri, H.J. Mohammed, N.S. Sani, M.I. Esa, B. A. Musawi, Malware Detection Using Deep Learning and Correlation-Based Feature Selection, Symmetry 15 (1) (2023) 123, https://doi.org/10.3390/sym15010123.

[76] A. Albakri, F. Alhayan, N. Alturki, S. Ahamed, S. Shamsudheen, Metaheuristics with Deep Learning Model for Cybersecurity and Android Malware Detection and Classification, Appl. Sci. 13 (4) (2023), https://doi.org/10.3390/app13042172.

[77] A. Daniel, R. Deebalakshmi, R. Thilagavathy, T. Kohilakanagalakshmi, S. Janakiraman, B. Balusamy, Optimal feature selection for malware detection in cyber physical systems using graph convolutional network, Comput. Electr. Eng. 108 (September 2022) (2023) 108689, https://doi.org/10.1016/j.compeleceng.2023.108689.

[78] A. Mahindru, A.L. Sangal, FSDroid:- A feature selection technique to detect malware from Android using Machine Learning Techniques, Multimed. Tools Appl. 80 (9) (2021) 13271–13323, https://doi.org/10.1007/s11042-020-10367-w.

[79] D.O. Şahin, Kural, Oğuz Emre, Akleylek, Sedat, Kılıç, A novel Android malware detection system: adaption of filter-based feature selection methods, J. Amb. Intellig. Humanized Comput. 14 (2023) 1243–1257, https://doi.org/10.1007/s12652-021-03376-6.

[80] D.Ö. Şahin, O.E. Kural, S. Akleylek, E. Kılıç, Permission-based Android malware analysis by using dimension reduction with PCA and LDA, J. Inf. Secur. Appl. 63 (October) (2021) 102995, https://doi.org/10.1016/j.jisa.2021.102995.

[81] C. Chimeleze, N. Jamil, R. Ismail, K.Y. Lam, J.S. The, J. Samual, C.A. Okeke, BFEDroid: A Feature Selection Technique to Detect Malware in Android Apps Using Machine Learning, Secur. Commun. Networks 2022 (1) (2022), https://doi.org/10.1155/2022/5339926.

[82] Y. Wu, M. Li, Q. Zeng, T. Yang, J. Wang, Z. Fang, L. Cheng, DroidRL: Feature selection for android malware detection with reinforcement learning, Comput. Secur. 128 (2023) 103126, https://doi.org/10.1016/j.cose.2023.103126.

[83] S. Garg, N. Baliyan, A novel parallel classifier scheme for vulnerability detection in Android, Comput. Electr. Eng. 77 (2019) 12–26, https://doi.org/10.1016/j.compeleceng.2019.04.019.

[84] S. Wang, Z. Chen, Q. Yan, B. Yang, L. Peng, Z. Jia, A mobile malware detection method using behavior features in network traffic, J. Netw. Comput. Appl. 133 (January) (2019) 15–25, https://doi.org/10.1016/j.jnca.2018.12.014.

[85] Ş. Bahtiyar, M.B. Yaman, C.Y. Altınığne, A multi-dimensional machine learning approach to predict advanced malware, Comput. Networks 160 (2019) 118–129, https://doi.org/10.1016/j.comnet.2019.06.015.

[86] L. Xiaofeng, J. Fangshuo, Z. Xiao, Y. Shengwei, S. Jing, P. Lio, ASSCA: API sequence and statistics features combined architecture for malware detection, Comput. Networks 157 (2019) 99–111, https://doi.org/10.1016/j.comnet.2019.04.007.

[87] E.M.B. Karbab, M. Debbabi, MalDy: Portable, data-driven malware detection using natural language processing and machine learning techniques on behavioral analysis reports, Digit. Investig. 28 (2019) S77–S87, https://doi.org/10.1016/j.diin.2019.01.017.

[88] W. Han, J. Xue, Y. Wang, Z. Liu, Z. Kong, MalInsight: A systematic profiling based malware detection framework, J. Netw. Comput. Appl. 125 (August 2018) (2019) 236–250, https://doi.org/10.1016/j.jnca.2018.10.022.

[89] A. Roy, D.S. Jas, G. Jaggi, K. Sharma, Android Malware Detection based on Vulnerable Feature Aggregation, Procedia Comput. Sci. 173 (2019) (2020) 345–353, https://doi.org/10.1016/j.procs.2020.06.040.

[90] D. Gupta, R. Rani, Improving malware detection using big data and ensemble learning, Comput. Electr. Eng. 86 (2020) 106729, https://doi.org/10.1016/j.compeleceng.2020.106729.

[91] E. Amer, I. Zelinka, A dynamic Windows malware detection and prediction method based on contextual understanding of API call sequence, Comput. Secur. 92 (2020), https://doi.org/10.1016/j.cose.2020.101760.

[92] R. Surendran, T. Thomas, S. Emmanuel, A TAN based hybrid model for android malware detection, J. Inf. Secur. Appl. 54 (2020) 102483, https://doi.org/10.1016/j.jisa.2020.102483.

[93] V.P. D, V. P, Detecting android malware using an improved filter based technique in embedded software, Microprocess. Microsyst. 76 (2020) 103115, https://doi.org/10.1016/j.micpro.2020.103115.

[94] J. Singh, J. Singh, Detection of malicious software by analyzing the behavioral artifacts using machine learning algorithms, Inf. Softw. Technol. 121 (January) (2020) 106273, https://doi.org/10.1016/j.infsof.2020.106273.

[95] R. Surendran, T. Thomas, S. Emmanuel, GSDroid: Graph Signal Based Compact Feature Representation for Android Malware Detection, Expert Syst. Appl. 159 (2020) 113581, https://doi.org/10.1016/j.eswa.2020.113581.

[96] I. Shhadat, B. Bataineh, A. Hayajneh, Z.A. Al-Sharif, The Use of Machine Learning Techniques to Advance the Detection and Classification of Unknown Malware, Procedia Comput. Sci. 170 (2019) (2020) 917–922, https://doi.org/10.1016/j.procs.2020.03.110.

[97] G. D'Angelo, M. Ficco, F. Palmieri, Association rule-based malware classification using common subsequences of API calls, Appl. Soft Comput. 105 (2021) 107234, https://doi.org/10.1016/j.asoc.2021.107234.

[98] N. Usman, S. Usman, F. Khan, M.A. Jan, A. Sajid, M. Alazab, P. Watters, Intelligent Dynamic Malware Detection using Machine Learning in IP Reputation for Forensics Data Analytics, Futur. Gener. Comput. Syst. 118 (2021) 124–141, https://doi.org/10.1016/j.future.2021.01.004.

[99] T. Panker, N. Nissim, Leveraging malicious behavior traces from volatile memory using machine learning methods for trusted unknown malware detection in Linux cloud environments, Knowledge-Based Syst. 226 (2021) 107095, https://doi.org/10.1016/j.knosys.2021.107095.

[100] V. Syrris, D. Geneiatakis, On machine learning effectiveness for malware detection in Android OS using static analysis data, J. Inf. Secur. Appl. 59 (May) (2021) 102794, https://doi.org/10.1016/j.jisa.2021.102794.

[101] S.K. Sasidharan, C. Thomas, ProDroid — An Android malware detection framework based on profile hidden Markov model, Pervasive Mob. Comput. 72 (2021) 101336, https://doi.org/10.1016/j.pmcj.2021.101336.

[102] Y. Wu, J. Shi, P. Wang, D. Zeng, C. Sun, DeepCatra : Learning Flow- and Graph-based Behaviors for Android Malware Detection, IET Information Security (2023) 1–12, https://doi.org/10.1049/ise2.12082.

[103] S.R.T. Mat, M.F.A. Razak, M.N.M. Kahar, J.M. Arif, A. Firdaus, A Bayesian probability model for Android malware detection, ICT Express 8 (3) (2022) 424–431, https://doi.org/10.1016/j.icte.2021.09.003.

[104] M.M. Alani, A.I. Awad, AdStop: Efficient flow-based mobile adware detection using machine learning, Comput. Secur. 117 (2022) 102718, https://doi.org/10.1016/j.cose.2022.102718.

[105] T.S. Urmila, Machine learning-based malware detection on Android devices using behavioral features, Mater. Today Proc. 62 (2022) 4659–4664, https://doi.org/10.1016/j.matpr.2022.03.121.

[106] D.Escudero García, N. DeCastro-García, A.L. Muñoz Castañeda, An effectiveness analysis of transfer learning for the concept drift problem in malware detection, Expert Syst. Appl. 212 (September 2022) (2023) 118724, https://doi.org/10.1016/j.eswa.2022.118724.

[107] M. Ahmed, N. Afreen, M. Ahmed, M. Sameer, J. Ahamed, An inception V3 approach for malware classification using machine learning and transfer learning, Int. J. Intell. Networks 4 (September 2022) (2023) 11–18, https://doi.org/10.1016/j.ijin.2022.11.005.

[108] H. Naeem, S. Dong, O.J. Falana, F. Ullah, Development of a Deep Stacked Ensemble With Process Based Volatile Memory Forensics for Platform Independent Malware Detection and Classification, Expert Syst. Appl. (2023) 119952, https://doi.org/10.1016/j.eswa.2023.119952.

[109] T. Tsafrir, A. Cohen, E. Nir, N. Nissim, Efficient feature extraction methodologies for unknown MP4-Malware detection using Machine learning algorithms, Expert Syst. Appl. 219 (January) (2023) 119615, https://doi.org/10.1016/j.eswa.2023.119615.

[110] F. Rustam, I. Ashraf, A.D. Jurcut, A.K. Bashir, Y. Bin Zikria, Malware detection using image representation of malware data and transfer learning, J. Parallel Distrib. Comput. 172 (2023) 32–50, https://doi.org/10.1016/j.jpdc.2022.10.001.

[111] N. Dabas, P. Ahlawat, P. Sharma, An Effective Malware Detection Method Using Hybrid Feature Selection and Machine Learning Algorithms, Arab. J. Sci. Eng. 48 (8) (2023) 9749–9767, https://doi.org/10.1007/s13369-022-07309-z.

[112] D.O. Sahin, S. Akleylek, E. Kilic, LinRegDroid: Detection of Android Malware Using Multiple Linear Regression Models-Based Classifiers, IEEE Access 10 (2022) 14246–14259, https://doi.org/10.1109/ACCESS.2022.3146363.

[113] H. Alomari, Q.M. Yaseen, M.A. Al-Betar, A Comparative Analysis of Machine Learning Algorithms for Android Malware Detection, Procedia Comput. Sci. 220 (2019) (2023) 763–768, https://doi.org/10.1016/j.procs.2023.03.101.

[114] S. Kumar, B. Janet, S. Neelakantan, Identification of malware families using stacking of textural features and machine learning, Expert Syst. Appl. 208 (July) (2022) 118073, https://doi.org/10.1016/j.eswa.2022.118073.

[115] H. juan Zhu, Y. Li, L. min Wang, V.S. Sheng, A multi-model ensemble learning framework for imbalanced android malware detection, Expert Syst. Appl. 234 (June) (2023) 120952, https://doi.org/10.1016/j.eswa.2023.120952.

[116] S. Seraj, S. Khodambashi, M. Pavlidis, N. Polatidis, MVDroid: an android malicious VPN detector using neural networks, Neural Comput. Appl. 35 (29) (2023) 21555–21565, https://doi.org/10.1007/s00521-023-08512-1.

[117] Y. Sung, S. Jang, Y.S. Jeong, J.H.(James J.) Park, Malware classification algorithm using advanced Word2vec-based Bi-LSTM for ground control stations, Comput. Commun. 153 (December 2019) (2020) 342–348, https://doi.org/10.1016/j.comcom.2020.02.005.

[118] S. Venkatraman, M. Alazab, R. Vinayakumar, A hybrid deep learning image-based analysis for effective malware detection, J. Inf. Secur. Appl. 47 (2019) 377–389, https://doi.org/10.1016/j.jisa.2019.06.006. Aug.

[119] W. Zhong, F. Gu, A multi-level deep learning system for malware detection, Expert Syst. Appl. 133 (2019) 151–162, https://doi.org/10.1016/j.eswa.2019.04.064.

[120] J. Kang, S. Jang, S. Li, Y.S. Jeong, Y. Sung, Long short-term memory-based Malware classification method for information security, Comput. Electr. Eng. 77 (2019) 366–375, https://doi.org/10.1016/j.compeleceng.2019.06.014.

[121] G. D'Angelo, M. Ficco, F. Palmieri, Malware detection in mobile environments based on Autoencoders and API-images, J. Parallel Distrib. Comput. 137 (2020) 26–33, https://doi.org/10.1016/j.jpdc.2019.11.001. Mar.

[122] X. Gao, C. Hu, C. Shan, B. Liu, Z. Niu, H. Xie, Malware classification for the cloud via semi-supervised transfer learning, J. Inf. Secur. Appl. 55 (October) (2020) 102661, https://doi.org/10.1016/j.jisa.2020.102661.

[123] K. Shaukat, S. Luo, V. Varadharajan, A novel deep learning-based approach for malware detection, Eng. Appl. Artif. Intell. 122 (February) (2023) 106030, https://doi.org/10.1016/j.engappai.2023.106030.

[124] H. Zhu, H. Wei, L. Wang, Z. Xu, V.S. Sheng, An effective end-to-end android malware detection method, Expert Syst. Appl. 218 (January) (2023) 119593, https://doi.org/10.1016/j.eswa.2023.119593.

[125] L. Saidia Fascí, M. Fisichella, G. Lax, C. Qian, Disarming visualization-based approaches in malware detection systems, Comput. Secur. 126 (2023), https://doi.org/10.1016/j.cose.2022.103062.

[126] M.K. Alzaylaee, S.Y. Yerima, S. Sezer, DL-Droid : Deep learning based android malware detection using real devices, Comput. Secur. 89 (2020), https://doi.org/10.1016/j.cose.2019.101663.

[127] D.O. Sahin, B.K. Yazar, S. Akleylek, E. Kilic, D. Giri, On the Android Malware Detection System Based on Deep Learning, in: ICAIAME 2021, 2023, https://doi.org/10.1007/978-3-031-09753-9.

[128] M. Waqar, S. Fareed, A. Kim, S.U.R. Malik, Malware Detection in Android IoT Systems Using Deep Learning, Computers, Materials & Continua (2023), https://doi.org/10.32604/cmc.2023.032984.

[129] W. Kanyongo, A.E. Ezugwu, Feature selection and importance of predictors of non-communicable diseases medication adherence from machine learning research perspectives, Informatics Med. Unlocked 38 (February) (2023) 101232, https://doi.org/10.1016/j.imu.2023.101232.

[130] Z. Guo, C. Yang, D. Wang, H. Liu, A novel deep learning model integrating CNN and GRU to predict particulate matter concentrations, Process Saf. Environ. Prot. 173 (December 2022) (2023) 604–613, https://doi.org/10.1016/j.psep.2023.03.052.

[131] E. Mbunge, B. Muchemwa, J. Batani, N. Mbuyisa, A review of deep learning models to detect malware in Android applications, Cyber Secur. Appl. 1 (February) (2023) 100014, https://doi.org/10.1016/j.csa.2023.100014.

[132] A.R. Javed, S.U. Rehman, M.U. Khan, M. Alazab, T.G. Reddy, CANintelliIDS: Detecting In-Vehicle Intrusion Attacks on a Controller Area Network Using CNN and Attention-Based GRU, IEEE Trans. Netw. Sci. Eng. 8 (2) (2021) 1456–1466, https://doi.org/10.1109/TNSE.2021.3059881.

[133] P. Dixit, S. Silakari, Deep Learning Algorithms for Cybersecurity Applications: A Technological and Status Review, Comput. Sci. Rev. 39 (2021) 100317, https://doi.org/10.1016/j.cosrev.2020.100317.

[134] H. Alkahtani, T.H.H. Aldhyani, Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications, Secur. Commun. Networks 2021 (2021), https://doi.org/10.1155/2021/3806459.

[135] X. Yuan, P. He, Q. Zhu, X. Li, Adversarial Examples: Attacks and Defenses for Deep Learning, IEEE Trans. Neural Networks Learn. Syst. 30 (9) (2019) 2805–2824, https://doi.org/10.1109/TNNLS.2018.2886017.

[136] S.P.R. M, P.K.R. Maddikunta, P. M, S. Koppu, T.R. Gadekallu, C.L. Chowdhary, M. Alazab, An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture, Comput. Commun. 160 (February) (2020) 139–149, https://doi.org/10.1016/j.comcom.2020.05.048.

[137] Z. Moti, S. Hashemi, H. Karimipour, Generative adversarial network to detect unseen Internet of Things malware, Ad Hoc Networks 122 (April) (2021), https://doi.org/10.1016/j.adhoc.2021.102591.
[138] P. Wang, Z. Wang, F. Ye, X. Chen, ByteSGAN: A semi-supervised Generative Adversarial Network for encrypted traffic classification in SDN Edge Gateway, Comput. Networks 200 (October) (2021) 108535, https://doi.org/10.1016/j.comnet.2021.108535.
[139] L. Taheri, A.F.A. Kadir, A.H. Lashkari, Extensible android malware detection and family classification using network-flows and API-calls, in: Proc. - Int. Carnahan Conf. Secur. Technol 2019-Octob, 2019, https://doi.org/10.1109/CCST.2019.8888430.
[140] Malware Memory Analysis, Canadian Institute for Cybersecurity, 2023. Accessed: Mar. 20[Online]. Available: https://www.unb.ca/cic/datasets/malmem-2022.html.
[141] A.H.L. Tristan Carrier, Princy Victor, Ali Tekeoglu, Detecting Obfuscated Malware using Memory Feature Engineering, in: The 8th International Conference on Information Systems Security and Privacy (ICISSP), 2022, p. 2022, https://doi.org/10.5220/0010908200003120.
[142] TECPERSON, "IoT Firmware Image Classification." Accessed: May 24, 2022. [Online]. Available: https://www.kaggle.com/datasets/datamunge/iot-firmware-image-classification.
[143] S. Kumar, B. Janet, DTMIC: Deep transfer learning for malware image classification, J. Inf. Secur. Appl. 64 (2022), https://doi.org/10.1016/j.jisa.2021.103063.
[144] O. James, A. Simon, S. Adebukola, Mal-Detect : An intelligent visualization approach for malware detection, J. King Saud Univ. Comput. Inf. Sci. 34 (5) (2022) 1968–1983, https://doi.org/10.1016/j.jksuci.2022.02.026.
[145] V. Verma, S.K. Muttoo, V.B. Singh, Multiclass malware classification via first- and second-order texture statistics, Comput. Secur. 97 (2020) 101895, https://doi.org/10.1016/j.cose.2020.101895.