Hindawi
Security and Communication Networks
Volume 2022, Article ID 8621083, 20 pages
https://doi.org/10.1155/2022/8621083

Research Article

A Systematic Overview of the Machine Learning Methods for Mobile Malware Detection

Yu-kyung Kim,[1] **Jemin Justin Lee**,[2] **Myong-Hyun Go**,[1] **Hae Young Kang**,[1] **and Kyungho Lee**[1]

[1]*Institute of Cyber Security & Privacy, Korea University, Seoul, Republic of Korea*
[2]*Center for Information Security Technology, Korea University, Seoul, Republic of Korea*

Correspondence should be addressed to Kyungho Lee; kevinlee@korea.ac.kr

Received 29 October 2021; Revised 6 December 2021; Accepted 12 June 2022; Published 22 July 2022

Academic Editor: Ilsun You

With the deployment of the 5G cellular system, the upsurge of diverse mobile applications and devices has increased the potential challenges and threats posed to users. Industry and academia have attempted to address cyber security challenges by implementing automated malware detection and machine learning algorithms. This study expands on previous research on machine learning-based mobile malware detection. We critically evaluate 154 selected articles and highlight their strengths and weaknesses as well as potential improvements. We explore the mobile malware detection techniques used in recent studies based on attack intentions, such as server, network, client software, client hardware, and user. In contrast to other SLR studies, our study classified the means of attack as supervised and unsupervised learning. Therefore, this article aims at providing researchers with in-depth knowledge in the field and identifying potential future research and a framework for a thorough evaluation. Furthermore, we review and summarize security challenges related to cybersecurity that can lead to more effective and practical research.

1. Introduction

Owing to the widespread popularity and rapid growth of various mobile applications for smartphones, cyberattacks through mobile applications have posed a serious threat [1–3]. Mobile malware attacks, such as phishing, repackaging, and application updates, are a constant threat to network providers, end-users, and app providers. The entire mobile operating system (OS) is vulnerable to cyberattacks, and approximately 87% of all Android smartphones are exposed to one or more fatal vulnerabilities [2]. Mobile malware poses a serious security threat to various applications, such as education, telecommunications, hospitals, and entertainment [4–6]. In other words, mobile malware attacks threaten cybersecurity in terms of confidentiality, integrity, and availability of data [3]. Established attack groups are capable of penetrating and destroying servers, networks, client software, client hardware, and mobile device users [7, 8].

Past studies on mobile malware have emphasized that government agencies have primarily focused on the gap within cyberspace[9–11]. However, these studies suggest that adopting new mitigation techniques are necessary. For instance, some evaluations are limited to certain malware, such as anomaly-based approaches, or some have failed to reveal the features required to train the classifier [12]. Scalability issues, such as limited computing and storage power to handle a large number of malware samples, require more attention [13].

Our goal is to reinvigorate research on these issues and reorient the practical requirements of cybersecurity domains. Therefore, we revisit the previous studies on machine learning-based mobile malware detection regarding unique requirements in cybersecurity domains. This study makes the following: contributions through an in-depth evaluation of the current and future solutions. Our approach leverages past studies on mobile malware detection studies that focus on evaluating datasets, detection techniques, means of attack, and evaluation metrics for system performance. We believe our study lays the foundation for future research and thesis that will support a larger research project. To the best of our knowledge, this is one of the first studies to perform a systematic literature review that will provide insight and

a crucial foundation for a foray into academic research. Furthermore, we were able to compare supervised and unsupervised learning for mobile malware based on the means of attack. We believe that by synthesizing the existing data, we were able to provide relevant insights for future researchers.

The remainder of this article is organized as follows: Section 2 describes the essential background for mobile cybersecurity, machine learning, and mobile malware. We describe relevant literature search methodologies that are essential for readers to systematically understand the accurate outlines of papers in Section 3. In Section 4, we investigate and analyze a machine learning-based mobile malware detection study. In Section 5, we conclude our article with discussions for future work.

2. Background

2.1. Mobile Cybersecurity. According to Certified Information Systems Auditor (CISA) [14], cybersecurity is a technology that protects networks, devices, and data from unauthorized access or criminal use, as well as refers to practices that ensure confidentiality, integrity, and availability of information [15]. Inadequate cybersecurity infrastructure could allow a malicious attacker to break into the system and spread malware, posing a serious risk. To reinforce cybersecurity, the following best practices should first be followed to minimize the risk of cyberattacks: for instance, keeping the software up to date, running the latest antivirus software, using strong passwords, changing default user names and passwords, implementing multilevel authentication (MFA), installing firewalls, and suspecting unexpected emails.

Past studies have focused on various aspects of the mobile cybersecurity [9–11, 16, 17]. Kang et al. [16] empirically analyzed the relationship between the Internet, mobile sellers, and service trade in Korea and observed that it has a positive effect. A vital aspect of information security research focuses on improving security protocols by encouraging users of information technology to adopt protective behaviors. Lemay et al. [17] proposed a model of the relationship between threat perception, anxiety level, and adaptive coping in college students based on the behavioral intention to learn about phishing.

As the number of mobile devices has increased across the world, the number of downloads of mobile device applications has also increased. These applications have developed into a means of causing personal information leakage or financial loss through junk mail or spam. In addition, attackers are capable of exploiting the vulnerabilities in Bluetooth-enabled devices to access privacy channels with higher-level clearances. Mobile malware is the main type of malware used in mobile device attacks, such as file manipulation, information leakage, financial loss, and device unavailability. There have been numerous studies conducted to address mobile cybersecurity [18–21]. "Location spoofing" refers to the act of falsely reporting the GPS location to other location-based applications. Wong et al. [18] proposed a behavior detection method using a gyroscope and accelerometer commonly mounted on mobile devices to prevent "location spoofing" and verified the authenticity of GPS data. Kholod et al. [19] studied various types of data distribution to improve the efficient and parallelized implementation of data mining in mobile cloud systems. La Marra et al. [20] conducted a study to detect malware in zero-day attacks by presenting D-BRIDEMAID, a reputation-based framework that can analyze Android applications. As security and performance requirements change, more mobile devices and web services require smaller and faster signatures [21]. To analyze mobile malware that is critical to cybersecurity, we must divide the analysis into areas such as servers, networks, client software, client hardware, and users.

2.2. Mobile Malware. Mobile malware inception began in the early 2000s [22]. Antivirus labs in Russia and Finland discovered Timfonica, the first known mobile virus, in 2000. Cabir, a notable mobile malware, was discovered in the Philippines in 2004. The malicious code from Timfonica infected mobile devices running Symbian OS, while the Cabir used a wireless Bluetooth signal to send a message "Carbe." The "CommWarrior" mobile malware discovered in 2005 was then spread to the multimedia messaging service (MMS). It was a malicious worm that attached a copy of its message to an MMS message and sent it to all contacts in the address book of the device. In 2010, the first SMS malicious code, "Trojan," affected Google's Android OS, sending particular SMS messages to specific numbers, causing financial loss as transactions were charged without the user's consent [23]. Most mobile malware is widely spread across Android-based mobile devices. Malware apps can be installed from unproven sources, such as third-party app stores and file-sharing websites. Thus, android-based applications are mostly targeted by the cyber threat groups. However, since Apple launched the iPhone in 2007, mobile malware targeting the iPhone has gradually emerged. Typically, mobile malware known as "IKee" is a malicious code that allows users to display images of pop stars from the 90s on the desktop of the iPhone, causing the platform structure to be arbitrarily changed.

Since 2004, mobile malware has spread rapidly. Mobile malware is installed on a terminal and carries out malicious attacks, such as stealing personal information, system damage, and remote control of mobile devices to induce user financial loss. Mobile malware can be classified into viruses, worms, Trojans, spyware, backdoors, and droppers. Viruses infect other files to spread, and worms are transmitted by SMS or MMS to replicate and spread themselves, destroying the OS. A Trojan is a malicious code that executes malicious behavior disguised as a normal program. Spyware is a malicious code that secretly collects information about an individual or organization or collects specific data without the user's consent. In terms of social engineering techniques, mobile malware is distributed in the following three methods: repackaging, application update, and phishing [24].

2.3. Machine Learning in Mobile Malware Detection. Machine learning started in 1966 with the ability to develop classification rules from experience [25]. Machine learning can detect mobile malware by learning various normal and malicious applications and detecting features on them. Machine learning can be classified as supervised learning [26], unsupervised learning [27], and reinforcement learning [28] as shown in Figure 1. Supervised learning, which learns from input and output values, is primarily used for classification and regression [26]. Unsupervised learning is used for clustering and compression and is learned only with input values [27]. Reinforcement learning is a behavioral psychology-based learning method for obtaining maximum rewards through agent-environment interactions [28]. Machine learning-based malware detection mainly uses supervised and unsupervised learning, with studies determining whether applications are normal, abnormal, or classified malware.

Machine learning should be identified regardless of classification, regression, or clustering. In addition, the collected data must be representative; thus, sample data must be collected and analyzed. The optimized data are then processed considering the limitations and potential errors of the sample data. To proceed with the learning process, after processing the data, extract the characteristics and apply the algorithm model based on the problem we want to solve with the data. At this time, a model parameter is obtained using the training data. Subsequently, the test data were used to evaluate the model in terms of accuracy, training speed, reliability, and generalization, and to determine the optimized model. In addition, it uses a new dataset to predict results and evaluate machine learning methods by solving real problems. The steps and approaches for detecting mobile malware based on machine learning are as follows: preparing data, feature extraction, training model, testing model, and deploying model. During the data preparation, the sample data are collected and preprocessed. The feature extraction is performed to reduce an initial set of data by identifying key features of the data for machine learning. Moreover, the training models are applied, and the testing model evaluates and optimizes the analysis. The deployment of the model evaluates the machine learning methods using a new dataset.

2.3.1. Algorithms of Supervised Learning. Support vector machine (SVM) [29–31] is an algorithm that can be used to classify data into a high-dimensional feature space in both linear classification and nonlinear classification. The model defines baselines for classification and is mainly used for data classification, such as pattern recognition and data analysis. When a new unclassified value appears, the classification identifies the side of its boundary. In other words, the algorithm classifies the data by categorizing the data to measure the distance between categories, obtain the central position value, and then calculate the hyperplane to judge the boundaries. As a result of the analysis of this study, majority of the studies employed the SVM algorithm [32–111].

Based on the nearest neighbor pattern classifications, the K-nearest neighbor (KNN) predicts new data using information from the nearest k of the existing data by finding the nearest k labeled samples [112–115]. It is an algorithm that determines the k elements closest to the input data within a specific space and classifies them into more matching groups. The KNN algorithm is efficient for classifying text documents [114], and recent studies have demonstrated that KNN is a suitable method for detecting denial of service (DoS) [116, 117].

The decision tree (DT) is a supervised learning model, a methodology used to analyze data to classify and predict patterns that may appear between classification and regression [118, 119]. It undergoes a series of decision-making processes by diverging from the top node to the bottom node, causing the heterogeneity between the nodes to increase, as depicted in Figure 2. Through this process, it classifies samples and regresses a binary division into classification, continuation, or numerical types. As a result of this study, the DT algorithm was used the fourth most [32, 37, 39, 44, 46, 47, 53, 56, 58, 59, 62, 64, 65, 68, 71, 72, 75, 76, 78, 85, 86, 92, 96, 98, 101, 111, 120–134].

The naive Bayesian (NB) is a supervised learning algorithm based on Bayes' theorem that is used for classification learning by multiplying prior probability information by the value of the "likelihood function" as measured through observation [135, 136]. However, this algorithm cannot be used if each probability violates the assumption that it is independent. This algorithm is a widely used classification technique, including spam mail filters, text classification, and sentiment analysis [137].

2.3.2. Algorithms of Unsupervised Learning. The K-means clustering algorithm [138], which is a clustering model of unsupervised learning, is an algorithm that groups the given data into k clusters. This method minimizes the variance of the distance differences between each cluster. "K" refers to the number of groups or clusters to be grouped from a given data. "Means" refers to the average distance between the center of each cluster and the data. As a result, the K-means algorithm was used less [47, 56, 68, 81, 97, 98, 111, 139].

2.3.3. Evaluation Metrics. To evaluate the performance of a machine learning model, before generating data, it is necessary to separate the training and evaluation sets into a training set and then verify the accuracy of the machine learning model with the evaluation set. The predictive power of classifiers with machine learning algorithms that can perform classification should be verified and evaluated. The machine learning model and pattern classification performance evaluation metrics include Accuracy, Precision, Recall, and F_1. Before explaining these four indicators, we must know the confusion matrix. The confusion matrix is a table for measuring predictive performance through training by comparing predictive and actual values. In other words, the evaluation metric presents the relationship

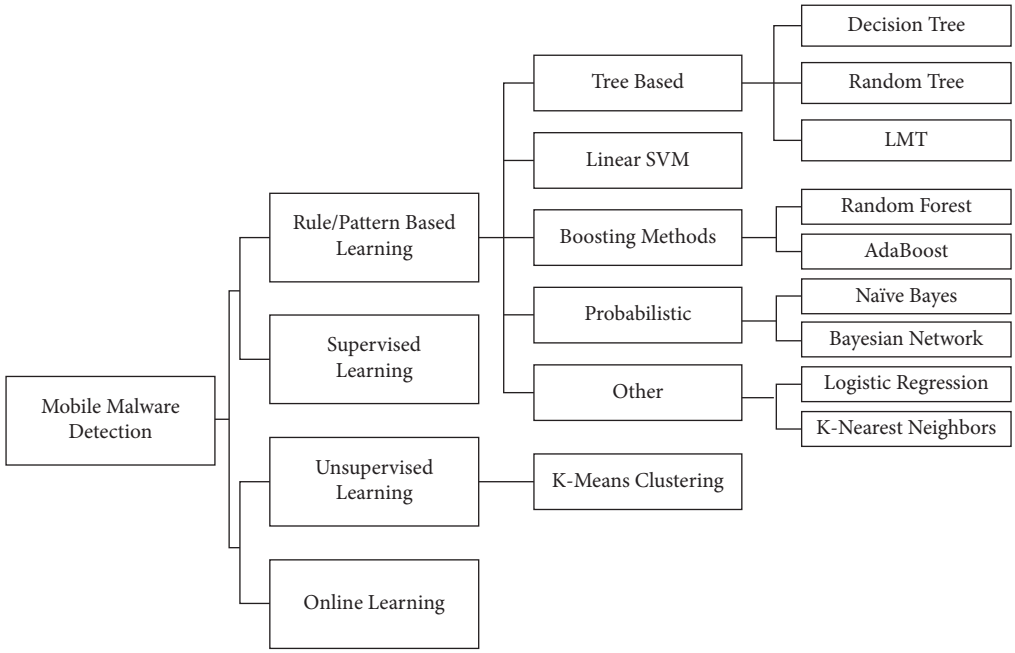

Figure 1: Algorithms used in mobile malware detection.

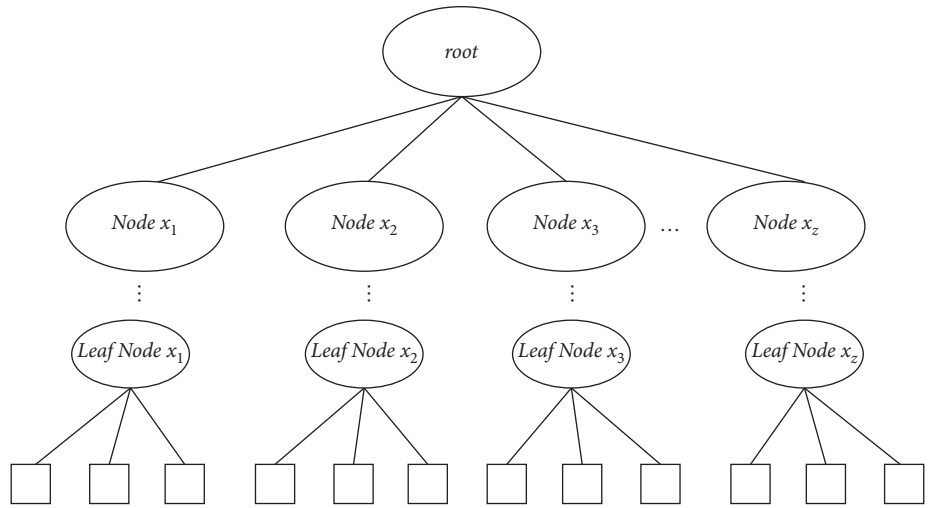

Figure 2: Decision tree classifier.

between the answer presented by the model and the actual answer as a factor and can be defined as four cases. These four cases are shown in Figure 2.

According to Figure 2, the target variable has two values: positive and negative. The columns represent the actual values of the target variable, and the rows represent the predicted values of the target variable. Based on the confusion matrix, we can derive values such as Accuracy, Precision, Recall, and F_1 [140]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{EP} + \text{TN} + \text{FN}}, \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{3}$$

$$F_1 = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})}. \tag{4}$$

Accuracy is defined by equation (1). This indicates the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the total number of predictions. However, problems arise owing to Accuracy paradox for predictive analysis. Therefore, it can be checked by Precision and Recall indicators that evaluate whether the "negative" ratio of real data provides the proper classification of situations that will occur with sparse possibilities, as depicted in Figure 3. Precision is defined in

equation (2). It is the number of correct positive results divided by the number of positive results predicted by the classifier. Precision is also known as the positive predictive value (PPV). Recall is defined by equation (3), which is the number of correct positive results divided by the number of all relevant samples. The Recall is also known as the true positive rate (TPR). Recall and Precision are indicators of the opposite concepts. The Accuracy of the model can be supplemented by checking the F_1, the harmonic mean of Precision and Recall. F_1 is defined by equation (4). Harmonic means is used to understand the model's performance by balancing both indicators when either of the Precision or Recall indicators is low or near zero.

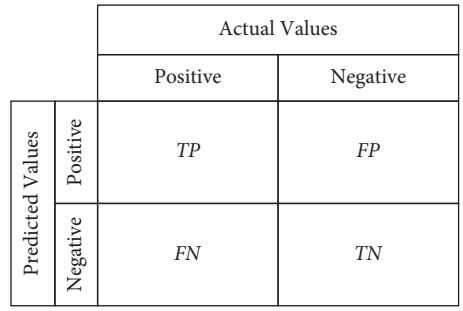

Figure 3: The confusion matrix.

3. Systematic Literature Collection

The methodology of our study is based on a structured literature review from 2016 to 2021, grounded in a systematic and method-based approach. The purpose of the systematic literature review was to assess the current research. The steps in the systematic literature review method are as follows: (1) research questions, (2) search process, (3) inclusive and exclusive criteria, (4) quality assessment, (5) data collection and data analysis, and (6) deviations from the protocol.

3.1. Research Questions. Although the general purpose of our study can be summarized in the analysis of mobile malware detection based on machine learning, this objective is explained in five specific research questions to gain a detailed understanding of the topic. The main purpose of our research questions is to analyze the number of studies on detecting mobile malware based on methodology over a specific period of time. We should recognize the strengths and limitations of this field of study.

The research questions addressed by our study are as follows:

(i) *RQ1*. How many studies on mobile malware and machine learning exist in the journal databases from 2016 to 2021?

(ii) *RQ2*. What research topics and types of mobile malware are being addressed?

(iii) *RQ3*. What are the limitations of the current research?

(iv) *RQ4*. What are the benefits and drawbacks of using and applying the methodology?

A specific period must be identified to help interpret how methodologies of mobile malware detection have evolved to address RQ1. The specific topics, methodology, and key types that differentiate them were considered for RQ2. Regarding RQ3 and RQ4, we visualize the benefits and drawbacks of previous studies on mobile malware detection based on machine learning.

3.2. Search Process. We followed a systematic methodology to investigate relevant studies that address subjects pertaining to the detection of mobile malware based on machine learning. According to Jamaluddin et al. [141]., However, we limit our review to studies from 2016 to 2021, as mobile malware detection has shown exponential growth during the past few years. The search process involves an outline of the most relevant bibliographic sources and search keywords.

A systematic research resource analysis was conducted using a search strategy. The main emphasis was on the detection of mobile malware. Depending on the research questions and the proposed theme, we present search queries that were used to identify the research for consideration. We entered queries for searching good quality research studies, including "Mobile," "Android," "Malware," "Detection," and "Machine Learning," which were nominated as the main keywords. We applied the Boolean operation, which uses conjunctions to combine or exclude queries in a search. The Boolean operators are OR, AND, and NOT, which are logical operators used to connect the search queries.

3.3. Inclusion and Exclusion Criteria. We used the queries ("Mobile" AND "Malware") OR ("Mobile" AND "Detection") OR ("Mobile" AND "Machine Learning") OR ("Android" AND "Malware") OR ("Android" AND "Detection") OR ("Android" AND "Machine Learning") independently on four databases (allintitle query) to gather the research papers at first. Many studies have been conducted on the detection of mobile malware as supervised learning. Therefore, we added the following additional queries to collect the specific research (allintext query): ("Mobile" AND "Malware" AND "Detection" AND "Supervised Learning") OR ("Mobile" AND "Malware" AND "Detection" AND "Unsupervised Learning") in Association for Computing Machinery (ACM) Digital Library and the Institute of Electrical and Electronics Engineers (IEEE) Xplore. Digital Bibliography and Library Project (DBLP) does not provide advanced search options; therefore, we excluded DBLP from the second search.

We excluded digital libraries, such as the Web of Science or Scopus, to intensively analyze papers in computer science [119]. Because the use of either Web of Science or Scopus for research evaluation may take biases that favor Natural Sciences and Engineering and Biomedical Research to the detriment of Social Sciences, we concentrated research on computer science. Our research papers found on Google

Scholar were excluded, because of duplication with papers found in other databases. We used various keywords to yield the most inclusive results. However, duplicate results were found among different databases. A substantial number of studies were duplicated from the Google Scholar database [142]. Therefore, we removed the results from the screening process. We screened for studies that focused on malware detection methods that use machine learning for the mobile environment. Furthermore, we removed publications that were not peer-reviewed.

The Preferred Reporting Items for Systematic Reviews and Meta-Analyzes (PRISMA) flow diagram is shown in Figure 4. A total of 12,587 studies were included in the search strategy. The inclusion and exclusion criteria were applied, as described above, to reduce their number to 154. A total of 154 studies were considered for the detection of mobile malware based on machine learning.

3.4. Quality Assessment. The journals to that the studies belonged were analyzed to determine if they were indexed in the Journal Citation Report (JCR) to assess the quality of the obtained literature results.

The used quality criteria were based on four quality assessment (QA) questions:

(i) *QA1*. Are the reviews on inclusion and exclusion criteria in the literature on mobile malware detection described and appropriate?

(ii) *QA2*. Is the literature search likely to have covered all relevant studies?

(iii) *QA3*. Did the reviewers assess the quality/validity of the included studies?

(iv) *QA4*. Were the basic data/studies adequately described?

The results of our research to the above quality evaluation question are as follows:

(i) *QA1*. Y(yes), the inclusion and exclusion criteria are explicitly defined above

(ii) *QA2*. Y(yes), we either searched four or more digital libraries and included additional search strategies or identified and referenced all journals addressing the topic of interest

(iii) *QA3*. Y (yes), we explicitly defined quality criteria as an index of peer-reviewed publications and extracted them from each primary research

(iv) *QA4*. Y (yes), information is presented regarding each research

3.5. Data Collection and Analysis. For our study, we used top research repositories as the main source to identify studies. Our study used four databases: DBLP, ACM Digital Library, IEEE Xplore, and Google Scholar. The reasons for selecting each database are as follows: the DBLP provides an index of peer-reviewed publications in computer science and trends in the publication scenario. However, the DBLP does not provide advanced search options. The ACM Digital Library is well known for the Turing Award, and the digital library primarily focuses on studies pertaining to the fields of computer science. The IEEE Xplore provides access to technical literature in electrical engineering, computer science, and electronics [143]. Google Scholar is the largest database of scholarly documents and accommodates approximately 100 million documents [144].

We selected studies that focused solely on detecting mobile malware using machine learning techniques and mobile malware classifiers. For a comprehensive understanding of mobile malware, we also benefited from real mobile malware samples. For this purpose, we collected six datasets, as shown in Table 1: (i) MalGenome [145], (ii) Drebin [146], (iii) M0Droid [147], (iv) CICMalDroid 2020 [149], (v) AndroZoo [150], and (vi) Android malware dataset [151]. Drebin used a known program to learn detection models based on static analysis. Therefore, it is essential to evaluate the number of samples in a family known to reliably detect this family. Furthermore, the presence of obfuscated or dynamically loaded malware on mobile devices cannot be ruled out.

MalGenome [145] is a dataset consisting of 1,260 Android malware samples, as listed in Table 1. Zhou et al. [145] collected 1260 Android malware samples from 49 different families and systematically collected them from various aspects, such as installation methods, activation mechanisms, and delivered malicious payloads. They performed a timeline analysis of findings based on collected malware samples and characterized them based on detailed behavior analysis, including installation, activation, and payload.

DREBIN [146] is a lightweight method that can automatically infer detection patterns and identify malware directly from smartphones. This methodology performs a comprehensive static analysis to extract feature sets from various sources and analyzes them in expressive vector space. This process first statically examines the Android application and extracts feature sets from the manifest and dex code of the application. They then geometrically analyzed the patterns and combinations of the features by matching the extracted feature sets to a joint vector space. This method used SVM techniques to identify malware by embedding a learning-based detection feature set. It should be noted that features contributing to malicious applications can be identified, and the detection process can be presented to users.

M0Droid [147] is an Android antimalware solution that analyzes system calls from Android applications on servers and generates signatures that are pushed to user devices for threat detection. The mobile malware detection model M0Droid uses behavioral attributes, such as file read requests or network access, to generate unique app signatures and uses signature normalization techniques. They proposed a solution to analyze and detect malware through behavior analysis and pattern recognition techniques with two categories of samples: malware and goodware datasets. M0Droid contains 1,530 malware samples and 49 malware families, as listed in Table 1.

CICMalDroid 2020 [149] contains a sample of 17,341 data for five Android applications: Adware, Banking, SMS, Riskware, and Benign, consisting of static and dynamic

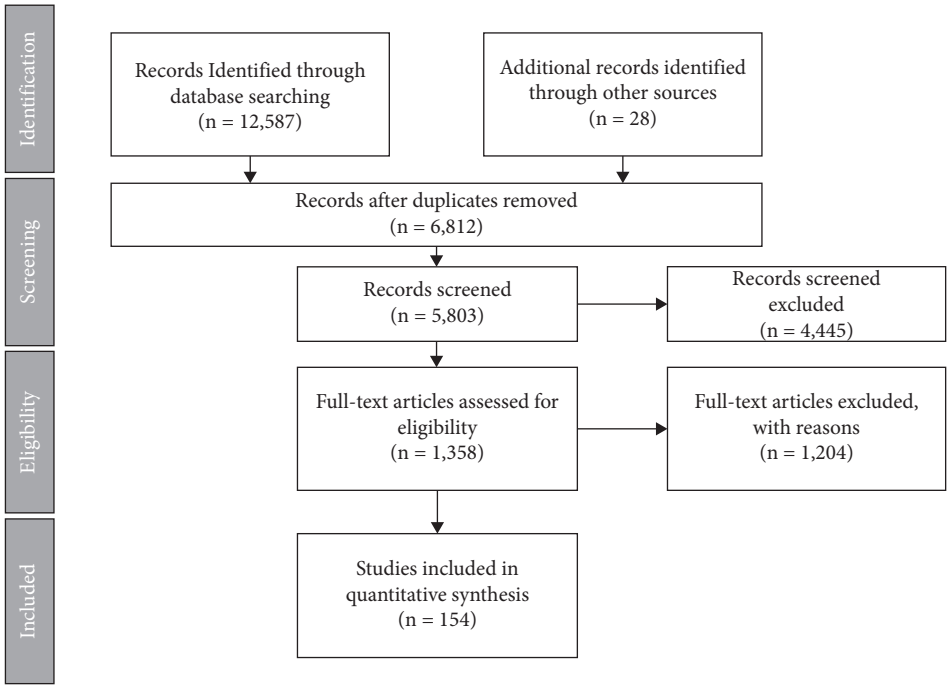

Figure 4: PRISMA flow diagram of the study.

Table 1: Mobile malware dataset.

Dataset	Date	Sample size	Malware family
MalGenome [145]	2011	1,260 malware	49
Drebin [146]	2012	5,560 malware	179
M0Droid [147]	2015	1,530 malware	153
CICMalDroid2017 [148]	2017	10,854 malware	42
CICMalDroid2020 [149]	2020	17,341 malware	191
AndroZoo [150]	2016	3,182,590 malware	Above 3,000
AMD [151]	2017	24,650 malware	71

features, as listed in Table 1. By collecting these data, Mahdavifar et al. [149] proposed an effective and efficient Android malware category classification system based on semisupervised deep neural networks. Although it contains a small number of labeled training samples, it can solve cost problems and efficiently specify categories of malware to help prioritize mitigation techniques. CICMalDroid 2017 [148] contains 10,854 samples, that contain 4,354 malware and 6,500 benign. These data were collected from the Google Play Market published in 2015, 2016, and 2017.

The AndroZoo dataset [150] contains more than 3,182,590 unique Android applications with a size of up to 20 TB or more, as listed in Table 2. These data demonstrate the importance of methodological problems and detection time when evaluating machine learning-based malware detector performance and detecting privacy leakage [150]. The Android dataset was collected from several sources, including the official Google Play Application Market, and currently includes 15,164,916 APKs. In addition, each APK was analyzed using different antivirus products to identify malware detection applications.

AMD datasets [151] were generated in 2016 with several malicious code samples. AMD datasets categorizes large datasets, including 24,650 malware app samples, into 135 variants belonging to 71 malware families [151] as listed in Table 2. This dataset groups malware samples with the same family of names and analyzes each family by classifying them into different variants using custom clustering. The AMD dataset performs a systematic and in-depth manual analysis of various malware samples to obtain behavioral information regarding malware.

The corresponding malware can be distinguished by classifying a dataset consisting of multiple malware families based on the characteristics of the manifest file called *AndroidManifest.xml*. Table 2 matches the features and feature sets of major malware families, such as FakeInstaller [152], DroidKungFu [153], GoldDream [154], and GingerMaster [155]. All Android applications must include *AndroidManifest.xml*, which provides data supporting the installation and later execution of the Android application. The information and data stored in this file can be efficiently retrieved on the device using the Android Asset Packaging Tool, which enables us to extract the sets, as listed in Table 2.

3.6. Deviations from the Protocol. Our systematic literature review was analyzed using the systematic approach explained above. The period for the research extraction is mainly from 2016 to October 2021, as the use of machine learning methodology for mobile malware detection has increased significantly owing to recent advances in artificial

Table 2: Mobile malware dataset and feature.

Malware family	Features	Feature set
FakeInstaller [152]	SendSMS	S7 suspicious API call
	SEND SMS	S2 requested permissions
	Android.hardware.telephony	S1 hardware components
	SendTextMessage	S5 restricted API calls
	READ PHONE STATE	S2 requested permissions
DroidKunFu [153]	SIG STR	S4 filtered intents
	System/bin/su	S7 suspicious API call
	BATTERY CHANGED ACTION	S4 filtered intents
	READ PHONE STATE	S2 requested permissions
	GetSubsriberId	S7 suspicious API call
GoldDream [154]	SendSMS	S7 suspicious API call
	Lebar.gicp.net	S8 network addresses
	DELETE PACKAGES	S2 requested permissions
	Android.provider.Telephony.SMS-RECEIVED	S4 filtered intents
	GetSubsriberId	S7 suspicious API call
GingerMaster [155]	USER PRESENT	S4 filtered intents
	GetSubsriberId	S7 suspicious API call
	READ PHONE STATE	S2 requested permissions
	System/bin/su	S7 suspicious API call
	HttpPost	S7 suspicious API call

intelligence and 5G. We attempted not to deviate from the protocols to minimize bias and other factors affecting the review study. However, there are worthy papers that were not reviewed in this study. The aforementioned papers were excluded since they were not available in the research databases that we used. We extracted the research papers written in English when searching for the research. Our study may have overlooked worthy papers written in other languages as a limitation.

4. Mobile Malware Detection

Previous systematic reviews have discussed mobile malware detection technology and methods to improve mobile security [156–162]. Feizollah et al. [156] reviewed 100 papers from 2010 to 2014, concentrating on the features of mobile malware detection. This review classified the available features into four categories: static, dynamic, hybrid, and application metadata. However, this review did not explain systematic research collection procedures. Senanayake et al. [157] conducted a systematic literature review. However, they analyzed papers by classifying them into static, dynamic, and hybrid analysis functions. According to Aslan et al. [158], known malware performs well with signature and heuristic-based detection approaches, whereas unknown and complex malware performs better with model inspection and cloud-based approaches. They made an insightful contribution where they attempted an approach to respond to a zero-day attack. However, there is no mention of the methods of mobile attacks. Most papers have studied methods for detecting mobile malware by classifying them into static, dynamic, and hybrid analyses. Analysis that classifies the analysis of machine learning-based methods into static, dynamic, and hybrid analyses is a limitation of existing studies. We conducted a review focusing on an attacker's means of attack or the goals of mobile malware. Therefore, our study greatly contributes to leading mobile cybersecurity through mobile malware detection techniques in a novel aspect that is different from existing studies.

4.1. Mobile Malware Attacks in Cybersecurity. Disruptive mobile malware threats regarding cybersecurity include stealing and leaking users' information, infringing on network security through botnet attacks, mobile banking attacks, ransomware attacks, and adware attacks [163]. This article categorizes the types of mobile malware attacks from a cybersecurity perspective into three categories: privacy leakage, banking and credit information leakage, and mobile system destruction.

4.1.1. Privacy Leakage. Information leakage in hardware-based attacks traverses data, helping mobile malware access secure repositories and eavesdropping privacy without notice [163, 164]. For example, there is intercepting mobile smart card communication and sensor sniffing. In a software-based mobile malware attack, information leakage is the monitoring and stealing of personal user information and transferring it to a malicious C2 server. In this process, malicious code conspires with other apps to build mobile app offerings and stealing data through inter-app communication, such as data exchange [164].

4.1.2. Banking and Credit Information Leakage: Mobile Banking and Ransomware. Mobile banking is an online banking system in which transactions are conducted through mobile applications. Mobile malware causes damage, such as stealing bank accounts or leaking credit card information in the mobile banking sector. For example, among mobile banking malware, banking Trojans collects personal user information and necessary credentials, stores

them in unrecognized sectors, and uploads the information collected to the C2 server when network connectivity becomes available, that is, intercepting bank-to-mobile validation information and digital certificates [164].

Ransomware is a form of malicious software that infects computer systems, restricts access, and demands a ransom. The number of ransomware attacks on the mobile devices has drastically increased during the past decade. In 2015, "*Trojan-Ransom.AndroidOS.Pletor*" mobile ransomware appeared, which, when installed, obtained root privileges and placed itself in the system document [164].

4.1.3. Mobile System Destruction: Botnet Attacks. A botnet is a software program designed to provide attackers control over the operation of infected devices without the user's consent. The bot is part of a botnet composed of multiple computers to be controlled by the botmaster, which has evolved into a severe cybersecurity threat [163]. Mobile hackers also collect and group compromised mobile devices to share attack payloads [164]. Cyberattacks are performed by generating botnets, such as spamming, information theft, server overload, billing fraud, and APT attacks.

4.2. Means of Mobile Malware. Mobile devices consist of an operating system, middleware, user interface, and software stack. Prior to the examination of the technology suitable detecting mobile malware, we must comprehend the structure of attack mobile malware. Attack targets were classified based on their mobile systems. Mobile malware attacks consist of servers (hosts), networks, client software, client hardware, and users [165]. The server's responsibility is to identify malicious behavior by comparing the behavior of newly installed applications with known traffic patterns. This is accomplished by aggregating reported data from various mobile devices and deriving a collaborative model representing the common traffic patterns of several users for each application. Alternatively, we identified malicious behavior with local models that were detected by analyzing the deviation of traffic patterns in installed applications [166]. Server- and host-based intrusion detection systems (IDSs) reside and monitor a single host system and collect and analyze events, such as file systems and system calls. Malware activities that have performed network overload attacks affect regular network behavior patterns; therefore, the activity of mobile malware can be detected by monitoring the network behavior of applications. Therefore, monitoring and analyzing traffic patterns in network-active applications is essential for developing practical solutions to prevent network overload [166]. Network-based IDSs collect and analyze traffic volumes, IP addresses, service ports, and forms of protocols to detect intrusion attempts. The responsibility of client software is to monitor applications already installed and running on mobile devices, teach user-specific local models, and detect deviations from observed normal behavior. Furthermore, the client software learns a local model to determine indicators, such as changes in users' behavior and updates resulting from new versions or malicious attacks to detect changes in the traffic patterns of applications [166].

4.3. Challenge of Mobile Malware Detection in Cybersecurity

4.3.1. Accuracy of Malware Detection. The accuracy in malware detection on mobile devices is based on static or dynamic analysis. Among mobile malware detection methods based on machine learning techniques, static analysis goes through the classification and selection process of sample data being detected and completes modeling after a learning process. After numerous tests, the best-performing prediction model was selected based on which new and variant malicious mobile malware was detected. Dynamic analysis can detect zero-day attacks or threats using self-learnable behavior analysis techniques. It is a method of learning and training user interaction tracking and the behavior of dynamic applications on mobile devices without server judgment. We first obtained information from real-world applications before the learning phase and feature values for each application labeled as malicious and normal. After generating the model through external learning tools, malicious behaviors performed based on the generated predictive model can be detected in real time and blocked according to each action-specific risk information.

4.3.2. Zero-Day Attack. Attackers do not solely focus on the mobile malware. Attackers find loopholes in existing mobile applications and vulnerabilities in the source code. Therefore, vulnerabilities in programs can occur due to mistakes during design and development and are often misused. Many antivirus products utilize signature-based detection methods that use signature-based methods to detect mobile malware. However, signature-based detection methods are complicated in identifying zero-day malware if an attacker is working on obfuscation.

4.3.3. Adversarial Training. Adversarial training can increase cybersecurity capabilities through training, which can be defended through aggressive and malicious attacks based on attack scenarios. Malware avoids the detection of anti-malware engines using obfuscation technology to deceive defenders. Leveraging adversarial machine learning is a method for solving behavior in which malware disguises itself as a problem-free positive feature representation. Adversarial attacks refer to security risks that can arise in adversarial environments owing to the vulnerabilities inherent in machine learning algorithms. Adversarial attacks that attack the confidentiality and integrity of information security in a series of machine learning processes include addiction, avoidance, model extraction, and learning data extraction attacks. Addiction attacks refer to breaking machine learning models by injecting malicious learning data, and avoidance attacks are deceiving machine learning models by disrupting data in the inference process of

machine learning models. Model extraction attacks and learning data extraction attacks refer to attacks that use reverse engineering to steal machine learning models or learning data. Adversarial training is a suitable method to defend against adversarial attacks. Adversarial machine learning refers to a technique used to deceive machine learning by automatically generating adversarial examples [167], which involves deliberately manipulating data fragments to induce machine learning models to make false predictions. It is a method to improve the resistance of machine learning by inputting predictable hacked data during the training phase of the machine learning process.

4.4. Techniques of Mobile Malware Detection. Traditionally, mobile malware detection technologies are signature-pattern DB-based detection technologies or cloud server-based detection technologies. The former consists of a structure in which clients check for malware using information supplied from the server by reflecting and detecting the results of analysis on the already distributed code in the database. The latter refers to a centralized inspection method that sends application information installed on client's device to the cloud server to determine whether it is malicious. This method has the advantage of not receiving DB from a special client-server every time and can reflect the analyzed results in real time.

These two mobile malware detection technologies have limitations in detecting and responding to mobile malware. Signature-pattern DB-based detection techniques have limitations that make detecting novel and variant malicious apps difficult. Mobile applications are always vulnerable to risks, as they present. Due to the recent upsurge of zero-day attacks and malicious attacks, mobile applications are always susceptible to various cyber threats. Yet, zero-day attacks on the cloud-based server are difficult to detect. Furthermore, the infrastructures' initial and operational costs are costly.

The machine learning-based detection algorithms seem suitable for performing static analysis for the malware. Malware detection in Android can be performed using a signature- and behavior-based detection methods. Yet, signature-based detection methods are unlikely to detect zero-day attacks, and behavior-based or anomaly-based detection methods are mainly used. Behavior-based detection methods use machine learning methods. To use machine learning, we must analyze the APK to extract features. Three techniques can be used to extract features: static, dynamic, and hybrid analyses. Static analysis does not run on a mobile device, but analyzes byte code and source code. Static analysis is less risky than dynamic analysis, since it does not use a runtime environment. On the contrary, dynamic analysis detects malware by analyzing the app through simulation. To collect information necessary to detect abnormal behavior in a mobile environment, a hybrid analysis method is required.

The malware classification phase during the process of mobile malware detection based on machine learning is an essential step in detection tasks. Malware-derived files are transformed into vectors used as training datasets in machine learning algorithms in the malware classification phase. The data used as training datasets were analyzed appropriately among the algorithms in machine learning and then inserted into the classification process. These methods analyze the training dataset to detect whether it is malicious or positive file.

Malware detection in Android can be performed using a signature- and behavior-based detection methods. However, signature-based detection methods are unlikely to detect zero-day attacks, and behavior-based or anomaly-based detection methods are primarily used. Behavior-based detection methods use machine learning methods. It is appropriate to detect known malicious codes using existing studies; however, it is difficult to immediately detect and respond to variants or new malicious codes. Therefore, machine learning-based detection technology is being studied for detecting and responding to variants or new malicious codes.

5. Results and Discussion

We uncovered several findings across each research question from the systematic literature review, as discussed in the following:

> *RQ1*. How many studies on mobile malware and machine learning exist in journal databases from 2016 to 2021?

A number of papers selected for this study are shown in Figure 5, which presents the studies selected for this review by the year of publication. We mainly focus on papers published between 2016 and 2021. We cover research appearing up to pre-2016, which is highly cited. We categorized and analyzed the literature related to machine learning-based mobile malware detection by year. The literature related to our study increased significantly between 2016 and 2019. The number of related studies has increased over the years; 250% increase from 2016 to 2017, 10.7% increase from 2017 to 2018, and 16.1% increase from 2018 to 2019. This could be due to the increasing use of mobile attacks. The number of related studies in 2021 is expected to increase beyond 2020. We confirm that machine learning is one of the methodologies for detecting mobile malware, which must be studied in the future:

> *RQ2*. What research topics and types of mobile malware are being addressed?

We conducted a literature study on machine learning-based mobile malware detection. We investigated a total of 154 machine learning-based mobile malware detection literature and conducted a frequency survey of the most frequently mentioned words, as shown in Figure 6. The most frequently mentioned word is "Malware" mentioned 14,017 times. The second most frequently mentioned word is "Application," which has been mentioned 13,210 times. The third most frequently mentioned word is "Feature," which has been mentioned 12,475 times. The fourth most frequently mentioned word is "Android," which has been mentioned 9,773 times. The fifth most frequently mentioned word is "Detection," which has been mentioned 8,764. The

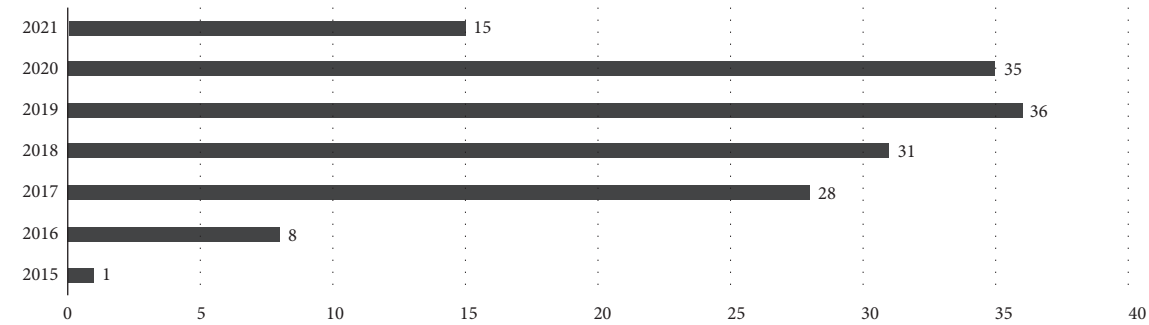

Figure 5: Number of papers related to mobile malware detection-based machine learning by year.

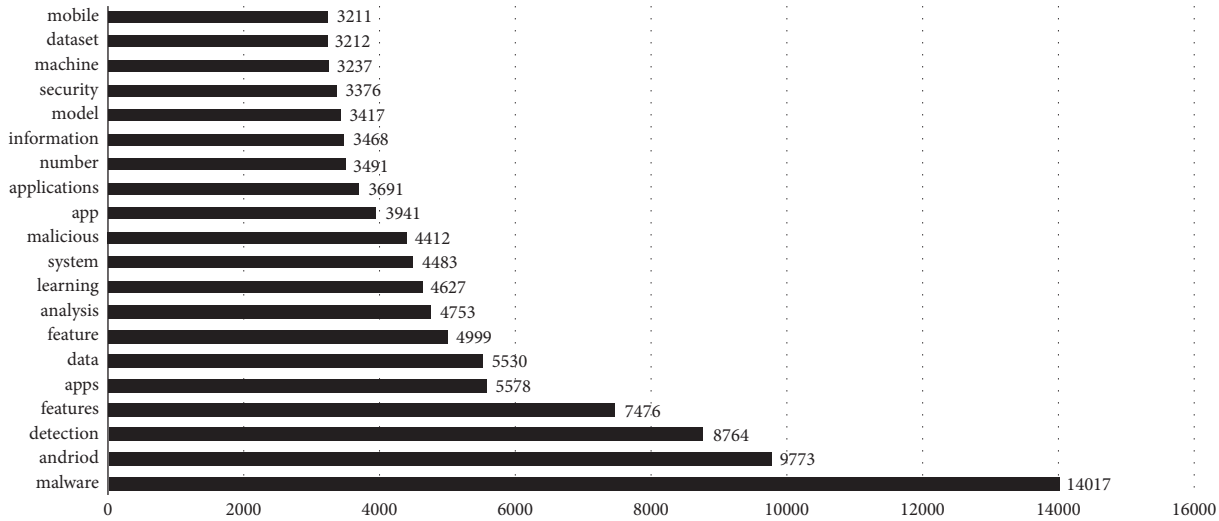

Figure 6: Distribution of the papers based on mobile malware detection.

following most frequently mentioned words are "Data," "Analysis," "Learning," "System," and "Malicious." In the analysis of our research, we list the top five words in the following order: "Malware," "Application," "Feature," "Android," and "Detection" are queries we must use to collect literature.

Using the evaluation metrics, Accuracy, Precision, and Recall were the most widely extracted evaluation metrics for machine learning. This is shown in Figure 7, along with the other extracted evaluation metrics in our literature review studies. Most of the evaluation metrics for machine learning analysis extracted the accuracy. In other words, most studies have focused on the accuracy of mobile malware detection. Improving the accuracy is important for mobile malware detection researchers. Receiver operating characteristic (ROC) or area under the ROC curve has low usage in evaluation metrics.

The dataset provides the input values of the model algorithm for generating the learning model. Therefore, the datasets used in machine learning must be reviewed and validated in advance. Proper dataset preparation highlights patterns of mobile malware, improves performance, and produces higher quality output values [168]. The datasets used in machine learning-based mobile malware detection studies to train the algorithms are shown in Figure 8. Google Play and Drebin are the most widely used datasets for mobile malware detection. The reason for the highest usage of the Google Play dataset is that it provides the largest data used to make an optimal model and can be utilized for a real dataset. Genome/MalGenome, VirusShare, and AndroZoo are other widely used datasets.

RQ3. What are the limitations of the current research?

We classified the detection method into supervised or unsupervised learning, as shown in Table 3 and Figure 9. Mobile malware attacks can be divided into five methods: (i) server, (ii) network, (iii) client software, (iv) client hardware, and (v) user.

We identified that the client software and client hardware have a higher percentage of studies than servers and networks. While we were not able to find many studies based on supervised learning that focused on server and network attacks, we were able to identify the studies that focused on the client software and client hardware. Chen et al. [34] proposed a new S-IDGC model that allowed users to fairly compare different types of classifiers by designing a comparative benchmark prototype system that integrated different types of machine learning classifiers for Android malicious traffic detection. This model referred to imbalance classification methods, including the synthetic minority

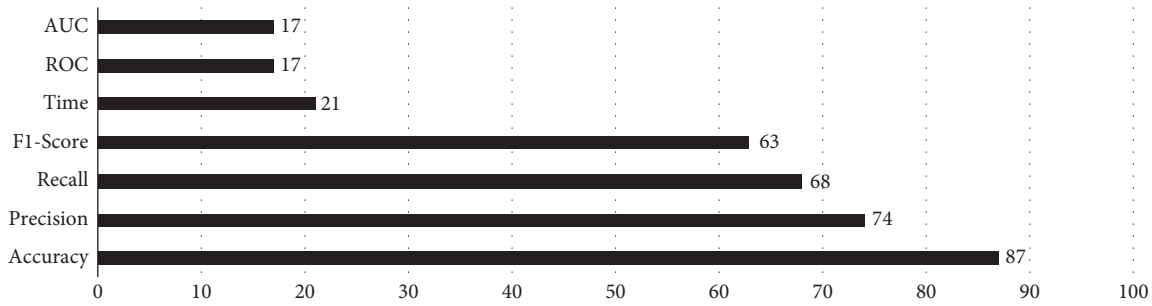

Figure 7: Evaluation metrics of machine learning models used in the review research studies.

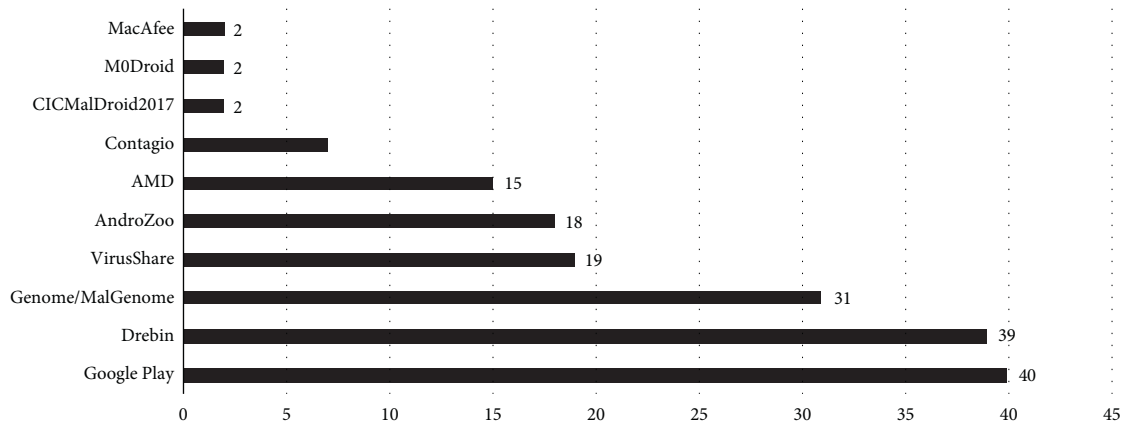

Figure 8: Usage of datasets for machine learning models used in the review research studies.

Table 3: Classification of the supervised and unsupervised learning by means of mobile malware attack.

Means of attack	Supervised learning	Unsupervised learning
Server	[46, 169]	–
Network	[34, 38, 40, 41, 50, 75, 93, 102, 107, 109, 120, 122, 123, 131, 134, 169–173]	[107]
Client software	[32, 35, 37, 42–45, 47–49, 51–74, 76–85, 88–92, 94–101] [103, 105, 106, 108–111, 121, 125–127, 130, 132, 133] [170, 174–196]	[47, 56, 63, 71, 81, 92, 97, 98, 100, 111, 184]
Client hardware	[33, 36, 39, 77, 86, 87, 124, 128, 129, 197–199]	[139, 200]
User	[104]	[201]

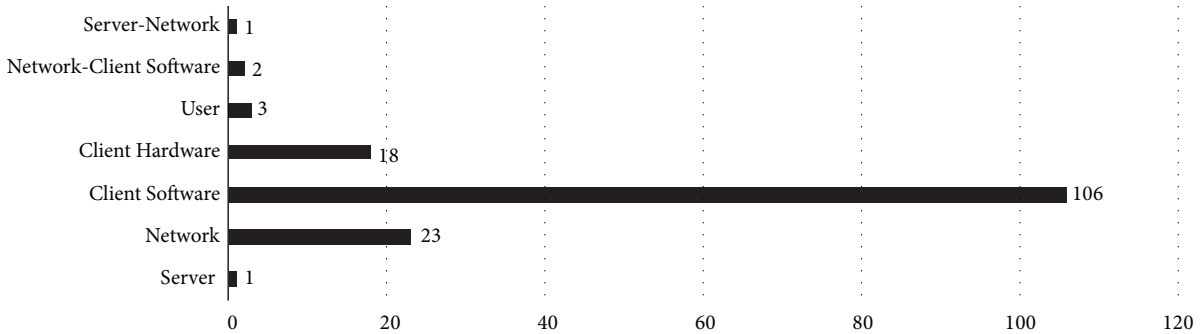

Figure 9: Means of attack used in the review research studies.

oversampling technique (SMOTE), SVM, SVM cost-sensitive (SVMCS), and C4.5 cost-sensitive (C4.5CS) methods. This model allowed users to compare the detection performance of different classification algorithms on the same dataset with the performance of a specific classification algorithm on different datasets. This study aimed at evaluating IDS performance using five classifiers: J48, DB, MLP, KNN, and RF [170]. The study assessed both the MalGenome and private datasets and identified that the BN and RD scored 99.7% and 93.03%, respectively, for the TPR. Egitmen et al. [39] used Android software with artificially generated text to classify modern Android malware, but applied a skip-gram technique configured for NLP to extract useful features. This study also demonstrated that the NLP-based static analysis approach for application source code has promising results. In conclusion, the accuracy was 95.64% without threatening system stability while running the target application.

While we were not able to find any study that utilized unsupervised learning that focused on server and network attacks, we were able to identify the studies that focused on the client software [56, 81, 185, 202] and client hardware [139, 200]. Amara et al. [81] improved abnormal-based detection technology by examining two factors that caused low accuracy in detection technology. This study extracted the main behavior of the application using a system called the filtering and abstraction process and characterized benign behavior using a machine learning classifier. This study also confirmed that the filtering and abstraction processes had a positive impact on the performance of the SVM and K-means models. Xu et al. [56] applied feature weights based on IG and PSO methods to measure the importance of features for machine learning classification. The proposed strategy achieved the highest accuracy in the machine learning model by increasing cluster diversity. Wu et al. [202] decomposed Android apps into manifest, Dalvik code, and basic library files to detect maliciousness and identify their families to analyze and classify Android malware applications efficiently. Therefore, MVI-Droid obtained an F1 score of 0.99 and an F1 score of 0.948 in multiple classifications. Cai et al. [63] proposed an Android malware detection technique called JOWMDroid, which is a static analysis based on feature weights and joint optimization of weight mapping and classifier parameters [77]. While majority of machine learning-based classifications provide a binary label for mobile users and app security analysts, there have been few to no studies that examine malicious behaviors for mobile applications. As such, XMAL was proposed to classify malware with high accuracy [202]. RevealDroid, a machine learning-based malware detection method, analyzed RevealDroid, a dataset that consisted of 54,000 malicious and benign apps [203]. The detection method used various features, such as Android API usage, reflection-based features, and features from native binaries of apps.

Systematic literature on mobile malware detection from the security perspective revealed three limitations: limited dataset, zero-day attack, and evaluation algorithms. There is a lack of good quality or a lack of diversity in the dataset used for the analysis of mobile malware detection. Wang et al. [122] proposed C4.5, a machine learning algorithm to identify Android malware, to achieve better detection rates in comparison with other detection approaches. The study also utilized the Drebin dataset and analyzed 8,312 benign apps and 5,560 malware samples. However, the limitation of this study is that the number of training data is small, and there are many unexamined features by analyzing only six TCP flow characteristics and four HTTP request characteristics. We must design methods that can detect zero-day attacks, which are not just those observed in the past. By analyzing the system calls of mobile applications called for a 1 s time with a host-based approach to detect mobile botnets and using induction machine learning models, Costa et al. [33] achieved high performance across different metrics. In addition, this study identified that reducing the dimensionality of the problem from 133 to 19 did not have a significant negative impact on performance. However, this study required new challenges to identify mobile botnets in real time and more diverse scenarios using multiple mobile devices. Sharma et al. [200] did not present a specific mechanism for detecting types of Android malware that antivirus software could not detect. The choice of algorithms is very important for detecting mobile malware. Mahindru et al. [46] examined the privilege-induced risk initiated by granting unnecessary privileges to these Android applications and utilized the least square support vector machine (LSSVM) learning approach linked through three unique kernel features: linear, radial basis, and polynomial. However, the malware detection model proposed in this study had a limitation in detecting only whether an application was malware or benign. Rasheed et al. [169] tested SMO, random tree, J48, naïve Bayes (NB), and LMT algorithms, and by following the best result to classify, the botnet attack was 85%. However, this study must improve algorithm classification by adding new subalgorithms to machine learning. Cai et al. [63] did not consider the correlation between features. Therefore, it was necessary to build joint features to improve the detection accuracy of malware in Android applications:

> *RQ4*. What are the benefits and drawbacks of using and applying the methodology?

The distribution status of the machine learning algorithms used for mobile malware detection was examined. SVM, random forest (RF), and NB were used as the top of the algorithm of widely studied machine learning models for mobile malware detection. Supervised learning algorithms are more popular than unsupervised learning algorithms. Figure 10 shows all the studied machine learning models used in the reviewed studies. Since the resource cost for the unsupervised learning algorithm to execute the model is high.

The SVM can be used for categorical or numerical prediction problems and is less likely to overfit with no effect on the error data. However, SVM requires multiple combinatorial tests to determine the optimal model, and the learning rate is slow. RF does not require data normalization but requires high computational power. NB can be predicted easily and quickly in multiclass

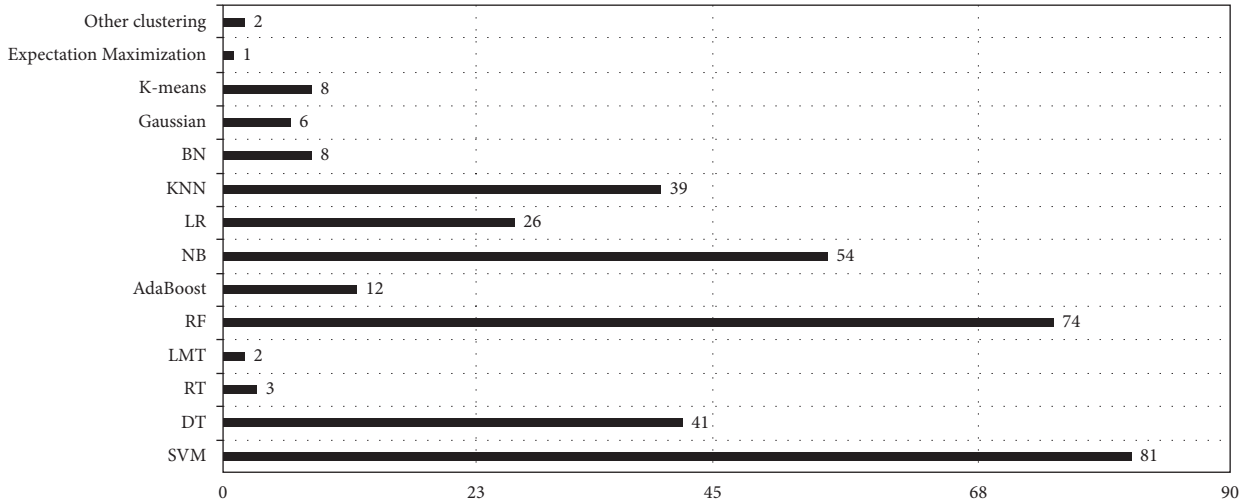

Figure 10: The algorithms used for mobile malware detection.

classification and requires less training data. However, other learning methods should be applied or considered where the assumption of independence does not hold. Overall, supervised learning algorithms have the advantage of providing labels along with the data when tested, making it easy to evaluate the performance of the model. In contrast, the purpose of unsupervised learning is to identify patterns in the data itself, and only data without labels are required. Although a label is not required, it is difficult to evaluate model performance when the label is not provided. Therefore, to study mobile malware detection, it is necessary to analyze the characteristics of data and algorithms in an optimal manner.

6. Conclusions

We examined the literature pertaining to machine learning-based mobile malware detection in cybersecurity. Our study focused on subjects, such as mobile system destruction and information leaks. We explored the mobile malware detection techniques utilized in recent studies based on attack intentions, such as server, network, client software, client hardware, and user. We identified several perspectives for future study. Our review can be utilized in future studies on these topics. Furthermore, we reviewed and summarized security challenges related to cybersecurity that could lead to improved and more practical planning. We aimed at reinvigorating research on these issues and reorienting the practical needs of cybersecurity domains. We performed a comprehensive examination of the previous literature on machine learning-based mobile malware detection in terms of unique requirements in cybersecurity domains.

The number of studies on the detection of machine learning-based mobile malicious code is steadily increasing. When analyzing the existing literature for the detection of mobile malware, researchers should refer to "queries used," "evaluation indicators," and "datasets," which are the analysis results for RQ2. In addition, in contrast to other SLR studies, our study classified the means of attack as supervised and unsupervised learning. Many studies have been conducted on clients, and there have been more studies using supervised learning algorithms than unsupervised learning. In other words, it can be observed that attackers have identified more vulnerabilities in software or hardware used by clients than in terms of servers or networks among means of attack and that this section is vulnerable to cybersecurity. In addition, it can be observed that supervised learning algorithms are more effective in detecting Android mobile malware than unsupervised learning algorithms. Therefore, researchers should conduct numerous detection studies in the client section using supervised learning algorithms to improve the cybersecurity of mobile devices.

To the best of our knowledge, a comprehensive evaluation of the adequacy of previous studies on machine learning-based mobile malware detection from a cybersecurity perspective has not been performed. We believe that the comprehensive evaluation from our study can help provide a foundation for future researchers to help underpin larger research projects. We were able to compare supervised learning and unsupervised learning detection methods for mobile malware. By synthesizing the existing data, we believe we were able to provide relevant insights for future researchers.

Data Availability

The data of analysis research papers used to support the findings of this study have been deposited in the GitHub repository (https://github.com/yukyungkim94/A-SLR-on-the-Mobile-Malware-Detection-Methods).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by a Korea Universitygrant (K2106341). This study was supported by a grant-in-aid of HANWHASYSTEMS.

References

[1] M. Talal, A. A. Zaidan, and B. B. Zaidan, "Comprehensive review and analysis of anti-malware apps for smartphones," *Telecommunication Systems*, vol. 72, no. 2, pp. 285–337, 2019.

[2] D.R. Thomas, A.R. Beresford, and A. Rice, "Security metrics for the android ecosystem," in *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, pp. 87–98, Denver, CO, USA, 2015.

[3] J. Jang-Jaccard and S. Nepal, "A survey of emerging threats in cybersecurity," *Journal of Computer and System Sciences*, vol. 80, no. 5, pp. 973–993, 2014.

[4] B. Martínez-Pérez, I. De La Torre-Díez, and M. López-Coronado, "Privacy and security in mobile health apps: a review and recommendations," *Journal of medical systems*, vol. 39, no. 1, pp. 1–8, 2015.

[5] G. Gui, Miao Liu, and Nei Kato, "6G: Opening new horizons for integration of comfort, security, and intelligence," *IEEE Wireless Communications*, vol. 27, no. 5, pp. 126–132, 2020.

[6] S.-M. Cheng, Pin-Yu Chen, Ching-Chao Lin, and Hsu-Chun Hsiao, "Traffic-aware patching for cyber security in mobile IoT," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 29–35, 2017.

[7] Y.-K. Kim, Jemin Justin Lee, and Myong-Hyun Go, "Analysis of the Asymmetrical Relationships between State Actors and APT Threat Groups," in *Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC)*, IEEE, Jeju, Korea (South), October 2020.

[8] R. Langner, "Stunt: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[9] G. Lykou, A. Anagnostopoulou, and D. Gritzalis, "Smart airport cybersecurity: Threat mitigation and cyber resilience controls," *Sensors*, vol. 19, no. 1, p. 19, 2019.

[10] J.F. Carías and Leire Labaka, "Defining a cyber resilience investment strategy in an industrial internet of things context," *Sensors*, vol. 19, no. 1, p. 138, 2019.

[11] J.F Carías, Marcos R. S. Borges, and Leire Labaka, "Systematic Approach to Cyber Resilience Operationalization in SMEs," *IEEE Access*, vol. 8, pp. 174200–174221, 2020.

[12] M. Naseer and J. F. Rusni, "Malware Detection: Issues and Challenges," *Journal of Physics Conference Series*, vol. 1807, Article ID 012011, 2021.

[13] A. Narayanan, Liu Yang, and Lihui Chen, "Adaptive and scalable android malware detection through online learning," in *Proceedings of the 2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 2484–2491, Vancouver, BC, Canada, July 2016.

[14] Cisa, *What is Cybersecurity?*, 2009, https://us-cert.cisa.gov/ncas/tips/ST04-001.

[15] J.M. Anderson, "Why we need a new definition of information security," *Computers & Security*, vol. 22, no. 4, pp. 308–313, 2003.

[16] M. Kang, "The Study on the Effect of the Internet and Mobile-Cellular on Trade in Services: Using the Modified Gravity Model," *J. Internet Serv. Inf. Secur*, vol. 10, no. 4, pp. 90–100, 2020.

[17] D.J. Lemay, R.B. Basnet, and T. Doleck, "Examining the Relationship between Threat and Coping Appraisal in Phishing Detection among College Students," *J. Internet Serv. Inf. Secur*, vol. 10, no. 1, pp. 38–49, 2020.

[18] S.K Wong and S.-M. Yiu, "Location spoofing attack detection with pre-installed sensors in mobile devices," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl*, vol. 11, no. 4, pp. 16–30, 2020.

[19] I. Kholod, A. Shorov, and S. Gorlatch, "Efficient Distribution and Processing of Data for Parallelizing Data Mining in Mobile Clouds," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl*, vol. 11, no. 1, pp. 2–17, 2020.

[20] A. La Marra, Antonio La Marra, and Fabio Martinelli, "D-BRIDEMAID: A Distributed Framework for Collaborative and Dynamic Analysis of Android Malware," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl*, vol. 11, no. 3, pp. 1–28, 2020.

[21] B. Sim and D. Han, "A study on the side-channel analysis trends for application to IoT devices," *J. Internet Serv. Inf. Secur*, vol. 10, pp. 2–21, 2020.

[22] K. Dunham, *Mobile malware attacks and defense*, Science direct, 2008.

[23] I. Kaspersky Lab & INTERPOL Joint Report, *Mobile Cyber Threats*, 2014, https://media.kaspersky.com/pdf/Kaspersky-Lab-KSN-Report-mobile-cyberthreats-web.pdf.

[24] Vasileios Kouliaridis, "A survey on mobile malware detection techniques," *IEICE Transactions on Information and Systems*, vol. E103-D, pp. 204–211, 2020.

[25] A.L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

[26] R. Reed and R.J. MarksIi, *Neural smiting: supervised learning in feedforward artificial neural networks*, MIT Press, Cambridge, MA, 1999.

[27] G.E. Hinton and T.J. Sejnowski, *Unsupervised learning: foundations of neural computation*, MIT Press, Cambridge, MA, 1999.

[28] Y. Li, "Deep reinforcement learning: An overview," 2017, https://arxiv.org/abs/1701.07274.

[29] B.E. Boser, I.M Guyon, and V.N Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, New York; NY, US, July 1992.

[30] W.S. Noble, "Support vector machine applications in computational biology," *Kernel methods in computational biology*, vol. 14, pp. 71–92, 2004.

[31] W.S. Noble, "What is a support vector machine?" *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

[32] L. Gong, Hao Lin, and Zhenhua Li, "Systematically Landing Machine Learning onto Market-Scale Mobile Malware Detection," *IEEE Transactions on Parallel and Distributed Systems*, 2020.

[33] V.G.T.D. Costa, Victor G. Turrisi Da Costa, and Sylvio Barbon Junior, "Mobile botnets detection based on machine learning over system calls," *International Journal of Security and Networks*, pp. 103–118, 2019.

[34] Z. Chen, Norihiro Okui, and Yasuaki Kobayashi, "Machine learning based mobile malware detection using highly imbalanced network traffic," *Information Sciences*, pp. 346–364, 2018.

[35] M.Z. Mas' ud et al., "A Comparative Study on Feature Selection Method for N-gram Mobile Malware Detection," *IJ Network Security*, pp. 727–733, 2017.

[36] G. Nguyen, Binh Minh Nguyen, and Dang Tran, "A heuristics approach to mine behavioural data logs in mobile malware detection system," *Data \& Knowledge Engineering*, vol. 115, pp. 129–151, 2018.

[37] Q. Zhou, Bouchaib Falah, and Sameer Abufardeh, "A novel approach for mobile malware classification and detection in Android systems," *Multimedia Tools and Applications*, pp. 3529–3552, 2019.

[38] E.A. Shams and A. Rizaner, "A novel support vector machine based intrusion detection system for mobile ad hoc networks," *Wireless Networks*, vol. 2018, pp. 1821–1829, Article ID 6726147, 2018.

[39] A. Egitmen, Irfan Bulut, R. Can Aygun, and A. Bilge Gunduz, "Combat mobile evasive malware via skip-gram-based malware detection," *Security and Communication Networks*, vol. 2020, p. 10, 2020.

[40] E. Mugabo and Q.-Y. Zhang, "Intrusion Detection Method Based on Support Vector Machine and Information Gain for Mobile Cloud Computing," *IJ Network Security*, pp. 231–241, 2020.

[41] S. Sharmeen, Shamsul Huda, and Jemal H. Abawajy, "Malware threats and detection for industrial mobile-IoT networks," *IEEE access*, vol. 6, pp. 15941–15957, 2018.

[42] M. Park, "A Framework for Identifying Obfuscation Techniques applied to Android Apps using Machine Learning," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, pp. 22–30, 2019.

[43] I. Martn and José Alberto Hernández, "Android malware characterization using metadata and machine learning techniques," *Security and Communication Networks*, 2018.

[44] Y. Bai, Zhenchang Xing, and Duoyuan Ma, "Comparative analysis of feature representations and machine learning methods in Android family classification," *Computer Networks*, vol. 184, Article ID 107639, 2021.

[45] K. Allix and Bissyand, "Empirical assessment of machine learning-based malware detectors for Android," *Empirical Software Engineering*, vol. 21, pp. 183–211, 2016.

[46] A. Mahindru and A.L. Sangal, "FSDroid:-A feature selection technique to detect malware from Android using Machine Learning Techniques," *Multimedia Tools and Applications*, vol. 80, pp. 13271–13323, 2021.

[47] N. Milosevic, A. Dehghantanha, and K.-K.R. Choo, "Machine learning aided Android malware classification," *Computers \& Electrical Engineering*, vol. 61, pp. 266–274, 2017.

[48] Z.-U. Rehman, Sidra Nasim Khan, and Khan Muhammad, "Machine learning-assisted signature and heuristic-based detection of malwares in Android devices," *Computers \& Electrical Engineering*, vol. 69, pp. 828–841, 2018.

[49] J. Qiu, Wei Luo, and Lei Pan, "Predicting the impact of Android malicious samples via machine learning," *IEEE Access*, vol. 7, pp. 66304–66316, 2019.

[50] G. Kirubavathi and R. Anitha, "Structural analysis and detection of android botnets using machine learning techniques," *International Journal of Information Security*, vol. 17, pp. 153–167, 2018.

[51] A. Salah, E. Shalabi, and W. Khedr, "A Lightweight Android Malware Classifier Using Novel Feature Selection Methods," *Symmetry*, vol. 12, p. 858, 2020.

[52] P. Palumbo, Luiza Sayfullina, and Dmitriy Komashinskiy, "A pragmatic android malware detection procedure," *Computers \& Security*, vol. 70, pp. 689–701, 2017.

[53] M. Yang, Xingshu Chen, and Yonggang Luo, "An Android Malware Detection Model Based on DT-SVM," *Security and Communication Networks*, vol. 2020, Article ID 8841233, 11 pages, 2020.

[54] L. Massarelli, Leonardo Aniello, and Claudio Ciccotelli, "AndroDFA: Android Malware Classification Based on Resource Consumption," *Information*, vol. 11, p. 326, 2020.

[55] F. Alswaina and K. Elleithy, "Android malware permission-based multi-class classification using extremely randomized trees," *IEEE Access*, vol. 6, pp. 76217–76227, 2018.

[56] Y. Xu, Chunhua Wu, and Kangfeng Zheng, "Computing adaptive feature weights with PSO to improve Android malware detection," *Security and Communication Networks*, vol. 2017, Article ID 3284080, 14 pages, 2017.

[57] J. Zhang, Zheng Qin, and Kehuan Zhang, "Dalvik opcodes graph based android malware variants detection using global topology features," *IEEE Access*, vol. 6, pp. 51964–51974, 2018.

[58] V.D. Priya and P. Visalakshi, "Detecting android malware using an improved filter based technique in embedded software," *Microprocessors and Microsystems*, vol. 76, 2020.

[59] K. Tian, Danfeng Yao, and Barbara G. Ryder, "Detection of repackaged android malware with code-heterogeneity features," *IEEE Transactions on Dependable and Secure Computing*, pp. 64–77, 2017.

[60] A.K. Singh, C.D. Jaidhar, and M.A.A. Kumara, "Experimental analysis of android malware detection based on combinations of permissions and API-calls," *Journal of Computer Virology and Hacking Techniques*, vol. 15, pp. 209–218, 2019.

[61] G. Peynirci, M. Eminaaolu, and K. Karabulut, "Feature Selection for Malware Detection on the Android Platform Based on Differences of IDF Values," *Journal of Computer Science and Technology*, vol. 35, pp. 946–962, 2020.

[62] R. Surendran, T. Thomas, and S. Emmanuel, "GSDroid: Graph Signal Based Compact Feature Representation for Android Malware Detection," *Expert Systems with Applications*, vol. 159, Article ID 113581, 2020.

[63] L. Cai, Y. Li, and Z. Xiong, "JOWMDroid: Android malware detection based on feature weighting with joint optimization of weight-mapping and classifier parameters," *Computers \& Security*, vol. 100, Article ID 102086, 2021.

[64] M. Cai, Yuan Jiang, and Cuiying Gao, "Learning features from enhanced function call graphs for Android malware detection," *Neurocomputing*, vol. 423, pp. 301–307, 2021.

[65] P. Liu, Weiping Wang, and Xi Luo, "NSDroid: efficient multi-classification of android malware using neighborhood signature in local function call graphs," *International Journal of Information Security*, vol. 20, pp. 1–13, 2020.

[66] A. Christianah, B. Gyunka, and A. Oluwatobi, "Optimizing Android Malware Detection Via Ensemble Learning," vol. 14, 2020.

[67] A. Kumar, Vinti Agarwal, and Shishir Kumar Shandilya, "PACER: Platform for Android Malware Classification, Performance Evaluation and Threat Reporting," *Future Internet*, vol. 12, p. 66, 2020.

[68] R. Taheri, Meysam Ghahramani, and Reza Javidan, "Similarity-based Android malware detection using Hamming distance of static binary features," *Future Generation Computer Systems*, vol. 105, pp. 230–247, 2020.

[69] V. Kouliaridis and Georgios Kambourakis, "Two Anatomists Are Better than One—Dual-Level Android Malware Detection," *Symmetry*, vol. 12, 2020.

[70] A. De Lorenzo, Fabio Martinelli, and Eric Medvet, "Visualizing the outcome of dynamic analysis of Android malware

with VizMal," *Journal of Information Security and Applications*, vol. 50, Article ID 102423, 2020.

[71] R. Kumar, Xiaosong Zhang, and Wenyong Wang, "A multimodal malware detection technique for Android IoT devices using various features," *IEEE access*, vol. 7, pp. 64411–64430, 2019.

[72] P. Feng, "A novel dynamic Android malware detection system with ensemble learning," *IEEE Access*, pp. 30996–31011, 2018.

[73] S. Garg and N. Baliyan, "A novel parallel classifier scheme for vulnerability detection in android," *Computers \& Electrical Engineering*, vol. 77, pp. 12–26, 2019.

[74] Y. Du, J. Wang, and Q. Li, "An android malware detection approach using community structures of weighted function call graphs," *IEEE Access*, vol. 5, pp. 17478–17486, 2017.

[75] J. Ribeiro, Firooz B. Saghezchi, and Georgios Mantas, "An autonomous host-based intrusion detection system for Android mobile devices," *Mobile Networks and Applications*, vol. 25, pp. 164–172, 2020.

[76] X. Jiang, Baolei Mao, and Jun Guan, "Android malware detection using fine-grained features," *Scientific Programming*, vol. 2020, Article ID 5190138, 13 pages, 2020.

[77] I. Almomani, Raneem Qaddoura, and Maria Habib, "Android Ransomware Detection Based on a Hybrid Evolutionary Approach in the Context of Highly Imbalanced Data," *IEEE Access*, vol. 9, pp. 57674–57691, 2021.

[78] X. Wang, Wei Wang, and Yongzhong He, "Characterizing Android apps' behavior for effective detection of malapps at large scale," *Future generation computer systems*, vol. 75, pp. 30–45, 2017.

[79] M. Yang, Shan Wang, and Zhen Ling, "Detection of malicious behavior in android apps through API calls and permission uses analysis," *Concurrency and Computation: Practice and Experience*, vol. 29, p. e4172, 2017.

[80] C. Zhu, Zhengwei Zhu, and Yunxin Xie, "Evaluation of Machine Learning Approaches for Android Energy Bugs Detection With Revision Commits," *IEEE Access*, vol. 7, pp. 85241–85252, 2019.

[81] A. Amamra, Jean-Marc Robert, and Andrien Abraham, "Generative versus discriminative classifiers for android anomaly-based detection system using system calls filtering and abstraction process," *Security and Communication Networks*, vol. 9, pp. 3483–3495, 2016.

[82] H.-J. Zhu, Tong-Hai Jiang, Bo Ma, and Zhu-Hong You, "HEMD: a highly efficient random forest-based malware detection framework for Android," *Neural Computing and Applications*, pp. 3353–3361, 2018.

[83] J.H. Abawajy and A. Kelarev, "Iterative classifier fusion system for the detection of Android malware," *IEEE Transactions on Big Data*, vol. 52, pp. 282–292, 2017.

[84] B. Ren, Chuanchang Liu, Bo Cheng, and Jie Guo, "Mobi-Sentry: Towards easy and effective detection of android malware on smartphones," *Mobile Information Systems*, vol. 2018, Article ID 4317501, 14 pages, 2018.

[85] B. Kang, Suleiman Y. Yerima, and Sakir Sezer, "N-gram opcodes analysis for android malware detection," 2016, https://arxiv.org/abs/1612.01445.

[86] S. Arshad, Munam A. Shah, Abdul Wahid, and Amjad Mehmood, "Samadroid: a novel 3-level hybrid malware detection model for android operating system," *IEEE Access*, vol. 6, pp. 4321–4339, 2018.

[87] T. Chen, Qingyu Mao, and Yimin Yang, "TinyDroid: a lightweight and efficient model for Android malware detection and classification," *Mobile information systems*, vol. 2018, Article ID 4157156, 9 pages, 2018.

[88] A. Demontis, Marco Melis, Battista Biggio, Davide Maiorca, and Daniel Arp, "Yes, machine learning can be more secure! a case study on android malware detection," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, pp. 711–724, 2017.

[89] X. Chen, Chaoran Li, Derui Wang, and Sheng Wen, "Android HIV: A study of repackaging malware for evading machine-learning detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 987–1001, 2019.

[90] H. Zhang, Y. Zheng, and L. Pan, "An efficient Android malware detection system based on method-level behavioral semantic analysis," *IEEE Access*, vol. 7, pp. 69246–69256, 2019.

[91] A. Narayanan, Mahinthan Chandramohan, Lihui Chen, and Yang Liu, "Context-aware, adaptive, and scalable android malware detection through online learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 1, pp. 157–175, 2017.

[92] H. Cai and N. Meng, "Droidcat: Effective android malware detection and categorization via app-level profiling," *IEEE Transactions on Information Forensics and Security*, pp. 1455–1470, 2018.

[93] H. Li et al., "Adversarial-example attacks toward android malware detection system," *IEEE Systems Journal*, pp. 653–656, 2019.

[94] C. Li et al., "Android malware detection based on factorization machine," *IEEE Access*, vol. 7, pp. 184008–184019, 2019.

[95] J. Li et al., "Significant permission identification for machine-learning-based android malware detection," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 3216–3225, 2018.

[96] F. Pierazzi et al., "A data-driven characterization of modern Android spyware," *ACM Transactions on Management Information Systems (TMIS)*, vol. 11, no. 1, pp. 1–38, 2020.

[97] B. Sun et al., "Detecting Android Malware and Classifying Its Families in Large-scale Datasets," *ACM Transactions on Management Information Systems (TMIS)*, vol. 13, no. 2, pp. 1–21, 2021.

[98] Y. Zhao et al., "On the Impact of Sample Duplication in Machine-Learning-Based Android Malware Detection," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 30, no. 3, pp. 1–38, 2021.

[99] H. Wang et al., "Understanding the purpose of permission use in mobile apps," *ACM Transactions on Information Systems (TOIS)*, vol. 35, no. 4, pp. 1–40, 2017.

[100] A. Salem, S. Banescu, and A. Pretschner, "Maat: Automatically analyzing virustotal for accurate labeling and effective malware detection," 2020, https://arxiv.org/abs/2007.00510.

[101] D. Su, Jiqiang Liu, Xiaoyang Wang, and Wei Wang, "Detecting Android locker-ransomware on chinese social networks," *IEEE Access*, vol. 7, pp. 20381–20393, 2018.

[102] H.D. Trinh, Engin Zeydan, and Lorenza Giupponi, "Detecting mobile traffic anomalies through physical control channel fingerprinting: A deep semi-supervised approach," *IEEE Access*, vol. 7, pp. 152187–152201, 2019.

[103] A. Alotaibi, "Identifying malicious software using deep residual long-short term memory," *IEEE Access*, vol. 7, pp. 163128–163137, 2019.

[104] Y. Nan and Z. Yang, "Identifying user-input privacy in mobile applications at a large scale," *IEEE Transactions on*

Information Forensics and Security, vol. 12, no. 3, pp. 647–661, 2016.

[105] Y.-C. Chen, Hong-Yen Chen, and Takeshi Takahashi, "Impact of Code Deobfuscation and Feature Interaction in Android Malware Detection," *IEEE Access*, vol. 9, pp. 123208–123219, 2021.

[106] C. Cilleruelo, L. De-Marcos, and J.-J. Martinez-Herráiz, "Malware Detection Inside App Stores Based on Lifespan Measurements," *IEEE Access*, vol. 9, pp. 119967–119976, 2021.

[107] H.D. Trinh and Angel Fernandez Gambin, "Mobile Traffic Classification through Physical Control Channel Fingerprinting: a Deep Learning Approach," *IEEE Transactions on Network and Service Management*, vol. 18, 2020.

[108] N. Lachtar, D. Ibdah, and A. Bacha, "The case for native instructions in the detection of mobile ransomware," *IEEE Letters of the Computer Society*, vol. 2, no. 2, pp. 16–19, 2019.

[109] H. Fu, X. Wang, and X. Liao, "Towards Automatic Detection of Nonfunctional Sensitive Transmissions in Mobile Applications," *IEEE Transactions on Mobile Computing*, vol. 18, 2020.

[110] Y. Lee, Xueqiang Wang, Xiaojing Liao, and XiaoFeng Wang, "Understanding Illicit UI in iOS apps Through Hidden UI Analysis," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[111] T. Kim et al., "A multimodal deep learning method for android malware detection using various features," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 773–788, 2018.

[112] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[113] N.S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[114] P. Soucy and G.W. Mineau, "A simple KNN algorithm for text categorization," in *Proceedings of the 2001 IEEE International Conference on Data Mining*, IEEE, San Jose, CA, USA, November 2001.

[115] L.E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[116] B.K. Samanthula, Y. Elmehdwi, and W. Jiang, "K-nearest neighbor classification over semantically secure encrypted relational data," *IEEE transactions on Knowledge and data engineering*, vol. 27, no. 5, pp. 1261–1273, 2014.

[117] F. Zhang, Hansaka Angel Dias Edirisinghe Kodituwakku, and J. Wesley Hines, "Multilayer data-driven cyber-attack detection system for industrial control systems based on network, system, and process data," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4362–4369, 2019.

[118] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*, Routledge, 2017.

[119] S.R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[120] S. Vimala, V. Khanaa, and C. Nalini, "A study on supervised machine learning algorithm to improvise intrusion detection systems for mobile ad hoc networks," *Cluster Computing*, vol. 22, pp. 4065–4074, 2019.

[121] H. Zhou, H.-f. Chai, and M.-l. Qiu, "Fraud detection within bankcard enrollment on mobile device based payment using machine learning," *Frontiers of Information Technology \& Electronic Engineering*, vol. 19, pp. 1537–1545, 2018.

[122] S. Wang et al., "A mobile malware detection method using behavior features in network traffic," *Journal of Network and Computer Applications*, pp. 15–25, 2019.

[123] L. Caviglione, Mauro Gaggero, Jean-Francois Lalande, and Wojciech Mazurczyk, "Seeing the unseen: Revealing mobile malware hidden communications via energy consumption and artificial intelligence," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 799–810, 2015.

[124] X. Wang and C. Li, "Android malware detection through machine learning on kernel task structures," *Neurocomputing*, vol. 453, pp. 126–150, 2021.

[125] S.Y. Yerima, M.K. Alzaylaee, and S. Sezer, "Machine learning-based dynamic analysis of Android apps with improved code coverage," *EURASIP Journal on Information Security*, pp. 1–24, 2019.

[126] F. Idrees, Muttukrishnan Rajarajan, Mauro Conti, Thomas M. Chen, and Yogachandran Rahulamathavan, "PIndroid: A novel Android malware detection system using ensemble learning methods," *Computers \& Security*, vol. 68, pp. 36–46, 2017.

[127] K. Bakour and H.M. nver, "VisDroid: Android malware classification based on local and global image features, bag of visual words and machine learning techniques," *Neural Computing and Applications*, vol. 33, pp. 3133–3153, 2021.

[128] H. Bai, Nannan Xie, Xiaoqiang Di, and Qing Ye, "FAMD: A Fast Multifeature Android Malware Detection Framework, Design, and Implementation," *IEEE Access*, vol. 8, pp. 194729–194740, 2020.

[129] K.K. Jogarah, Keshav Soopaul, Yogesh Beeharry, and Visham Hurbungs, "Hybrid machine learning algorithms for fault detection in android smartphones," *Transactions on Emerging Telecommunications Technologies*, vol. 29, Article ID e3272, 2018.

[130] S. Garg, S.K. Peddoju, and A.K. Sarje, "Network-based detection of Android malicious apps," *International Journal of Information Security*, pp. 385–400, 2017.

[131] A.V. Mbaziira, J. Diaz-Gonzales, and M. Liu, "Deep learning in detection of mobile malware," *Journal of Computing Sciences in Colleges*, vol. 36, no. 3, pp. 80–88, 2020.

[132] S. Seneviratne et al., "Spam mobile apps: Characteristics, detection, and in the wild analysis," *ACM Transactions on the Web (TWEB)*, vol. 11, no. 1, pp. 1–29, 2017.

[133] M. Rahman et al., "Search rank fraud and malware detection in Google Play," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 6, pp. 1329–1342, 2017.

[134] J. Hua, Z. Shen, and S. Zhong, "We can track you if you take the metro: Tracking metro riders using accelerometers on smartphones," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 286–297, 2016.

[135] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," *Machine learning*, vol. 29, no. 2, pp. 103–130, 1997.

[136] C. Manning, *I. Introduction*, 1988.

[137] M.H. Arif, Jianxin Li, Muhammad Iqbal, and Kaixu Liu, "Sentiment analysis and spam detection in short informal text using learning classifier systems," *Soft Computing*, vol. 22, no. 21, pp. 7281–7291, 2018.

[138] J.A. Hartigan and M.A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[139] J. Ribeiro et al., "Hidroid: prototyping a behavioral host-based intrusion detection and prevention system for android," *IEEE Access*, vol. 8, pp. 23154–23168, 2020.

[140] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[141] J. Jamaluddin, N. Zotou, R. Edwards, and P. Coulton, "Mobile phone vulnerabilities: a new generation of malware," in *Proceedings of the IEEE International Symposium on Consumer Electronics*, pp. 199–202, Reading, UK, September 2004.

[142] S. Khan, Xiufeng Liu, Kashish A. Shakil, and Mansaf Alam, "A survey on scholarly data: From big data perspective," *Information Processing \& Management*, vol. 53, no. 4, pp. 923–944, 2017.

[143] D. Hull, S.R. Pettifer, and D.B. Kell, "Defrosting the digital library: bibliographic tools for the next generation web," *PLoS computational biology*, vol. 4, no. 10, Article ID e1000204, 2008.

[144] M. Khabsa and C.L. Giles, "The number of scholarly documents on the public web," *PloS one*, vol. 9, no. 5, Article ID e93949, 2014.

[145] X.J. Yajin Zhou, "Dissecting android malware: Characterization and evolution," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, pp. 95–109, San Francisco, CA, USA, May 2012.

[146] A. Daniel, Michael Spreitzenbarth, and Malte Hübner, "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, pp. 23–26, February 2014.

[147] Mohsen Damshenas, Ali Dehghantanha, Kim-Kwang Raymond Choo, and Ramlan Mahmud, "M0Droid: An Android Behavioral-Based Malware Detection Model," pp. 141–157, 2015.

[148] A.H. Lashkari, Andi Fitriah A. Kadir, and Laya Taheri, "Toward developing a systematic approach to generate benchmark android malware datasets and classification," in *Proceedings of the 2018 International Carnahan Conference on Security Technology (ICCST)*, pp. 1–7, Montreal, QC, Canada, October 2018.

[149] Samaneh Mahdavifar, Andi Fitriah Abdul Kadir, and Rasool Fatemi, "Dynamic Android Malware Category Classification using Semi-Supervised Deep Learning," in *Proceedings of the 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pp. 515–522, Calgary, AB, Canada, August 2020.

[150] Kevin Allix, Tegawendé F. Bissyandé, and Jacques Klein, "AndroZoo: Collecting Millions of Android Apps for the Research Community," pp. 468–471, 2016.

[151] S. Wei, Zedong Zhang, and Shasha Li, "Calibrating Network Traffic with One-Dimensional Convolutional Neural Network with Autoencoder and Independent Recurrent Neural Network for Mobile Malware Detection," *Security and Communication Networks*, vol. 2021, Article ID 6695858, 10 pages, 2021.

[152] F. Ruiz, *Fakeinstaller leads the attack on android phones*, p. 2012, 2016.

[153] X. Jiang, *Security alert: new droidkungfu variant–AGAIN!–Found in Alternative Android Markets*, 2011.

[154] X. Jiang: *Security Alert*, New Android malware–GoldDream–found in alternative app markets. Retrieved June, vol. 17, 2011.

[155] X. Jiang, *Security alert: Gingermaster*, 2011.

[156] A. Feizollah et al., "A review on feature selection in mobile malware detection," *Digital investigation*, vol. 13, pp. 22–37, 2015.

[157] J. Senanayake, H. Kalutarage, and M.O. Al-Kadri, "Android Mobile Malware Detection Using Machine Learning: A Systematic Review," *Electronics*, vol. 10, no. 13, p. 1606, 2021.

[158] Ö.A. Aslan and R. Samet, "A comprehensive review on malware detection approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020.

[159] Z. Wang, Q. Liu, and Y. Chi, "Review of android malware detection based on deep learning," *IEEE Access*, vol. 8, pp. 181102–181126, 2020.

[160] K. Liu, S. Xu, and G. Xu, "A review of android malware detection approaches based on machine learning," *IEEE Access*, vol. 8, pp. 124579–124607, 2020.

[161] M. Rushanan, Aviel D. Rubin, Denis Foo Kune, and Colleen M. Swanson, "Sok: Security and privacy in implantable medical devices and body area networks," in *Proceedings of the 2014 IEEE symposium on security and privacy*, IEEE, Berkeley, CA, USA, May 2014.

[162] Y. Acar, Michael Backes, Sven Bugiel, Sascha Fahl, Patrick McDaniel, and Matthew Smith, "Sok: Lessons learned from android security research for appified software platforms," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP)*, IEEE, San Jose, CA, USA, May 2016.

[163] Q. Attia, K. Ahmad, and C. Victor, "Mobile malware attacks: Review, taxonomy & future directions," *Future Generation Computer Systems*, vol. 97, pp. 887–909, 2019.

[164] Z.Y. Ping Yan, "A survey on dynamic mobile malware detection," *Software Quality Journal*, vol. 26, pp. 891–919, 2018.

[165] Y. K. Kim, Myong-Hyun Go Jemin Justin Lee, Haeyoung Kang, and Kyungho Lee, "A Systematric Literature Review on the Mobile Malware Detection Methods," in *Proceedings of the the 5th International Symposium on Mobile Internet Security (MobiSec 2021)*, Jeju Island, South Korea, October 2021.

[166] A. Shabtai, L. Tenenboim-Chekina, and D. Mimran, "Mobile malware detection through analysis of deviations in application network behaviorn," *Computer & Security*, vol. 43, pp. 1–18, 2014.

[167] Al-Dujaili Abdullah, Alex Huang, and Erik Hemberg, "Adversarial deep learning for robust detection of binary encoded malware," pp. 76–82, 2018, https://www.researchgate.net/publication/326854723_Adversarial_Deep_Learning_for_Robust_Detection_of_Binary_Encoded_Malware.

[168] B. James and J.C. Fraley, "The promise of machine learning in cybersecurity," in *Proceedings of the SoutheastCon 2017*, pp. 1–6, Concord, NC, USA, March 2017.

[169] M.M. Rasheed, A.K. Faieq, and A.A. Hashim, "Android Botnet Detection Using Machine Learning," *Ing \'e nierie des Syst \`e mes d'Information*, vol. 25, 2020.

[170] F.A. Narudin, Ali Feizollah, Nor Badrul Anuar, and Abdullah Gani, "Evaluation of machine learning classifiers for mobile malware detection," *Soft Computing*, vol. 20, pp. 343–357, 2016.

[171] M. Abuthawabeh and K. Mahmoud, "Enhanced android malware detection and family classification using conversation-level network traffic features," *International Arab Journal of Information Technology*, vol. 17, pp. 607–614, 2020.

[172] K.-C. Kim, Eunbyeol Ko, and Jinsung Kim, "Intelligent Malware Detection Based on Hybrid Learning of API and

ACG on Android," *J. Internet Serv. Inf. Secur.*, pp. 39–48, 2019.

[173] S. Wu, Pan Wang, Xun Li, and Yong Zhang, "Effective detection of android malware based on the usage of data flow APIs and machine learning," *Information and software technology*, vol. 75, pp. 17–25, 2016.

[174] M. Alazab, Mamoun Alazab, and Andrii Shalaginov, "Intelligent mobile malware detection using permission requests and API calls," *Future Generation Computer Systems*, vol. 107, pp. 509–521, 2020.

[175] H. Kim, Taejoo Cho, and Gail-Joon Ahn, "Risk assessment of mobile applications based on machine learned malware dataset," *Multimedia Tools and Applications*, vol. 77, pp. 5027–5042, 2018.

[176] P. Vinod, A. Zemmari, and M. Conti, "A machine learning based approach to detect malicious android apps using discriminant system calls," *Future Generation Computer Systems*, vol. 94, pp. 333–350, 2019.

[177] J. Jung, H. Kim, S. J. Cho, and S. Han, "Efficient Android Malware Detection Using API Rank and Machine Learning," *J. Internet Serv. Inf. Secur.*, pp. 48–59, 2019.

[178] L.C. Navarro, Alexandre K.W. Navarro, and André Grégio, "Leveraging ontologies and machine-learning techniques for malware analysis into Android permissions ecosystems," *Computers \& Security*, vol. 78, pp. 429–453, 2018.

[179] I. Martn, J.A. Hernndez, and S. de los Santos, "Machine-Learning based analysis and classification of Android malware signatures," *Future Generation Computer Systems*, vol. 97, pp. 295–305, 2019.

[180] X. Wang, D. Zhang, and X. Su, "Mlifdect: android malware detection based on parallel machine learning and information fusion," *Security and Communication Networks*, vol. 2017, Article ID 6451260, 14 pages, 2017.

[181] Y. Ye and L. Wu, "A Risk Classification Based Approach for Android Malware Detection," *Tiis*, vol. 11, pp. 959–981, 2017.

[182] R. Surendran, T. Thomas, and S. Emmanuel, "A TAN based hybrid model for android malware detection," *Journal of Information Security and Applications*, vol. 54, Article ID 102483, 2020.

[183] F. Shang, Yalin Li, and Xiaolin Deng, "Android malware detection method based on naive Bayes and permission correlation algorithm," *Cluster Computing*, vol. 21, pp. 955–966, 2018.

[184] A. Appice, G. Andresini, and D. Malerba, "Clustering-aided multi-view classification: A case study on Android malware detection," *Journal of Intelligent Information Systems*, vol. 55, pp. 1–26, 2020.

[185] A. Firdaus, Nor Badrul Anuar, Ahmad Karim, and Mohd Faizal Ab Razak, "Discovering optimal features using static analysis and a genetic search based method for Android malware detection," *Frontiers of Information Technology \& Electronic Engineering*, vol. 19, pp. 712–736, 2018.

[186] X. Xiao, Xianni Xiao, Yong Jiang, and Xuejiao Liu, "Identifying Android malware with system call co-occurrence matrices," *Transactions on Emerging Telecommunications Technologies*, vol. 27, no. 5, pp. 675–684, 2016.

[187] A. Ananya, A. Aswathy, T. R. Amal, P. G. Swathy, and P. Vinod, "SysDroid: a dynamic ML-based android malware analyzer using system call traces," *Cluster Computing*, vol. 23, pp. 1–20, 2020.

[188] D. Hu, Zhongjin Ma, Xiaotian Zhang, and Peipei Li, "The concept drift problem in Android malware detection and its solution," *Security and Communication Networks*, vol. 2017, Article ID 4956386, 13 pages, 2017.

[189] C. Wang, Zhiyuan Li, Xiuliang Mo, Hong Yang, and Yi Zhao, "An android malware dynamic detection method based on service call co-occurrence matrices," *Annals of Telecommunications*, vol. 72, pp. 607–615, 2017.

[190] A. Sharma and S.K. Sahay, "Group-wise classification approach to improve android malicious apps detection accuracy," 2019, https://arxiv.org/pdf/1904.02122.pdf.

[191] Y. Zhang, W. Ren, and T. Zhu, "SaaS: A situational awareness and analysis system for massive android malware detection," *Future Generation Computer Systems*, vol. 95, pp. 548–559, 2019.

[192] Z. Ma, H. Ge, Y. Liu, and M. Zhao, "A combination method for android malware detection based on control flow graphs and machine learning algorithms," *IEEE access*, pp. 21235–21245, 2019.

[193] L. Wei, W. Luo, and J. Weng, "Machine learning-based malicious application detection of android," *IEEE Access*, vol. 5, pp. 25591–25601, 2017.

[194] S.Y. Yerima, S. Sezer, and Droidfusion, "A novel multilevel classifier fusion approach for android malware detection," *IEEE transactions on cybernetics*, vol. 49, pp. 453–466, 2018.

[195] L. Yu, Xiapu Luo, Chenxiong Qian, and Shuai Wang, "Enhancing the description-to-behavior fidelity in android apps with privacy policy," *IEEE Transactions on Software Engineering*, vol. 44, no. 9, pp. 834–854, 2017.

[196] S. Aonzo, Alessio Merlo, Mauro Migliardi, Luca Oneto, and Francesco Palmieri, "Low-resource footprint, data-driven malware detection on android," *IEEE Transactions on Sustainable Computing*, vol. 5, no. 2, pp. 213–222, 2017.

[197] F. Martinelli, F. Meraldo, and V. Nardone, "Model checking and machine learning techniques for HummingBad mobile malware detection and mitigation," *Simulation Modelling Practice and Theory*, Article ID 102169, 2020.

[198] A. Narayanan, M. Chandramohan, and L. Chen, "A multi-view context-aware approach to Android malware detection and malicious code localization," *Empirical Software Engineering*, vol. 23, pp. 1222–1274, 2018.

[199] H. Papadopoulos, N. Kumar, and R. Kumar, "Android malware detection with unbiased confidence guarantees," *Neurocomputing*, pp. 3–12, 2018.

[200] S. Sharma et al., "The Paradox of Choice: Investigating Selection Strategies for Android Malware Datasets Using a Machine-learning Approach," *Communications of the Association for Information Systems*, vol. 46, p. 26, 2020.

[201] M.B. Neria, N.-S. Yacovzada, and I. Ben-Gal, "A risk-scoring feedback model for webpages and web users based on browsing behavior," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 4, pp. 1–21, 2017.

[202] Q. Wu, Miaomiao Li, Xueling Zhu, and Bo Liu, "Mviidroid: A multiple view information integration approach for android malware detection and family identification," *IEEE MultiMedia*, pp. 48–57, 2020.

[203] J. Garcia, M. Hammad, and S. Malek, "Lightweight, obfuscation-resilient detection and family identification of android malware," in *Proceedings of the ACM Transactions on Software Engineering and Methodology (TOSEM)*, pp. 1–29, Association for Computing Machinery, New York, NY, USA, 2018.