Exploring Machine Learning in Malware Analysis: Current Trends and Future Perspectives

Noura Alyemni, Mounir Frikha
College of Computer Sciences & Information Technology,
King Faisal University, Al-Ahsa 31982, Saudi Arabia

***Abstract*—Sophisticated cyberattacks are an increasing concern for individuals, businesses, and governments alike. Detecting malware remains a significant challenge, particularly due to the limitations of traditional methods in identifying new or unexpected threats. Machine Learning (ML) has emerged as a powerful solution, capable of analyzing large datasets, recognizing complex patterns, and adapting to rapidly changing attack strategies. This paper reviews the latest advancements in machine learning for malware analysis, shedding light on both its strengths and the challenges it faces. Additionally, it explores the current limitations of these approaches and outlines future research directions. Key recommendations include improving data preprocessing techniques to reduce information loss, utilizing distributed computing for greater efficiency, and maintaining balanced, up-to-date datasets to enhance model reliability. These strategies aim to improve the scalability, accuracy, and resilience of ML-driven malware detection systems.**

***Keywords*—Machine learning; malware analysis; cybersecurity**

I. Introduction

Malware evolves and adapts continuously as computer systems and internet connections continue to expand [1]. The interconnected nature of devices allows malware to spread rapidly, resulting in significant cybersecurity risks. In a sense, malware is similar to a digital virus; it is a sneaky program designed to harm your computer or network [2]. This term encompasses a variety of harmful programs, including viruses, worms, trojans, ransomware, adware, and others [3].

As cyberattacks become more sophisticated, there is an increasing need for advanced malware detection and analysis techniques. However, traditional methods face limitations in performance accuracy and often fail to detect unexpected malware variants. In malware analysis, techniques from a variety of fields are used, including program analysis and network analysis [4]. By examining malicious samples, analysts aim to gain a comprehensive understanding of malware behavior and how it evolves over time.

Researchers have developed various methods for malware detection, which can be broadly divided into two groups: signature-based techniques and machine learning (ML)-based techniques. Signature-based methods rely on recognizing predefined patterns from known malware, while ML-based approaches use algorithms to analyze both benign and malicious samples [5]. This allows ML models to detect both familiar threats and new unpredictable ones. The adaptability of ML-based techniques makes them more suitable for malware detection.

The application of machine learning in malware detection offers promising solutions by adapting to new and evolving threats. However, while machine learning offers significant potential, existing research often examines individual techniques in isolation, without providing a cohesive view of their combined strengths and weaknesses. Furthermore, practical challenges such as mitigating adversarial attacks, managing computational efficiency, and addressing dataset imbalances in real-world applications remain underexplored. These gaps highlight the need for a more integrated and comprehensive approach to fully realize the potential of machine learning in malware detection. ML-based methods enable systems to learn and improve from experience without requiring explicit programming for each task. Unlike signature-based techniques, which depend on predefined malware signatures, ML-based methods are more effective in identifying emerging threats. The effectiveness of these models depends heavily on the quality of features and training data, making them adaptable to the constantly changing nature of malware.

This paper aims to present a comprehensive overview of current trends in machine learning for malware analysis, including descriptions, challenges, and future directions. Specifically, the research aims to answer these questions:

1) What are the key trends in machine learning-based malware analysis techniques?
2) What are the challenges and issues associated with each of these trends?
3) What future research directions in this field require further exploration?

The paper is organized as follows. Section II provides an overview of machine learning in malware analysis. In Section III, we present the methodology used in our research. Section IV describes different trends in malware analysis using ML, followed by challenges associated with each trend in Section V. Finally, suggestions for future directions, countermeasures, and conclusions are discussed in Sections VI and VII.

II. Role of Machine Learning in Malware Analysis

Machine learning has become a vital component in malware detection and analysis, offering solutions to the challenges posed by traditional methods. Its ability to identify unique patterns, adapt to emerging threats, and process vast amounts of data has positioned it as a cornerstone technology in the fight against cybercrime.

One of the key strengths of machine learning is its scalability. Unlike traditional malware analysis techniques, which

depend on manual processes that are both time-consuming and error-prone, machine learning algorithms can evaluate millions of files in just seconds [6]. This ability to quickly and efficiently identify potential threats is crucial in a landscape where the volume and complexity of malware are growing exponentially .

Another significant advantage of machine learning is its adaptability. As cybercriminals continuously develop sophisticated malware and zero-day attacks exploiting vulnerabilities that have not yet been identified machine learning models are uniquely equipped to detect hidden anomalies and respond to novel attack patterns [7]. Models that focus on analyzing dynamic behaviors are particularly effective at staying ahead of these evolving threats, making machine learning an indispensable tool in cybersecurity.

Pattern recognition is another area where machine learning excels. By analyzing the code, behavior, and attributes of malware, these models can uncover intricate patterns that would likely go unnoticed by human analysts. This capability is especially important for identifying zero-day malware, which exploits previously unknown vulnerabilities [8]. Moreover, the automation provided by machine learning frees security analysts to focus on higher-level tasks, such as strategic threat intelligence, thereby improving an organization's overall response to cyberattacks.

To explore how machine learning strengthens malware detection, the next section delves into the primary approaches to malware analysis, including static, dynamic, and hybrid techniques.

A. Overview of Malware Analysis Approaches

Understanding how malware operates, what it targets, and the potential damage it can cause is critical for developing effective defenses. Malware analysis helps achieve this by examining the behavior, structure, and impact of malicious programs. Over the years, several methods have been created to analyze and detect malware, each tailored to address evolving threats. This section discusses the primary approaches: static, dynamic, and hybrid analysis.

- Static Analysis: Static analysis involves inspecting the structure of a program without running it. This approach identifies key attributes of executable files, such as memory usage and file sections, to understand the malware's properties. It is often divided into basic and advanced techniques. Basic static analysis focuses on simple characteristics like file size, type, and header information, using tools such as PEiD, BinText, MD5deep, and PEview [9]. Advanced static analysis takes a deeper dive into the code itself, analyzing commands and instructions in detail to uncover malware's hidden functionality [10]. Machine learning often utilizes features extracted during static analysis, including opcode sequences, file headers, and structural patterns, to build models capable of identifying malware patterns. These features allow models to distinguish between malicious and legitimate software.
- Dynamic Analysis: Dynamic analysis examines the behavior of malware as it executes, often in a controlled environment such as a sandbox or virtual machine. This method provides insights into how malware operates in real-world scenarios while keeping the host machine protected from infection. Tools like Process Monitor, API Monitor, Process Explorer, Regshot, and Wireshark are commonly used to observe basic malware behaviors [11]. Advanced dynamic analysis goes further by using debugging tools like OllyDbg and WinDbg, allowing analysts to step through code execution, modify parameters, and examine detailed system interactions. Once the analysis is complete, the environment is reset to its original state to ensure safety [12]. Behavioral data collected during dynamic analysis, such as API calls, system interactions, and network traffic patterns, plays a crucial role in training machine learning models. These insights help create algorithms that detect both known and previously unseen malware.
- Hybrid Analysis: Hybrid analysis combines static and dynamic techniques to provide a more comprehensive understanding of malware. It begins by analyzing the code and structure without executing it and then proceeds to observe its behavior in a controlled environment. This dual approach overcomes many limitations of using static or dynamic methods alone [13]. Features generated from both static and dynamic analysis, such as opcode sequences, behavioral patterns, and system interaction logs, are integrated into machine learning models. This combination enhances the adaptability and accuracy of malware detection frameworks, making them more effective against evolving threats.

Each of these methods has its strengths and weaknesses, as shown in Table I, and their integration with machine learning offers promising advancements in malware detection [14].

TABLE I. COMPARISON OF MALWARE ANALYSIS APPROACHES

Approaches	Advantages	Disadvantages
Static Analysis	• Quick analysis • Low resource usage • Multi-path analysis • Enhanced security • High accuracy	• Difficulty analyzing obfuscated and encrypted malware • Limited ability to detect unknown malware
Dynamic Analysis	• Analysis of obfuscated and encrypted malware • Superior accuracy compared to static analysis • Detection of known and unknown malware	• Slow and insecure • High resource usage • Time-consuming and vulnerable • Limited code analysis
Hybrid Analysis	• Result in more accurate result	• Requires significant time and resources • High level of complexity

Static, dynamic, and hybrid analysis approaches provide valuable features that significantly enhance machine learning

models used in malware detection. By integrating these methods into machine learning workflows, we can improve detection accuracy and tackle the challenges posed by increasingly sophisticated and evolving malware threats.

III. Research Strategy

This paper uses a systematic literature review (SLR) to explore the role of machine learning in malware detection. The goal is to gather a comprehensive set of relevant studies. The PRISMA 2020 framework (see Fig. 1) was followed to ensure a transparent and systematic approach to selecting and evaluating research articles.

The review focused on journal articles and conference papers published in the past four years. Databases such as IEEE Xplore, ScienceDirect, SpringerLink, and Google Scholar were searched using keywords like "machine learning" AND "malware analysis," "AI-based malware detection," and "deep learning" AND "malware classification". The initial search retrieved 550 records, 500 from primary databases and 50 from secondary sources. After removing 100 duplicates, 450 unique papers remained.

Next, the studies were screened by reviewing their titles and abstracts. This step eliminated 350 papers that were not relevant, lacked full-text availability, were limited to abstracts, or were in languages other than English. The remaining 100 papers were reviewed in full, resulting in the exclusion of 70 papers due to insufficient relevance or methodological quality. Finally, 30 studies were selected based on their alignment with the research scope and their focus on recent challenges or innovative approaches in ML-based malware detection. The selection process is illustrated in Fig. 1.

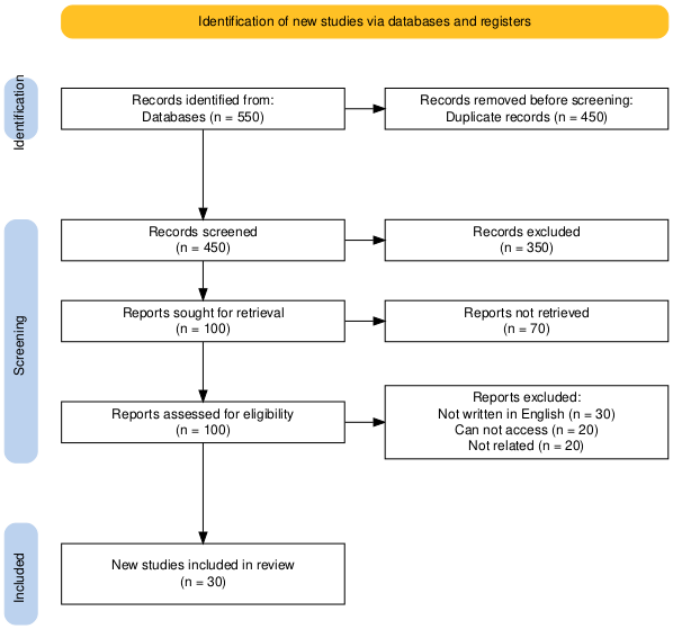

Fig. 1. Selection of papers for review using PRISMA model.

IV. Trends in Malware Analysis Using Machine Learning

Machine learning has revolutionized malware detection by improving both accuracy and efficiency. Researchers have concentrated on three key approaches: deep learning, transfer learning, and explainable AI (XAI). Each of these techniques brings its own advantages and challenges, working together to tackle the complex demands of malware detection by striking a balance between precision, resource efficiency, and transparency. This section explores these methods, highlighting their applications and contributions to advancing malware detection.

A. Deep Learning-Based Malware Analysis

Deep learning is a sophisticated branch of machine learning that uses deep artificial neural networks to find hidden patterns and intricate correlations in data. These networks are made up of linked layers of cells that hierarchically learn representations directly from raw input data, simplifying the process and eliminating the need for manual feature engineering [15]. This automatic feature extraction allows deep learning models to efficiently manage large volumes of data, making them vital in fields such as image processing, healthcare and cybersecurity. Deep learning creates several levels of abstraction using supervised and unsupervised algorithms, facilitating complex analysis and decision-making. This has led to its broad acceptance in a variety of sectors [16].

Rhode et al. [17] investigated Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, for early-stage malware prediction. The authors extracted static features from Portable Executable (PE) files, a common format used by Windows applications, and utilized the LSTM network to simulate the sequential nature of file execution, as a result of which the model was able to capture both short-term and long-term dependencies. By focusing on early-stage behaviors, their model predicted whether a file was malicious before it fully executed, preventing malware before it spreads. Study results demonstrated that LSTM networks are capable of learning temporal patterns, crucial for understanding malware behavior over time. In contrast to traditional machine learning models that rely on static analysis, Rhode et al.'s approach reduced the vulnerability window by detecting malicious intent earlier with LSTMs. In their study, they found that the RNN-based model outperformed traditional techniques such as decision trees and SVMs, which require more data to identify malware. Elayan and Mustafa [18], developed a deep learning-based approach for detecting Android malware. Using gated recurrent units (GRUs), they analyzed static features from Android apps, including API calls and permissions. Their model achieved a high accuracy of 98.2% on the CICAndMal2017 dataset, which supports the effectiveness of deep learning in identifying malicious Android apps.

Catak et al. [19], developed a sequential model using deep learning for analyzing Windows EXE files. Researchers gathered and analyzed a dataset of non-malicious and malicious EXE files and extracted API call sequences. They employed a Long Short-Term Memory (LSTM) network to model the sequential nature of these API calls, which enabled the model

to detect temporal dependencies and patterns indicative of malicious behavior. Based on the dataset, the LSTM model was trained to distinguish malicious from benign EXE files with an impressive accuracy of 98.2%. As a result, deep learning has been demonstrated to be effective in detecting and classifying malware in Windows environments. McDole et al. [20] investigate the application of deep learning techniques for behavioral malware analysis in cloud Infrastructure-as-a-Service (IaaS) environments. The study shows how deep learning models are effective at analyzing malware behavior, making them more effective at detecting sophisticated attacks on cloud-based infrastructures. Similarly, Ravi et al. [21] developed a multi-view attention-based deep learning framework for malware detection in smart healthcare systems. By accurately identifying malicious activities and taking into account the unique operational dynamics of healthcare networks, their work demonstrates that deep learning plays a critical role in ensuring security within healthcare settings.

Calik Bayazit et al. [22] conducted a comprehensive comparative analysis of deep learning models for Android malware detection. They used the Drebin dataset, a publicly accessible collection of benign and malicious Android apps, to evaluate the performance of various models, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). They trained and evaluated the models effectively by extracting static features such as permissions, API calls, and opcode sequences from the apps. In this study, the results demonstrated that hybrid architectures, combining CNNs and RNNs, outperformed individual models, providing evidence that deep learning can enhance Android security.

Ibrahim et al. [23] proposed a malware detection method for Android applications that combines static analysis and deep learning. They extracted key features, including two newly defined features, from the applications. These features were then used as input for a custom-developed deep learning model. The method was evaluated using a classified dataset of Android apps. The extracted features included permissions, API calls, services, broadcast receivers, opcode sequences, application size, and fuzzy hash.

Patil and Deng [24] demonstrated the superior performance of deep learning (DL) networks over traditional machine learning models in malware analysis. They developed a neural network-based framework that achieved high accuracy in classifying malware. The researchers attributed the improved performance to the backpropagation and gradient descent mechanisms employed in DL, which enhance accuracy, true positive rate, and reduce false positive rate.

Rodrigo et al. [25] developed a hybrid machine learning model for Android malware detection. The model consisted of three fully connected neural networks: one for static features, one for dynamic features, and one for a combination of both. When trained on individual features, the static network achieved 92.9% accuracy and the dynamic network achieved 81.1% accuracy. However, the hybrid model, combining both static and dynamic features, outperformed the individual models with an accuracy of 91.1%. This suggests that a hybrid approach, considering both static and dynamic characteristics, is more effective for detecting Android malware.

Obaidat et al. [26] proposed the Jadeite framework to detect Java-based malware by combining image analysis and behavior analysis with deep learning. The framework uses Java bytecode to create grayscale images that represent malware and identify malicious behavior in real time. Jadeite is composed of three primary components. The first component is the Bytecode Transformation Engine, which converts Java bytecode into grayscale images so malware can be visualized. The second is the Feature Extraction Engine that extracts critical features from bytecode. In addition to the two grayscale images and extracted features, the CNN Classifier Engine analyzes the entire file to determine whether it is malicious or benign using a Convolutional Neural Network (CNN) model.

Although these techniques have offered optimal results in modeling intricate patterns of different malware, these algorithms heavily depend on high-quality datasets and are fairly susceptible to adversarial inputs, as will be explained below. Moreover, based on the papers examined, CNNs emerge as the most often used deep learning technology for malware detection, recognized for their capacity to quickly extract features from binary or grayscale representations. As a result of this capability, CNNs are particularly effective at analyzing image-based malware. Additionally, in order to detect dangerous patterns in code or behavior, recurrent neural networks and LSTMs are frequently used for sequential data analysis. Other approaches, such as deep neural networks, help advance the area of malware detection by identifying intricate linkages in data. While deep learning has revolutionized malware detection, its practical deployment still faces significant challenges, as detailed in the next section.

Although deep learning excels at identifying complex patterns, its application often demands large datasets and significant computational resources, which can limit its practicality. To address these constraints, transfer learning has emerged as a promising alternative by reusing pre-trained models to enhance efficiency.

B. Transfer Learning-Based Malware Analysis

Transfer learning allows knowledge from one domain to be applied in another, minimizing the need for large training datasets and heavy computational demands. This method has garnered considerable interest in malware analysis for its ability to efficiently address challenges related to data scarcity and resource constraints. Researchers have demonstrated its potential in improving malware detection, particularly when datasets are limited—a common challenge in real-time applications [27].

A number of advantages can be derived from the use of transfer learning in malware analysis. First, it greatly minimizes the quantity of training data necessary. Because the model begins with pre-learned information fewer malware-specific samples are required to attain high accuracy. Additionally, Transfer learning improves feature representation by combining abstract and complicated patterns learnt during pre-training. Moreover, the process is computationally efficient which reduces the time and resources needed to train a model from the beginning [28], [29].

Chen et al. [30] demonstrated the effectiveness of transfer learning for static malware classification by adapting pre-trained CNNs to malware-specific datasets. By treating mal-

ware binaries as images, their approach significantly reduced training time while maintaining high accuracy. Bhodia et al. [31] employed VGG16, a deep learning model pre-trained on ImageNet, for malware image classification. Fine-tuning the model on malware-specific datasets improved detection accuracy and showed particular promise in identifying zero-day malware attacks.

Prima and Bouhorma [32] leveraged transfer learning to adapt CNN models for malware detection, converting binary malware files into grayscale images. Their results highlighted the efficiency of transfer learning in resource-constrained environments. Similarly, Ahmed et al. [33] proposed a framework that combines transfer learning with convolutional neural networks to classify malware binaries. They used data augmentation and fine-tuning techniques, which enhanced detection accuracy while reducing computational demands.

Zhao et al. [34] extended the concept of transfer learning by developing a multi-channel framework that visualizes malware binaries as images. By fine-tuning pre-trained CNN models for malware detection, their study emphasized the importance of integrating diverse data channels to improve robustness. Panda et al. [35] investigated transfer learning in IoT environments by using pre-trained models to classify malware image representations. They introduced preprocessing techniques to standardize input sizes but noted the challenge of information loss during the conversion process.

Ngo et al. [36] introduced a hybrid approach that combines transfer learning with static and dynamic feature analysis. Their method minimized computational overhead while improving malware detection accuracy, particularly for obfuscated malware. Tasyurek and Arslan [37] developed RT-Droid, a real-time Android malware detection framework based on transfer learning. By examining static features like API calls and permissions, their framework achieved 98.6% accuracy, demonstrating its effectiveness in real-time scenarios.

Transfer learning techniques such as grayscale image, multi-channel frameworks, and the combination of static-dynamic features have also improved malware detection. These methods enable the models to fine-tune with ease for malware related tasks without the need for large amounts of datasets or computational resources. However, preprocessing requirements, input standardization challenges, and dataset imbalances remain significant obstacles.

However, challenges such as information loss during preprocessing, dataset imbalances, and the complexity of fine-tuning pre-trained models must be addressed for transfer learning to realize its full potential in malware detection.

C. Explainable AI-Based Malware Analysis

Explainable Machine Learning (XAI) is a strong ally to improve the transparency and reliability of malware detection systems. Additionally, XAI helps explain how complex machine learning algorithms make decisions based on identifying key features and patterns [38]. A growing field of research has focused on explainability, aimed at clarifying and simplifying machine learning reasoning and decision-making processes. Explainability methods clarify how ML models work, assisting developers and users in understanding their behavior [39].

A variety of explainable AI methods including SHAP and LIME, provide interpretable explanations for model outputs which assist analysts in understanding how classifications are made. This transparency improves decision-making helps refine malware detection methods and supports the development of stronger more reliable malware detection systems [40]. Moreover, XAI aids in identifying potential inaccuracies in the models and data leading to fairer and more balanced systems. With XAI, the factors that influence a model's decision are clarified resulting in fewer false positives and negatives ultimately improving malware detection accuracy and effectiveness.

Ladarola et al. [41] developed a deep learning model for classifying malware families based on their visual representations, achieving 93.4% accuracy. They used LIME to explain model decisions and activation maps to assess model reliability, identify biases, and improve robustness. Alani and Awad [42] proposed PAIRED, an efficient Android malware detection system using XML techniques. By extracting static features from applications, PAIRED achieved an accuracy rate of over 98% while consuming minimal resources. SHAP values were utilized to explain the decision-making process, enhancing the transparency and interpretability of their model.

Liu et al. [43] focused on the interpretability of machine learning models for Android malware detection. They examined the internal workings of models, including decision trees, to identify key features and patterns involved in malware classification. Similarly, Kinkead et al. [44] utilized LIME to enhance the interpretability of CNN-based predictions. Their study validated the consistency between CNN's feature selection and LIME's interpretability framework, showcasing the utility of LIME in corroborating CNN-based malware detection.

According to H. Manthena [45], many malware analysis models lack transparency, making them difficult to trust. This problem was addressed by integrating XML techniques, such as KernelSHAP, TreeSHAP, and DeepSHAP, into online malware detection. These techniques evaluated performance metrics, improved interpretability, and improved the trustworthiness of security systems. In another study, Manthena et al. [46] developed a malware detection system using SHAP to reveal the inner workings of CNNs and Feedforward Neural Networks (FFNNs). The system provided insights into model predictions, improving transparency and trust in the results.

Lu and Thing [47] proposed an Android malware detection framework employing three model explanation methods: Modern Portfolio Theory (MPT), SHAP, and LIME. These methods were compared based on their ability to provide explanations, with MPT demonstrating utility in analyzing adversarial samples. Additionally, Pan et al. [48] developed a hardware-assisted malware detection framework using regression-based explainable machine learning techniques to overcome prediction inaccuracies and lack of transparency.

Sharma et al. [49] designed a traffic analysis-based malware detection system based on traffic analysis that utilizes human-readable network traffic features. Decision tree-based models were employed, enabling more interpretable malware detection. Iadarola et al. [50] proposed an interpretable approach for detecting and categorizing Android malware fami-

lies. By visually representing malware as images and feeding them into an explainable deep learning model, their system achieved both high performance and transparency.

Explainable machine learning techniques such as SHAP, LIME, and XML are very effective in increasing the interpretability and transparency of malware detection systems. These methods contribute to enhancing the interpretability of the factors behind the decisions and therefore enhancing the reliability and accuracy of the results. However, challenges such as scalability in real-time environments and computational overhead prevent broad adoption.

XAI plays a vital role in improving transparency and reliability in malware detection, but it struggles with challenges like real-time scalability, computational demands, and the trade-off between performance and interpretability.

Table II provides a summary of the analyzed papers.

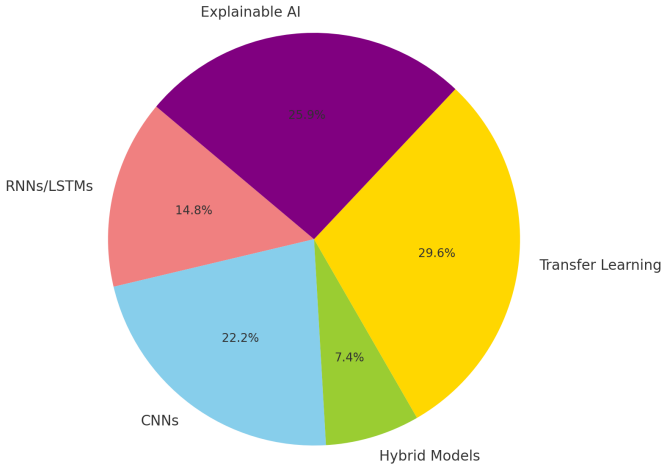

Fig. 2. Proportion of machine learning techniques used in malware detection studies.

Fig. 2 shows the use of various machine learning techniques in malware detection from the studies examined in the paper. Techniques like transfer learning and explainable XAI are prominently represented, indicating their increasing significance in addressing challenges such as limited resources and improving model interpretability. Deep learning methods, including CNNs and RNNs/LSTMs, also play a vital role in identifying complex patterns and managing sequential data, reflecting their foundational importance. Although hybrid models are less commonly employed, they showcase the potential of combining multiple approaches to achieve higher detection accuracy. This analysis underscores the diversity of methodologies adopted in malware detection research and their continuing progress.

While machine learning has great potential to enhance malware detection, it also comes with notable challenges. Deep learning demands significant computational resources, transfer learning contends with issues like imbalanced datasets, and explainable AI poses integration complexities. Tackling these obstacles is crucial to advancing malware detection methods. The next section delves into these challenges, highlighting gaps in current approaches and exploring opportunities for improvement.

V. Limitations of Current Approaches

Although machine learning techniques have made significant advances in malware detection, there are still many limitations, affecting their practicality and scalability. Based on the studies reviewed, this section discusses the limitations of deep learning, transfer learning, and explainable AI.

A. Limitations of Deep Learning

Deep learning techniques have revolutionized malware detection, yet they are not without challenges:

- Rhode et al. [17] emphasized that their LSTM-based approach for early-stage malware detection heavily relied on large, high-quality datasets, limiting its practical applicability in real-world environments. Moreover, LSTMs are computationally intensive, susceptible to noise and adversarial attacks, and often face challenges in generalizing to previously unseen malware families. The complexity of interpreting LSTM models further complicates their adoption, as it undermines trust and impedes effective debugging
- Elayan and Mustafa [18] observed that their GRU-based model for Android malware detection, despite its high accuracy, posed challenges in resource-limited environments such as mobile devices or IoT systems. The model's computational demands resulted in higher energy consumption, longer processing times, and reduced battery efficiency on mobile devices. Addressing this issue may involve developing lightweight architectures or employing model compression techniques to enhance its suitability for resource-constrained settings.
- Catak et al. [19] highlighted the effectiveness of LSTMs in analyzing sequential data but noted their vulnerability to adversarial attacks. Subtle, malicious perturbations in input data can easily deceive LSTMs, resulting in misclassification. This weakness poses a significant security threat in practical applications, as attackers can exploit it to bypass detection mechanisms.
- McDole et al. [20] observed that while deep learning models provide scalability and flexibility in cloud environments, they come with high computational costs. This results in increased latency, elevated operational expenses, and lower energy efficiency, rendering them unsuitable for real-time applications with strict latency demands, such as autonomous vehicles or industrial control systems.
- Ravi et al. [21] identified that their multi-view attention framework, while effective, heavily relies on specialized hardware such as GPUs or TPUs for efficient training and inference. This dependence limits its usability in resource-constrained settings, including IoT and edge devices, where computational resources and memory are restricted. Additionally, the need for specialized hardware can elevate both deployment

TABLE II. RELATED WORK ANALYSIS

Ref.	Addressed Problems	Machine learning techniques used
[17]	Early detection of malware to predict malicious behavior in its initial stages	RNNs, LSTM
[18]	Detection of Android malware using deep learning methods for increased accuracy	CNNs
[19]	Sequential analysis of malware behavior through API call patterns in Windows executable	RNNs, LSTMs
[20]	Behavioral analysis of malware in cloud IaaS environments	CNNs, RNNs
[21]	Malware detection in smart healthcare systems using a multi-view attention-based approach	Attention-based DL Framework, Multi-view models
[22]	Comparative evaluation of deep learning techniques for detecting Android malware	CNNs, RNNs, DL models
[23]	Automatic detection of Android malware using static analysis techniques combined with deep learning	Static Analysis with Deep Neural Networks (DNNs)
[24]	Analysis of malware using a combination of traditional machine learning and deep learning methods	Machine Learning (Random Forest, SVM), CNNs
[25]	Development of a hybrid model for detecting malware on Android devices by combining multiple techniques	Hybrid Model combining Decision Trees and Neural Networks
[26]	Detection of Java-based malware using a combination of image-based and behavior-based features	CNN
[30]	Static malware classification by leveraging pre-trained models to improve accuracy	Transfer Learning with Deep Neural Networks
[31]	Malware classification using image-based representations of malware and transfer learning	Transfer Learning with CNNs
[32]	Malware classification leveraging pre-trained models for enhanced detection	Transfer Learning
[33]	Malware classification by leveraging the Inception V3 architecture and transfer learning	Transfer Learning with Inception V3
[34]	Visual malware classification using a multi-channel approach combined with transfer learning	Transfer Learning with CNNs
[35]	Malware detection in IoT environments through image-based transfer learning techniques	Transfer Learning
[36]	Efficient malware detection using combined static and dynamic features enhanced by transfer learning	Transfer Learning
[37]	Real-time Android application analysis utilizing transfer learning for malware detection	Transfer Learning with CNN Models
[41]	Understanding deep learning predictions in image-based malware detection using activation maps	Deep Learning with Activation Maps
[42]	Lightweight and explainable approaches for Android malware detection	Lightweight Explainable AI for Android Malware Detection
[43]	Enhancing understanding of Android malware detection models performance through explainable AI approaches	Explainable AI applied to Android Malware Detection
[44]	Improving interpretability of CNNs in Android malware detection	CNNs, Explainability
[45]	Development of explainable machine learning frameworks for malware analysis	Explainable Machine Learning
[46]	Providing insights into black-box models used for online malware detection, improving transparency and trust	Explainable Machine Learning
[47]	Providing explanations for predictions of AI-based malware detectors, especially for malicious Android apps	Explainable AI for Malware Detection
[48]	Utilizing hardware-assisted techniques to detect malware with explainable machine learning models	Explainable Machine Learning
[49]	Developing an extensible and explainable system for analyzing network traffic and detecting malware	TTP-based Explainable Systems, Machine Learning
[50]	Enhancing interpretability in deep learning models for detecting and categorizing mobile malware families	Deep Learning, Interpretability

costs and energy consumption, posing challenges for broader adoption.

- Calik Bayazit et al. [22] highlighted the effectiveness of hybrid architectures that leverage the advantages of various deep learning models. However, these architectures tend to be highly complex, presenting challenges in terms of training, optimization, and efficient deployment. Additionally, identifying the ideal combination of models and hyperparameters for a given task can be both time-intensive and computationally demanding.
- Ibrahim et al. [23] observed that integrating static analysis with deep learning improved malware detection accuracy. Despite its benefits, this approach often demands considerable domain expertise to effectively derive and refine features from static analysis outputs. Additionally, the integration of static analysis tools with deep learning models presents challenges in complexity and resource requirements, necessitating significant computational power and specialized infrastructure.
- Patil and Deng [24] highlighted that, although deep learning models outperform traditional approaches, their high training costs and demanding hardware requirements pose significant challenges. These scalability limitations restrict their usability in resource-constrained settings and complicate their application in scenarios requiring frequent retraining, such as adapting to emerging threats.
- Rodrigo et al. [25] observed that while their hybrid model, which integrates static and dynamic features, enhanced malware detection accuracy, it also introduced increased inference time. This limitation makes the approach less practical for real-time applications with strict latency demands, such as intrusion detection systems and real-time threat monitoring.
- Obaidat et al. [26] emphasized the effectiveness of their CNN-based method for detecting Java-based malware through visual bytecode representations. However, the conversion of bytecode into visual formats may result in information loss, which can hinder the model's ability to accurately identify subtle behavioral patterns and nuances of malware.

B. Limitations of Transfer Learning

Transfer learning has shown to be effective in reducing training time and resource consumption, but it faces the following challenges:

- Chen et al. [30] noted that converting malware binaries into image representations for the use of convolutional neural networks can result in substantial information loss. This reduction in critical data may compromise the model's accuracy and its capacity to effectively analyze complex malware behaviors and traits.
- Bhodia et al. [31] highlighted that, although transfer learning demonstrated high accuracy in malware detection, its effectiveness is significantly affected by class imbalance in the training dataset. When certain malware classes are underrepresented, the resulting models may become biased, leading to reduced generalization capabilities for unseen samples belonging to these underrepresented categories.
- Prima and Bouhorma [32] noted that although transfer learning provides an effective initial framework, it often necessitates substantial fine-tuning on specific malware datasets. This process can be both computationally intensive and time-consuming, demanding considerable resources and potentially delaying the quick deployment and adaptation needed to address emerging threats.
- Zhao et al. [34] highlighted the effectiveness of multi-channel frameworks in combining diverse data sources. However, merging information from channels like static analysis, dynamic analysis, and network traffic introduces considerable computational overhead and complexity. Effectively processing and integrating these varied data streams necessitates meticulous optimization of both the model architecture and the training methodology.
- Panda et al. [35] emphasized the difficulties of preprocessing and standardizing input formats for IoT malware detection. The variation among IoT devices and the diversity of malware samples introduce significant challenges in creating consistent input structures, which can complicate data preprocessing workflows and potentially affect the overall performance of the models.
- Ngo et al. [36] demonstrated that integrating static and dynamic features within transfer learning models enhances accuracy. However, this methodology presents notable challenges, including increased training complexity, extended training durations, and difficulties in fine-tuning hyperparameters. Furthermore, the combined use of static and dynamic analysis may lead to longer inference times, potentially hindering the system's efficiency in real-time scenarios.
- Tasyurek and Arslan [37] highlighted that their RT-Droid system demonstrated effectiveness in real-time Android malware detection. However, maintaining its efficacy in the face of rapidly evolving malware requires frequent updates to its models and feature sets. This ongoing need for retraining and redeployment poses significant challenges, including increased resource demands and operational complexity.

C. Limitations of Explainable AI

Explainable AI techniques have enhanced the interpretability of malware detection models, but there are still several limitations:

- Ladarola et al. [41] demonstrated that LIME effectively enhances interpretability by offering localized explanations of model predictions. However, its substantial computational demands render it impractical

for real-time applications with strict latency constraints, such as online malware detection or intrusion detection systems.

- Alani and Awad [42] highlighted the utility of SHAP values in offering valuable insights into the elements shaping model predictions, thereby improving transparency. However, incorporating SHAP value computations into resource-limited environments, such as mobile or edge devices, poses challenges due to the substantial computational resources required for their calculation.
- Liu et al. [43] observed that decision trees, despite their inherent interpretability, encounter scalability challenges when dealing with large-scale malware datasets. The expansion in the number of features and data samples can significantly increase the tree's complexity, resulting in prolonged training durations, higher memory usage, and reduced overall efficiency.
- Kinkead et al. [44] demonstrated the effectiveness of LIME in explaining CNN-based malware detection model predictions. However, they identified scalability as a significant limitation, particularly when handling large and complex malware datasets. This constraint poses challenges for its practical application in real-world scenarios requiring swift analysis and explanation of extensive malware samples.
- Lu and Thing [47] investigated various explainability techniques, including MPT. However, they identified that MPT has shortcomings in effectively handling adversarial attacks. Such adversarial examples, designed to exploit vulnerabilities in the model, can undermine the reliability of explainability methods, resulting in distorted or inaccurate interpretations.
- Pan et al. [48] introduced a hardware-assisted framework aimed at enhancing the transparency and interpretability of deep learning models for malware detection. Despite its advantages, the reliance on specialized hardware restricts its use in general-purpose systems. Additionally, this dependency may elevate deployment costs, presenting a barrier to broader implementation.
- Manthena et al. [46] highlighted that SHAP enhances the interpretability of deep learning models by generating feature importance scores. However, calculating SHAP values introduces substantial computational overhead, which can adversely affect real-time system performance. This limitation poses a bottleneck in high-throughput malware analysis workflows, hindering their efficiency in time-sensitive applications.
- Sharma et al. [49] emphasized the effectiveness of decision-tree-based models in traffic analysis and malware detection. However, these models are highly susceptible to obfuscation techniques, which are employed by malware developers to modify the code's structure while retaining its functionality. Such obfuscation methods can compromise the model's ability to accurately detect and classify malware, posing a significant challenge in maintaining detection reliability.

The analyzed papers demonstrate that considerable progress has been made in employing machine learning approaches to detect malware; still, various limitations make it difficult to implement the obtained outcomes.

1) Deep Learning:
   - Datasets must be large and high quality to be effective in environments with limited data.
   - Vulnerable to adversarial attacks that manipulate model predictions.
   - The computational requirements make it difficult to deploy in resource-constrained systems.
2) Transfer Learning:
   - In preprocessing steps, such as converting malware binaries into images, losing information can be increased.
   - Dataset imbalances affect model generalizability.
   - Fine-tuning pre-trained models is computationally expensive and time-intensive.
3) Explainable AI (XAI):
   - High computational overhead deter scalability for real time applications.
   - Finding a balance between transparency and efficiency is still difficult.
   - Compatibility with existing security systems is a high level of integration and thus calls for domain-specific solutions.

While these challenges present significant hurdles, they also highlight critical areas that require further exploration and innovation. Overcoming these barriers is vital to realize the full potential of machine learning in malware detection. With advancements in techniques such as preprocessing optimization, improved dataset balancing, enhanced computational efficiency, and seamless system integration, limitations can be addressed effectively. The next section delves into specific strategies and emerging possibilities that aim to enhance the scalability, reliability, and transparency of machine learning-based malware detection systems.

VI. Future Directions and Countermeasures

Detecting and analyzing malware has made significant progress; however, a number of challenges still exist. This section presents potential future research directions and actionable countermeasures for improving machine learning-based malware detection systems' scalability, robustness, and transparency.

A. Future Directions

1) Deep Learning:

- Federated Learning for Privacy: Federated learning enables collaborative model training while ensuring data privacy by retaining data on individual devices, thus reducing the risk of sensitive information exposure. However, privacy concerns persist, prompting ongoing research into methods such as differential privacy to enhance protection and address these challenges [51].

- Hybrid Architectures: Combining CNNs and RNNs allows for combining their complementary capabilities, with CNNs excelling at identifying spatial patterns and RNNs adept at analyzing sequential data [67]. This integration enables the model to effectively capture both spatial and temporal relationships within malware datasets, offering enhanced accuracy and robustness in malware detection.
- Optimized Lightweight Models: Design models tailored for resource-constrained environments such as IoT devices or edge platforms by employing techniques like model pruning, quantization, and knowledge distillation. These methods significantly reduce model size and computational demands while maintaining acceptable levels of accuracy. Sze et al. [68] highlighted the value of optimizing deep neural networks for embedded systems, showcasing how such strategies can enhance energy efficiency and make models more suitable for real-time malware detection in low-power, latency-sensitive scenarios.

2) Transfer Learning:

- Improved Preprocessing Techniques: Advancing preprocessing methods is essential to retain critical features while minimizing the loss of information during data transformation. Techniques such as adaptive feature scaling and intelligent data augmentation can strike a balance, ensuring that key data characteristics are preserved for better model accuracy [69]. This approach has proven beneficial in applications where maintaining high-dimensional data integrity is crucial.
- Cross-Domain Adaptability: Developing models that perform effectively across varying domains, such as IoT and cloud infrastructures, is a vital research direction. Leveraging strategies like domain adaptation and transfer learning can enable these models to generalize efficiently across diverse environments, addressing discrepancies in data distributions and ensuring consistent performance [70].
- Streamlined Fine-Tuning Processes: Streamlining fine-tuning procedures is critical to enhance efficiency and performance. Automated tools such as AutoML and hyperparameter optimization frameworks can simplify this process by automating the search for optimal model parameters [72]. This reduces manual intervention and significantly improves the model's overall effectiveness.
- Standardized Datasets for Malware Detection: Ensuring the availability of standardized and balanced datasets is essential for reliable evaluation and benchmarking of malware detection models. These datasets should include diverse malware samples and simulate real-world scenarios to enhance the generalizability and robustness of the models [74].

3) Explainable AI:

- Efficient Explanation Models: Creating lightweight XAI frameworks tailored for real-time applications is essential. These models should focus on reducing computational overhead while delivering clear, actionable insights. Streamlining algorithms like SHAP or LIME for efficient processing can make XAI more applicable in scenarios requiring immediate decision-making.
- Adaptive Explanations: Develop Dynamic explainability frameworks are crucial for addressing evolving malware patterns. By continuously learning and adapting to new threats, these systems can provide context-specific explanations that remain relevant over time. Such adaptability ensures that cybersecurity measures evolve in tandem with emerging challenges.
- Integration with Security Frameworks: Modular XAI tools designed for seamless integration with existing cybersecurity systems can significantly enhance decision-making [73]. These tools can act as plug-and-play components, working in harmony with established security workflows to improve detection accuracy and transparency .

B. Countermeasures

1) Deep Learning:

- Defensive Mechanisms for Adversarial Inputs: Building robust defenses against adversarial attacks is essential for bolstering the security of machine learning models. A widely adopted technique is adversarial training, which incorporates adversarial examples into the training process to enhance the model's resilience. For example, the study [52] introduces an Ulam-stability-based method that significantly improves model robustness against such attacks. Another promising method involves using an anti-adversarial module, as outlined in [53]. This approach applies targeted counter-adversarial treatments to input samples, effectively reducing the impact of adversarial perturbations. By employing these advanced techniques, machine learning models can achieve heightened resistance to adversarial inputs, ultimately increasing their reliability in security-critical applications.
- Augmentation Techniques for Data: Data augmentation has emerged as a vital technique to address limitations in dataset sizes. By using GANs, synthetic but realistic data can be generated, significantly enriching training datasets. For instance, [62] discuss how GANs can create synthetic malware samples, which help balance datasets and improve the robustness of machine learning models in detecting malware. This approach not only reduces reliance on large datasets but also enhances model generalizability by diversifying training inputs.
- Improving Adversarial Resilience: Adversarial training has become essential in strengthening models against adversarial attacks. For example, Madry et al. propose incorporating adversarial samples during the training process to improve a model's robustness [63]. By simulating potential attacks, this method ensures that models can better resist manipulation, making them more reliable in security-critical environments

- Leveraging Hardware Solutions: To address the computational demands of deep learning models, leveraging specialized hardware like Tensor Processing Units (TPUs) and GPUs has proven effective. Jouppi et al. demonstrate the use of TPUs to accelerate deep learning tasks, showing how such hardware can reduce training times and energy consumption while maintaining high performance [64].

2) Transfer Learning:

- Handling Dataset Imbalances: Addressing dataset imbalances is crucial for developing effective machine learning models. Techniques such as oversampling the minority class and generating synthetic data have proven effective in mitigating these imbalances. For instance, the Synthetic Minority Over-sampling Technique (SMOTE) creates synthetic samples by interpolating between existing minority instances, thereby enhancing model performance on imbalanced datasets [54]. Moreover, recent advancements have introduced methods like Localized Random Affine Shadowsampling (LoRAS), which oversamples from an approximated data manifold of the minority class, addressing limitations associated with traditional techniques [55]. Through the implementation of these strategies, models can achieve better balance and improved prediction accuracy.

- Optimized Preprocessing Workflows: Effective preprocessing is critical to ensuring the success of machine learning models in malware detection. Optimizing these workflows not only preserves essential data features but also reduces computational overhead, enabling efficient and scalable model deployment. Techniques such as feature selection and dimensionality reduction, as presented in [56], can streamline preprocessing by focusing on the most informative attributes while discarding redundant data. Additionally, leveraging automated preprocessing pipelines, as highlighted in [57], can dynamically adapt preprocessing strategies to diverse datasets and application requirements.

- Distributed Training Systems: Distributed training systems enable the efficient processing of large datasets and complex machine learning models by leveraging the computational power of multiple machines. This approach not only reduces resource bottlenecks but also accelerates the training process, making it ideal for scaling malware detection models to meet real-world demands. For instance, distributed frameworks such as Apache Spark and TensorFlow Distributed offer robust architectures for handling extensive data and computations across multiple nodes [58], [59]. These systems optimize training by partitioning tasks, balancing workloads, and parallelizing computations. Additionally, advancements in federated learning and edge computing can complement distributed systems, enabling secure and decentralized training of models without compromising data privacy [60], [61].

3) Explainable AI:

- Real-Time XAI Models: Real-time XAI frameworks are essential for applications requiring rapid decision-making. Simplified versions of SHAP and LIMEcan reduce computational overhead, enabling real-time processing. Accordingly, in the study by [65] real-time SHAP implementation demonstrated effective trade-offs between interpretability and speed, ensuring timely insights without significant computational delays.

- Combining Explanation Approaches: Integrating localized explanation strategies, like LIME, with global methods, such as SHAP, provides a comprehensive understanding of model decisions. This hybrid approach balances detailed insights with overarching trends, improving both interpretability and model validation. A study by [66] highlights the effectiveness of combining explanation techniques to enhance trust in machine learning models without compromising accuracy

According to the outlined future directions and countermeasures, researchers can go a long way in enhancing the detection of malware. It seeks to optimise the ML approaches to increase their applicability on the current and emerging complex cybersecurity challenges.

VII. Conclusions

This paper offers a detailed review of the latest trends and challenges in applying machine learning to malware detection and analysis, with a focus on its increasing role in combating complex cyber threats. Machine learning has shown great promise as a versatile tool, providing scalability, adaptability, and improved pattern recognition for identifying and analyzing malware. However, significant challenges remain, including vulnerabilities to adversarial attacks, biases in datasets, and a lack of transparency in many deep learning models.

By exploring methods such as deep learning, transfer learning, and explainable AI, this review highlights both their strengths and the challenges they face, including high computational requirements and reliance on feature extraction. These obstacles underscore the need for innovative approaches to improve the effectiveness and dependability of machine learning systems in malware detection.

To overcome these limitations, this paper proposes several novel strategies, such as leveraging distributed computing, refining preprocessing methods, and enhancing the integration of explainability techniques. As a result of these advancements, machine learning models will become more robust, efficient, and transparent, ensuring their effectiveness in addressing malware threats as they evolve.

Funding

This work was funded by King Faisal University, Saudi Arabia [Grant No. KFU250089].

Acknowledgment

This work was supported through the Annual Funding track by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [Grant No. KFU250089].

CONFLICTS OF INTEREST

All authors declare no conflict of interest.

REFERENCES

[1] N. Z. Gorment, A. Selamat, L. K. Cheng, and O. Krejcar, "Machine learning algorithm for malware detection: Taxonomy, current challenges and future directions," *IEEE Access*, 2023.

[2] M. S. Akhtar and T. Feng, "Malware analysis and detection using machine learning algorithms," *Symmetry*, vol. 14, no. 11, pp. 2304, 2022.

[3] M. S. Akhtar and T. Feng, "IOTA-based anomaly detection machine learning in mobile sensing," *EAI Endorsed Transactions on Creative Technologies*, vol. 9, pp. 172814, 2022, doi:10.4108/eai.9-12-2022.172814.

[4] D. Ucci, L. Aniello, and R. Baldoni, "Survey of machine learning techniques for malware analysis," *Computers & Security*, vol. 81, pp. 123–147, 2019, doi:10.1016/j.cose.2018.11.001.

[5] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, pp. 160, 2021.

[6] R. Komatwar and M. Kokare, "A survey on malware detection and classification," *Journal of Applied Security Research*, pp. 1–31, Aug. 2020.

[7] O. Aslan, M. Ozkan-Okay, and D. Gupta, "A review of cloud-based malware detection system: Opportunities, advances and challenges," *European Journal of Engineering and Technology Research*, vol. 6, no. 3, pp. 1–8, Mar. 2021.

[8] R. Komatwar and M. Kokare, "Retracted article: A survey on malware detection and classification," *Journal of Applied Security Research*, vol. 16, no. 3, pp. 390–420, 2021.

[9] M. Sikorski and A. Honig, *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. San Francisco, CA, USA: No Starch Press, 2012.

[10] O. Aslan and A. A. Yilmaz, "A new malware classification framework based on deep learning algorithms," *IEEE Access*, vol. 9, pp. 87936–87951, 2021.

[11] O. Aslan and R. Samet, "Investigation of possibilities to detect malware using existing tools," in *Proc. IEEE/ACS 14th Int. Conf. Computer Systems and Applications (AICCSA)*, 2017, pp. 1277–1284.

[12] M. Ijaz, M. H. Durad, and M. Ismail, "Static and dynamic malware analysis using machine learning," in *Proc. 16th Int. Bhurban Conf. Applied Sciences and Technology (IBCAST)*, 2019, pp. 687–691.

[13] N. Tarar, S. Sharma, and C. R. Krishna, "Analysis and classification of android malware using machine learning algorithms," in *Proc. 3rd Int. Conf. Inventive Computation Technologies (ICICT)*, 2018, pp. 738–743.

[14] V. Rao and K. Hande, "A comparative study of static, dynamic and hybrid analysis techniques for android malware detection," *Int. J. Eng. Dev. Res.*, vol. 5, no. 2, pp. 1433–1436, 2017.

[15] U. H. Tayyab *et al.*, "A survey of the recent trends in deep learning based malware detection," *Journal of Cybersecurity and Privacy*, vol. 2, no. 4, pp. 800–829, 2022.

[16] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[17] M. Rhode, P. Burnap, and K. Jones, "Early-stage malware prediction using recurrent neural networks," *Computers & Security*, vol. 77, pp. 578–594, 2018.

[18] O. N. Elayan and A. M. Mustafa, "Android malware detection using deep learning," *Procedia Computer Science*, vol. 184, pp. 847–852, 2021.

[19] F. O. Catak *et al.*, "Deep learning-based sequential model for malware analysis using Windows EXE API calls," *PeerJ Computer Science*, vol. 6, pp. e285, 2020.

[20] A. McDole *et al.*, "Deep learning techniques for behavioral malware analysis in cloud IaaS," in *Malware Analysis Using Artificial Intelligence and Deep Learning*, Springer, 2021, pp. 269–285.

[21] V. Ravi *et al.*, "A multi-view attention-based deep learning framework for malware detection in smart healthcare systems," *Computer Communications*, vol. 195, pp. 73–81, 2022.

[22] E. C. Bayazit *et al.*, "Deep learning-based malware detection for Android systems: A comparative analysis," *Tehnicki Vjesnik*, vol. 30, no. 3, pp. 787–796, 2023.

[23] M. Ibrahim *et al.*, "A method for automatic Android malware detection based on static analysis and deep learning," *IEEE Access*, vol. 10, pp. 117334–117352, 2022.

[24] R. Patil and W. Deng, "Malware analysis using machine learning and deep learning techniques," in *Proc. 2020 SoutheastCon*, vol. 2, pp. 1–7.

[25] C. Rodrigo, S. Pierre, R. Beaubrun, and F. El Khoury, "A hybrid machine learning-based malware detection model for Android devices," *Cybersecurity and Data Science*, pp. 194, 2021.

[26] I. Obaidat, M. Sridhar, K. M. Pham, and P. H. Phung, "Jadeite: A novel image-behavior-based approach for Java malware detection using deep learning," *Computers & Security*, vol. 113, pp. 102547, 2022.

[27] R. Ribani and M. Marengoni, "A survey of transfer learning for convolutional neural networks," in *Proc. 32nd SIBGRAPI Conf. Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, 2019, pp. 47–57.

[28] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Artificial Neural Networks and Machine Learning–ICANN 2018*, 2018, pp. 270–279.

[29] H. M. K. Barznji, "Transfer learning as a new field in machine learning," *Int. Archives Photogrammetry, Remote Sensing Spatial Information Sciences*, vol. 44, pp. 343–349, 2020.

[30] L. Chen, "Deep transfer learning for static malware classification," *arXiv preprint arXiv:1812.07606*, 2018.

[31] N. Bhodia, P. Prajapati, F. Di Troia, and M. Stamp, "Transfer learning for image-based malware classification," *arXiv preprint arXiv:1903.11551*, 2019.

[32] B. Prima and M. Bouhorma, "Using transfer learning for malware classification," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 44, pp. 343–349, 2020.

[33] M. Ahmed, N. Afreen, M. Ahmed, M. Sameer, and J. Ahamed, "An Inception V3 approach for malware classification using machine learning and transfer learning," *International Journal of Intelligent Networks*, vol. 4, pp. 11–18, 2023.

[34] Z. Zhao, S. Yang, and D. Zhao, "A new framework for visual classification of multi-channel malware based on transfer learning," *Applied Sciences*, vol. 13, no. 4, pp. 2484, 2023.

[35] P. Panda, O. K. CU, S. Marappan, S. Ma, and D. V. Nandi, "Transfer learning for image-based malware detection for IoT," *Sensors*, vol. 23, no. 6, pp. 3253, 2023.

[36] M. V. Ngo, T. H. Truong, D. Rabadi, J. Y. Loo, and S. G. Teo, "Fast and efficient malware detection with joint static and dynamic features through transfer learning," in *Proc. Int. Conf. Applied Cryptography and Network Security*, pp. 503–531, 2023.

[37] M. Tasyurek and R. S. Arslan, "Rt-droid: A novel approach for real-time Android application analysis with transfer learning-based CNN models," *Journal of Real-Time Image Processing*, vol. 20, no. 3, pp. 55, 2023.

[38] U. Bhatt, A. Xiang, S. Sharma, *et al.*, "Explainable machine learning in deployment," in *Proc. 2020 Conf. Fairness, Accountability, and Transparency*, 2020, pp. 648–657.

[39] S. M. Lundberg *et al.*, "Explainable machine-learning predictions for the prevention of hypoxaemia during surgery," *Nature Biomedical Engineering*, vol. 2, no. 10, pp. 749, 2018.

[40] X. Zhong, B. Gallagher, S. Liu, *et al.*, "Explainable machine learning in materials science," *npj Computational Materials*, vol. 8, no. 1, pp. 204, 2022.

[41] G. Ladarola, F. Mercaldo, F. Martinelli, and A. Santone, "Assessing deep learning predictions in image-based malware detection with activation maps," in *Proc. Security and Trust Management: 18th Int. Workshop, STM 2022*, vol. 13867, pp. 104, 2023.

[42] M. M. Alani and A. I. Awad, "Paired: An explainable lightweight Android malware detection system," *IEEE Access*, vol. 10, pp. 73214–73228, 2022.

[43] Y. Liu, C. Tantithamthavorn, L. Li, and Y. Liu, "Explainable AI for Android malware detection: Towards understanding why the models perform so well?" in *Proc. IEEE 33rd Int. Symp. Software Reliability Engineering (ISSRE)*, pp. 169–180, 2022.

[44] M. Kinkead, S. Millar, N. McLaughlin, and P. O'Kane, "Towards explainable CNNs for Android malware detection," *Procedia Computer Science*, vol. 184, pp. 959–965, 2021.

[45] H. Manthena, *Explainable Machine Learning Based Malware Analysis*, Ph.D. dissertation, North Carolina Agricultural and Technical State University, 2022.

[46] H. Manthena, J. C. Kimmel, M. Abdelsalam, and M. Gupta, "Analyzing and explaining black-box models for online malware detection," *IEEE Access*, vol. 11, pp. 25237–25252, 2023.

[47] Z. Lu and V. L. Thing, "How does it detect a malicious app? Explaining the predictions of AI-based malware detector," in *Proc. IEEE 8th Int. Conf. Big Data Security on Cloud (BigDataSecurity)*, pp. 194–199, 2022.

[48] Z. Pan, J. Sheldon, and P. Mishra, "Hardware-assisted malware detection using explainable machine learning," in *Proc. IEEE 38th Int. Conf. Computer Design (ICCD)*, pp. 663–666, 2020.

[49] Y. Sharma, S. Birnbach, and I. Martinovic, "Radar: A TTP-based extensible, explainable, and effective system for network traffic analysis and malware detection," 2023.

[50] G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Towards an interpretable deep learning model for mobile malware detection and family identification," *Computers & Security*, vol. 105, pp. 102198, 2021.

[51] Y. Bi, H. Wang, J. Liu, and X. Zhang, "Enabling privacy-preserving cyber threat detection with federated learning," *arXiv preprint arXiv:2404.05130*, 2023.

[52] K. Yan, L. Yang, Z. Yang, and W. Ren, "Enhancing adversarial robustness through stable adversarial training," *Symmetry*, vol. 16, no. 10, p. 1363, 2024.

[53] Z. Qin, G. Liu, and X. Lin, "Enhancing model robustness against adversarial attacks with an anti-adversarial module," in *Pattern Recognition and Computer Vision (PRCV 2023)*, Lecture Notes in Computer Science, vol. 14433, Springer, 2023, pp. 66–78.

[54] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[55] B. Kovács, I. Bagyinszki, and J. Abonyi, "LoRAS: Localized Random Affine Shadowsampling to Address Class Imbalance," *Applied Sciences*, vol. 9, no. 16, pp. 3334, 2019, doi:10.3390/app9163334.

[56] J. Smith and A. Kumar, "Efficient Feature Selection for Malware Detection Using Recursive Feature Elimination," *Journal of Cybersecurity*, vol. 10, no. 3, pp. 45–60, 2021.

[57] L. Wang, M. Zhang, and H. Liu, "AutoML-Driven Preprocessing for Scalable Malware Detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 2, pp. 173–183, 2022.

[58] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Apache Spark: A Unified Engine for Big Data Processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[59] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, et al., "TensorFlow: A System for Large-Scale Machine Learning," in *Proc. 12th USENIX Symp. Operating Systems Design and Implementation (OSDI)*, pp. 265–283, 2016.

[60] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proc. 20th Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282, 2017.

[61] P. Kairouz, B. McMahan, D. Alistarh, et al., "Advances and Open Problems in Federated Learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[62] J. Wang, W. Xu, J. Chen, and S. Liu, "Data Augmentation for Deep Learning Using Generative Adversarial Networks: A Review," *IEEE Access*, vol. 9, pp. 141061–141076, 2021.

[63] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2018.

[64] N. P. Jouppi, C. Young, N. Patil, D. Patterson, and G. Agrawal, "In-Datacenter Performance Analysis of a Tensor Processing Unit," in *Proc. 44th Annual Int. Symp. Computer Architecture (ISCA)*, 2017, pp. 1–12.

[65] X. Zhong, B. Gallagher, S. Liu, et al., "Explainable machine learning in materials science," *npj Computational Materials*, vol. 8, no. 1, pp. 204, 2022.

[66] U. Bhatt, A. Xiang, S. Sharma, et al., "Explainable machine learning in deployment," in *Proc. 2020 Conf. Fairness, Accountability, and Transparency*, pp. 648–657, 2020.

[67] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[68] V. Sze, Y. Chen, T. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[69] D. Jha, K. W. Liang, and T. Singh, "Advances in preprocessing techniques for deep learning applications," *IEEE Access*, vol. 8, pp. 34512–34523, 2020.

[70] W. Li, L. Wang, and E. H. Xing, "Domain adaptation in the era of deep learning," *Nature Machine Intelligence*, vol. 1, no. 6, pp. 335–346, 2019.

[71] B. Zoph and Q. V. Le, "AutoML: A method for efficient hyperparameter optimization," *Proceedings of the National Academy of Sciences*, vol. 115, no. 25, pp. 5862–5869, 2018.

[72] B. Zoph and Q. V. Le, "AutoML: A method for efficient hyperparameter optimization," *Proceedings of the National Academy of Sciences*, vol. 115, no. 25, pp. 5862–5869, 2018.

[73] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-López, D. Molina, R. Benjamins, and F. Herrera, "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[74] H. S. Anderson and P. Roth, "EMBER: An open dataset for training and evaluating machine learning models on malware detection," *arXiv preprint arXiv:1804.04637*, 2018.