Machine Learning in Malware Analysis: Current Trends and Future Directions

Safa Altaha[1], Khaled Riad[2]
College of Computer Sciences & Information Technology,
King Faisal University, Al-Ahsa 31982, Saudi Arabia[1]
Computer Science Department, College of Computer Sciences & Information Technology,
King Faisal University, Al-Ahsa 31982, Saudi Arabia[2]
Mathematics Department-Faculty of Science, Zagazig University, Zagazig 44519, Egypt[2]

***Abstract*—Malware analysis is a critical component of cybersecurity due to the increasing sophistication and the widespread of malicious software. Machine learning is highly significant in malware analysis because it can process huge amounts of data, identify complex patterns, and adjust to changing threats. This paper provides a comprehensive overview of existing work related to Machine Learning (ML) methods used to analyze malware along with a description of each trend. The results of the survey demonstrate the effectiveness and importance of three trends, which are: deep learning, transfer learning, and XML techniques in the context of malware analysis. These approaches improve accuracy, interpretability, and transparency in detecting and analyzing malware. Moreover, the related challenges and Issues are presented. After identifying these challenges, we highlight future directions and potential areas that require more attention and improvement, such as distributed computing and parallelization techniques which can reduce training time and memory requirements for large datasets. Also, further investigation is needed to develop image resizing techniques to be used during the visual representation of malware to minimize information loss while maintaining consistent image sizes. These areas can contribute to the enhancement of machine learning-based malware analysis.**

Keywords*—Malware; malware analysis; machine learning; deep learning; transfer learning

I. Introduction

The increasing demand for computer system usage and internet connectivity results in the continuous evolution and change of malware [1]. As more people and businesses rely on computers and the internet in their daily activities, the potential attack surface for malware actors and attackers expands, giving them more opportunities to exploit vulnerabilities and compromise systems. Moreover, the widespread adoption of the internet has connected millions of devices, which allows malware to spread rapidly. Malicious actors can use the internet to distribute malware through many channels, such as: email attachments, malicious websites, or social media platforms. Malware describes any software or collection of instructions that are designed to intentionally affect computer systems, businesses, or users by causing harm. [2]. The term "malware" includes a wide range of threats, which include viruses, worms, trojan horses, ransomware, adware, and various other types of malicious software [3]. The detection and analysis of malware has consistently been a major concern and a challenging issue due to limitations in analysis methodologies, performance accuracy, and approaches that fail to identify unforeseen malware attacks. Malware analysis involves the utilization of techniques from diverse fields such as program analysis and network analysis. Its purpose is to examine malicious samples in order to gain a comprehensive understanding of various aspects, including their behavior and the transformations they undergo over time [4]. Recently, researchers have introduced various techniques for malware analysis. These techniques are typically classified into two groups: the first one is the signature-based techniques and the second one is ML-based techniques. The first techniques depend on predefined patterns or signatures of known malware samples to identify and detect malicious software. ML-based techniques for malware detection utilize ML algorithms to analyze benign and malicious malware samples. By examining these samples, the algorithms generate learning patterns that can be used to detect both known and unpredictable and new malware. This ability makes ML-based approaches a preferred choice for malware detection. ML is a core element of artificial intelligence, it typically enables systems to automatically learn and improve from experience, without requiring explicit programming for each task [5], [6]. Compared to signature-based techniques, which rely on predefined signatures, ML-based techniques are more efficient in identifying new malware [7]. This is because the accuracy of ML models depends on the features used and the training set, allowing them to learn and detect the changing characteristics of malware.

This paper aims to provide a comprehensive and up-to-date overview of existing trends related to ML used to analyze, detect, and classify malware, which includes a description of each trend, its related challenges, and issues. In addition, we include future direction suggestions. This research paper attempts to address the following questions:

1) What are the trends related to malware analysis mechanisms that use machine learning?
2) What are the potential issues and challenges related to each trend related to malware analysis mechanism?
3) What could be the future directions of research in this domain that need more investigations?

The rest of the paper is organized as follows. Section II introduces the three basic approaches to malware analysis. Section III presents the search strategy. Followed by section IV which describes different trends in malware analysis using ML. Section V states some challenges related to each trend. In

Section VI, some future directions and work are highlighted. Finally, Section VII concludes this work.

II. Malware Analysis Approaches

Three basic approaches to malware analysis and detection include static, dynamic, and hybrid analysis. All of them play significant roles in the overall malware analysis process and each contributes to the overall malware analysis process in different ways.

Static analysis involves inspecting the structure of an executable file without executing it. The executable file has various static attributes, such as distinct sections and memory compactness. The static analysis involves two parts: basic and advanced. Basic static analysis focuses on the basic properties and features of the malware to gain an initial understanding of its characteristics. File size, file type, and header information are extracted using various tools and examined during basic static analysis. After basic static analysis, advanced static analysis techniques can be applied to gain a deeper understanding of the malware's behavior and capabilities. The advanced static analysis features require more in-depth tools to extract and uncover the actual behavior of malware. Advanced static analysis involves investigating the program commends in detail [8]. The static analysis faces challenges in detecting obfuscated malware, as it is unable to effectively analyze packed samples. This limitation has been highlighted by Komatwar and Kokare [9].

Dynamic analysis involves executing the program instructions and examining the behavior of malware. To secure the machine from being affected by the malware, the dynamic analysis is conducted within an isolated environment, like a virtual machine or sandbox. Dynamic analysis can be divided into two parts: basic and advanced. In basic dynamic analysis, monitoring tools are used to examine the behaviors of malware. On the other hand, advanced dynamic analysis involves the use of debugging tools. These debuggers enable the analysts to run individual commands, with the ability to modify parameters and variables [8]. During dynamic analysis, the software operates in an environment where it has full access to all resources. At the conclusion of malware execution, the controlled environment is restored to its default state, which was captured at the beginning of the environment setup. An agent within the controlled environment logs the software's behavior [10], [11]. Unlike static analysis, dynamic analysis can deal with obfuscation and detect new malware.

Hybrid analysis is an approach that combines both static analysis and dynamic analysis techniques to detect and analyze malware. This method begins by initially analyzing the malware statically, examining its code and structure without executing it. Then, dynamic analysis is applied to enhance the overall analysis further. By incorporating dynamic analysis, the hybrid approach overcomes the limitations of applying static or dynamic analysis alone. It allows for a more comprehensive understanding of the malware's behavior. The combination of both analysis approaches enhances the overall analysis process and improves the effectiveness of malware detection and analysis [12].

Table I presents the pros and the cons of each approach [13],[14].

TABLE I. Comparison between Malware Analysis Approaches

Approaches	Pros	Cons
Static analysis	Require less time and less power and memory consumption.	Can't detect obfuscation and unknown malware.
Dynamic analysis	The ability to detect unknown malware	Consumes higher amount of resources, unsafe, and require more time.
Hybrid analysis	Result in more accurate result.	Higher complexity and higher cost.

Based on a study that was conducted by Gorment et al. [1] for ML algorithms for malware detection, they found out that most contributions on this domain use static analysis with a percentage of 53.3% of the related studies, while dynamic analysis accounted for 28.9% and followed by hybrid analysis with percentage 17.8%. This demonstrates the high effectiveness of static analysis as it is fast and safe. In addition, it has a small false positive rate compared to dynamic analysis [15].

III. Research Strategy

This section outlines the methodology employed in this study, including the search strategy and the criteria used to determine the final set of papers included in the analysis. Google Scholar and other databases such as IEEE and ResearchGate are used to search for existing related literature. We have included research papers that are related to Malware analysis and ML. After that, the duplicated papers were excluded. Also, some papers were excluded for other reasons, such as they are not relevant, the whole paper is not available, only the abstract is provided, and some papers are written in foreign languages.

To make sure that we include recent and relevant research, we selected papers published within the past four years. Moreover, priority was given to papers that focused on ideas or objectives that have been the focus in recent years and had not been explored by researchers in previous years. The selected publications specifically emphasized contributions related to malware analysis or detection in general, with a particular focus on ML-based methods for malware analysis. By following this methodology, the goal of this study is to gather a comprehensive and up-to-date collection of literature that addresses the intersection of malware analysis and ML.

IV. Trends in Malware Analysis using Machine Learning

Numerous techniques, trends, and strategies have been proposed for malware analysis and detection that make use of ML. This section explores trends related to this domain that was introduced and proposed by researchers in recent years as the field of malware analysis and ML is dynamic and continuously improving.

A. Deep Learning-based Malware Analysis

DL is an advanced subset of ML that brings ML closer to the field of artificial intelligence. It enables the modeling of complicated relationships and concepts by employing multiple layers of representation [16]. The motivation behind DL is the function and organization of the human brain and its interconnected network of neurons. It consists of multiple

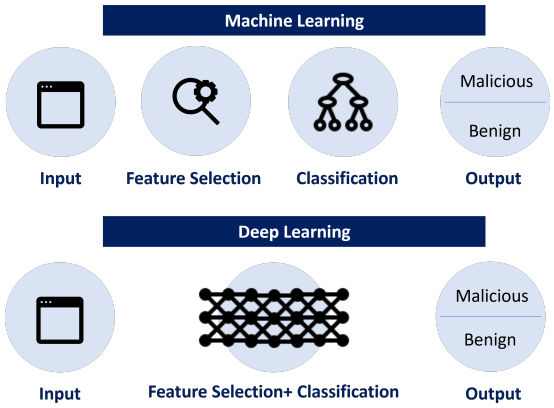

Fig. 1. Difference between deep learning and machine learning.

layers of connected nodes, called neurons or units. Each neuron takes inputs and then performs some computation to produce an output that is used as input to the next layer. The layers are organized hierarchically. In DL, each layer of a neural network learns more abstract concepts from the data. The higher layers build based on the representations learned by the lower layers. DL does not require explicit feature engineering or preprocessing of the data. Instead, it automatically extracts relevant features and representations directly from the raw data [17].

The motivation behind the adoption of DL in many fields is the need to organize and analyze massive volumes of data efficiently. Moreover, DL models have the capability to learn and extract relevant features directly from the raw input data during the training process [18], which is the main difference between ML and deep learning as shown in Fig. 1. It is commonly preferred and used in many domains, including image processing, speech processing, healthcare, and the rapidly expanding field of cybersecurity, which has seen a surge in demand for DL techniques [15].

Rhode et al. [19] proposed a Recurrent Neural Network (RNN) model for malware analysis and prediction. Their focus was on examining the possibility of predicting that an executable file is malicious or normal using a brief snapshot of behavioral data. They found out that employing an ensemble of RNN allows for accurate classification of whether an executable is malicious or benign within the initial five seconds of executing the file, achieving an impressive accuracy rate of 94%. Elayan and Mustafa [20] made use of two static features of Android applications, which are Application Program Interface (API) and permission, to present an approach for detecting malware in Android applications. Gated Recurrent Units (GRUs) were used, which is a type of RNN. GRU consisted of three blocks: input block, middle block, and output block. The approach achieved good results, correctly predicting 98.0% of the dataset. It showed high scores in both recall and precision, successfully detecting 99.2% of the malware samples.

Catak et al. [21] developed a classification method using Long Short-Term Memory (LSTM). LSTM is a DL method that was developed as an improvement over RNN. LSTM was specifically designed to address the limitations of RNN. The malware is analyzed and detected based on the API class that the Windows operating system makes; this is used as a representation of the patterns of malicious software. The classifier's result indicates a high level of accuracy, reaching up to 95%. McDole et al. [22] introduced a study focusing on the analysis of malware detection techniques in cloud Infrastructure-as-a-Service (IaaS) environments, along with exploring the potential of using CNN. The proposed malware detection by Ravi et al. [23] used a CNN to detect malware in smart healthcare systems for both Windows and Android. The proposed approach achieved an accuracy of 98% in the Windows dataset and 97% in the Android dataset.

Bayazit et al. [24] developed comparative systems that is based on DL for malware detection, where they employed various approaches and compared the results of each approach. The system used both Static and dynamic analysis with different ML and DL classifiers. Moreover, A comparative analysis is conducted, comparing traditional ML algorithms such as Decision Trees, Random Forests, and DL algorithms including LSTM, CNN-LSTM, ANN, and Multilayer Perceptron (MLP). They found that LSTM achieved a high accuracy rate of 98.75% in static analysis classification. Additionally, the CNN-LSTM deep learning algorithm showed a high accuracy rate of 95.26% in dynamic analysis classification.

İbrahim et al. [25] introduced an approach that used static analysis and the most important features from Android applications, including two newly proposed features. These features are then used as input for an API DL model specifically developed for this purpose. The proposed method is implemented and evaluated using a classified dataset of Android applications. They focused on extracting the following features: permissions, API calls, services, broadcast receivers, and opcode sequences. Furthermore, they introduced two new static features: application size and fuzzy hash.

Patil and Deng [26] showed how accuracy could be improved using DL networks rather than the traditional ML models by introducing a neural networks-based framework for malware analysis that achieved high accuracy. The findings from the experiment indicated that the DL-based malware classification method achieves high accuracy in classification. Moreover, they suggested that the backpropagation and gradient descent mechanisms in DL help improve accuracy, true positive rate, and reduced false positive rate.

Rodrigo et al. [27] designed a hybrid ML model for Android malware detection. The model contained three fully connected neural networks: The first network focused on static features and achieved an accuracy of 92.9% when trained on 840 static attributes. The second network is designed for dynamic analysis and achieved an accuracy of 81.1% when trained on 3722 dynamic attributes. The last network combined both static and dynamic features, resulting in a hybrid model that achieved an accuracy of 91.1% when trained on 7081 static and dynamic attributes. This demonstrates that the hybrid analysis performs better than using static and dynamic features.

Obaidat et al. [28] presented an approach combining static analysis techniques with advanced image-based DL algorithms to improve the accuracy of malware detection in Java bytecode. The proposed approach, called Jadeite, is designed to classify Java bytecode files as malicious or benign using a combination

of supervised DL and static analysis. It takes as input a JAR file that contains the Java bytecode and consists of three main components. The first component is the Bytecode Transformation Engine, which converts the Java bytecode file into a grayscale image. The second component is the Feature Extraction Engine, which takes the features from the bytecode file. Finally, the CNN classifier Engine takes both the grayscale image and the features obtained from the previous steps. It employs a CNN model to detect if the file is malicious or benign.

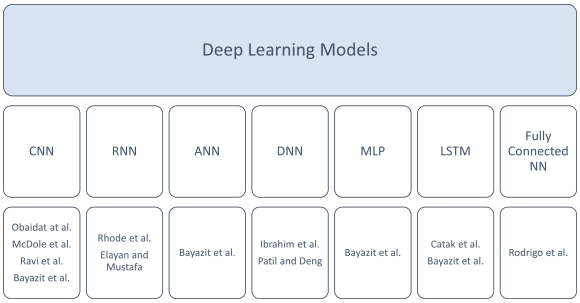

Fig. 2. Deep learning models used in the research papers.

Fig. 2 presents the deep learning techniques used by the authors in the research papers. The Figure shows that the most commonly used deep learning technique is CNN which indicates its effectiveness and efficiency in malware analysis. Following CNN, RNN is the next commonly used technique, followed by DNN and LTSM. Below is a comparative analysis these DL algorithms in malware detection:

- **CNNs** can learn the key and important features without human involvement, which is its main advantage. it is commonly used in computer vision fields for image processing and pattern recognition. Hence, it has been successful in image-based malware detection by extracting features from binary or grayscale representations of malware [29].
- **RNNs** are usually used for analyzing sequential data, making them applicable for code-based malware detection. They can capture dependencies and patterns in code instructions, enabling the identification of malicious behavior [30].
- **DNNs** also known as feedforward neural networks, can learn complex patterns and relationships in the input data, making them capable of detecting malware based on learned representations. Wang et al. [31] have stated that recent studies have shown that using DNNs in malware detection allows for the recognition of abstract patterns from extensive collections of malware samples. This enhances the ability to detect various types of malware, providing a more comprehensive approach.
- **LSTMs** are a type of RNN that can effectively capture long-term dependencies in sequential data, making them suitable for code-based malware analysis [30].

B. Transfer Learning-based Malware Analysis

The concept of transfer learning has been explored in the literature since the 1990s, using different names such as learning to learn, life-long learning, and knowledge transfer [32]. It is a valuable technique in ML that addresses the challenge of limited training data. It allows us to use knowledge gained from a source domain and apply it to a target domain, even when the training and test data are dependent and not identically distributed. This approach is particularly helpful in domains where improving performance is difficult due to a lack of enough training data [33]. The main difference between traditional ML and transfer learning lies in the treatment of tasks and the use of previous knowledge. In traditional ML, each task is treated independently, and the model needs to learn from scratch for each task. There is no sharing or transfer of knowledge between tasks, and the model starts from the beginning for each new task. On the other hand, transfer learning involves taking knowledge and insights obtained from previous tasks or domains and transferring them to the learning process of a new task. Instead of starting from scratch, the model can benefit from the knowledge extracted from other source tasks [34]. Fig. 3 shows that the traditional ML learns the task from scratch. This means that they are trained on specific datasets related to the task. On the other hand, transfer learning uses the knowledge gained from a learning system and applies it to another learning system [35].

Transfer learning offers several advantages, including faster learning and reduced reliance on large training data. By applying knowledge from related tasks or domains, the learning process becomes quicker because the model is built based on existing knowledge. Also, the need for a large amount of training data is reduced as the model can generalize effectively by taking advantage of the existing knowledge [32].

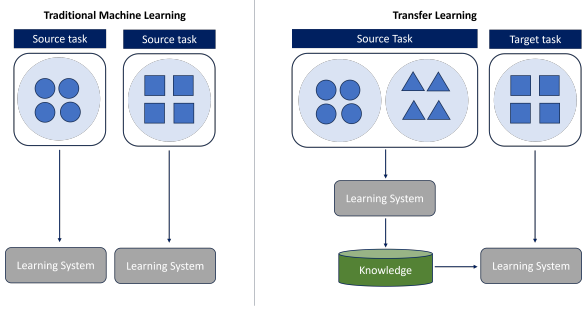

Fig. 3. Learning process of traditional ML and transfer learning.

Chen [36] introduced an approach that used deep transfer learning for static malware classification. He showed that the proposed technique has better performance compared to training from scratch and other traditional ML models. Moreover, the transfer learning scheme speeds up the training phase in deep neural networks while maintaining high performance in classification.

Bhodia et al. [37] used transfer deep learning for malware analysis and detection based on image analysis. The authors converted executable files into images and used DL models for image recognition. To train the models, they used transfer

learning by applying existing DL models that have been trained before on large image datasets. Transfer DL result was compared to a simpler k-Nearest Neighbor approach (k-NN). In certain cases, the k-NN learning technique had better performance than the proposed models. However, in simulated zero-day experiments, the proposed models had better performance compared to k-NN. Ahmad et al. [38] proposed an approach for classifying malware into nine different classes. They treated the malware binaries as image data and applied various ML and DL techniques. Logistic Regression, ANN, CNN, transfer learning on CNN, and LSTM models were used to achieve the classification results. Additionally, they employed transfer learning with InceptionV3 for training, which showed better results, particularly when compared to the LSTM model.

Acharya et al. [39] presented a transfer learning approach for efficient Android malware detection. First, the authors performed malware detection using the traditional ML models such as CNN and then they used the transfer learning technique to compare between them. They transferred the relevant features and information from a model that was trained before to a target model, and it was found that transfer learning resulted in low computational costs.

Prima and Bouhorma [40] suggested a framework for classifying malware using transfer learning. The proposed approach used pre-trained DL models that have been trained on large image datasets. In this framework, the input is a grayscale image that represents the malicious program. Then, this image is passed over a block of convolutional layers. The output is the class of the malware. According to their findings, Prima and Bouhorma stated that CNN shows better performance compared to traditional ML techniques, particularly in domains like image classification. Zhao et al. [41] proposed a malware classification method that incorporates transfer learning, multi-channel image vision features, and CNN. The methodology involves several steps. First, they extracted features from malware samples and converted them into grayscale images of three different types. To ensure a consistent size, they process the grayscale image sizes using an algorithm. Next, they synthesized the three grayscale images into three-dimensional RGB images. These RGB images are than used for training and classification.

Panda et al. [42] used two malware image datasets, namely Malevis and MalImg. For preparing the datasets, they performed pre-processing and resized each image to 224x224 pixels. Since the datasets were relatively small, they applied a technique to increase the amount of data available for training. To evaluate the accuracy of the proposed model, they used the MalImg dataset as a baseline. On the other hand, the Malevis dataset was used to build a transfer learning on CNN. The approach involved developing a transfer learning model from scratch. The extracted features are used as input to three neural network models: the autoencoder, the gated recurrent unit, and the multi-layer perceptron. The result of the multi-layer perceptron model is the final classification by taking the output from the gated recurrent unit model as its input. Tasyurek and Arslan [43] proposed a fast and accurate model that is based on CNN. The model converts the features obtained from the manifest file to an RGB format image. Then, these images are used to train the model with the transfer learning technique. The experiment results in high accuracy equal to 98.3% and quick prediction due to the use of transfer learning.

He et al. [44] claimed that the traditional ML models are not effective in accurately identifying previously unknown and zero-day malware using Hardware-Based Malware Detection. So, they proposed a Hardware-Based Malware Detection based on deep neural networks and transfer learning. the proposed solution which is based on image-based hardware events showed better performance than the existing ML methods. It achieves a high detection rate of 97% at runtime, using only the top four hardware events. Additionally, the solution maintains a low false positive rate and does not require any hardware redesign overhead. Ngo et al. [45] proposed a model that extracts both static and dynamic features for detecting malware. Then, they presented a technique for transferring knowledge that is obtained from a big source model, which is trained on the previously extracted features, to a small target model. The authors stated that the proposed model reduces the time required for prediction.

C. Explainable Machine Learning-based Malware Analysis

In order to create ML models that are reliable to humans, researchers have discussed various techniques for explaining and interpreting these models to users. This field of study, known as "explainability," focuses on reasoning and decision-making processes employed by ML models. Explainability includes any technique that helps users or developers understand the behavior of ML models [46].

The term "XAI" (eXplainable Artificial Intelligence) was introduced by the Defense Advanced Research Projects Agency (DARPA) in 2017. Since then, it has gained popularity across various fields, including healthcare, transportation, legal, finance, military, and scientific research. XML refers to the development and application of ML models and algorithms that are transparent, interpretable, and able to provide understandable explanations for their predictions or decisions. XML aims to bridge the gap between the complexity of traditional ML models and the need for human interpretability and understanding. In traditional ML, models such as deep neural networks can be complex and hard to understand. They operate as black boxes, making predictions without providing any explanation for the underlying reasoning or factors that affect the output. This lack of transparency can be a challenging issue, especially in domains where trust, accountability, and interpretability are important, such as healthcare, finance, and legal systems [47]. In contrast, the goal of XML techniques is to make models more transparent by providing explanations for their predictions.

Alani and Awad [48] introduced a lightweight Android malware detection system that applies of XML techniques. The system uses static features extracted from applications to classify the application as malicious or benign. The results of the experiment show an accuracy rate of over 98%, while the system remains lightweight and consumes little device's resources. Moreover, the classifier model is interpreted using Shapley Additive Explanation (SHAP) values. Liu et al. [49] also have proposed Android malware detection based on XAL. This study takes a different approach compared to other research. Instead of focusing on evaluating how well ML classifiers detect malware and identify the causes, it applies

XML approaches. The goal is to understand what ML-based models learn during training. They stated that authors should have a better understanding of how ML models work, rather than just focusing on improving the accuracy. ladarola et al. [50] presented a DL model that is based on images for detecting which family the malware belongs to. Moreover, they explained the system prediction by using LIME as the explanation method. They achieved high accuracy equal to 93.4%.

Manthena [51] stated that many existing studies in the field of malware analysis lack transparency and explainability regarding the predictions made by their models. This absence of information about the models' decisions is a significant issue, particularly in the context of malware analysis. Manthena presented an online malware analysis by applying XML, such as KernalSHAP, TreeSHAP, and DeepSHAP to analyze and evaluate the reliability of the performance metrics. By doing so, she emphasized the importance of applying XML techniques in the context of online malware detection. While Manthena used SHAP as an XML technique, Kinkead et al. [52] used Local Interpretable Model-Agnostic Explanations (LIME) to explain the predictions of a ML model. They have developed a method that used CNN to identify significant locations within an Android app's opcode sequence. These identified locations are believed to play a key role in detecting malware. Secondly, they have conducted a comparison between the locations highlighted by CNN and the locations identified as important by LIME. This comparison allows us to evaluate the consistency between the two methods in identifying significant locations for malware detection. After conducting the experiment, they found out that the positions in the opcode sequences classified as malicious by CNN closely match those classified as most malicious by LIME.

Lu and Thing [53] proposed an android malware detection using three model explanation methods, which are Modern Portfolio Theory (MPT), SHAP, and LIME. They conducted an experiment to compare these methods regarding the explanation ability. The result of the experiment showed that the MPT is considered valuable to security analysis as it can be used to determine the reasons that the classifiers are fooled by the adversarial samples. Iadarolaa et al. [54] proposed a method for Android malware detection and family identification that depends on representing applications as images. These images are then used as input to an explainable DL model specifically designed by the authors. The purpose of this approach is to provide a transparent and interpretable solution for detecting and identifying malware in Android applications. The proposed methodology can be divided into two main parts. The first part involves training the model using appropriate techniques and data to achieve perfect performance. The second part of the methodology is responsible for the interpretability of the model's learning process. This aspect emphasizes understanding and explaining how the model makes predictions or decisions. Pan et al. [55] proposed a solution that focuses on overcoming the limitations of current malware detection methods, which include prediction inaccuracy and a lack of transparency. To address these two challenges, they have developed a hardware-assisted malware detection framework using an XML algorithm based on regression.

Manthena et al. [56] addressed the block box characteristics found in many ML and DL models such as CNN and Feed-Forward Neural Net (FFNN) by proposing a malware detection system that is trained on an online dataset and using SHAP in order to explain the outcome of the model. Sharma et al. [57] proposed an explainable system for malware detection using traffic analysis. Explainability in this model is gained by using features of the network traffic that are understandable by humans and interpretable ML decision trees for detecting malware.

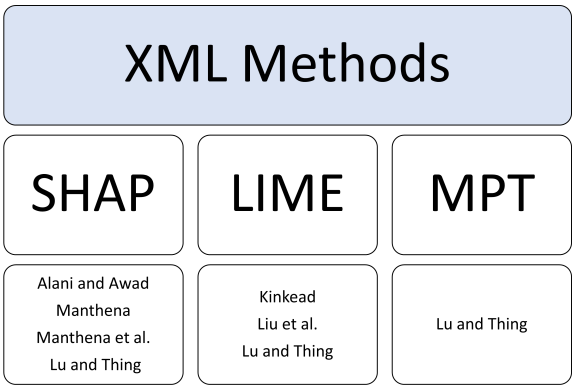

Fig. 4. XML Methods.

Fig. 4 shows that the most commonly used methods for model explanation in malware analysis are SHAP and LIMA. The main difference between these methods is that SHAP provides a general explanation, whereas LIME produces local reasoning. This means that SHAP provides an explanation regarding the performance of the model across all samples. LIMA Interpret the prediction of a model for a specific sample[58].

Table II provides a summary of the analyzed papers for the reader and researcher to have a clear understanding.

D. Major Findings and Discussion

This section discusses some key findings based on the surveyed works in the above section. According to the previous section, the majority of current studies make use of DL models. The use of DL models allows for effective detection and classification of malware based on complex patterns and features. Additionally, transfer learning has been widely applied in malware analysis, using pre-trained models on large datasets to enhance the performance of malware detection systems. Moreover, XML techniques have gained attention in the field of malware analysis. XML methods provide an understanding of the decision-making process of the models, that ensure transparency and interpretability. Overall, the surveyed works show the effectiveness and importance of using DL, transfer learning, and XML techniques in the context of malware analysis.

The method of converting the samples or the files to an image before processing is used by many researchers based on surveyed work, such as [28], [37], [38], [40], [41], [42], [43], [44], [50], [54].

TABLE II. SUMMARY OF RECENT WORKS

Authors	Pub. year	Focus/Objective	Platform	Machine-learning techniques used
Rhode et al. [19]	2018	Performing early detection of Maliciousness of a file within the initial 5 seconds of its execution.	Windows 7 Executable file	RNN
Elayan and Mustafa [20]	2021	Utilizing DL techniques, specifically the GRU architecture of Recurrent Neural Networks to detect malware in Android applications.	Android	
Obaidat et al. [28]	2022	Identifying Java bytecode malware programs through static analysis and image-based DL classification.	Java program	CNN
McDole et al [22]	2021	Analyzing malware in a cloud environment by utilizing DL.	Cloud IaaS	
Ravi et al. [23]	2022	Proposing a method for malware detection on Windows and Android operating system in smart healthcare systems.	Windows and Android OS	
Catak et al. [21]	2020	Using LSTM for malware analysis based on API calls in Windows operating systems.	Windows OS	LSTM
Bayazit et al. [24]	2023	Present and compare the use of static and dynamic analysis in DL-based malware binary classification.	Android	LSTM, CNN-LSTM, ANN and MLP .
Rodrigo et al. [27]	2021	Using hybrid ML for Android malware detection.	Android	Fully connected neural networks
İbrahim et al. [25]	2022	Using static analysis for android malware detection.	Android	Deep learning
Patil and Deng [26]	2020	Comparing the accuracy of the DL approach and traditional ML in malware analysis.	-	
Kinkead et al. [52]	2021	Using Explainable CNNs for developing Android Malware Detection method.	Android	.
ladarola et al. [50]	2023	Explinable DL model on images for Malware fimaly detection.	-	Explainable CNNs
Pan et al. [55]	2020	Addressing two limitations: inaccuracy in predictions and lack of transparency	-	Explainable RNN
Liu et al. [49]	2022	Researchers should focus on having a better understanding of how ML models work, rather than just focusing on improving the accuracy of the models.	Android	Explainable machine learning
Manthena [51]	2022	Addressing the lack of Explainable ML approaches for online malware analysis.	-	

Alani and Awad [48]	2022	Explaining the reasons behind the selected features using Shapley additive explanation values to ensure that the high accuracy of the classifier come from explainable conditions.	Android	Explainable machine learning
Manthena et al. [56]	2023	Addressing the block box issue that found in many Ml and DL models	-	
Lu and Thing [53]	2022	Designing Android malware analysis and Comparing the explanation between different explanation methods.	Android	
Sharma et al. [57]	2023	Explainable malware detection using network traffic features	-	Explainable Decision tree
Iadarolaa et al. [54]	2021	Using explainable deep learning model for detecting malware and identifying their family.	mobile	Explainable deep learning
Chen [36]	2019	Demonstrating that transfer learning outperforms training from scratch.	-	Transfer deep learning
Bhodia et al. [37]	2019	Malware detection and classification based on image analysis	Executable files	
Prima and Bouhorma [40]	2020	CNN have better performance compared to traditional learning techniques.	-	
Panda et al. [42]	2023	Developing a transfer learning model from scratch	IoT	Transfer learning on CNN
Acharya et al. [39]	2023	Using transfer learning for mobile malware detection to lower the computational cost.	Mobile	
Zhao et al. [41]	2023	Solving the problem of existing DL model for malware detection which is long training time	-	
Ahmad et al. [38]	2023	Classifying malware, with nine different class using CNN-transfer learning model	-	
Ngo et al. [45]	2023	Addressing the limitations of feature aggregation while using static and dynamic features and transferring knowledge from big to small models.	-	
Tasyurek and Arslan [43]	2023	Fast CNN-based transfer learning for malware analysis	Android	
He et al. [44]	2022	Using a deep neural network and transfer learning for detecting zero-day malware.	-	Transfer learning on DNN

Usually, the malware sample comes as an executable file in binary format. To perform malware representation, the byte in the binary file is converted to pixels in the image containing textural patterns which results in better visualization of the malware [59], [60]. Malware representation of a file before processing it is very common in malware analysis files because by using this approach, researchers avoid the need for and dependency on features engineering process and methods [61], [62]. Moreover, it allows for a visual representation of the malware and incorporates computer vision with this domain [63].

V. Limitations and Challenges

Over time, various trends in malware detection and analysis have emerged, each with its limitations and drawbacks. As a result, researchers have shifted their focus to alternative technologies that aim to detect malware in real-time with minimal false positives and increase detection and classification accuracy. This section discusses the challenges and limitations associated with each trend based on the analyzed papers in the previous sections.

A. Limitations and Challenges Found in Papers using Deep Learning

- Rhode et al. [19] used RNN for early-stage malware detection assuming that the malware will execute the malicious activities within the first five seconds. But what if the attacker or adversaries are aware that the first 5 seconds of a file's execution are used to determine whether it is malicious or not, they can manipulate the file's behavior to avoid detection. By introducing long periods of sleep or inactivity at the beginning of a malicious file, so, the attacker can trick the system into classifying the file as safe. We suggest incorporating additional features or mechanisms in the system to capture and analyze behavior beyond the initial 5 seconds, such as by extracting and analyzing a broader range of features, like system calls, or network activity to gain a deeper understanding of the file's intent.
- Obaidat et al. [28] used DL with a large dataset for Java malware detection. DL architectures rely heavily on supervised learning, which requires a large amount of labeled samples to train the model effectively. As mentioned by Kadam and Vaidya in [64], using a small number of samples does not allow the model to learn the underlying features accurately in the training stage, and processing such a large amount of data in DL requires extensive training time and needs high processing power.
- The dataset used by Catak et al. [21] has an unequal distribution of instances for each malware Category. For example, there are 1001 rows labeled as Worms and only 379 samples for Adware. This may affect the performance of the model in classification [65]. To address this issue, we can use data resampling techniques. Data resampling is used for class-imbalance problems which aim to balance the distribution of instances across different classes by either increasing the number of instances in the minority class (oversampling) or decreasing the number of instances in the majority class (undersampling). This helps to mitigate the impact of class imbalance on the model's training process [66].

B. Limitations and Challenges Found in Papers using Transfer Learning

- In the work of Panda et al. [42], the transfer learning model may struggle to handle malware images with varying sizes. Inconsistencies in image sizes can lead to challenges in the model's ability to learn meaningful features and patterns. Resizing or standardizing the images to a fixed size is typically required as a pre-processing step, which can add complexity and potential loss of information. Moreover, the model proposed in this paper is incapable of detecting unseen malware. this may lead to a prolonged classification process because of the need for pre-processing the malware images before classifying the malware.
- Niu et al. [67] and Pan and Yang [68] mentioned that transfer learning suffers from an issue called negative transfer. Negative transfer refers to a phenomenon in which the application of transfer learning leads to a decrease in performance or some impact on the target task. It occurs when the knowledge learned from the source domain is not relevant or compatible with the target task. So, instead of benefiting the target task, the transferred knowledge may introduce noise or incorrect assumptions that affect the model's capability to generalize and make accurate predictions. We can mitigate the risk of negative transfer by using an ensemble of models trained with different source domains. By combining different sources of transferred knowledge, the ensemble can take advantage of the strengths of each model and reduce the impact of negative transfer.
- Based on paper of Prima and Bouhorma [40] and Panda et al. [42], We can conclude that transfer learning requires large datasets. Even though the use of a large dataset can achieve high accuracy, it is time-consuming and may require more computational resources [69]. Incremental learning could partially solve this problem. Instead of training the model on the entire large dataset at once, incremental learning can be employed to train the model on smaller subsets of data sequentially, gradually increasing the complexity of the task. This way, computational resources can be used more efficiently, and the time required for training can be reduced.

C. Limitations and Challenges Found in Papers using XML

- The work proposed by Alani and Awad [48] has been acknowledged for its accuracy and efficiency, requiring minimal time for malware detection. However, a limitation of this approach is its inability to effectively detect unknown and obfuscated malware. This limitation arises from the fact that the proposed method relies on static features for malware detection. Static features typically refer to characteristics

extracted from the file or code without considering dynamic behaviors or run-time information. Applying dynamic analysis along with static analysis can provide more valuable information regarding the behavior of malware during execution because dynamic analysis can capture actions such as file system modifications, network communications, process interactions, and system calls, which can help identify malicious activities and detect previously unknown or obfuscated malware.

- In many cases, there exists a trade-off between model performance and interpretability as stated by Antoniadi et al. [70] and Arrietaet al.[71]. This means that as models become more interpretable, their predictive accuracy may decrease. Conversely, highly accurate models may be less interpretable. making a balance between interpretability and performance is a challenge in XML, as it requires finding the right level of transparency without decreasing the accuracy.
- The split of the dataset in Kinkead et al. [52] for training, validation, and testing was not fair. A balanced split is typically recommended to make sure that both the training and testing samples have a representative distribution of malware and benign samples. the best resampling technique for this case is cross-validation which is considered the standard resampling technique for splitting the dataset into training, validation, and testing sets [72].
- Unlike accuracy or precision, interpretability lacks standard evaluation metrics. Measuring the quality of explanations is still an active area of research. The absence of widely accepted evaluation criteria makes it challenging to compare and evaluate different XML methods [73] [74].

VI. Future Directions

In the previous section, several issues and limitations that need to be addressed to develop a system that has the ability to detect, analyze, and classify malware efficiently are listed. In this section, we will highlight some areas for future work that researchers can use to help mitigate the mentioned issues:

A. The use of a Moderate-sized and Balanced Dataset

As mentioned in the previous section some research papers used very large datasets and other papers suffered from unbalanced datasets. This problem is because the researchers believe that larger datasets lead to higher accuracy and reduce bias. While this belief is common, it is important to consider that there are potential challenges linked with very large datasets. By working with a dataset of moderate size, computational resources and time required for training models can be reduced. Another issue that needs to be taken care of is the use of up-to-date datasets because it is important for maintaining the relevance and applicability of the study in real-time scenarios.

B. Domain Distance Measure

Addressing the issue of negative transfer in transfer learning is an important research challenge that needs to be addressed in the future. Negative transfer leads to worse performance compared to not transferring at all. Therefore, it is essential to find ways to prevent negative transfer from happening in transfer learning. An accurate measurement of domain distance is another important research aspect that needs attention to solve this problem. When applying transfer learning techniques, it is essential to rate the similarity or dissimilarity between the source and target domains correctly. Existing approaches for measuring domain distance often rely on assumptions that may not capture the true underlying relationships between domains. This can cause inappropriate transfer of knowledge and performance degrading. Developing robust and accurate methods for evaluating domain distance will enable better identification of relevant knowledge for transfer and make the adaptation of models to new domains more effective [67].

C. Image Resizing and Standardization

Resizing images to a fixed size is a common preprocessing step to address inconsistencies in image sizes. This step ensures that the input images have a consistent size and format, which is essential for many ML models. However, it's important to carefully consider the impact of resizing on the loss of information [75]. So further investigation regarding resizing techniques to minimize information loss while maintaining a consistent and reasonable image size for efficient processing is needed.

D. Distributed Computing and Parallelization

To facilitate the training process and reduce the time required for large dataset processing, researchers should consider making use of distributed computing techniques and parallelization. This involves distributing the workload across multiple computing resources, such as GPUs or multiple machines, to train the model in parallel. Parallelization can significantly reduce the training time and reduce the memory resource requirements [76]

E. Comparative Evaluations

We recommend more future research regarding evaluating the quality of the explanation of an XML. One way to solve this is by comparative evaluations. It means comparing different XML methods on fixed benchmarks or datasets. By applying multiple XML methods to the same dataset or task, their performance in terms of interpretability can be compared. This can be done using qualitative analyses by experts or by developing quantitative metrics that consider different elements of interpretability.

VII. Conclusions

Malware analysis is a critical component of cybersecurity due to the increasing sophistication and the widespread of malicious software. Understanding malware is key to developing strong defenses. Malware analysis helps identify and classify different types of malware, which makes it easier to detect and prevent future attacks. ML plays a key role in malware analysis due to its ability to analyze large amounts of data and detect complex patterns. In this paper, we provide a survey of existing trends related to malware analysis using

ML including a description of each trend. The surveyed works show the effectiveness and importance of applying DL, transfer learning, and XML techniques in the context of malware analysis. These approaches contribute to improved accuracy, interpretability, and transparency in detecting and analyzing malware. Moreover, the challenges and limitations related to each trend are explored. Based on the survey results, we also provide some future directions to be investigated that have the potential to shape the future of malware analysis. These areas may offer exciting opportunities for further improvement in the field in order to overcome the challenges faced by the researchers.

Despite the valuable insights this paper provides, it is important to acknowledge its limitations. The sample size was relatively small, which may limit the generalizability of the findings. Future research should aim to replicate these findings with larger samples.

ACKNOWLEDGMENT

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [Grant No. 5520].

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] U. V. Nikam and V. M. Deshmuh, "Performance evaluation of machine learning classifiers in malware detection," in *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*. IEEE, 2022, pp. 1–5.

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

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

[5] I. H. Sarker, A. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big data*, vol. 7, pp. 1–29, 2020.

[6] I. Sarker, "Machine learning: algorithms, real-world applications and research directions. sn comput sci 2: 160," 2021.

[7] M. T. Ahvanooey, Q. Li, M. Rabbani, and A. R. Rajput, "A survey on smartphones security: software vulnerabilities, malware, and attacks," *arXiv preprint arXiv:2001.09406*, 2020.

[8] Ö. Aslan and A. A. Yilmaz, "A new malware classification framework based on deep learning algorithms," *Ieee Access*, vol. 9, pp. 87 936–87 951, 2021.

[9] 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.

[10] M. Ijaz, M. H. Durad, and M. Ismail, "Static and dynamic malware analysis using machine learning," in *2019 16th International bhurban conference on applied sciences and technology (IBCAST)*. IEEE, 2019, pp. 687–691.

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

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

[13] N. K. Gyamfi and E. Owusu, "Survey of mobile malware analysis, detection techniques and tool," 11 2018, pp. 1101–1107.

[14] V. Rao and K. Hande, "A comparative study of static, dynamic and hybrid analysis techniques for android malware detection," *International Journal of Engineering Development and Research*, vol. 5, no. 2, pp. 1433–1436, 2017.

[15] U.-e.-H. Tayyab, F. B. Khan, M. H. Durad, A. Khan, and Y. S. Lee, "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] N. Rusk, "Deep learning," *Nature Methods*, vol. 13, no. 1, pp. 35–35, 2016.

[18] Karthick Jonagadla, "Deep learning in financial markets," https://www.quantace.in/deep-learning-application-financial-markets/, 2023, accessed: November 9, 2023.

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

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

[21] F. O. Catak, A. F. Yazı, O. Elezaj, and J. Ahmed, "Deep learning based sequential model for malware analysis using windows exe api calls," *PeerJ Computer Science*, vol. 6, p. e285, 2020.

[22] A. McDole, M. Gupta, M. Abdelsalam, S. Mittal, and M. Alazab, "Deep learning techniques for behavioral malware analysis in cloud iaas," *Malware analysis using artificial intelligence and deep learning*, pp. 269–285, 2021.

[23] V. Ravi, M. Alazab, S. Selvaganapathy, and R. Chaganti, "A multi-view attention-based deep learning framework for malware detection in smart healthcare systems," *Computer Communications*, vol. 195, pp. 73–81, 2022.

[24] E. Calik Bayazit, O. Koray Sahingoz, and B. Dogan, "Deep learning based malware detection for android systems: A comparative analysis," *Tehnički vjesnik*, vol. 30, no. 3, pp. 787–796, 2023.

[25] M. İbrahim, B. Issa, and M. B. Jasser, "A method for automatic android malware detection based on static analysis and deep learning," *IEEE Access*, vol. 10, pp. 117 334–117 352, 2022.

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

[27] C. Rodrigo, S. Pierre, R. Beaubrun, and F. B. El Khoury, "A hybrid machine learning-based malware detection model for android devices. electronics 2021, 10, 2948," 2021.

[28] 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, p. 102547, 2022.

[29] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, pp. 1–74, 2021.

[30] P. Maniriho, A. N. Mahmood, and M. J. M. Chowdhury, "A survey of recent advances in deep learning models for detecting malware in desktop and mobile platforms," *arXiv preprint arXiv:2209.03622*, 2022.

[31] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia, X. Xing, X. Liu, and C. L. Giles, "Adversary resistant deep neural networks with an application to malware detection," in *Proceedings of the 23rd ACM sigkdd international conference on knowledge discovery and data mining*, 2017, pp. 1145–1153.

[32] R. Ribani and M. Marengoni, "A survey of transfer learning for convolutional neural networks," in *2019 32nd SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)*. IEEE, 2019, pp. 47–57.

[33] 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: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*. Springer, 2018, pp. 270–279.

[34] H. M. K. Barznji, "Transfer learning as new field in machine learning," 2020.

[35] M. Ranaweera and Q. H. Mahmoud, "Virtual to real-world transfer learning: A systematic review," *Electronics*, vol. 10, no. 12, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/12/1491

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

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

[38] 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. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666603022000252

[39] S. Acharya, U. Rawat, and R. Bhatnagar, "A computationally inexpensive method based on transfer learning for mobile malware detection," in *Proceedings of Fourth International Conference on Computer and Communication Technologies: IC3T 2022*. Springer, 2023, pp. 263–274.

[40] 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.

[41] 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, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/4/2484

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

[43] 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. 1–17, 2023.

[44] Z. He, A. Rezaei, H. Homayoun, and H. Sayadi, "Deep neural network and transfer learning for accurate hardware-based zero-day malware detection," in *Proceedings of the Great Lakes Symposium on VLSI 2022*, 2022, pp. 27–32.

[45] M. V. Ngo, T. Truong-Huu, D. Rabadi, J. Y. Loo, and S. G. Teo, "Fast and efficient malware detection with joint static and dynamic features through transfer learning," in *International Conference on Applied Cryptography and Network Security*. Springer, 2023, pp. 503–531.

[46] U. Bhatt, A. Xiang, S. Sharma, A. Weller, A. Taly, Y. Jia, J. Ghosh, R. Puri, J. M. Moura, and P. Eckersley, "Explainable machine learning in deployment," in *Proceedings of the 2020 conference on fairness, accountability, and transparency*, 2020, pp. 648–657.

[47] X. Zhong, B. Gallagher, S. Liu, B. Kailkhura, A. Hiszpanski, and T. Y.-J. Han, "Explainable machine learning in materials science," *npj Computational Materials*, vol. 8, no. 1, p. 204, 2022.

[48] M. M. Alani and A. I. Awad, "Paired: An explainable lightweight android malware detection system," *IEEE Access*, vol. 10, pp. 73 214–73 228, 2022.

[49] Y. Liu, C. Tantithamthavorn, L. Li, and Y. Liu, "Explainable ai for android malware detection: Towards understanding why the models perform so well?" in *2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2022, pp. 169–180.

[50] G. ladarola, F. Mercaldo1, F. Martinelli, and A. Santone, "Assessing deep learning predictions in image-based malware detection with activation maps," in *Security and Trust Management: 18th International Workshop, STM 2022, Copenhagen, Denmark, September 29, 2022, Proceedings*, vol. 13867. Springer Nature, 2023, p. 104.

[51] H. Manthena, "Explainable machine learning based malware analysis," Ph.D. dissertation, North Carolina Agricultural and Technical State University, 2022.

[52] 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.

[53] Z. Lu and V. L. Thing, ""how does it detect a malicious app?" explaining the predictions of ai-based malware detector," in *2022 IEEE 8th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*. IEEE, 2022, pp. 194–199.

[54] 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, p. 102198, 2021.

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

[56] 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. 25 237–25 252, 2023.

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

[58] K. Safjan, "Explaining ai - the key differences between lime and shap methods," *Krystian's Safjan Blog*, 2023.

[59] "An enhancement for image-based malware classification using machine learning with low dimension normalized input images," *Journal of Information Security and Applications*, vol. 69, p. 103308, 2022.

[60] A. Bensaoud, N. Abudawaood, and J. Kalita, "Classifying malware images with convolutional neural network models," *International Journal of Network Security*, vol. 22, no. 6, pp. 1022–1031, 2020.

[61] M. U. Demirezen, "Image based malware classification with multimodal deep learning," *International Journal of Information Security Science*, vol. 10, no. 2, pp. 42–59, 2021.

[62] Z. Zhao, D. Zhao, S. Yang, L. Xu *et al.*, "Image-based malware classification method with the alexnet convolutional neural network model," *Security and Communication Networks*, vol. 2023, 2023.

[63] B. Saridou, I. Moulas, S. Shiaeles, and B. Papadopoulos, "Image-based malware detection using & alpha;-cuts and binary visualisation," *Applied Sciences*, vol. 13, no. 7, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/7/4624

[64] S. Kadam and V. Vaidya, "Review and analysis of zero, one and few shot learning approaches," in *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) held in Vellore, India, December 6-8, 2018, Volume 1*. Springer, 2020, pp. 100–112.

[65] V. Khullar, M. Angurala, K. D. Singh, P. Prasant, V. Pabbi, and M. Veeramanickam, "Exploring methods for dealing with class imbalances in supervised machine learning structured datasets," in *2023 3rd International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS)*. IEEE, 2023, pp. 209–214.

[66] A. Estabrooks, T. Jo, and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," *Computational intelligence*, vol. 20, no. 1, pp. 18–36, 2004.

[67] S. Niu, Y. Liu, J. Wang, and H. Song, "A decade survey of transfer learning (2010–2020)," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, 2020.

[68] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[69] H. O. Ikromovich and B. B. Mamatkulovich, "Facial recognition using transfer learning in the deep cnn," *Open Access Repository*, vol. 4, no. 3, pp. 502–507, 2023.

[70] A. M. Antoniadi, Y. Du, Y. Guendouz, L. Wei, C. Mazo, B. A. Becker, and C. Mooney, "Current challenges and future opportunities for xai in machine learning-based clinical decision support systems: a systematic review," *Applied Sciences*, vol. 11, no. 11, p. 5088, 2021.

[71] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information fusion*, vol. 58, pp. 82–115, 2020.

[72] B. Vrigazova, "The proportion for splitting data into training and test set for the bootstrap in classification problems," *Business Systems Research: International Journal of the Society for Advancing Innovation and Research in Economy*, vol. 12, no. 1, pp. 228–242, 2021.

[73] A. Das and P. Rad, "Opportunities and challenges in explainable artificial intelligence (xai): A survey," *arXiv preprint arXiv:2006.11371*, 2020.

[74] L. Longo, R. Goebel, F. Lecue, P. Kieseberg, and A. Holzinger, "Explainable artificial intelligence: Concepts, applications, research challenges and visions," in *International cross-domain conference for machine learning and knowledge extraction*. Springer, 2020, pp. 1–16.

[75] J. J. Luke, R. Joseph, and M. Balaji, "Impact of image size on accuracy and generalization of convolutional neural networks," *Int. J. Res. Anal. Rev.(IJRAR)*, vol. 6, no. 1, pp. 70–80, 2019.

[76] K. Zhu, H. Wang, H. Bai, J. Li, Z. Qiu, H. Cui, and E. Chang, "Parallelizing support vector machines on distributed computers," *Advances in neural information processing systems*, vol. 20, 2007.