frontiers | Frontiers in Physics

TYPE Review
PUBLISHED 28 March 2024
DOI 10.3389/fphy.2024.1349463

OPEN ACCESS

EDITED BY
Amin Ul Haq,
University of Electronic Science and Technology of China, China

REVIEWED BY
Rajesh Kumar,
University of Electronic Science and Technology of China, China
Ikram Ud Din,
The University of Haripur, Pakistan
Nuzhat Naqvi,
Mohi-ud-Din Islamic University, Pakistan

*CORRESPONDENCE
Tapan Kumar Das,
tapan.das@vit.ac.in

RECEIVED 04 December 2023
ACCEPTED 15 March 2024
PUBLISHED 28 March 2024

CITATION
Redhu A, Choudhary P, Srinivasan K and Das TK (2024), Deep learning-powered malware detection in cyberspace: a contemporary review.
Front. Phys. 12:1349463.
doi: 10.3389/fphy.2024.1349463

Deep learning-powered malware detection in cyberspace: a contemporary review

Ananya Redhu[1], Prince Choudhary[1], Kathiravan Srinivasan[1] and Tapan Kumar Das[2]*

[1]School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India, [2]School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, Vellore, India

This article explores deep learning models in the field of malware detection in cyberspace, aiming to provide insights into their relevance and contributions. The primary objective of the study is to investigate the practical applications and effectiveness of deep learning models in detecting malware. By carefully analyzing the characteristics of malware samples, these models gain the ability to accurately categorize them into distinct families or types, enabling security researchers to swiftly identify and counter emerging threats. The PRISMA 2020 guidelines were used for paper selection and the time range of review study is January 2015 to Dec 2023. In the review, various deep learning models such as Recurrent Neural Networks, Deep Autoencoders, LSTM, Deep Neural Networks, Deep Belief Networks, Deep Convolutional Neural Networks, Deep Generative Models, Deep Boltzmann Machines, Deep Reinforcement Learning, Extreme Learning Machine, and others are thoroughly evaluated. It highlights their individual strengths and real-world applications in the domain of malware detection in cyberspace. The review also emphasizes that deep learning algorithms consistently demonstrate exceptional performance, exhibiting high accuracy and low false positive rates in real-world scenarios. Thus, this article aims to contribute to a better understanding of the capabilities and potential of deep learning models in enhancing cybersecurity efforts.

KEYWORDS

artificial intelligence, cyberspace data security, deep learning, malware detection, network security

1 Introduction

This comprehensive review delves into the burgeoning role of deep learning (DL) models in the face of the ever-evolving menace of malware in cyberspace. Malware represents a continuously evolving cybersecurity threat, and traditional detection technologies often struggle to keep up with the rapid creation of new malware types [1]. However, deep learning models have gained significance in this field due to their ability to automatically learn features from large datasets [2]. Deep learning models also possess the remarkable capability to adapt to emerging threats by learning from extensive and diverse datasets [3]. They excel in extracting intricate and subtle features within malware samples, a task that may be challenging for rule-based or signature-based systems. This feature extraction prowess contributes to heightened accuracy in distinguishing between benign and malicious files, thereby reducing false positives that can disrupt legitimate operations. Moreover, deep learning models offer speed, efficiency, scalability, and

continuous improvement, making them invaluable tools for real-time or near-real-time detection and response in the dynamic landscape of cybersecurity [4]. Figure 1 illustrates different categories in malware analysis.

DL models can reliably categorize malware samples into numerous families or types by analyzing their individual properties, assisting security researchers and practitioners in recognizing and responding to emerging threats more efficiently [5]. This review aims to provide an in-depth understanding of various DL architectures utilized in this field, including Recurrent Neural Networks (RNNs), Deep Autoencoders (DAEs), Long Short-Term Memory (LSTM) networks, Deep Neural Networks (DNNs), Deep Belief Networks (DBNs), Deep Convolutional Neural Networks (CNNs), and Deep Generative Models (such as Generative Adversarial Networks or GANs). RNNs are designed for sequential data processing and can capture dependencies in data over time. They are commonly used in tasks like natural language processing and speech recognition. Deep DAEs are utilized for unsupervised learning and data compression. They comprise an encoder and a decoder and find applications in feature learning and anomaly detection. LSTMs, a type of RNN, have specialized memory cells that capture long-term dependencies in data. They are particularly effective in sequential tasks where retaining context is crucial. DNNs consist of multiple layers of interconnected neurons and are employed for supervised learning tasks like image and speech recognition. They form the core of many deep learning applications. DBNs are generative models composed of multiple layers of stochastic, latent variables. They are used in tasks such as feature learning, collaborative filtering, and dimensionality reduction. CNNs are designed for processing grid-like data, such as images, and use convolutional layers to automatically learn spatial hierarchies of features. They find wide applications in image and video analysis. Deep Generative Models, including GANs, are capable of generating data rather than classifying it. GANs, for example, consist of a generator and a discriminator that compete in a game, resulting in the generation of realistic data. They are often used in image generation and data augmentation. This review investigates these models in terms of their unique capabilities and applications in the field of cybersecurity.

1.1 Limitations of previous reviews

In recent research, various issues and challenges related to malware detection using data mining have been extensively explored [6]. One significant challenge is the imbalance of classes within datasets, which affects the accuracy and robustness of malware detection models. Additionally, the need for open and public benchmarks, the emergence of concept drift, and the concerns surrounding adversarial learning techniques all pose significant obstacles to the effectiveness of these detection mechanisms. Furthermore, the interpretability of models remains a critical concern, impeding the deployment of reliable and understandable solutions. When it comes to Cyber-Physical System (CPS) malware detection, the complexity of different malware classes and their numerous variants ma kes detection even more challenging [7]. The rise of Advanced Persistent Threats (APTs) adds another layer of sophistication, demanding advanced strategies to combat coordinated and purposeful attacks. Analyzing malware, including static and dynamic aspects, presents difficulties in understanding and identifying malware, necessitating robust detection strategies. While signature-based and behavior-based methods offer distinct advantages, they also face challenges related to accuracy and efficiency in classifying programs as malicious or benign [6].

An examination of the strengths and weaknesses of signature-based and behavior-based malware detection reveals that each method has its own merits and shortcomings [6]. Signature-based detection is fast and efficient but struggles to detect polymorphic malware, whereas behavior-based detection excels in identifying unconventional attacks but faces challenges regarding storage and time complexity. Ransomware detection and prediction techniques have received significant attention, particularly in the context of machine learning methods [8]. However, there has been a lack of emphasis on predicting ransomware, and identified shortcomings in real-time protection and 0-day ransomware identification highlight the need for more comprehensive approaches. Adversarial machine learning exploitation and concept drift further complicate the landscape of machine learning models in this domain.

Deep Learning (DL)-based malware detection frameworks encounter several challenges, including data imbalance, interpretability issues, susceptibility to adversarial attacks, the need for regular updates, and difficulties in achieving cross-platform detection [9]. Efficient feature extraction techniques and the recognition of new characteristics in 0-day malware add further complexity to the development and deployment of DL models. Deep learning for 0-day malware detection and classification focuses on learning paradigms, feature types, benchmark datasets, and evaluation metrics. API/System calls are the most common type of feature, and prevalent benchmark datasets include Drebin. Evaluation metrics encompass Accuracy, Precision, Recall, F1-score, False Positive Rate, False Negative Rate, Area Under the Curve, and Evasion rate.

In the domain of Android malware detection using machine learning, challenges and advancements in static analysis have been explored [10]. Machine learning techniques applied to features extracted through static analysis have shown varying degrees of success, relying on tools like APK Tool and Androguard for decompiling and analyzing APK files. The challenges posed by adversarial attacks for PE (Portable Executable) malware are multifaceted. Adversarial attacks in both feature-space and problem-space encounter difficulties in maintaining the format, executability, and maliciousness of PE files. The taxonomy of attacks includes white-box attacks, where the attacker has full knowledge of the model, and black-box attacks, where limited or no knowledge of the model's internals presents additional challenges [11]. Feature-space attacks involve direct manipulation of features, while problem-space attacks entail altering the actual inputs, such as PE files. These challenges highlight the need for robust defenses against adversarial threats in the context of malware detection. Table 1 provides a comparison with previous review papers with a similar focus.

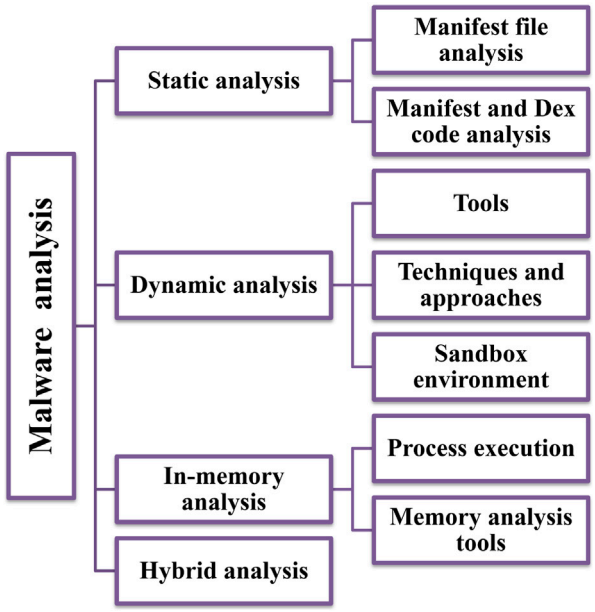

FIGURE 1
Categories in malware analysis.

1.2 Motivation and objectives of this review

The rapidly evolving landscape of cybersecurity presents an ongoing challenge, particularly in the realm of malware detection. Traditional methods struggle to keep pace with the relentless creation of new malware variants. They struggle to keep up with evolving threats, making deep learning's ability to autonomously extract features from vast datasets crucial. Deep Learning models excel in discerning intricate patterns within malware, offering scalability, efficiency, and continuous improvement. As a result, there's a pressing need for innovative solutions that can adapt to emerging threats and provide robust protection against cyberattacks.

This review aims to delve into the burgeoning role of deep learning models in combating malware threats in cyberspace. It provides a thorough exploration of various deep learning architectures and their applications in malware detection. By analyzing the strengths and limitations of each model, the review offers valuable insights to researchers and practitioners seeking to harness deep learning techniques for cybersecurity. Recognizing the dynamic nature of cyber threats, the review also sheds light on the evolving landscape of malware and the increasing sophistication of cyber-attacks. It identifies future research directions, emphasizing the need for innovative DL-based solutions that can adapt to dynamic malware behavior and effectively counter adversarial attacks.

1.3 Contributions of this review

The main contributions of this article are as follows:

a) This review provides a critical assessment of the existing literature in the field of deep learning-powered malware detection in cyberspace. Further, it helps to identify gaps and areas for improvement, guiding future research directions and ensuring a more comprehensive understanding of the subject matter.
b) This review extends the scope of traditional approaches to encompass the rapidly growing threat landscape targeting mobile devices. This expansion of focus ensures that the review remains relevant and up-to-date with emerging trends in cybersecurity, providing insights into the unique challenges and opportunities presented by mobile malware.
c) By including recent tools in malware analysis and detection, this review offers readers a comprehensive overview of the current state-of-the-art technologies and methodologies available for combating malware threats. This enables researchers and practitioners to stay abreast of the latest advancements in the field and make informed decisions when selecting and implementing detection tools and techniques.
d) By incorporating a diverse range of tools in malware detection, including behavioral analysis tools, threat intelligence platforms, deception tools, and memory forensic tools, this review provides a holistic perspective on the multifaceted nature of malware detection. This ensures that readers gain insights into the various approaches and methodologies employed in the detection and analysis of malware, enhancing their understanding of the complexities involved in combating cyber threats.
e) By highlighting open challenges in the field of malware detection using deep learning, this review identifies areas where further research and development are needed to address existing gaps and limitations. This stimulates discussion and collaboration within the research community, fostering innovation and driving progress towards more effective and robust solutions for malware detection using deep learning techniques.

2 Survey methodology

Figure 2 illustrates the process of article selection for this review, adhering to the PRISMA guidelines [13]. A comprehensive search for deep learning models in malware analysis and detection was conducted in three databases, namely, Google Scholar, Scopus, and Web of Science, spanning from January 2015 to December 2023. The search string "Cyberspace, Deep Learning, and Malware Detection" was employed to collect relevant articles. Inclusion and exclusion criteria were applied to determine the articles to be included in the review. Specifically, the articles had to be written in English, published in peer-reviewed journals or conferences, and relevant to both malware analysis and detection and deep learning. During the initial stage, 900 non-duplicate articles were obtained, as depicted in Figure 2. Following the screening of titles and abstracts, 457 articles were excluded. Subsequently, 171 articles for which full-text reports could not be retrieved were also removed from consideration. Additionally, 272 articles were assessed for eligibility, leading to the removal of 133 articles with incomplete information. Finally, a total of 139 articles met all the criteria and were selected for this review.

TABLE 1 Comparison with previous review articles with a similar background. (✓: Yes and ✗: No).

Reference	Year	Summary of the main contributions	Deep learning	Open challenges	Future directions
Our Review	-	The review examines the effectiveness of Deep Learning models in detecting malware, as well as highlighting the existing challenges and potential future opportunities	✓	✓	✓
[9]	2023	This work presents a survey of deep learning techniques for 0-day malware detection and classification, elaborating on the taxonomy of resilient techniques for 0-day attacks	✓	✓	✗
[7]	2023	The review discusses the significant impacts of malware threats on Cyber-Physical Systems and explores the application of nature-inspired metaheuristic algorithms as a means to counter these threats	✗	✗	✓
[8]	2023	This work offers a thorough overview of the evolution, taxonomy, and research related to ransomware. It specifically focuses on the challenges and detection techniques within the realm of cybersecurity	✓	✗	✓
[10]	2022	This work provides a critical review of machine learning approaches used for Android malware detection. It covers various learning methods and their organization based on feature use	✓	✗	✓
[11]	2022	The review discusses the utilization of Indicators of Compromise (IOCs), machine learning methods, and deep learning-based methods in tools and anti-malware products	✓	✓	✓
[12]	2020	This survey provides a detailed overview of traditional machine learning methods, including their challenges and limitations in the field. It also highlights recent trends, particularly in deep learning, and discusses open research issues	✓	✓	✗
[6]	2018	This work offers a systematic and detailed survey of malware detection mechanisms that utilize data mining techniques. It classifies the approaches into two categories: signature-based methods and behavior-based methods	✓	✓	✗

3 Deep learning-powered malware detection in cyberspace

Deep learning (DL) models are highly proficient in autonomously learning features from extensive datasets, making them particularly suitable for detecting malware in the digital realm. By thoroughly analyzing malware samples, DL models acquire the capability to accurately categorize them into distinct families or types.

DL models undergo training using comprehensive sets of extracted attributes, including elements such as opcode sequences, API calls, and system calls. This training empowers the models to differentiate intricate patterns that distinguish malware from benign software. Consequently, these well-trained models can be deployed to classify new and previously unknown samples, providing a powerful tool for robust detection and in-depth analysis of malware. Figure 3 illustrates the current taxonomy of deep learning models for malware detection in cyberspace. Additionally, Table 2 provides a summary of research conducted on deep learning models for malware detection in cyberspace.

3.1 Recurrent neural networks

Recurrent Neural Networks (RNNs) play a significant role in the field of malware detection in cyberspace due to their ability to handle sequential input data. In the context of malware detection, RNNs are useful for assessing system calls, API calls, and network traffic generated by software applications to identify potentially harmful activities. System calls provide insights into a program's actions within the system, allowing the detection of deviations from normal software behavior that may indicate malicious activity. API calls reveal how a program interacts with the underlying system, enabling the identification of specific APIs used for malicious purposes, such as modifying system settings. Network traffic data is crucial for detecting malware that communicates with external servers, attempts data exfiltration, or engages in suspicious data exchange over the network.

RNNs excel at analyzing sequences of data and capturing temporal dependencies in the behavior of potentially malicious software. They are particularly effective at processing sequential data encountered in malware analysis, such as sequences of system calls, API calls, or network traffic generated by software. By being exposed to sequential data, RNNs become adept at discerning correlations and patterns that indicate malware behavior. They take sequential data as input, working through it one element at a time and updating their internal state based on the observed data. This mechanism allows RNNs to capture temporal dependencies and patterns in the data, which is essential for understanding the dynamic nature of malware. The hidden state within RNNs serves as a form of memory, retaining information about previous observations and enabling the contextualization of past events while predicting the current one. However, these strengths are counterbalanced by challenges inherent in its application. There is a limitation that Kaspersky malware family classification criteria of the malware sample used for analysis in this paper may not be

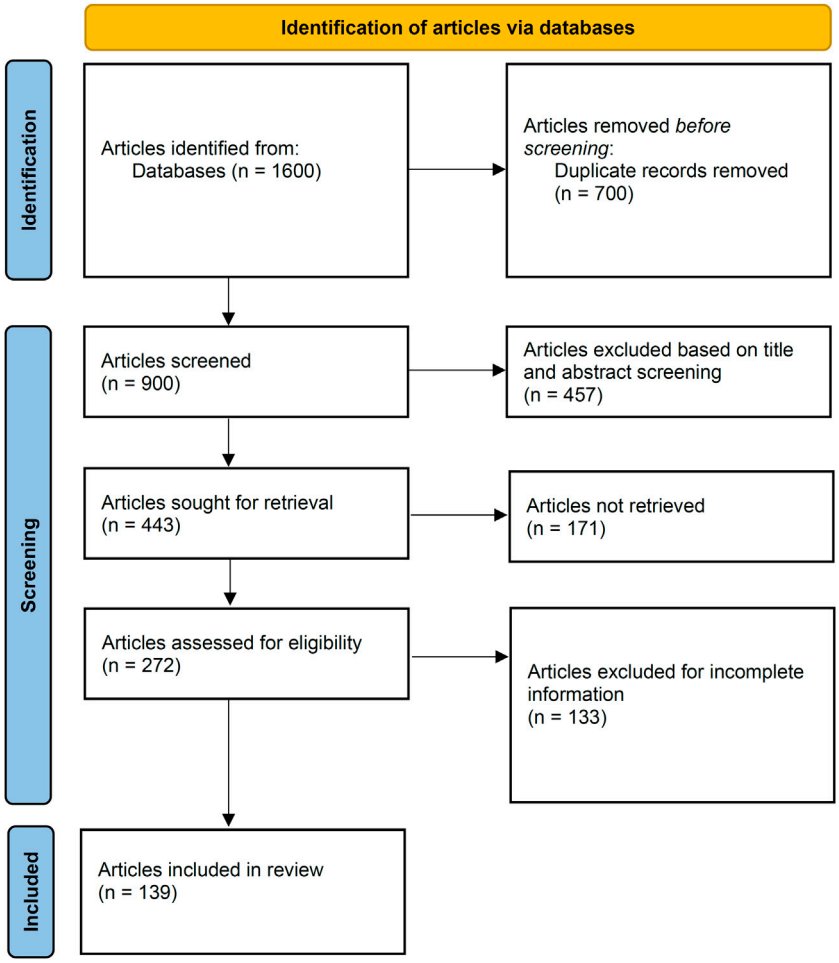

FIGURE 2
Selection of articles adhering to the PRISMA guidelines.

accurate. Only the types and order of the APIs were taken into consideration that were called when extracting patterns for APIs called by malware and evaluating them. Since the API itself is higher-level than the machine code or assembly in the computer, the performance may be improved if the semantic criteria and semantic distinction of malware API to be extracted [1].

To train RNNs for malware detection, historical data is used to adjust their internal parameters. Backpropagation through time is employed to update the model's weights and biases based on prediction errors, allowing the them to learn patterns associated with malware. Additionally, their temporal modeling capabilities enable the identification of anomalous behavior trends within an application over time, facilitating the early detection of novel or previously undiscovered strains of malware. The ability of RNNs to capture nuanced and evolving behavioral patterns makes them a valuable tool in malware detection. They enhance security by providing a dynamic, adaptable, and context-aware approach to identifying malicious software, especially in the face of rapidly changing cybersecurity threats. RNNs are effective at identifying evasive and polymorphic malware, which employ techniques to avoid detection and continually change their code to generate different variants. RNNs can tackle these challenges by recognizing deviations from normal behavior and analyzing the evolving patterns in the code.

Addressing data imbalance in training RNNs demands a strategic approach. One method involves data augmentation, wherein synthetic data is generated by introducing variations to the existing minority class samples, thereby enriching the dataset. Additionally, employing sampling techniques such as oversampling (replicating minority class samples) or undersampling (reducing the number of majority class samples) can help balance the dataset distribution. Moreover, integrating cost-sensitive learning proves effective by assigning varying costs to misclassification errors across different classes, thereby accommodating the imbalance and enhancing model performance. These strategies collectively empower RNNs to navigate the challenges posed by skewed data distributions, ultimately fostering more robust and accurate predictions.

Experiments were conducted with 787 malware samples belonging to nine families. In the experiments that were carried out, representative API call patterns of nine malware families on 551 samples were extracted as a training set and performed

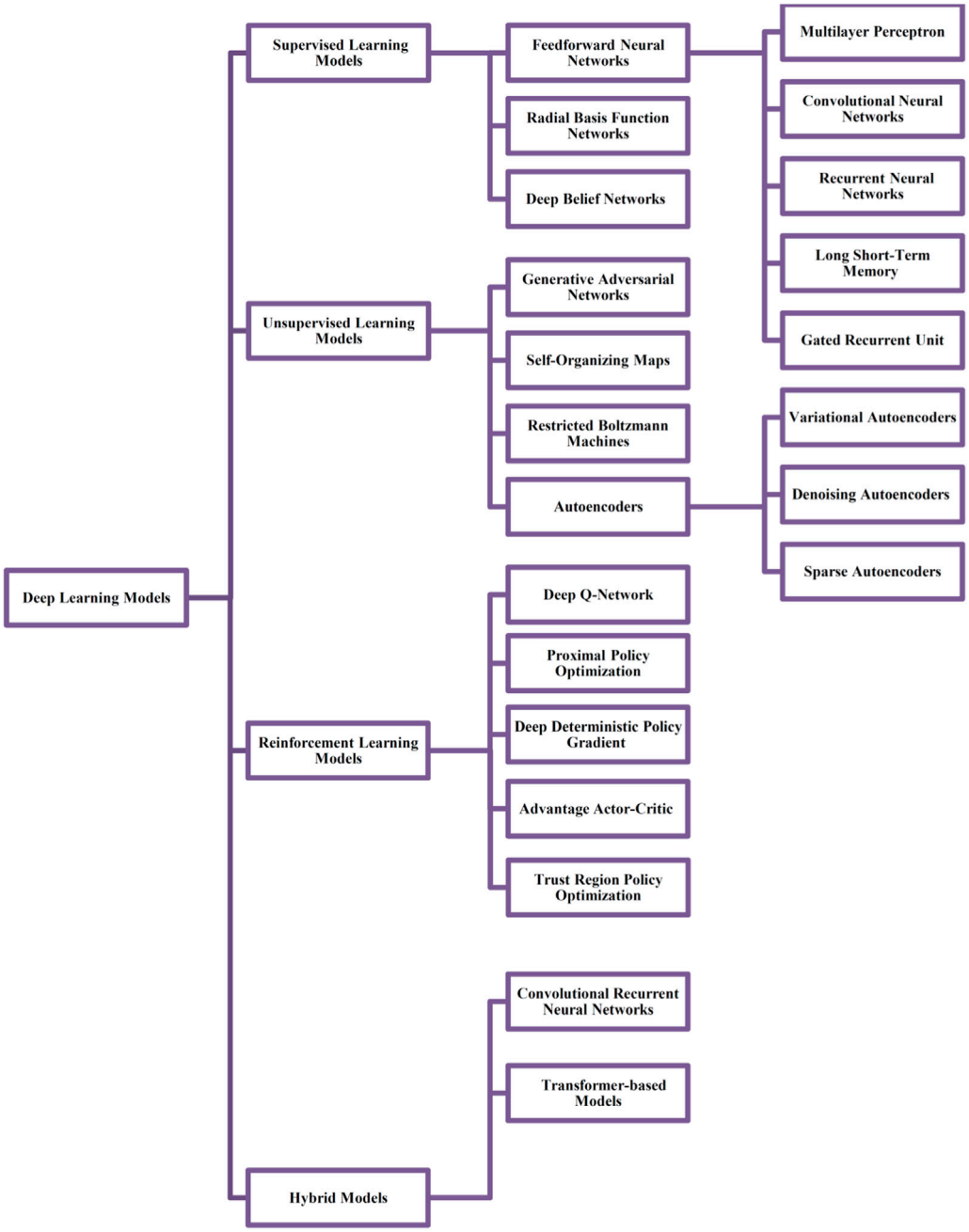

FIGURE 3
Current Deep learning Models for Malware Detection in Cyberspace–Taxonomy.

classification on the 236 samples as a test set. Classification accuracy results using API call patterns extracted from RNN were measured as 71% on average. The results show the feasibility of our approach using RNN to extract representative API call pattern of malware families for malware family classification. First, the similarities of the representative API call patterns with extracted from each family and the API call sequences of the malware belonging to the test set are compared. Then, top three representative API call patterns were selected with the highest similarity compared to each malware in the test set and compare the top three family results with the correct answer. Jaccard similarity coefficient was used as the similarity measure [1].

Experimental results using a balanced dataset showed 83% accuracy and a 0.44 loss, which outperformed the baseline model in terms of the minimum loss. The imbalanced dataset's accuracy was 98%, and the loss was 0.10, which exceeded the state-of-the-art model's accuracy. This demonstrates how well the suggested model can handle malware classification [23].

One successful application of RNNs in malware detection is dynamic behavioral analysis. This involves analyzing software

TABLE 2 Details of works on deep learning-powered models for malware detection in cyberspace.

Reference	Malware type	DL models used	Brief focus	Key contributions	Limitations	Performance metrics
[5]	Smart Vehicular Network malware	Duelling Deep Q Learning	Detect abnormal network traffic and classification of attack	• Improve detection accuracy using a supervised machine learning task based on agent-environment interaction using a modified duelling DQN model • The algorithm is modified such that it can pick a supervisor to implement an interaction mechanism as a supervised algorithm for action, state, and reward • Demonstrates the efficacy of the suggested invasion detection strategy and the enhancement in classification performance over conventional ML algorithms	• Low Recall compared to existing systems	• The experimental results showed that SIMPLE achieves an accuracy of 90% in a 5-way classification task on new malware families
[14]	IoT botnet	CCR-ELM	Lightweight framework to detect IoT botnet and botnet clusters	• Framework for detecting IoT botnets and botnet clusters. The framework works with data regarding automated behaviour • The Zeek Network Analysis Framework is used for reassembling the network flow. Every network flow produces 27 statistical and behavioural records pertaining to network communication, application-level protocols, and payload exchanged • For the botnet detection portion of ELM, overall efficacy improves as the number of concealed nodes rises. A botnet family's behaviour may vary based on the device it infects and the stage at which it is deployed	• Requires more memory space with the addition of data	• The proposed ensemble method achieves the best outcome with the highest accuracy 99.9%, compared to state-of-the-art machine learning, deep learning, and ensemble models
[15]	Worms	CNN Model	Anomaly based intrusion detection, classification of an event as malignant or benign	• In context of accuracy and recall, the devised model outperforms popular models such as NB, J48, RF, Bagging, and Adaboost	• Cannot distinguish fuzzy attacks • Multiclass classification accuracy has room for improvement	• Support Vector Machine (SVM), and AdaBoost algorithms and they achieved the highest accuracy rate of 99,80% with the Decision Tree classifier
[16]	Malware Detection in Fog Computing	CNN	Optimization of detection and classification mechanism for malware detection in Fog Computing	• Structured and powerful malware detection system can be deployed in fog computing by utilising a feature reduction ability that takes a screenshot of a file and converts it into an image and gives a new way for feature reduction by reading only a specific number of bytes per 1 KB of data and splitting an image into chunks, which divides a large file into fixed-size output images • The training of a model involves the inclusion of disturbance to improve the model's accuracy. This method obtained a 97.2% success rate, used 16 times fewer features than other approaches, and was able to manage enormous files	• When detecting files with specific extensions such as.7zip, which have a fewer number of files in the training dataset, the model performs poorly • Model detects chunks of large file as malware even if it is benign due to lack of adequate samples in training dataset.	• They used Convolutional Neural Network (CNN) algorithm for classification and achieved an accuracy rate of 94%
[17]	Android malware	• SERLA • SimHash and CNN	The framework uses disassembly technology to produce bytes file and asm file for each executable file and a special matrix generation technique to produce three 256x 256 square matrices. The three matrices are then utilised as the three channels of an RGB image and combined to create a colour image. Furthermore, to improve the discriminative power of the RGB images, we apply adaptive histogram equalization processing utilizing the CLAHE (Contrast Limited Adaptive Histogram Equalization) data augmentation technique. In conjunction with the oversampling method for training neural networks, trained models for malware detection and family classification are ultimately obtained	• Proposed a comprehensive detection and classification framework for malware that can convert executable files into their corresponding bytes and asm files. Therefore, we create a steady dataset containing both normal software samples and malware samples. This dataset can be utilised for a wider range of malware detection experiment categories. • A novel approach is introduced for data representation, which leverages binaries and word vectors derived from both bytes' files and asm files. This innovative method aims to extract comprehensive information from software samples, enabling a more holistic understanding of the data. It considers the characteristics of more aspects of the data samples and can provide more valuable assistance for the training of the detection model, thereby enhancing the detection performance.	• The labelling method used cannot filter normal software with 100% accuracy. • Compilation configuration is also one of the noticeable issues. Different compilation configurations will make the code with the same function compiled into different assembly files, which can cause wrong classification of the detection model	• The experimentation on a recent data set which includes 11,120 applications showed that an accuracy of 97% on average can be achieved. • SWORD obtained an accuracy of 94.2% in experiments on a data set containing 2000 Android samples from various sources

(Continued on following page)

10.3389/fphy.2024.1349463

TABLE 2 (*Continued*) Details of works on deep learning-powered models for malware detection in cyberspace.

Reference	Malware type	DL models used	Brief focus	Key contributions	Limitations	Performance metrics
				• In conjunction with the data augmentation technique in computer vision, an optimised deep neural network SERLA based on SEResNet50, Bi-LSTM, and an attention mechanism is developed for malware detection. Compared to other neural network malware detection models and even state-of-the-art methods, our model is superior in all evaluation metrics, as demonstrated by experimental results		
[18]	Malware	Dragonfly-based DGDBN	Cloud security and malware detection	• Development of a novel Dragonfly-based Genetic Deep Belief Network (DGDBN) technique for safeguarding VMs in cloud environments	• Limited information on dataset and real-world testing	• The authors prove that such gradient approximation mechanism allows the objective function to converge to optima with probability 1, where in their experiments only a 2% accuracy loss is observed on average on GoogleLeNet training
[19]	Android malware	DAE-CNN (Deep Autoencoder-CNN)	Large-scale Android malware detection	• The research proposes a hybrid model combining a Deep Autoencoder (DAE) and Convolutional Neural Network (CNN) to enhance large-scale Android malware detection	• The research discusses the model's performance in terms of accuracy and training time reduction but does not specify if it was tested in a real-world environment or against a broader range of Android malware. • Real-world testing is crucial to validate the model's practical effectiveness	• Showed an excellent performance with an overall detection accuracy of 99.3% for Probe, Remote to Local, Denial of Service and User to Root type of attacks
				• To enhance efficiency, the research introduces DAE as a pre-training method for CNN. This approach significantly reduces training time, specifically an 83% reduction compared to CNN-S		
				• The model incorporates ReLU activation functions, sparsity rules, and the combination of convolutional and pooling layers with the full-connection layer to enhance feature extraction capability		
[20]	Botnet Malware	Khaos (DGA Model)	Domain Generation Algorithm (DGA) for botnets	• The research introduces Khaos, a novel Domain Generation Algorithm (DGA) for generating domain names used in botnet command and control (C&C) servers. Khaos is designed to enhance anti-detection capabilities	• The study does not address the adaptability of Khaos to evolving malware threats	• Two image conversion methods, byteplot and space-filling curves, were used to represent the malware samples, and a ResNet-50 architecture was used to train models on the image datasets. The models were then tested against a projected gradient descent attack. It was found that without GAN-generated data, the models' prediction performance drastically decreased from 93%–95%–4.5% accuracy
				• The study leverages neural language models and the Wasserstein Generative Adversarial Network (WGAN) to design Khaos. By mimicking real domain names through the arrangement of syllables and acronyms, Khaos aims to create domain names that are challenging for detection	• The dynamic nature of botnets and malware requires continuous innovation to stay ahead of detection methods	
[4]	Cybersecurity Threats	EDRBM (Ensemble Deep Restricted Boltzmann Machine)	Classification of cybersecurity threats in large-scale networks	• The research introduces Ensemble Deep Restricted Boltzmann Machine (EDRBM) as a novel deep learning model for the classification of cybersecurity threats in large-scale network environments. This represents a significant advancement in the field of threat detection	• The content does not delve into the data and computational resources needed for implementing EDRBM in large-scale network environments	• When tested on all the features of the NSL-KDD data set, the deep learning method obtaines very low result compared with the mentioned methods, but when it is tested on six features, the method in terms of accuracy metric gets the high result and composed of 75.75%
				• EDRBM is applied to classify cybersecurity threats, with a specific emphasis on malware attacks. It serves as a classification model capable of differentiating between benign and malicious network traffic flowsets	• Real-world implementation may require significant resources, and these requirements should be considered	
[21]	Malware Families	Shallow Convolutional Neural Network (CNN)	Assessing the vulnerability of malware classifier to dead code insertion and adversarial attacks	• Introducing a Double Q-network-powered framework to induce misclassification in malware families	• Lack of discussion about the impact of these attacks in real-world settings	• The model outperforms other CNN architecture by a significant margin on the accuracy metric, where 98.85% was achieved on both datasets, Benign and Malicious PE Files dataset and MalwareDataSet, and 98.37% was achieved on the Classification of Malwares dataset.
				• Training an AI agent to insert dead code instructions into malware samples	• No information on the resource requirements for deploying the proposed framework	
				• Demonstrating significant classification accuracy reduction in malware classifier	• Ethical and legal aspects related to malware manipulation and evasion are not addressed	
				• Achieving 100% evasion rate for specific malware families	• Specific limitations associated with the proposed framework are not discussed	

(Continued on following page)

10.3389/fphy.2024.1349463

TABLE 2 (*Continued*) Details of works on deep learning-powered models for malware detection in cyberspace.

Reference	Malware type	DL models used	Brief focus	Key contributions	Limitations	Performance metrics
				• Enhancing evasion success rates through deep reinforcement learning with the Double Q-learning algorithm • Validating results on the Portable Executable files dataset for reproducibility		
[22]	Steganography	Convolutional Neural Networks (CNN) and Extreme Learning Machines (ELM)	Training machine learning models for malware classification based on features obtained without disassembly or code execution	• Introducing a method to visualize malware samples as images for classification • Evaluating two machine learning techniques: CNNs and ELMs • Demonstrating that ELMs achieve comparable accuracy to CNNs with significantly faster training times • Showing that ELMs and CNNs perform well with both one-dimensional and two-dimensional data	• Lack of specific information about the malware types or dataset used in the experiments • No discussion of the real-world applicability or performance of the proposed approach • The paper does not address the potential limitations of image-based malware classification in terms of accuracy or security	• The average accuracy for the unweighted model is 96.5%, while for the weighted model we obtain a slight improvement at 97.7%

behavior, such as file operations, system calls, registry accesses, and network interactions, to identify potentially malicious activities. They can create behavioral profiles of software by observing and analyzing its interactions with the operating system and external resources. These profiles can be compared against known malware behavior patterns to identify potential threats. RNNs excel in anomaly detection, crucial for identifying malware that exhibits unusual or unexpected behavior. By continuously learning and adapting to evolving malware behavior, RNN-based systems can update their models to detect novel threats, making them effective against 0-day attacks. They also reduce false positives by focusing on behavioral analysis, prioritizing potential threats for security professionals to investigate. RNNs can consider the context of each action within a software's behavior, distinguishing legitimate activities from malicious ones. Given their proficiency in processing ordered data, RNNs are well-suited for tasks involving time series data. They adapt to various application domains, including speech recognition, language modeling, translation, and image captioning, where sequential data analysis is crucial [1].

3.2 Deep autoencoder

Deep Autoencoders (DAEs) have ascended as potent tools in the fight against malware, particularly within the realm of unsupervised learning. DAEs are a type of neural network that encodes high-dimensional input into a lower-dimensional representation and then decodes it back into its original format. They serve as valuable tools for uncovering the inherent characteristics and patterns exhibited by benign software applications. By training on extensive datasets of benign applications, DAEs learn and internalize the defining traits of harmless programs.

Labeled datasets are crucial in training DAEs for malware detection. These datasets play a pivotal role in imparting the model with the ability to discern nuanced patterns that distinguish benign software from malicious counterparts. Operating within a supervised learning framework, they are trained on input-output pairs derived from labeled datasets. Each input represents features extracted from both benign and malicious samples, enabling the model to comprehend the distinctive characteristics associated with each class. This process contributes to the model's generalization ability, allowing it to recognize common patterns indicative of malware across different variations and instances, ensuring effectiveness on previously unseen data.

Addressing data imbalance in training deep DAEs and variational autoencoders (VAEs) involves employing several strategic approaches. One method involves leveraging the inherent generative capacity of VAEs to counter data scarcity by generating synthetic data. This technique helps balance the class distribution, enhancing the model's ability to learn from underrepresented classes. Additionally, adversarial training can be utilized to foster robustness against imbalances. By exposing the model to adversarial examples, it learns to create more resilient representations, mitigating the impact of data imbalance. These strategies collectively empower them to handle skewed datasets effectively, improving their capacity to generalize and learn meaningful representations across all classes.

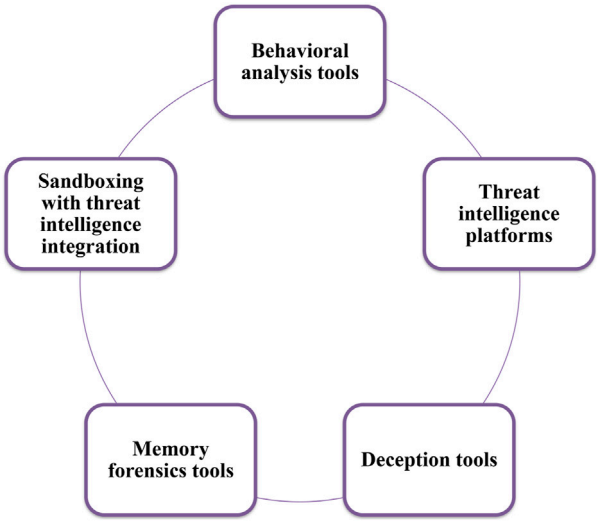

FIGURE 4
Categories of recent tools in malware analysis and detection.

When presented with novel applications, DAEs can assess whether they deviate significantly from the learned benign patterns. This assessment is made possible by evaluating the reconstruction error generated during the decoding process. A high reconstruction error indicates a substantial departure from the expected benign behavior, raising suspicion of potential malicious activity. They can also be seamlessly integrated with other machine learning algorithms, enhancing the comprehensiveness of malware detection strategies.

The approach without autoencoder, both precision and recall are 99 Percentage for just the Bi-LSTM model in detecting malicious activities in cyber security. Average precision and recall of the performed model with autoencoder is 93% [24].

The architectural framework of DAEs consists of two pivotal stages: encoding and decoding. In the decoding phase, the compressed representation is reconstructed back to its original form, with each network layer performing a distinctive transformation on the input data. Their adaptability benefits applications like natural language processing (NLP), picture recognition, identity verification, and data reduction.

While DAEs have ascended as potent tools in the fight against malware, deploying them for real-world malware detection comes with various challenges. Scalability is a significant challenge, as training and deploying DAEs at scale can strain organizational infrastructure. Efficient scaling becomes essential as datasets grow in size and models become more complex. The demand for computational resources, especially during training, poses another challenge. Organizations must address the need for processing power, which can lead to longer inference times and increased operational costs. Real-time analysis, crucial for timely malware identification, can be challenging with DAEs, particularly those with complex architectures that struggle to achieve low-latency predictions. Imbalances in real-world malware datasets pose challenges related to biased models favoring the majority class, resulting in suboptimal detection of less common or emerging malware variants. The interpretability and explainability of DAEs, often seen as black-box models, become critical in a production setting to gain the trust of security analysts and stakeholders [24].

In malware detection, a comparative analysis reveals distinctive strengths and weaknesses among DAEs, traditional machine learning algorithms, and other deep learning approaches. DAEs excel in unsupervised feature learning and automatically capture intricate representations vital for complex tasks. Their ability to detect anomalies without labeled data is advantageous for identifying new malware variants. Traditional machine learning algorithms, like Decision Trees, offer superior interpretability and computational efficiency but rely on manual feature engineering and have limitations in anomaly detection [19]. Deep learning approaches, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), excel in spatial and temporal contexts but require large labeled datasets and can be computationally intensive. Factors like interpretability, data availability, and analysis requirements play a crucial role in choosing among DAEs, traditional algorithms, or other deep learning models. DAEs, with their focus on unsupervised feature learning and anomaly detection, stand out in the malware detection toolkit, each approach presenting unique strengths tailored to the demands of the cybersecurity landscape.

Deep Autoencoders (DAEs) find practical application in unsupervised learning for discerning inherent traits within benign software. They leverage extensive datasets of benign applications to learn characteristic features. Through this learning process, DAEs become adept at identifying deviations from these established norms, effectively flagging potential malware presence. By analyzing and detecting anomalies in software behavior, DAEs serve as a valuable tool in the continuous battle against cybersecurity threats, enabling proactive identification of suspicious activities and potential threats [25].

Unsupervised learning presents a promising avenue for discerning the inherent traits of malware, and its efficacy lies in the ability to perform effective feature learning without relying on labeled data. However, despite its potential, DAEs encounter notable challenges.

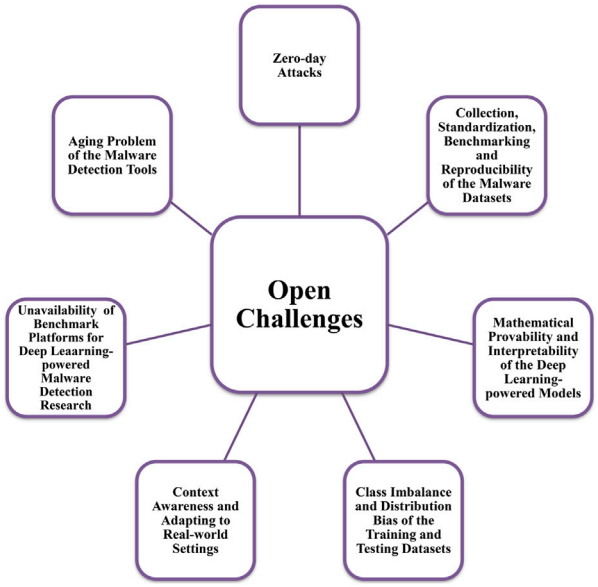

FIGURE 5
Open challenges–deep learning-powered malware detection in cyberspace.

Scalability poses a significant hurdle during both training and deployment phases, demanding innovative solutions to handle the complexities of large-scale data. Additionally, the vulnerability to adversarial attacks presents a pressing concern, necessitating robust defense mechanisms to fortify these models. Furthermore, integrating unsupervised learning methodologies with existing security infrastructure proves to be a challenging task, demanding a concerted effort to harmonize these disparate elements effectively. Thus, while holding considerable promise, the practical implementation of unsupervised learning in identifying malware characteristics necessitates a strategic approach to mitigate these formidable challenges.

3.3 LSTM

The Long Short-Term Memory (LSTM) architecture has demonstrated its effectiveness in virus detection due to its ability to identify long-term dependencies within sequential data. LSTMs are a type of recurrent neural network (RNN) that use memory cells and gates to control the flow of information within the network. This makes them valuable for analyzing sequences of system calls, API calls, or network traffic generated by applications, especially in the context of malware detection in cyberspace. By processing sequential data using LSTM networks, patterns and correlations indicative of malicious behavior can be discovered. The following techniques are used in this scenario:

a) Sequence Encoding: System call sequences are encoded into numerical vectors, where each system call is represented as an integer or a one-hot encoded vector. This encoding enables effective processing by the LSTM network.
b) Sequence Padding: Sequences are often padded or truncated to a fixed length to ensure uniform input lengths. This step is crucial for creating consistent input for the LSTM.
c) LSTM Architecture: LSTM layers are utilized to capture the temporal dependencies and the order of system calls within the encoded sequences. LSTMs excel at modeling long-range dependencies, making them well-suited for this task.
d) Output Classification: The output of the LSTM layer is typically connected to a classification layer responsible for distinguishing between benign and malicious behavior based on the patterns learned from the system call sequences.

Like system call analysis, LSTM networks can be used for analyzing sequences of API calls to detect malware in cyberspace. The techniques involved are similar to those used in system call analysis, including sequence encoding, padding, LSTM architecture, and output classification. Additionally, they can also be utilized to model the behavior of an application over time, enabling the identification of anomalous activities that may indicate the presence of a new or previously undiscovered malware strain [25]. For enhanced malware detection strategies, LSTMs can also be seamlessly integrated with other machine learning techniques, such as Convolutional Neural Networks (CNNs).

The development of the LSTM architecture was primarily motivated by the need to address the vanishing gradient problem present in standard neural networks [26]. This problem arises when each connection within a network has its individual weight that remains unchanged over time, leading to training difficulties. As a type of RNN, LSTMs leverage memory cells and gating mechanisms to effectively control the flow of information, making them well-suited for the analysis of sequences involving system calls, API calls, or network traffic while identifying patterns and correlations indicative of malicious behavior.

A high-quality dataset consisting of 2,060 benign and memory-resident programs was created. In other words, the dataset contains 1,287,500 labeled sub-images cut from the MRm-DLDet transformed ultra-high resolution RGB images. MRm-DLDet was

implemented for Windows 10, and it performs better than the latest methods, with a detection accuracy of up to 98.34%. Twelve diferent neural networks were trained and the F-measure up to 99.97% [27].

To address the challenges posed by data imbalance in LSTM training, several strategies are employed. One approach involves employing data augmentation techniques, which entail generating synthetic data by introducing variations to the existing minority class samples. This method aids in balancing the dataset and providing the model with more diverse instances to learn from. Another valuable strategy involves leveraging sampling techniques such as oversampling or undersampling. Oversampling involves replicating minority class samples to balance the class distribution, while undersampling focuses on reducing the number of majority class samples. These methods help create a more equitable representation of classes within the dataset, enabling the LSTM to learn effectively from both the majority and minority classes. Furthermore, adopting cost-sensitive learning techniques proves beneficial. By assigning varying costs to misclassification errors in different classes, the model can account for the imbalance and prioritize accurate classification of the minority class. This approach ensures that the LSTM places appropriate emphasis on correctly identifying instances from both the majority and minority classes, thereby enhancing overall performance despite data imbalances [26].

LSTM's mastery in capturing temporal dependencies enables a deep comprehension of malware's dynamic behavior over time. However, this strength comes with inherent challenges. The resource-intensive nature of training poses a significant obstacle, while vulnerability to adversarial attacks is a critical concern. Additionally, the constant need for updates to align with the ever-evolving array of malware variants presents an ongoing demand. Despite these hurdles, the methodology's proficiency in unraveling intricate data sequences remains a promising frontier in the realm of deciphering and combating malware conduct.

3.4 Deep neural network

Deep Neural Networks (DNNs) have gained prominence in virus detection due to their remarkable capacity to comprehend complex patterns and data characteristics. DNNs represent a class of artificial neural networks characterized by numerous interconnected layers of nodes. These layers collaboratively process incoming data, ultimately yielding predictions or classifications. In the realm of malware detection in cyberspace, DNNs are trained on extensive datasets encompassing both benign and malicious programs, enabling them to discern the fundamental attributes and patterns inherent to each class.

Leveraging the knowledge acquired from this training, DNNs can effectively categorize new applications as either benign or malicious based on their intrinsic characteristics. They exhibit versatility in assessing diverse forms of input data, spanning system calls, API calls, and network traffic, making them adaptable to various malware detection scenarios. For instance, in API call pattern recognition, Convolutional Neural Networks (CNNs), a type of DNN, can effectively analyze sequences for anomaly detection or malware identification by representing each API call as a feature vector. In network traffic analysis, DNNs, including CNN architectures, excel at detecting spatial patterns within data for intrusion detection, often extracting features from packet headers or payloads. Additionally, CNNs prove valuable in image-based malware detection, where they process images of executable files to identify malicious code patterns. Furthermore, the ability of DNNs to integrate multiple forms of input data in multimodal threat analysis, combining features from system calls, API patterns, and network traffic, highlights their capability for comprehensive threat assessment [28].

The utilization of transfer learning allows DNNs trained on one malware classification task to be fine-tuned for related challenges, showcasing their adaptability and knowledge transfer capabilities in cybersecurity applications. The training process for DNNs typically involves backpropagation, a technique that seeks to minimize the loss function's value through the gradient descent approach. The training process involves several key stages, starting with the initialization of weights and biases, a critical step that establishes the foundation for effective learning [29]. As the input data undergoes forward propagation, traversing through the network's layers, activations are computed, and the output is generated based on the current parameters. Simultaneously, the loss function calculates the disparity between the predicted output and actual labels, providing a quantifiable metric for the network's performance. Backpropagation follows, utilizing the chain rule to compute gradients and propagate errors backward through the network. This process enables the network to discern the contribution of each weight to the overall error. Subsequently, gradient descent optimization adjusts the weights and biases to minimize the loss function, guiding the network toward optimal configurations for proficient malware detection. The entire sequence of forward propagation, loss calculation, backpropagation, and weight updates iterate over multiple epochs, allowing the network to progressively refine its parameters. These iterative adjustments enhance the network's capacity to generalize and effectively identify previously unseen malware variants, underscoring its effectiveness in the realm of cybersecurity. Resampling methods such as oversampling, undersampling, or using techniques like SMOTE (Synthetic Minority Over-sampling Technique) can rebalance the dataset, ensuring equal representation of classes. Additionally, adjusting class weights during training serves as a means to penalize misclassifications of the minority class more heavily, allowing the model to prioritize learning from the underrepresented data. These strategies collectively aim to mitigate the impact of data imbalance, enabling DNNs to better generalize and make more accurate predictions across all classes in the dataset.

A deep neural network based malware detection system that Invincea has developed, achieves a usable detection rate at an extremely low false positive rate and scales to real world training example volumes on commodity hardware. Their system achieves a 95% detection rate at 0.1% false positive rate (FPR), based on more than 400,000 software binaries sourced directly from our customers and internal malware databases [28].

To generate predictions or classifications, DNNs meticulously process data through their interconnected layers of nodes. When applied to malware detection, these networks can be trained on extensive datasets containing both benign and malicious programs, equipping them with the knowledge needed

to distinguish between the two categories. By synergizing DNNs with other machine learning techniques, such as Deep Autoencoders (DAEs) or Long Short-Term Memory (LSTM) networks, a more comprehensive and robust approach to malware identification can be achieved. Deep Autoencoders, as unsupervised models, play a pivotal role in feature learning and extraction, providing compact and meaningful representations of input data, particularly valuable in high-dimensional spaces like raw system call sequences or network traffic patterns. This enhances the model's robustness against various malware variants by capturing latent features and anomalies during pre-training on unlabeled data. On the other hand, Long Short-Term Memory Networks excel in capturing temporal dependencies and sequences, crucial for understanding dynamic aspects of malware behavior over time. Integrated into a DNN architecture, LSTMs contribute temporal context awareness, enabling the model to discern evolving patterns exhibited by sophisticated malware. Ensemble learning techniques, such as stacking or bagging, further amplify the model's robustness by combining the strengths of DNNs, DAEs, and LSTMs [28]. The ensemble approach leverages the diversity of information captured by each component, resulting in a more accurate and resilient model less sensitive to noise and outliers. Additionally, the utilization of transfer learning facilitates knowledge transfer from related tasks, such as feature learning or sequential modeling, enhancing the DNN's generalization performance in malware identification.

While leveraging DNNs has significantly propelled the field of malware detection towards greater accuracy and efficiency, their application comes with inherent limitations and challenges. Firstly, scalability issues pose a substantial hurdle, as training large-scale DNNs demands significant computational resources and can be financially burdensome. The complexity of DNN architectures, coupled with the extensive data required for effective training, exacerbates this challenge, particularly for organizations with limited computational capabilities. Secondly, interpretability challenges impede the widespread adoption of DNNs in malware detection in cyberspace. DNNs are often considered black-box models, lacking transparency in their decision-making processes. In intricate tasks like malware detection, understanding the rationale behind a specific decision is crucial for building trust and ensuring alignment with the expectations of security experts. Adversarial attacks constitute another formidable challenge. DNNs are susceptible to intentional manipulations of input data by malicious actors, leading to misclassifications and compromising the reliability of malware detection systems. Such attacks pose a significant security risk, requiring robust defenses to mitigate their impact [29]. Furthermore, the issue of data imbalance within malware datasets complicates the generalization performance of DNNs. Imbalances, where certain types of malware are underrepresented, can result in model biases towards prevalent classes, leading to suboptimal detection of less common or emerging malware variants. Lastly, the lack of explainability in DNNs' decision-making processes hinders their integration into security workflows. The opacity of these models makes it challenging for security analysts to comprehend the basis for a classification, impeding effective collaboration between automated systems and human experts.

3.5 Deep Belief Network

Deep Belief Networks (DBNs) are powerful tools in the realm of malware detection in cyberspace. These neural networks excel at capturing intricate patterns and features within vast datasets, making them invaluable for identifying malicious software. DBNs are particularly effective in analyzing software behavior and identifying anomalies or suspicious activities. They have the ability to autonomously discover relevant features, which is advantageous in the context of rapidly evolving malware. By processing various aspects of software behavior, such as system calls, API calls, or network traffic patterns, DBNs can differentiate between normal software operations and potentially harmful ones. This approach allows for the detection of previously unseen malware strains or novel attack techniques, making them a critical component of modern cybersecurity systems. The versatility and adaptability of DBNs in handling large and diverse datasets make them an essential tool for protecting against the ever-growing landscape of malware threats [2].

End-to-end deep learning architectures, specifically Bidirectional Long Short-Term Memory (BiLSTM) neural networks, are employed for the static behavior analysis of Android bytecode. Unlike conventional malware detectors that rely on handcrafted features, this system autonomously extracts insights from opcodes. This approach demonstrates the superiority of deep learning models over traditional machine learning methods, offering a promising solution to safeguard Android users from malicious applications [30].

Researchers have also explored the suitability of deep learning models for mobile malware detection. They utilize a deep neural network (DNN) implementation called DeepLearning4J (DL4J), which successfully identifies mobile malware with high accuracy rates. The study suggests that adding more layers to the DNN models improves their accuracy in detecting mobile malware, showcasing the feasibility of using DNNs for continuous learning and anticipating new types of attacks [22].

An anti-malware system that uses customized learning models, which are sufficiently deep, and are end to end deep learning architectures report an accuracy of 0.999 and an F1-score of 0.996 on a large dataset of more than 1.8 million Android applications [2]. The SOFS-OGCNMD system achieves system's average accuracy is 98.28%, average precision is 98.65%, recall is 98.53%, and F1-Score is 98.47 [22].

In addition, a method has been proposed to address the challenges of malware detection in Cyber-Physical Systems (CPS) within the Internet of Things (IoT). The model, called Snake optimizer-based feature selection with optimum graph convolutional network for malware detection (SOFS-OGCNMD), demonstrates remarkable results in accuracy, precision, recall, and F1-Score, outperforming recent models and contributing to the protection of CPS and IoT systems from evolving cyber threats [18].

Furthermore, a system has been designed to enhance the security of power systems through a Deep Belief Network (DBN)-based malware detection system. This system deconstructs malicious code into opcode sequences, extracts feature vectors, and utilizes DBN classifiers to categorize malicious code. It effectively utilizes unlabeled data for training and outperforms other classification algorithms in terms of accuracy. The research showcases the

potential of DBNs for enhancing malware detection accuracy and reducing feature vector dimensions, thereby contributing to safeguarding power systems from cyber threats.

3.6 Deep convolutional neural network

The realm of malware detection in cyberspace, particularly in the context of evolving cyber threats, is experiencing a surge in innovative approaches driven by deep learning and convolutional neural networks (CNNs). Deep Convolutional Neural Networks (DCNNs) have emerged as robust and efficient technologies for detecting malware. Their ability to automatically extract complex features from various forms of data makes them exceptionally well-suited for this task. In the context of malware analysis, CNNs excel at processing and identifying malicious patterns within binary code, enabling the detection of known malware strains and even the discovery of novel threats. These networks are particularly effective in detecting malware through the analysis of file content and structure, which includes identifying suspicious code segments and unusual behaviors. Deep CNNs offer a significant advantage in terms of adaptability as they can be trained on diverse and evolving datasets to keep up with the continuous evolution of malware. Additionally, their capacity to handle large-scale data and discern subtle variations in binary files enables the identification of both prominent and more subtle malicious patterns. They are at the forefront of malware detection, contributing to the defense against the ever-growing sophistication of cyber threats [3].

DCNNs have proven to be highly effective in tasks related to image processing, excelling in capturing intricate spatial hierarchies and patterns. Their performance, often measured through precision, recall, and F1-score, is particularly notable in image classification tasks, especially when trained on extensive and diverse datasets. When compared to traditional machine learning models like SVM or Random Forest, DCNNs consistently outshine them in image-related tasks, showcasing a superior ability to discern complex patterns. Transfer learning models, such as VGG16 or ResNet, compete strongly with them, benefiting from pre-trained networks on large datasets. However, DCNNs, especially those incorporating transfer learning architectures, often emerge as leaders, demonstrating heightened precision, recall, and F1-score by leveraging their effective feature extraction capabilities. In tasks involving sequential data, where Recurrent Neural Networks (RNNs) excel in capturing temporal dependencies, DCNNs maintain their superiority in scenarios where spatial features hold more significance, as seen in image-related tasks. Ensemble models, combining various techniques, present a competitive alternative, sometimes matching or exceeding DCNNs' performance, particularly in cases where diverse models contribute to improved generalization. These models find a significant application in the financial sector, particularly in credit scoring. In credit scoring, financial institutions aim to predict the probability of a loan applicant defaulting on a loan. This prediction is based on a myriad of factors including credit history, income, employment status, and others. Ensemble models combine various machine learning models like Decision Trees, Logistic Regression, and Neural Networks to assess these factors. In the real world, this translates to more accurate credit scoring, which helps financial institutions in reducing the risk of loan defaults while approving more loans for credit-worthy applicants.

The evaluation of DCNNs on diverse datasets is indispensable for gauging their generalizability and robustness across various malware types. Assessing these models on multiple datasets provides crucial insights into their adaptability and real-world performance. Key considerations include exploring malware variability, addressing imbalances in datasets, accounting for temporal aspects in malware evolution, cross-domain evaluation to assess adaptability, examining scenarios involving transfer learning, evaluating resilience against adversarial attacks, accounting for geographical variations in malware prevalence, ensuring versatility in handling different feature representations, and maintaining consistent evaluation metrics such as precision, recall, F1-score, and area under the ROC curve. This comprehensive approach to evaluation enables researchers and practitioners to develop DCNN models that can effectively navigate the dynamic and complex landscape of malware detection, ensuring their efficacy across diverse and evolving cybersecurity scenarios.

An advanced intelligent IoT malware detection model proposed based on deep learning and ensemble learning algorithms, called DEMD-IoT achieves the best outcome with the highest accuracy 99.9%, compared to state-of-the-art machine learning, deep learning, and ensemble models [3]. The 4L-DNN model outperforms other DNN architecture by a significant margin on the accuracy metric, where 98.85% was achieved on both datasets, Benign and Malicious PE Files dataset and Malware Dataset, and 98.37% was achieved on the Classification of Malwares dataset [31].

To address the challenges related to malware detection, the DEMD-IoT model leverages the power of deep learning and ensemble learning techniques. It comprises a stack of three one-dimensional convolutional neural networks (1D-CNNs) tailored to analyze IoT network traffic patterns. The model also features a meta-learner, utilizing the Random Forest algorithm, to integrate results and produce the final prediction. DEMD-IoT's advantages lie in its ensemble strategy to enhance performance and the use of hyperparameter optimization to fine-tune base learners. Notably, it employs 1D-CNNs, avoiding the complexity of preprocessing phases. Empirical evaluation on the IoT-23 dataset demonstrates that this ensemble method outperforms other models, achieving a remarkable accuracy of 99.9% [31].

Another study introduces a web-based malware detection system centered on deep learning, specifically a one-dimensional convolutional neural network (1D-CNN). Unlike traditional methods, it focuses on static features within portable executable files, making it ideal for real-time detection. The 1D-CNN architecture, tailored for these executable files, facilitates efficient feature extraction. Comparisons with state-of-the-art methods across diverse datasets confirm the model's superiority. As malware poses a significant security threat, this web-based system offers user-friendly malware detection, reducing vulnerability to cyberattacks and benefiting individuals and organizations alike. The study emphasizes the importance of deploying deep learning models in web-based applications to enhance usability and accessibility [32].

In another paper, the authors propose an efficient neural network model, EfficientNetB1, for classifying malware families using image representations of malware at the byte level. By

employing computer vision techniques, they aim to detect sophisticated and evolving malware. The evaluation of various pretrained CNN models highlights the importance of minimizing computational resource consumption during training and testing. EfficientNetB1 achieves an impressive accuracy of 99% in classifying malware classes, requiring fewer network parameters compared to other models. This work contributes to the field of cybersecurity by providing a novel approach that combines efficient neural network models and diverse image representation methods for accurate and resource-efficient malware classification [33].

To address the persistent challenge of malware detection in Windows systems, a Convolutional Neural Network (CNN)-based approach is used. It leverages the execution time behavioral features of Portable Executable (PE) files to identify and classify elusive malware. The approach was evaluated using a dataset comprising MIST files, generating images from N-grams selected by various Feature Selection Techniques. Results from 10-fold cross-validation tests showcase the remarkable malware detection accuracy, particularly when employing N-grams recommended by the Relief Feature Selection Technique. In comparison to other machine learning-based classifiers, this CNN-based approach outperforms them, offering a promising solution to enhance malware detection in Windows systems.

3.7 Deep generative models

Deep Generative Models offer a promising avenue for enhancing malware detection techniques in cyberspace. These models operate by generating synthetic data that mimics the characteristics of malicious code, thereby providing an innovative approach to detect malware. By leveraging techniques such as Variational Autoencoders (VAEs) or Generative Adversarial Networks (GANs), deep generative models can create artificial malware samples to diversify training datasets. This augmentation helps improve the robustness of malware detection systems, enabling them to recognize new and evolving threats. These models can also be employed in anomaly detection, identifying deviations from normal software behavior, which often indicates the presence of malware. Furthermore, they can generate features that enhance feature-based malware detection in cyberspace. Their adaptability and ability to generate data like malicious code samples contribute to strengthening the overall cybersecurity landscape, offering a proactive approach to identifying and combating malware threats [34].

In contrast, state-of-the-art methods and alternative approaches encompass signature-based detection, heuristic-based detection, and traditional machine learning models. Signature-based methods are efficient in identifying known malware through predefined patterns but face limitations in detecting novel threats. Heuristic approaches rely on rules and behavioral patterns, demonstrating adaptability but may produce false positives or negatives. Traditional machine learning models, while interpretable and computationally efficient, are constrained by the need for manual feature engineering and may struggle with high-dimensional data. Interpretability favors signature-based and heuristic-based methods, as well as certain traditional machine learning models, over deep generative models. However, the adaptability to novel threats is shared by deep generative models and heuristic-based approaches, distinguishing them from the limitations of signature-based methods. Deep generative models, with their strengths in unsupervised learning and anomaly detection, offer a promising avenue for addressing challenges posed by evolving and novel malware threats.

The two features extracted from the data with their respective characteristics are concatenated and entered into the malware detector of a hybrid deep generative model. By using both features, the proposed model achieves an accuracy of 97.47%, resulting in the state-of-the-art performance. [34]. A model which was verified by extensive experiments on the benchmark datasets KDD'99 and NSL-KDD effectively identifies normal and abnormal network activities. It achieves 99.73% accuracy on the KDD'99 dataset and 99.62% on the NSL-KDD dataset [35].

To tackle the challenge of detecting obfuscated malware, which often employs techniques like null value insertion and code reordering to evade traditional detection methods, a deep generative model is proposed. This model combines both global and local features by transforming malware into images to capture global characteristics efficiently and extracting local features from binary code sequences. By fusing these two types of features, the model achieves an impressive accuracy of 97.47% [36]. A novel approach is also introduced that leverages generative adversarial networks (GANs) for plausible malware training and augmentation. By training a discriminator using malware images generated by GAN models, the framework enhances the robustness of detection against 0-day malware. This eliminates the need for inefficient malware signature analysis, reducing signature complexity. The study emphasizes the importance of understanding 0-day malware features through explainable AI techniques and suggests future work on expanding the framework's applicability [35]. Despite the inherent black-box nature of these models, Explainable AI (XAI) provides a set of methodologies to shed light on their decision-making processes. Layer-wise Relevance Propagation (LRP) assigns relevance scores to input features, aiding in the identification of crucial patterns for 0-day malware features. Saliency maps highlight significant regions in input data, offering interpretability by emphasizing key areas in images or sequences. Integrated Gradients calculates feature attribution, providing nuanced insights into how variations in input features contribute to the identification of 0-day malware characteristics. Local Interpretable Model-agnostic Explanations (LIME) generates faithful interpretations by perturbing input instances, creating surrogate models for better understanding. Attention mechanisms focus on relevant parts of input sequences, aiding in the interpretation of the importance of different elements, particularly beneficial for sequential data. Counterfactual explanations generate alternative instances, showcasing the impact of input feature variations on model predictions, enhancing understanding of 0-day malware identification. Rule-based explanations extract decision rules approximating the behavior of deep generative models, offering a simplified representation for accessibility and understanding by security analysts. These explainability methods collectively contribute to a more transparent and interpretable framework, allowing analysts to dissect and comprehend the decision-making processes of deep generative models in the complex domain of 0-day malware detection.

To tackle network intrusion detection, where high-dimensional data, the scarcity of labeled samples, and real-time detection pose challenges, the proposed solution utilizes deep learning. It employs a multichannel Simple Recurrent Unit (SRU) model that outperforms traditional LSTM algorithms in efficiency and accuracy. To address the scarcity of labeled samples, a generative adversarial model (DCGAN) is used to generate training data, significantly improving system detection rates and reducing false alarms. The paper introduces efficient data preprocessing and demonstrates an impressive detection accuracy of 99.73% on KDD datasets. The SRU-based approach offers real-time intrusion detection capabilities and enhances network security [20].

While offering unique strengths, Deep Generative Models also come with several drawbacks in the context of malware detection in cyberspace. One key limitation lies in their interpretability, as these models are often perceived as black-box systems, making it challenging for security analysts to comprehend and trust their decision-making processes. Moreover, the computational intensity required for training, stemming from complex architectures and large datasets, poses practical challenges for deployment, particularly in resource-constrained environments. Data dependency is another drawback, with deep generative models relying on substantial amounts of labeled data for effective training. Acquiring diverse and representative datasets for various malware types can be logistically challenging, considering the dynamic nature of the cybersecurity landscape. Additionally, these models are vulnerable to adversarial attacks, similar to other deep learning approaches, which pose a threat to their reliability in real-world scenarios. The need for large-scale training data is a practical concern, as optimal performance often hinges on access to extensive and diverse datasets. Adapting to the dynamic nature of cybersecurity threats is another limitation, requiring frequent updates and retraining to effectively address new malware variants. Incorporating domain knowledge or expert-defined rules into the learning process can be difficult for deep generative models, hindering their ability to leverage human expertise in refining malware detection.

3.8 Deep Boltzmann machine

Deep Boltzmann Machines (DBMs) are powerful tools in malware detection in cyberspace. These deep learning algorithms excel in capturing intricate patterns within large datasets. When used for malware detection, they analyze binary code or behavioral data to identify malicious patterns and anomalies. By modeling the complex relationships between features, they effectively distinguish between benign and malicious software. DBMs offer the advantage of unsupervised learning, making them adept at uncovering novel and previously unseen malware variants. They can identify subtle and evolving threat vectors, making them crucial in the battle against constantly changing malware. Additionally, DBMs can be used for feature extraction, reducing data dimensionality and enhancing the efficiency of other detection algorithms.

Compared to other malware detection techniques, including traditional machine learning algorithms, Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs), each approach brings its unique attributes. Traditional algorithms are known for their interpretability but may require manual feature engineering. CNNs excel in spatial feature extraction for image-based tasks. RNNs outperform DBMs in handling sequential data and capturing temporal dependencies.

In the field of cybersecurity, DBMs play a pivotal role in bolstering defenses and ensuring the early identification of emerging malware threats [37]. A multi-objective RBM model aims to improve robustness and data classification accuracy. This study addresses challenges such as dataset imbalance, complex deep learning network models, and the need for multiple objectives. It leverages non-dominated sorting genetic algorithms (NSGA-II) to tackle imbalanced malware families. The proposed model, in conjunction with NSGA-II, significantly enhances data classification accuracy within HetNets, demonstrating its effectiveness in safeguarding data fusion processes [38].

To tackle dimensionality, Subspace-based Restricted Boltzmann Machines (SRBM) introduce a novel approach that combines RBMs with subspace learning. SRBM efficiently reduces feature dimensionality while considering non-linear feature relationships. Compared to other methods like PCA and Stacked Auto Encoder (SAE), SRBM stands out with significant improvements in performance metrics, enhancing efficiency and accuracy in Android malware detection [4].

To explore the application of deep learning in the detection of Denial of Service (DoS) attacks, a deep Gaussian-Bernoulli-type RBM is introduced with additional layers, optimizing hyperparameters for improved detection accuracy. This deep RBM model supports continuous data and demonstrates superior accuracy when compared to alternative RBM models, such as Bernoulli-Bernoulli RBM. The study underscores the importance of developing systems capable of detecting malicious behavior within network traffic, particularly in the context of DoS attacks [39].

In order to confirm the effect of the proposed method (RBM + NSGA-II) on the accuracy of data classification, the recall rate values with five other methods are compared. The methods are GIST + KNN, GIST + SVM, GLCM + KNN, GLCM + SVM, and DRBA, and the recall rates are 91.7, 91.4, 92.3, 93, and 94.5 percent, respectively [37]. The method proposed here has a recall rate of 95.83 percent. Next, by comparing the values of loss, recall rate, and false alarm rate with, it was found that the proposed multi-objective RBM model has loss values (loss = 0.083, 0.080, and 0.086) and recall rate values (recall = 88.64, 93.48, and 95.83 percent) are all better in three different resolutions, and the value of FPR at 50 × 50 resolution is slightly worse (FPR = 12.5 percent is greater than 11.50 percent) [37].

Obtaining and labeling appropriate training data for Deep Boltzmann Machines (DBMs) in malware detection presents multifaceted challenges with implications for model generalizability and real-world applicability. Firstly, imbalances in class distribution within malware datasets pose a challenge, potentially leading to biased model training and diminished effectiveness in detecting less common malware types. Annotating malware samples is resource-intensive, and the dynamic cybersecurity landscape introduces new variants regularly, contributing to limitations in dataset size and timeliness. This can hinder the model's capacity to generalize to evolving threats. Moreover, inherent biases in malware datasets

from different sources create potential limitations. Models trained on biased datasets may struggle to generalize across different contexts, impacting performance when faced with malware variants from underrepresented sources or regions. The active involvement of malicious actors in crafting adversarial samples further complicates the training process. Adversarial samples, intentionally manipulated to deceive the model, can compromise the robustness and reliability of the DBM in real-world scenarios. Additionally, the heterogeneous nature of malware, ranging from simple to highly sophisticated attacks, presents a challenge in capturing this diversity within a single training dataset. A lack of diversity may result in a model that struggles to identify novel and sophisticated malware types, further constraining its efficacy in practical, real-world scenarios.

3.9 Deep reinforcement learning

Deep reinforcement learning (DRL) plays a crucial role in enhancing malware detection by introducing innovative approaches to address evolving cybersecurity challenges. This advanced technique utilizes artificial intelligence and deep learning algorithms to train intelligent agents that learn to make decisions based on interactions with malware samples. These agents can determine optimal sequences of actions to modify malware, making it more difficult for anti-malware engines to detect. DRL is particularly effective in scenarios where traditional machine learning approaches struggle, especially in dealing with adversarial attacks. By allowing the agents to iteratively interact with malware, it is possible to enhance the agility and evasiveness of malware, making detection more challenging. This approach empowers researchers and cybersecurity professionals to proactively combat cyber threats, adapt to new evasion techniques, and continuously strengthen their malware detection systems [40].

In comparison to established methods, signature-based detection techniques prove effective in identifying known malware patterns, offering computational efficiency and a well-established presence in cybersecurity practices. Heuristic-based approaches leverage rules and behavioral patterns, adapting to new threats through heuristic updates. While computationally efficient, heuristics may generate false positives or negatives based on predefined rules. Traditional machine learning models, such as Support Vector Machines (SVMs) or Random Forests, provide interpretability and efficiency but may struggle with complex relationships in data due to their reliance on manual feature engineering. Analysis of these approaches reveals the superiority of DRL techniques in sequential decision-making tasks and adaptability to dynamic environments, addressing limitations seen in signature-based and traditional machine learning methods.

One method explores the evolution from traditional signature-based methods to machine learning-based algorithms for malware detection. While machine learning approaches have significantly improved detection accuracy, they remain vulnerable to adversarial attacks. The study delves into the creation of adversarial samples to test the resilience of these systems, particularly focusing on binary file modification. It discusses the complexities involved in avoiding corruption of the binary and the need to strengthen the defenses of machine learning models. The research highlights the ongoing need to enhance the robustness of malware classifiers against adversarial attacks. Another study introduces a novel framework called DQEAF (Deep Q-Learning for Evading Anti-Malware Engines), which employs DRL to bypass anti-malware engines. This framework trains an artificial intelligence agent to iteratively interact with malware samples and determine optimal sequences of non-destructive actions that modify the samples, enabling them to evade detection. The study emphasizes the effectiveness of this approach, achieving a 75% success rate in evading detection by anti-malware engines, particularly in the context of Portable Executable (PE) samples [40].

Alternative approaches delve into network security and leverage Software-Defined Networking (SDN) to optimize traffic analysis through Deep Packet Inspection (DPI). One such approach utilizes deep reinforcement learning, specifically Deep Deterministic Policy Gradient (DDPG), to intelligently allocate sampling resources in SDN-capable networks. The goal is to capture malicious network flows while minimizing the load on multiple traffic analyzers. The study showcases the efficacy of this approach in achieving more efficient traffic monitoring and cyber threat detection, highlighting the importance of data-driven decisions in traffic sampling [41]. Additionally, the vulnerability of a leading malware classifier to dead code insertion is explored, and a framework employing deep reinforcement learning, specifically a Double Q-network, is introduced to induce misclassification in the classifier. An intelligent agent, trained through a convolutional Q-network, strategically inserts NOP instructions into malware code sequences. The results demonstrate a significant reduction in the classifier's accuracy, showcasing the potential for evasion using the dead code insertion technique.

One of the primary performance metrics of DRLs lies in the rewards earned over time, depicting the agent's learning progress by maximizing cumulative rewards through interactions with an environment. The reported values for the performance comparison were an average of 500 iterations with a 95% confidence interval. Parameter values used for network topologies. traffic steering overheads than other methods while maintaining a load-balancing of traffic analyzers over 88% [41].

DRL models exhibit remarkable adaptability to new and unknown malware samples, making them valuable assets in the ever-changing landscape of cybersecurity. Their adaptability arises from the models' ability to learn optimal strategies through dynamic interactions with their environment, mirroring real-world cybersecurity scenarios effectively. In the realm of malware detection, DRL models such as Deep Q Networks (DQN) or Proximal Policy Optimization (PPO) excel in sequential decision-making tasks. This capability proves vital in scenarios where the identification process involves a series of actions and responses, enabling these models to learn optimal sequences of actions for effective detection and response to emerging threats. The models exhibit a remarkable feature learning capability, automatically extracting relevant patterns from raw input data. This reduces the reliance on predefined features or signatures, facilitating adaptability as the models can discover novel patterns associated with new malware samples without explicit feature engineering.

DRL models shine in 0-day threat detection, showcasing their prowess in identifying previously unseen and unknown threats. By learning from the dynamics of the environment and comprehending the underlying patterns of normal and malicious behavior, DRL models can adeptly adapt to emerging threats that lack historical data or predefined signatures. The support for continuous learning allows the models to stay current with the evolving threat landscape, ensuring they can effectively counter emerging risks. Reinforcement learning agents within DRL can dynamically adjust their policies based on feedback from the environment, enabling the model to update its knowledge as it encounters new malware samples. This dynamic policy adjustment significantly enhances the model's ability to handle unknown threats effectively.

Tackling data imbalance is pivotal for effective model training. One key strategy involves reward balancing, a technique aimed at adjusting reward mechanisms to address the imbalance between minority and majority classes. This approach seeks to ensure that the learning process does not disproportionately favor the majority class while neglecting the minority. By fine-tuning the reward system, the algorithm can be guided to allocate appropriate attention to underrepresented scenarios, encouraging the model to learn from these instances as rigorously as from the dominant ones. This balance fosters a more comprehensive understanding of the environment, enabling the reinforcement learning agent to make informed decisions across diverse situations. By strategically adjusting reward structures, DRL algorithms can overcome data imbalance challenges, ultimately enhancing their adaptability and performance in complex real-world scenarios.

While this model holds promise for malware detection in cyberspace, its practical use faces notable challenges. High computational requirements, especially for complex models like Deep Q Networks and Proximal Policy Optimization, pose a constraint, particularly in resource-constrained environments. Additionally, the demand for substantial training data raises concerns about data efficiency, affecting performance when labeled malware samples are limited. Lengthy training times of DRL models, particularly deep neural networks, can hinder timely deployment in dynamic cybersecurity scenarios. The black-box nature of DRL models presents interpretability challenges, making it difficult to understand the decision-making processes and the features crucial for malware detection in cyberspace. Moreover, sample inefficiency, sensitivity to hyperparameters, and difficulties in generalizing across diverse malware variants further limit the effectiveness of DRL. Vulnerability to adversarial attacks adds another layer of concern, as intentional manipulations could compromise the reliability of the model. Deploying DRL models at scale in complex network environments requires addressing scalability challenges. Ethical considerations, especially regarding privacy and potential misuse, necessitate compliance with regulatory frameworks for responsible deployment. Balancing these challenges is crucial for unlocking the full potential of DRL in the realm of cybersecurity.

3.10 Extreme Learning Machine

Extreme Learning Machine (ELMs) are increasingly used in malware detection due to their versatility and efficiency. ELMs excel in feature extraction, making them suitable for processing various data types crucial for malware analysis. Their single hidden layer with randomized weight assignments enables them to process many features quickly, which is beneficial for comprehensive malware detection in cyberspace. While this allows for quick processing of many features, it may not capture complex relationships and dependencies in the data as effectively as models with multiple hidden layers and optimized weight assignments. ELMs are particularly favored for their fast-training process, as they do not involve iterative weight optimization. This speed and their ability to handle diverse data types make ELMs a valuable tool in the ongoing battle against malware [42].

While this may be beneficial for efficiency, it could also limit the model's ability to fine-tune and improve its performance over time. Iterative weight optimization techniques, such as backpropagation, are commonly used in other machine learning models to refine the model's predictions and achieve higher accuracy. Research in this field addresses the pressing challenges posed by malware, with a primary focus on improving accuracy, automation, and efficiency in the detection process. Data Imbalance can pose a significant problem in ELM training but several strategies can help address this challenge. Resampling techniques like oversampling or undersampling methods are effective approaches. Oversampling involves increasing the instances of the minority class, while undersampling reduces the instances of the majority class, aiming to balance the dataset's representation. This helps prevent the model from being biased toward the majority class. Another valuable strategy is weighted learning, where different weights are assigned to samples based on their class. By assigning higher weights to minority class samples and lower weights to majority class samples, the learning process becomes more balanced, allowing the model to better discern patterns from the less represented class.

Notably, one of these studies introduces an innovative approach in the form of a Two-hidden-layered Extreme Learning Machine (TELM), which departs from conventional backpropagation techniques to offer a streamlined and faster approach to malware detection in cyberspace. This approach incorporates dependencies of malware sequence elements, effectively enhancing the accuracy of classification while dramatically reducing both training and detection time. The practical implications of this study are profound, particularly in safety-critical systems such as healthcare and the Internet of Things (IoT), where rapid and reliable malware detection is imperative [43].

While the Two-hidden-layered Extreme Learning Machine (TELM) and the Gauss-Mapping Black Widow Optimization with Deep Learning Enabled Android Malware Classification (GBWODL-AMC) demonstrate efficacy in malware detection, they exhibit specific limitations and trade-offs. TELM, characterized by its two-hidden-layer extreme learning architecture, faces challenges in interpretability due to the inherent complexity of deep learning models. The model's decision-making processes might be challenging to decipher, potentially impacting the trust users place in its outputs. Additionally, TELM's performance is contingent on the availability of sufficient labeled training data, making it susceptible to constraints in scenarios where obtaining diverse datasets is difficult. The computational intensity of training TELM models, especially with larger datasets and intricate

architectures, can pose challenges in resource-constrained environments.

On the other hand, GBWODL-AMC, which integrates Gauss-Mapping Black Widow Optimization, introduces its own set of limitations. The model's effectiveness is tied to the appropriateness of the chosen optimization technique, and its dependency on this specific strategy may limit its applicability across diverse problem domains. Hyperparameter sensitivity poses a trade-off, requiring careful experimentation to select optimal values and avoid reduced model performance. Deep learning components within GBWODL-AMC are susceptible to potential overfitting, especially when dealing with complex datasets, which may hinder the model's generalization to new, unseen data. Similar to TELM, the model's interpretability may be compromised due to the inherent complexity of deep learning architectures.

It is common for Android malware to employ code obfuscation techniques to evade detection. In response, a cutting-edge model, the Gauss-Mapping Black Widow Optimization with Deep Learning Enabled Android Malware Classification (GBWODL-AMC), is introduced. This model combines novel feature selection techniques with deep extreme learning, and through meticulous parameter optimization, it achieves a remarkable accuracy rate of up to 98.95%. The significance of this research extends to the realm of mobile device security, providing a promising solution for more effectively combating Android malware.

Another technique delves into the critical domain of detecting obfuscated malware within network traffic. It introduces the MalHyStack hybrid classification model, a powerful fusion of machine learning algorithms and deep learning [44]. This model, through the incorporation of feature subset selection and a balanced dataset, achieves exceptional accuracy rates that surpass existing models. The broader implication of this research is evident in its ability to combat obfuscated malware efficiently while maintaining a high degree of accuracy.

In comparing Extreme Learning Machines (ELMs) and Convolutional Neural Networks (CNNs) for malware detection across a broader range of datasets, each approach exhibits distinct strengths and weaknesses. ELMs are characterized by their fast training times, simplicity, and non-iterative training, making them efficient for large datasets and resource-constrained scenarios. However, ELMs may face challenges in capturing hierarchical features and may require manual feature engineering, limiting their suitability for complex data, such as images or sequences. On the other hand, CNNs excel in spatial feature extraction, hierarchical representation learning, and end-to-end learning, making them particularly effective for image-based malware detection tasks [42]. Their ability to automatically learn hierarchical representations from raw data eliminates the need for extensive manual feature engineering. Nonetheless, CNNs come with computational intensity during training, interpretability challenges due to their black-box nature, and a dependency on large labeled datasets, posing challenges in data acquisition. When applied to image-based malware detection, ELMs may perform well under resource constraints, while CNNs are likely to outperform ELMs due to their proficiency in spatial feature extraction. In handling sequential data like API calls or network traffic, ELMs may struggle with temporal dependencies and might require additional feature engineering, whereas CNNs, with modifications like 1D convolutions or recurrent layers, offer a more robust performance. For multimodal data encompassing a combination of images and sequences, ELMs may need careful feature engineering, while CNNs, capable of processing both types of data, provide a more comprehensive solution. In terms of generalization across diverse malware datasets, ELMs might face challenges, especially with complex hierarchical features. In contrast, CNNs, with their inherent capacity for hierarchical representation learning, demonstrate potential for better generalization across a broad spectrum of malware datasets [43].

In the realm of malware detection, enhancing the interpretability of deep learning models holds immense importance for establishing trust, comprehending model predictions, and gaining insights into classification decisions. To achieve this, several key approaches have emerged. Feature visualization techniques enable the understanding of the specific characteristics or patterns that the model identifies as indicative of malware. Techniques like activation maximization or gradient-based methods visualize salient features, such as sequences of system calls or network traffic patterns. Saliency maps, generated through methods like Grad-CAM, highlight crucial regions in the input data that influence the model's output, shedding light on the importance of different input features in model decisions. Attention mechanisms, prevalent in models like transformers, aid in understanding the model's processing of sequential data by visualizing attention weights, indicating the elements of input sequences that hold more significance [44]. Transforming complex models into interpretable rule-based systems, accomplished through rule extraction algorithms or decision tree induction, simplifies model logic and facilitates comprehension. Layer-wise relevance propagation helps attribute relevance to input features by discerning the contribution of different model layers to the final decision. Model distillation aims to simplify complex models while retaining performance, training smaller, more interpretable models to mimic the behavior of intricate deep learning models. Techniques like Integrated Gradients or SHAP values quantify the impact of each input feature on model output, providing a clear understanding of feature importance. Moreover, domain-specific visualization tools tailored for malware analysis offer interactive dashboards or tools for security analysts, enabling intuitive exploration of model decisions and deeper insights into malware behavior.

3.11 Attention models

In the ongoing battle against malware, traditional detection methods are struggling to keep pace with the constant innovation of cybercriminals. Attention models, a powerful deep learning technique, are emerging as a game-changer. These models do not treat all aspects of a file equally; instead, they learn from vast datasets of malware and benign software to identify the most critical features–the red flags that scream "malicious." The polymorphism in malicious components has deteriorate the situation, as malicious files, which essentially belong to the same malware "family" and have the same form of malicious behavior, are constantly modified, or obfuscated using various strategies to make them appear to be many different files [45]. By focusing on these key elements, attention models can achieve higher accuracy in detecting

both known and unknown malware strains, while also reducing the number of innocent files mistakenly flagged as threats. This ability to adapt and learn makes them invaluable in the fight against ever-evolving cyber threats. Traditional methods often suffer from a high rate of false positives, mistakenly quarantining harmless files. Existing gray image based malware detection and classification approaches are primarily based on conventional machine learning or deep learning with Convolutional Neural Networks (CNNs). GIST + kNN pioneers the application of machine learning in malware classification on the Malimg dataset. Subsequent studies relied on features extracted from PCA features, N-gram application programming interface (API) sequences, opcodes, control flow graphs, text semantics of network traffic and URLs, system calls, OS-level information flow and the network activities of the malware. While these advancements have broadly elevated the field, they require manual feature design, deep foundational knowledge, and even the construction of complex network system environments for detection and classification [45]. Attention models, by placing less weight on irrelevant features, can significantly reduce these false alarms. This translates to less wasted time and resources for security teams, allowing them to focus on genuine threats. Additionally, some attention models offer a degree of explainability. They can provide insights into why a particular file was classified as malware, helping security professionals understand the attacker's techniques and potentially identify vulnerabilities that need to be addressed.

The attention-based feature extraction method allows malicious code analysts to only analyze parts of malicious code based on the features extracted by the attention-based feature extraction method, rather than analyzing the entire malicious code. This is expected to considerably reduce the efforts required by malicious code analysts [46]. An implementation of the ARI cell with LSTM networks, called ARI-LSTM enhances the LSTM cell by incorporating ARI mechanism within the cell, and use sthe resulting neural network for sequence learning with ransomware. Through evaluation on a ransomware dataset for the Windows operating system environment, it is seen that ARI-LSTM improves the performance of an LSTM in detecting ransomware from emulation sequences [47].

Cross-dataset experiments conducted on the Windows and Android datasets, with an accuracy of 90.64% on cross-dataset detection of the android [45]. The attention-based model yielded an accuracy that was approximately 12% and 5% higher than those of the CNN-based and SC-LSTM-based models, respectively [46].

However, attention models are not without their challenges. Training these complex algorithms requires significant resources. Large, diverse datasets of malware samples are essential for them to learn and adapt effectively. Additionally, the computational cost of training and deploying these models can be substantial. Finally, while some models offer explanations, their inner workings can be intricate, requiring expertise to fully comprehend. Despite these challenges, the potential of attention models is undeniable. Their ability to learn, adapt, and focus on the most critical features makes them a powerful weapon in the fight against malware. As these models continue to evolve and become more accessible, they hold the promise of a future where cyber defenses are more agile and effective, constantly learning and adapting to the ever-changing threat landscape.

3.12 Summary and interpretability of deep learning models

Choosing the optimal model for malware detection hinges on several factors, including dataset characteristics, feature requirements, and performance expectations. Among the considered models, each possesses unique strengths. Recurrent Neural Networks (RNNs) excel in capturing temporal dependencies, making them suitable for sequential data. Deep Autoencoders prove effective in learning hierarchical representations, particularly for anomaly detection. Long Short-Term Memory (LSTM) networks, designed for sequential data, demonstrate prowess in handling long-term dependencies. Deep Neural Networks (DNNs) are versatile, capable of learning complex non-linear mappings. Deep Belief Networks (DBNs) are adept at unsupervised learning and hierarchical representation learning.

Deep Convolutional Neural Networks (CNNs) are well-suited for image-based data, capturing spatial hierarchies effectively. Deep Generative Models can generate new samples, aiding in understanding data distribution. Deep Boltzmann Machines are suitable for unsupervised learning and complex dependency modeling. Deep Reinforcement Learning is designed for tasks involving agent-environment interactions and policy learning. Extreme Learning Machines (ELMs) stand out for their fast training, simplicity, and good generalization.

For malware detection, a combination or ensemble approach may prove effective. Models like CNNs can extract features from binary files or images, while RNNs or LSTMs capture temporal dependencies in malware behavior. Unsupervised learning models like autoencoders or DBNs can aid in anomaly detection, identifying novel malware patterns. Experimentation and evaluation on specific datasets are crucial to determine the most effective model, and regular updates are essential to adapt to evolving malware threats.

4 Comparisons with non deep learning models

In the realm of malware detection, traditional non-deep learning methods like signature-based detection heavily rely on predefined patterns, making them susceptible to 0-day threats. Heuristic-based approaches utilize rules but can struggle to adapt to evolving tactics. Behavioural analysis, while effective, faces scalability issues and might overlook subtle anomalies. Non-deep learning machine learning algorithms such as Decision Trees and SVMs demand expert feature engineering and struggle with high-dimensional data complexity [7].

In contrast, deep learning models provide a range of distinct advantages over these traditional methods. They autonomously learn intricate features from raw data, bypassing the limitations of handcrafted features in traditional methods. Their adaptability to diverse data types and capacity to generalize to new, unseen malware variants outshine the rigidity of traditional approaches, often reliant on frequent updates [8]. Deep learning excels in capturing complex patterns and relationships, particularly in large-scale datasets, surpassing traditional methods in nuanced pattern recognition tasks.

Furthermore, deep learning's scalability and automation in handling large datasets streamline feature extraction, whereas traditional methods may encounter scalability limitations. However, while traditional methods often boast interpretability due to explicit rule-based decisions, deep learning models' complex architectures render them less interpretable, though ongoing efforts aim to enhance interpretability through emerging techniques. Additionally, while deep learning can learn from imbalanced data, it requires specific strategies to effectively manage class imbalance, a challenge that traditional methods also encounter, often necessitating sophisticated sampling or weighting techniques [6, 10–12].

5 Smartphone applications in malware analysis and detection

Smartphone applications have emerged as valuable assets in the field of malware analysis and detection. Numerous tools have been designed for both Android and iOS platforms, leveraging the computing power and connectivity of smartphones to enhance the capabilities of security professionals and organizations. These applications play a pivotal role in scanning and analyzing mobile apps for known malware signatures, identifying behavioral anomalies and vulnerabilities. This aids in the early detection of potentially harmful applications and helps prevent device compromise [48].

One key feature of these applications is real-time monitoring, which keeps a vigilant eye on network traffic, system activity, and app behavior. This continuous surveillance is crucial for identifying suspicious or malicious activities on mobile devices, enabling prompt alerts to users or administrators about unusual behaviors or interactions with known malicious domains. Integration with Mobile Device Management (MDM) solutions further enhances the functionality of these applications. MDM allows organizations to manage and secure mobile devices remotely, enforce security policies, deploy updates, and, if necessary, remotely wipe compromised devices. This integration is particularly beneficial for enterprises looking to safeguard their mobile device ecosystem [49].

Furthermore, some of these smartphone applications incorporate threat intelligence feeds, providing access to the latest information on mobile-specific threats and indicators of compromise. This integration significantly improves detection and response capabilities against emerging threats. Another aspect of these applications is app reputation scanning [50]. They assess mobile applications based on various factors, including the source, required permissions, and code behavior. This enables users and administrators to make informed decisions about app installation and usage.

Behavioral analysis is another advanced feature offered by some applications, where they monitor the interactions of mobile apps with device resources and the network. This method effectively unveils hidden or obfuscated malicious activities that may not be evident through static analysis alone. Additionally, there is a focus on user education within these applications. They provide tips and information about potential security risks and best practices for safe mobile device usage, empowering users to become more aware and vigilant regarding their security [51].

These applications often include the capability to detect rooting or jailbreaking of devices. Rooted or jailbroken devices are more susceptible to security risks, and detecting such modifications is crucial for alerting users and administrators to potential tampering or compromise. The contribution of smartphone applications to malware analysis and detection is increasingly significant, especially as mobile devices become more prevalent targets for cybercriminals. These applications empower users and organizations to proactively defend their smartphones and the sensitive data they contain. As the mobile threat landscape evolves, the importance of these applications in ensuring mobile security and privacy becomes even more critical [52].

6 Recent tools in malware analysis and detection

Machine learning has become a popular approach for malware detection due to its proficiency in identifying patterns and anomalies in large datasets [53–55]. Various algorithms, such as Random Forest, Support Vector Machines, and Neural Networks, are employed to analyze features extracted from executable files, including binary code, API calls, and file metadata, effectively detecting malware. Dynamic analysis tools, which involve executing malware in a controlled environment to observe its behavior, have also seen significant advancements. Modern tools offer capabilities like automated sandboxing and advanced code instrumentation, allowing for real-time monitoring of system and network activities and analysis of the malware's actions. Examples of these tools include Cuckoo Sandbox, Any. Run, and Hybrid Analysis.

Behavioral analysis is another critical area, focusing on how malware behaves upon execution. By using advanced techniques to detect abnormal behavior patterns, such as process injection and privilege escalation, these tools can identify malicious actions, enabling the detection of previously unknown malware. Memory forensics tools have become increasingly sophisticated, with tools like Volatility enabling analysts to extract and analyze information from a system's RAM. These tools are crucial for uncovering hidden processes, rootkits, and other memory-resident malware. YARA rules have gained popularity for creating custom patterns to identify specific malware characteristics. These rules, defined by security professionals, are instrumental in both static and dynamic analysis phases. In response to malware authors developing sandbox-evading techniques, analysts have improved sandbox environments to mimic real systems more closely and developed methods to detect sandbox detection techniques. Blockchain technology is also being utilized to create immutable and transparent threat intelligence databases. This innovation aids in the secure sharing and distribution of malware indicators, facilitating quicker detection and response to emerging threats.

Deep learning, including techniques like convolutional and recurrent neural networks, is increasingly applied in malware analysis. These methods are capable of learning intricate patterns and behaviors, enhancing the accuracy of identifying malicious code and activities. Zero-Day Vulnerability Scanners are evolving to identify vulnerabilities that might be exploited by malware. These tools employ a range of techniques, such as static analysis and

fuzzing, to detect 0-day vulnerabilities. Finally, with the rise of IoT devices, specialized tools and techniques are emerging for IoT-specific malware analysis. These tools focus on the unique characteristics and communication patterns of IoT devices, aiding in the detection and analysis of potential threats in this growing domain. Figure 4 illustrates the categories of recent tools in malware analysis and detection.

6.1 Behavioral analysis tools

Behavioral analysis is a crucial approach in malware analysis and detection, focusing on how malware behaves when executed in a controlled environment [56–59]. This technique observes the dynamic actions and interactions of malware with the host system and network, allowing for the detection of malicious behavior that may not be evident through static analysis alone. The core of behavioral analysis involves creating a dynamic execution environment, commonly known as a sandbox, where malware samples can be safely executed. This environment replicates the target system, enabling the malware to run without causing harm to the actual host.

Within this setting, various tools monitor different aspects of the malware's behavior, including file system interactions, registry modifications, process creation, and network communication. These tools log system calls, API functions, and other activities to meticulously track the sequence of events. Behavioral analysis tools also establish behavioral signatures that define what constitutes normal system behavior. By comparing the observed actions of the malware against these signatures, abnormal and potentially malicious behavior can be identified, such as attempts to encrypt files or establish unauthorized network connections. Additionally, heuristic algorithms and anomaly detection techniques are applied to the collected data. These algorithms search for patterns that deviate from the expected norm, flagging activities that indicate malicious intent. This approach is particularly effective in detecting previously unknown malware. Once the analysis is complete, these tools generate comprehensive reports detailing the malware's behavior, its impact on the system, and indicators of compromise (IOCs).

6.2 Threat intelligence platforms

Threat Intelligence Platforms (TIPs) are crucial for enhancing malware analysis and detection. They offer a structured framework for collecting, aggregating, analyzing, and disseminating threat intelligence. These platforms are invaluable for cybersecurity professionals and organizations as they provide critical insights to proactively defend against emerging threats [60–63]. A key function of TIPs is aggregating data from diverse sources, including feeds, internal logs, open-source intelligence, and proprietary databases. This data includes indicators of compromise (IOCs) such as malware signatures, IP addresses, domain names, and file hashes, thus providing a comprehensive view of potential threats. TIPs also play a vital role in normalizing and enriching this raw threat data, ensuring consistency and actionability. They standardize different data formats and add contextual information, such as source reputation and known malware families, thereby enhancing the quality and relevance of the intelligence. Moreover, TIPs employ sophisticated algorithms to correlate and analyze the collected data. This process involves identifying patterns, trends, and anomalies that may indicate malware infections or other malicious activities. Advanced techniques like machine learning and data analytics are often utilized to uncover previously unknown threats.

In terms of incident response, TIPs provide real-time alerts and playbooks for security teams. They enable automated actions based on received intelligence, such as blocking malicious IP addresses or isolating infected devices, thereby facilitating swift and effective responses to threats. Furthermore, these platforms promote the sharing of threat intelligence within trusted networks and information-sharing communities. This collaboration allows organizations to benefit from collective insights and strengthen their overall security posture. Standards like STIX/TAXII are often employed to facilitate the exchange of information. TIPs offer a high degree of customization and can be tailored to meet the specific needs of an organization. They often integrate with existing security tools and infrastructures, such as SIEMs, firewalls, and endpoint protection systems, to provide automated responses and adaptability to the ever-evolving threat landscape.

Moreover, the storage of historical threat data by TIPs is crucial for trend analysis and retrospective investigations. This historical perspective aids in identifying long-term patterns and understanding how cybercriminal tactics evolve over time. TIPs assist organizations in compliance reporting by maintaining detailed records of threat intelligence and incident response activities. These records are essential for meeting regulatory requirements and facilitating audits, thus playing a critical role in organizational compliance strategies.

6.3 Deception tools

Deception tools, a relatively new but increasingly vital component in the cybersecurity landscape, have demonstrated remarkable effectiveness in malware analysis and detection. These tools are designed to create a deceptive environment within a network with the aim of misleading, confusing, and ultimately trapping malicious actors and malware. They employ various strategies and technologies to achieve this goal [64–66]. A common technique used in deception tools is the deployment of honeypots and honeynets, which essentially act as decoy systems created to mimic real assets within a network. These systems are designed to be enticing to attackers, drawing them in and piquing their interest. Honeypots can range in complexity, from low-interaction versions that emulate services and applications at a basic level, to high-interaction variants that closely simulate real systems, thus enticing attackers further into the deception.

In addition to these decoy systems, deception tools also generate counterfeit data and services. This includes forged documents, credentials, and network shares that appear legitimate and attractive to attackers. When attackers engage with this deceptive content, the tools capture their actions, enabling detailed analysis of their tactics, techniques, and procedures (TTPs). Advanced simulation techniques are another facet of deception tools, going beyond simple emulation. These tools can mimic actual network

behavior, including simulating user actions, generating realistic traffic patterns, and replicating the unique "personality" of a network. This level of sophistication makes the deceptive environment more convincing and effective.

One of the key advantages of deception tools is their ability to provide early detection of potential threats. When attackers or malware interact with these deceptive elements, they inadvertently trigger alerts, notifying security teams of the presence of a threat. This enables rapid investigation and response, helping to mitigate risks more efficiently. Beyond mere detection, these tools play a critical role in attribution and analysis. By examining how attackers interact with the decoys, security teams can gain valuable insights into their methodologies. This understanding is crucial for developing more effective countermeasures against future attacks. Deception tools are also adept at luring and containing malware. They can create simulated vulnerabilities or backdoors specifically designed to be exploited by malware, allowing for the isolation and detailed analysis of the malicious code. This capability is particularly useful for studying malware behavior and developing strategies to neutralize it.

Another significant advantage of deception tools is their ability to minimize false positives. By focusing on interactions with the deceptive elements, these tools reduce the volume of irrelevant alerts, streamlining the workload of security teams and enhancing the overall efficiency of malware detection. The most advanced deception tools are adaptive, capable of evolving over time based on observed attacker behavior. They continuously refine and update their deceptive elements to make them even more convincing, ensuring they remain effective against evolving threats and sophisticated attackers. This adaptive nature underscores the dynamic and proactive approach of deception tools in the ongoing battle against cyber threats.

6.4 Memory forensics tools

Memory forensics tools are essential in the field of malware analysis and detection, providing cybersecurity experts with the means to examine a computer's volatile memory (RAM) for indications of malicious activities [67–69]. In the overall process of forensic analysis, memory analysis plays a crucial role since malware often resides in memory to avoid detection and maintain its presence. These tools facilitate the retrieval of memory dumps from live systems or capture memory images from forensic images, which include active processes, data structures, and code present at the time of acquisition. They enable detailed analysis of these memory dumps, focusing on identifying running processes, their memory footprints, and associated threads. This analytical process is crucial for identifying suspicious or unauthorized applications that may be operating covertly.

A significant function of memory forensics tools is malware detection. They are skilled at scanning memory for known malicious signatures, patterns, or behaviors. This includes identifying injected or obfuscated code, rootkits, and other forms of memory-resident malware that are notoriously difficult to detect through conventional means. Furthermore, these tools are instrumental in uncovering evidence of API hooking and function call redirection, tactics commonly employed by malware to intercept and manipulate system calls. This capability is crucial for understanding the extent of the malware's control over a system. Detection of rootkits is another vital aspect of memory analysis, as rootkits are designed to be invisible to traditional file-based forensics. Memory forensics tools can reveal hidden processes, files, and network connections by exploring memory structures.

Advanced memory forensics tools even extend their capabilities to the examination of kernel memory, a critical area where essential system data and structures reside. Analyzing this segment can provide deep insights into the inner workings of the operating system and any potential manipulations by malware. These tools can also analyze memory dumps to extract information about active network connections and related data, assisting in the identification of malicious network communications. This analysis of network activity is crucial for understanding how malware communicates and potentially exfiltrates data. Timeline reconstruction is another crucial feature offered by memory forensics tools. By analyzing memory dumps over a period of time, analysts can piece together a timeline of events, revealing the sequence in which processes were initiated and actions were executed. This is particularly helpful in understanding the development and spread of a malware infection within a system. The extensibility of many memory forensics tools through plugin support enhances their utility significantly. Analysts can utilize custom scripts or leverage pre-built plugins to automate and refine the analysis process, making these tools even more powerful in combating sophisticated malware threats.

6.5 Sandboxing with threat intelligence integration

Sandboxing with integrated threat intelligence represents a sophisticated and effective approach to malware analysis and detection by combining isolated environments with up-to-date threat intelligence [70–72]. This method offers a comprehensive and dynamic means of understanding and identifying malicious software. At its core, sandboxing involves executing potentially malicious code within a controlled and isolated environment, such as a virtual machine or container. This setup closely mimics a real system, encouraging malware to demonstrate its full functionality and intentions. During the sandboxing process, malware is allowed to run freely, enabling real-time capture of its behavior, including file system interactions, registry changes, network communications, and process activities. This dynamic analysis provides valuable insights into the malware's execution flow, evasion tactics, and persistence mechanisms.

The integration of threat intelligence feeds is a crucial aspect of this approach. These feeds are constantly updated sources of information, providing the latest data on known threats, indicators of compromise (IOCs), malware signatures, and other relevant security details. By incorporating these feeds into the sandboxing environment, it becomes possible to retrieve up-to-date threat information instantly. Consequently, the behavior of the analyzed code can be cross-referenced with this data, facilitating the detection of matches with known malware. The sandbox also

plays a vital role in IOC detection by scrutinizing the behaviors and attributes of the executed code against the IOCs from threat intelligence feeds. A match suggests that the code under analysis exhibits characteristics typical of known malicious software. Upon detecting a match or suspicious behavior, the sandbox generates alerts and detailed reports. These reports provide valuable information about the malware's actions, potential impact, and IOCs, which are crucial for further investigation and mitigation efforts. Additionally, integrating sandboxing with incident response tools can trigger automated responses, such as isolating affected systems, blocking malicious domains, or generating tickets for human analysts to investigate further.

Another advantage of this integrated approach is its support for historical analysis. The threat intelligence within the sandbox allows for checking previously analyzed samples for known threats, aiding in the identification of recurring attack patterns and related malware families. By combining sandboxing with threat intelligence integration, organizations gain a proactive, responsive, and insightful method for conducting malware analysis and detection. This approach not only aids in identifying and mitigating known threats but also plays a crucial role in addressing emerging threats by leveraging the latest intelligence data in real-time while closely monitoring the behavior of suspicious code in a secure and controlled environment.

7 Open challenges

In the field of malware detection using deep learning, there are several challenges that need to be addressed and promising avenues for future research [23, 73–85]. Figure 5 illustrates the open challenges associated with the deep learning-powered malware detection in cyberspace. One of the main challenges is the need to enhance the resilience of deep learning models against adversarial attacks, which are increasingly employed by malware authors to evade detection. Additionally, it is crucial to develop interpretable models that shed light on the decision-making processes of these models. Real-time detection, particularly in streaming environments, is becoming imperative to swiftly identify and counteract malware propagation. As the volume of malware samples and feature spaces continues to expand, scalability concerns must be addressed [44, 86–98].

Another important area for exploration is the development of techniques for few-shot and zero-shot learning, which can facilitate the detection of new and previously unseen malware strains. This capability is crucial in the ever-evolving threat landscape. The fusion of data from multiple sources, privacy-preserving methods for sharing labeled malware samples, and ethical considerations are also significant areas for research [14, 75, 98–110]. Improving malware detection accuracy can be achieved through efficiency in model architectures, seamless integration with existing security systems, cross-domain transfer learning, hybrid models that combine different deep learning architectures, and automated feature engineering methods. User education and awareness also play a pivotal role in reducing inadvertent installation or interaction with malware [21, 111–117]. Finally, collaborative threat intelligence platforms that enable information sharing among organizations represent a promising approach to collectively strengthen defenses against malware. Figure 3 illustrates the open challenges in deep learning-powered malware detection in cyberspace.

The field of deep learning-powered malware detection encompasses various challenges and solutions [118–123]. One significant challenge is handling 0-day attacks, as deep learning models traditionally rely on historical data and struggle against novel, unseen threats [124–128]. To address this, techniques such as transfer learning and anomaly detection should be employed to enhance the models' ability to detect new threats. Another area of concern is the collection, standardization, benchmarking, and reproducibility of malware datasets. The lack of standardized datasets and evaluation metrics hinders fair comparisons between different deep learning models. Overcoming this challenge requires the establishment of standardized benchmarks and datasets for malware analysis, as well as promoting open data sharing and collaboration within the research community.

The mathematical provability and interpretability of deep learning-powered models also pose challenges. These models, especially neural networks, are often considered "black boxes," making their decision-making processes opaque. It is essential to develop interpretable models or techniques that explain the predictions of deep learning models to ensure transparency and trust in malware detection systems. Additionally, class imbalance and distribution bias in training and testing datasets can significantly impact model performance. Imbalanced datasets tend to bias models towards the majority class, resulting in poor performance on minority classes that are often crucial in malware detection. Techniques like oversampling, undersampling, or synthetic data generation, along with tailored evaluation metrics, are vital for addressing this issue. Adapting to real-world settings and maintaining context awareness is another hurdle.

Deep learning models may struggle to adapt to rapidly changing environments, leading to potential obsolescence [129–136]. Developing dynamic models capable of continuous learning from new data and adapting to evolving threat landscapes is a solution to this problem. The lack of benchmark platforms for deep learning-powered malware detection research also hampers progress and collaboration in the field. Establishing such benchmark platforms and encouraging competitions can foster innovation and the development of more effective malware detection solutions. The aging problem of malware detection and classification tools is an ongoing challenge. As attackers evolve their tactics and techniques, malware detection tools often become less effective over time. To address this, continuous research and development are necessary to keep these tools updated and capable of identifying new attack vectors.

8 Future research directions

Deep learning techniques are revolutionizing malware detection, offering innovative approaches to tackle the complexity and sophistication of modern cyber threats. Graph Neural Networks (GNNs) excel in comprehending intricate relationships within graph-structured data, enabling a deeper understanding of malware behavior patterns often missed by traditional models. Transformer-based architectures, renowned for their success in natural language processing, hold promise in capturing temporal dependencies within sequences of system or API calls, potentially enhancing the comprehension of malware behavior.

The emergence of meta-learning techniques empowers models to swiftly adapt to new malware variants or unseen attack patterns, bolstering the adaptability and generalization of detection systems. Self-supervised learning, by training models on unlabeled data, unveils latent features and anomalies within malware, potentially improving identification accuracy. Federated learning, a collaborative approach, allows multiple devices or organizations to jointly train models without compromising data privacy, leading to more robust and accurate malware detection systems. A prime example of the effectiveness of Federated Learning is its application in improving predictive text and autocorrect features on smartphones. This technology involves training an algorithm across multiple decentralized devices (or servers) holding local data samples, without exchanging them. This method is used by major tech companies to enhance their keyboard applications. In this scenario, each smartphone has a local model that learns from the user's typing behavior. Instead of sending individual data points (like the words typed) back to a central server, the smartphone computes an update to the model based on the local data and only sends this model update back to the server. This way, the central model gets trained over time with the aggregated updates from millions of users, without ever having access to specific examples from any individual's data. This preserves privacy while still benefiting from the collective learning of all users.

Adversarial robustness techniques aim to fortify models against attacks, ensuring the reliability of malware detection systems in the face of adversarial threats. Continual learning techniques enable models to evolve with changing environments, incorporating new malware behaviors while retaining the ability to detect historical attack patterns. Finally, Explainable AI (XAI) techniques enhance the interpretability of models, fostering trust and aiding cybersecurity experts in comprehending model decisions. These emerging deep learning techniques collectively promise to elevate the efficacy and resilience of malware detection systems, offering a more comprehensive defense against evolving cyber threats.

The advancement of technology will significantly contribute to the progress of research in malware detection using deep learning models [137–140]. In Explainable Artificial Intelligence, the focus should be on enhancing the interpretability and transparency of deep learning models for cybersecurity experts. This entails developing neural network architectures that are easier to understand and techniques for generating explanations of model predictions in a human-readable format. Additionally, in Generative Artificial Intelligence, there is a need to explore how generative models like GANs and VAEs can be utilized to generate synthetic malware samples. These samples can be used to train deep learning models, allowing them to mimic the creativity of malware authors and enabling more robust model training and testing. Moreover, in the context of the Internet of Everything (IoE), deep learning models can be applied to analyze and secure interconnected devices and networks. It is crucial to address the unique challenges and vulnerabilities that arise in malware detection within the IoE ecosystem.

The limitations inherent in singular deep learning models for malware detection necessitate exploration of more advanced methodologies. Hybrid and ensemble techniques present promising avenues for enhanced threat coverage and resilience. These approaches can synergistically combine the strengths of deep learning architectures, such as convolutional neural networks (CNNs) and long short-term memory networks (LSTMs), with established methods like rule-based pre-filtering and feature engineering. For instance, domain knowledge may be leveraged to extract salient features from code and network traffic, which can then be fed into CNNs for automated learning of complex representations. Subsequently, LSTMs can analyze the remaining data for intricate temporal sequences indicative of novel malware, reducing computational burden and focusing resources on potential threats. Ensemble techniques further diversify the defensive landscape by combining diverse deep learning models trained on disparate data representations. Meta-learning algorithms can then orchestrate the collective predictions of these models, resulting in enhanced generalizability and improved resilience against evasion attempts.

However, the dynamic nature of the malware landscape demands agile solutions. Continuous learning techniques empower models to dynamically update their knowledge base with incoming data and emerging threats, obviating the need for complete retraining. Incremental learning approaches, such as online learning with memory replay, enable models to continuously learn from new data points while retaining past knowledge, mitigating the risk of catastrophic forgetting. Curriculum learning further facilitates this process by gradually exposing the model to more complex malware samples, building a robust foundation for accurate real-world detection. Additionally, meta-learning techniques can equip models with the ability to learn how to learn quickly on new tasks, enabling rapid adaptation to novel malware variants.

9 Conclusion

This article delves into the realm of deep learning models for malware detection in cyberspace, highlighting their significance and contributions to the field of cybersecurity. Deep learning models have emerged as powerful tools in combating malware, offering unparalleled potential in automatically learning features from vast datasets. However, it is crucial to acknowledge the limitations of current deep learning techniques in malware detection. These limitations include the vulnerability of deep learning models to adversarial attacks and the necessity of large, labeled datasets for effective training. Future directions in this field could involve exploring federated learning techniques to enhance privacy and reduce reliance on centralized data collection. Additionally, combining multiple deep learning approaches, such as ensemble models, could further enhance detection capabilities, particularly against evolving and sophisticated malware threats. The impact of deep learning on malware detection in cyberspace has been substantial. These models have revolutionized the field by providing accurate and efficient means of categorizing malware into distinct families or types. They empower security researchers and practitioners to swiftly identify and counter emerging threats, ultimately strengthening cybersecurity practices. The diversity of deep learning architectures, including Recurrent Neural Networks (RNNs) and Deep Convolutional Neural Networks (DCNNs), has expanded the range of applications in malware detection, making them a critical tool in the ongoing battle against evolving cyber threats. As cybersecurity concerns continue to grow, deep learning emerges as a viable option for advancing the state of the art in malware identification and analysis.

Author contributions

AR: Data curation, Formal Analysis, Investigation, Writing–original draft, Writing–review and editing. PC: Data curation, Formal Analysis, Investigation, Writing–original draft, Writing–review and editing. KS: Conceptualization, Data curation, Investigation, Methodology, Software, Supervision, Visualization, Writing–original draft, Writing–review and editing. TD: Funding acquisition, Project administration, Validation, Writing–review and editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The author(s) declared that they were an editorial board member of Frontiers, at the time of submission. This had no impact on the peer review process and the final decision.

Publisher's note

References

1. Kwon I, Im EG. Extracting the representative API call patterns of malware families using recurrent neural network. In: Proceedings of the Proceedings of the International Conference on Research in Adaptive and Convergent Systems; September 20-23, 2017; ACM: Krakow Poland (2017). p. 202–7.

2. Amin M, Tanveer TA, Tehseen M, Khan M, Khan FA, Anwar S. Static malware detection and attribution in android byte-code through an end-to-end deep system. *Future generation Comput Syst* (2020) 102:112–26. doi:10.1016/j.future.2019.07.070

3. Nobakht M, Javidan R, Pourebrahimi A. DEMD-IoT: a deep ensemble model for IoT malware detection using CNNs and network traffic. *Evolving Syst* (2023) 14(3): 461–77. doi:10.1007/s12530-022-09471-z

4. Imamverdiyev Y, Abdullayeva F. Deep learning method for denial of service attack detection based on restricted Boltzmann machine. *Big data* (2018) 6(2):159–69. doi:10.1089/big.2018.0023

5. Eckhart M, Ekelhart A. Digital twins for cyber-physical systems security: state of the art and outlook. In: Biffl S, Eckhart M, Lüder A, Weippl E, editors. *Security and quality in cyber-physical systems engineering*. Cham: Springer International Publishing (2019). p. 383–412.

6. Souri A, Hosseini R. A state-of-the-art survey of malware detection approaches using data mining techniques. *Hum Cent Comput Inf Sci* (2018) 8:3. doi:10.1186/s13673-018-0125-x

7. Malik MI, Ibrahim A, Hannay P, Sikos LF. Developing resilient cyber-physical systems: a review of state-of-the-art malware detection approaches, gaps, and future directions. *Computers* (2023) 12:79. doi:10.3390/computers12040079

8. Razaulla S, Fachkha C, Markarian C, Gawanmeh A, Mansoor W, Fung BCM, et al. The age of ransomware: a survey on the evolution, taxonomy, and research directions. *IEEE Access* (2023) 11:40698–723. doi:10.1109/ACCESS.2023.3268535

9. Deldar F, Abadi M. Deep learning for zero-day malware detection and classification: a survey. *ACM Comput Surv* (2023) 56(2):1–37. doi:10.1145/3605775

10. Ali M, Hassen HR, Lones MA, Zantout H. An in-depth review of machine learning based Android malware detection. *Comput Security* (2022) 121:102833. doi:10.1016/j.cose.2022.102833

11. Tayyab U-e.-H, Khan FB, Durad MH, Khan A, Lee YS. A survey of the recent trends in deep learning based malware detection. *J Cybersecur Priv* (2022) 2:800–29. doi:10.3390/jcp2040041

12. Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: research developments, trends and challenges. *J Netw Comp Appl* (2020) 153:102526. doi:10.1016/j.jnca.2019.102526

13. Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ* (2021) 372:n71. doi:10.1136/bmj.n71

14. Subrahmanyam SSB, Goutham P, Ambati VKR, Bijitha CV, Nath HV. A hybrid method for analysis and detection of malicious executables in IoT network. *Comput Security* (2023) 132:103339. doi:10.1016/j.cose.2023.103339

15. Jain M, Andreopoulos W, Stamp M. Convolutional neural networks and extreme learning machines for malware classification. *J Comp Virol Hacking Tech* (2020) 16: 229–44. doi:10.1007/s11416-020-00354-y

16. GulatasKilinc HH, Zaim AH, Aydin MA. Malware threat on edge/fog computing environments from Internet of Things devices perspective. *IEEE Access* (2023) 11: 33584–606. doi:10.1109/ACCESS.2023.3262614

17. Zhang N, Xue J, Ma Y, Zhang R, Liang T, Tan YA. Hybrid sequence-based Android malware detection using natural language processing. *Int J Intell Syst* (2021) 36(10):5770–84. doi:10.1002/int.22529

18. Chen X. Power system malware detection based on deep belief network classifier. In: 2022 6th International Conference on Green Energy and Applications (ICGEA); 4th to 6th March 2022; Singapore (2022). p. 245–9.

19. He N, Wang T, Chen P, Yan H, Jin Z. An android malware detection method based on deep autoencoder. In: Proceedings of the Proceedings of the 2018 artificial intelligence and cloud computing conference; July 2 2018 to July 7 2018; San Francisco, CA, USA (2018). p. 88–93.

20. Reilly C, O Shaughnessy S, Thorpe C. Robustness of image-based malware classification models trained with generative adversarial networks. In: Proceedings of the 2023 European Interdisciplinary Cybersecurity Conference; June 14 - 15, 2023; Stavanger, Norway (2023). p. 92–9.

21. Shu L, Dong S, Su H, Huang J. Android malware detection methods based on convolutional neural network: a survey. *IEEE Trans Emerging Top Comput Intelligence* (2023) 7:1330–50. doi:10.1109/tetci.2023.3281833

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

23. Almaleh A, Almushabb R, Ogran R. Malware API calls detection using hybrid logistic regression and RNN model. *Appl Sci* (2023) 13(9):5439. doi:10.3390/app13095439

24. Rezvy S, Petridis M, Lasebae A, Zebin T. Intrusion detection and classification with autoencoded deep neural network. In: Proceedings of the Innovative Security Solutions for Information Technology and Communications: 11th International Conference, SecITC 2018; November 8–9, 2018; Bucharest, Romania (2019). p. 142–56.

25. D'Angelo G, Ficco M, Palmieri F. Malware detection in mobile environments based on Autoencoders and API-images. *Comput.* (2020) 137:26–33. doi:10.1016/j.jpdc.2019.11.001

26. Alotaibi A. Identifying malicious software using deep residual long-short term memory. *IEEE Access* (2019) 7:163128–37. doi:10.1109/ACCESS.2019.2951751

27. Liu J, Feng Y, Liu X, Zhao J, Liu Q. MRm-DLDet: a memory-resident malware detection framework based on memory forensics and deep neural network. *Cybersecurity* (2023) 6(1):21. doi:10.1186/s42400-023-00157-w

28. Saxe J, Berlin K. Deep neural network based malware detection using two dimensional binary program features. In: Proceedings of the 2015 10th International Conference on Malicious and Unwanted Software (MALWARE); Oct. 20 2015 to Oct. 22 2015; Fajardo, PR, USA (2015). p. 11–20.

29. Li D, Wang Z, Xue Y. Deepdetector: android malware detection using deep neural network. In: Proceedings of the 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE); IEEE; 22-23 June 2018; Paris, France (2018). p. 184–8.

30. Mercaldo F, Santone A. Deep learning for image-based mobile malware detection. *J Comp Virol Hacking Tech* (2020) 16(2):157–71. doi:10.1007/s11416-019-00346-7

31. Alqahtani A, Azzony S, Alsharafi L, Alaseri M. Web-based malware detection system using convolutional neural network. *Digital* (2023) 3(3):273–85. doi:10.3390/digital3030017

32. Chaganti R, Ravi V, Pham TD. Image-based malware representation approach with EfficientNet convolutional neural networks for effective malware classification. *J Inf Security Appl* (2022) 69:103306. doi:10.1016/j.jisa.2022.103306

33. Sl SD, Jaidhar CD. Windows malware detector using convolutional neural network based on visualization images. *IEEE Trans Emerging Top Comput* (2019) 9(2):1057–69. doi:10.1109/TETC.2019.2910086

34. Kim JY, Cho SB. Obfuscated malware detection using deep generative model based on global/local features. *Comput Security* (2022) 112:102501. doi:10.1016/j.cose.2021.102501

35. Yang J, Li T, Liang G, He W, Zhao Y. A simple recurrent unit model based intrusion detection system with DCGAN. *IEEE Access* (2019) 7:83286–96. doi:10.1109/access.2019.2922692

36. Won DO, Jang YN, Lee SW. PlausMal-GAN: plausible malware training based on generative adversarial networks for analogous zero-day malware detection. *IEEE Trans Emerging Top Comput* (2022) 11(1):82–94. doi:10.1109/tetc.2022.3170544

37. Cui Z, Zhao Y, Cao Y, Cai X, Zhang W, Chen J. Malicious code detection under 5G HetNets based on a multi-objective RBM model. *IEEE Netw* (2021) 35(2):82–7. doi:10.1109/mnet.011.2000331

38. Liu Z, Wang R, Japkowicz N, Tang D, Zhang W, Zhao J. Research on unsupervised feature learning for android malware detection based on restricted Boltzmann machines. *Future Generation Comp Syst* (2021) 120:91–108. doi:10.1016/j.future.2021.02.015

39. Jayashree R. Enhanced classification using restricted Boltzmann machine method in deep learning for COVID-19. *Understanding COVID-19: role Comput intelligence* (2022) 425–46. doi:10.1007/978-3-030-74761-9_19

40. Pandey S, Kumar N, Handa A, Shukla SK. Evading malware classifiers using RL agent with action-mask. *Int J Inf Security* (2023) 22(6):1743–63. doi:10.1007/s10207-023-00715-w

41. Kim S, Yoon S, Lim H. Deep reinforcement learning-based traffic sampling for multiple traffic analyzers on software-defined networks. *IEEE Access* (2021) 9:47815–27. doi:10.1109/access.2021.3068459

42. Jahromi AN, Hashemi S, Dehghantanha A, Choo KKR, Karimipour H, Newton DE, et al. An improved two-hidden-layer extreme learning machine for malware hunting. *Comput Security* (2020) 89:101655. doi:10.1016/j.cose.2019.101655

43. Aldehim G, Arasi MA, Khalid M, Aljameel SS, Marzouk R, Mohsen H, et al. Gauss-mapping black Widow optimization with deep extreme learning machine for android malware classification model. *IEEE Access* (2023) 11:87062–70. doi:10.1109/access.2023.3285289

44. Roy KS, Ahmed T, Udas PB, Karim ME, Majumdar S. MalHyStack: a hybrid stacked ensemble learning framework with feature engineering schemes for obfuscated malware analysis. *Intell Syst Appl* (2023) 20:200283. doi:10.1016/j.iswa.2023.200283

45. He Y, Kang X, Yan Q, Li E. ResNeXt+: attention mechanisms based on ResNeXt for malware detection and classification. *IEEE Trans Inf Forensics Security* (2023) 19:1142–55. doi:10.1109/tifs.2023.3328431

46. Choi S, Bae J, Lee C, Kim Y, Kim J. Attention-based automated feature extraction for malware analysis. *Sensors* (2020) 20(10):2893. doi:10.3390/s20102893

47. Agrawal R, Stokes JW, Selvaraj K, Marinescu M. Attention in recurrent neural networks for ransomware detection. In: ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP); May 12-17, 2019; Brighton, UK (2019). p. 3222–6.

48. Alkahtani H, Aldhyani THH. Artificial intelligence algorithms for malware detection in android-operated mobile devices. *Sensors* (2022) 22:2268. doi:10.3390/s22062268

49. Krzysztoń M, Bok B, Lew M, Sikora A. Lightweight on-device detection of android malware based on the koodous platform and machine learning. *Sensors* (2022) 22:6562. doi:10.3390/s22176562

50. Lu K, Cheng J, Yan A. Malware detection based on the feature selection of a correlation information decision matrix. *Mathematics* (2023) 11:961. doi:10.3390/math11040961

51. Lee J, Jang H, Ha S, Yoon Y. Android malware detection using machine learning with feature selection based on the genetic algorithm. *Mathematics* (2021) 9:2813. doi:10.3390/math9212813

52. Cañadas AM, Mendez OM, Vega JDC. Algebraic structures induced by the insertion and detection of malware. *Computation* (2023) 11:140. doi:10.3390/computation11070140

53. Singh AK, Taterh S, Mitra U. An efficient tactic for analysis and evaluation of malware dump file using the volatility tool. *SN COMPUT SCI* (2023) 4:457. doi:10.1007/s42979-023-01844-8

54. Amira A, Derhab A, Karbab EB, Omar N. A survey of malware analysis using community detection algorithms. *ACM Comput Surv* (2023) 56(2):1–29. doi:10.1145/3610223

55. Pereberina A, Kostyushko A, Tormasov A. An algorithm for scheduling of threads for system and application code split approach in dynamic malware analysis. *J Comput Virol Hack Tech* (2023) 19:459–68. doi:10.1007/s11416-023-00473-2

56. Hashida Haidros Rahima Manzil S. Detection approaches for android malware: taxonomy and review analysis. *Expert Syst Appl* (2024) 238(Part F):122255. doi:10.1016/j.eswa.2023.122255

57. Kara I. Fileless malware threats: recent advances, analysis approach through memory forensics and research challenges. *Expert Syst Appl* (2023) 214:119133. doi:10.1016/j.eswa.2022.119133

58. Celdrán AH, Sánchez PMS, Castillo MA, Bovet G, Pérez GM, Stiller B. Intelligent and behavioral-based detection of malware in IoT spectrum sensors. *Int J Inf Secur* (2023) 22:541–61. doi:10.1007/s10207-022-00602-w

59. Bhat P, Behal S, Dutta K. A system call-based android malware detection approach with homogeneous and heterogeneous ensemble machine learning. *Comput Security* (2023) 130:103277. doi:10.1016/j.cose.2023.103277

60. Sun N, Ding M, Jiang J, Xu W, Mo X, Tai Y, et al. Cyber threat intelligence mining for proactive cybersecurity defense: a survey and new perspectives. *IEEE Commun Surv Tutorials* (2023) 25(3):1748–74. doi:10.1109/COMST.2023.3273282

61. Turner A, McCombie S, Uhlmann A. Ransomware-bitcoin threat intelligence sharing using structured threat information expression. *IEEE Security and Privacy* (2023) 21(03):47–57. doi:10.1109/MSEC.2022.3166282

62. Sai Charan PV, Ratnakaram G, Chunduri H, Mohan Anand P, Kumar Shukla S. DKaaS: DARK-KERNEL as a service for active cyber threat intelligence. *Comput Security* (2023) 132:103329. doi:10.1016/j.cose.2023.103329

63. Lin P-C, Hsu W-H, Lin Y-D, Hwang R-H, Wu H-K, Lai Y-C, et al. Correlation of cyber threat intelligence with sightings for intelligence assessment and augmentation. *Computer Networks* (2023) 228:109736. doi:10.1016/j.comnet.2023.109736

64. Sajid MSI, Wei J, Al-Shaer E, Qi D, Abdeen B, Khan L. SymbSODA: configurable and verifiable orchestration automation for active malware deception. *ACM Trans Priv Secur* (2023) 26(4):1–36. doi:10.1145/3624568

65. El-Kosairy A, Abdelbaki N. Deception as a service: intrusion and ransomware detection system for cloud computing (IRDS4C). *Adv Comp Int* (2023) 3:9. doi:10.1007/s43674-023-00056-0

66. Ganfure GO, Wu C -F, Chang Y -H, Shih W -K. RTrap: trapping and containing ransomware with machine learning. *IEEE Trans Inf Forensics Security* (2023) 18:1433–48. doi:10.1109/TIFS.2023.3240025

67. Liu J, Feng Y, Liu X, Zhao J, Liu Q. MRm-DLDet: a memory-resident malware detection framework based on memory forensics and deep neural network. *Cybersecurity* (2023) 6:21. doi:10.1186/s42400-023-00157-w

68. Daghmehchi Firoozjaei M, Samet S, Ghorbani AA. Parent process termination: an adversarial technique for persistent malware. *J Cyber Security Tech* (2023) 1–26. doi:10.1080/23742917.2023.2246229

69. Naeem H, Dong S, Falana OJ, Ullah F. Development of a deep stacked ensemble with process based volatile memory forensics for platform independent malware detection and classification. *Expert Syst Appl* (2023) 223:119952. doi:10.1016/j.eswa.2023.119952

70. Chen T, Zeng H, Lv M, Zhu T. CTIMD: cyber threat intelligence enhanced malware detection using API call sequences with parameters. *Comput Security* (2024) 136:103518. doi:10.1016/j.cose.2023.103518

71. Ilca LF, Lucian OP, Balan TC. Enhancing cyber-resilience for small and medium-sized organizations with prescriptive malware analysis, detection and response. *Sensors* (2023) 23:6757. doi:10.3390/s23156757

72. Geng JX, Wang J, Fang Z, Zhou Y, Wu D, Ge W. A Survey of strategy-driven evasion methods for PE malware: transformation, concealment, and attack. *Comput Security* (2023) 137:103595. doi:10.1016/j.cose.2023.103595

73. Ilca LF, Lucian OP, Balan TC. Enhancing cyber-resilience for small and medium-sized organizations with prescriptive malware analysis, detection and response. *Sensors* (2023) 23(15):6757. doi:10.3390/s23156757

74. Vasani V, Bairwa AK, Joshi S, Pljonkin A, Kaur M, Amoon M. Comprehensive analysis of advanced techniques and vital tools for detecting malware intrusion. *Electronics* (2023) 12(20):4299. doi:10.3390/electronics12204299

75. Singh A, Ikuesan RA, Venter H. MalFe—malware feature engineering generation platform. *Computers* (2023) 12(10):201. doi:10.3390/computers12100201

76. Zhang S, Wu J, Zhang M, Yang W. Dynamic malware analysis based on API sequence semantic fusion. *Appl Sci* (2023) 13(11):6526. doi:10.3390/app13116526

77. Taher F, AlFandi O, Al-kfairy M, Al Hamadi H, Alrabaee S. DroidDetectMW: a hybrid intelligent model for android malware detection. *Appl Sci* (2023) 13(13):7720. doi:10.3390/app13137720

78. Akhtar MS, Feng T. Evaluation of machine learning algorithms for malware detection. *Sensors* (2023) 23(2):946. doi:10.3390/s23020946

79. Taher F, Al Fandi O, Al Kfairy M, Al Hamadi H, Alrabaee S. A proposed artificial intelligence model for android-malware detection. *Informatics* (2023) 10(3):67. doi:10.3390/informatics10030067

80. Alhashmi AA, Darem AA, Alashjaee AM, Alanazi SM, Alkhaldi TM, Ebad SA, et al. Similarity-based hybrid malware detection model using API calls. *Mathematics* (2023) 11(13):2944. doi:10.3390/math11132944

81. Herrera-Silva JA, Hernández-Álvarez M. Dynamic feature dataset for ransomware detection using machine learning algorithms. *Sensors* (2023) 23(3):1053. doi:10.3390/s23031053

82. Lockett A, Chalkias I, Yucel C, Henriksen-Bulmer J, Katos V. Investigating IPTV malware in the wild. *Future Internet* (2023) 15(10):325. doi:10.3390/fi15100325

83. Nachaat Mohamed. Current trends in AI and ML for cybersecurity: a state-of-the-art survey. *Cogent Engineering* (2023) 10:2. doi:10.1080/23311916.2023.2272358

84. Sun H, Shu H, Kang F, Guang Y. ModDiff: modularity similarity-based malware homologation detection. *Electronics* (2023) 12(10):2258. doi:10.3390/electronics12102258

85. Fedorchenko E, Novikova E, Fedorchenko A, Verevkin S. An analytical review of the source code models for exploit analysis. *Information* (2023) 14(9):497. doi:10.3390/info14090497

86. Buriro A, Buriro AB, Ahmad T, Buriro S, Ullah S. MalwD&C: a quick and accurate machine learning-based approach for malware detection and categorization. *Appl Sci* (2023) 13(4):2508. doi:10.3390/app13042508

87. Djenna A, Bouridane A, Rubab S, Marou IM. Artificial intelligence-based malware detection, analysis, and mitigation. *Symmetry* (2023) 15(3):677. doi:10.3390/sym15030677

88. Cha HJ, Yang HK, Song YJ, Kang AR. Intelligent anomaly detection system through malware image augmentation in IIoT environment based on digital twin. *Appl Sci* (2023) 13(18):10196. doi:10.3390/app131810196

89. Babbar H, Rani S, Sah DK, AlQahtani SA, Kashif Bashir A. Detection of android malware in the Internet of Things through the K-nearest neighbor algorithm. *Sensors* (2023) 23(16):7256. doi:10.3390/s23167256

90. Gazzan M, Sheldon FT. Opportunities for early detection and prediction of ransomware attacks against industrial control systems. *Future Internet* (2023) 15(4):144. doi:10.3390/fi15040144

91. Khalid O, Ullah S, Ahmad T, Saeed S, Alabbad DA, Aslam M, et al. An insight into the machine-learning-based fileless malware detection. *Sensors* (2023) 23(2):612. doi:10.3390/s23020612

92. Ba'abbad I, Batarfi O. Proactive ransomware detection using extremely fast decision tree (efdt) algorithm: a case study. *Computers* (2023) 12(6):121. doi:10.3390/computers12060121

93. Zhang S, Hu C, Wang L, Mihaljevic MJ, Xu S, Lan T. A malware detection approach based on deep learning and memory forensics. *Symmetry* (2023) 15(3):758. doi:10.3390/sym15030758

94. Saridou B, Moulas I, Shiaeles S, Papadopoulos B. Image-based malware detection using α-cuts and binary visualisation. *Appl Sci* (2023) 13(7):4624. doi:10.3390/app13074624

95. Alabrah A. A novel neural network architecture using automated correlated feature layer to detect android malware applications. *Mathematics* (2023) 11(20):4242. doi:10.3390/math11204242

96. Lu J, Ren X, Zhang J, Wang T. CPL-net: a malware detection network based on parallel CNN and LSTM feature fusion. *Electronics* (2023) 12(19):4025. doi:10.3390/electronics12194025

97. Aboaoja FA, Zainal A, Ali AM, Ghaleb FA, Alsolami FJ, Rassam MA. Dynamic extraction of initial behavior for evasive malware detection. *Mathematics* (2023) 11(2):416. doi:10.3390/math11020416

98. Deng L, Wen H, Xin M, Li H, Pan Z, Sun L. Enimanal: augmented cross-architecture IoT malware analysis using graph neural networks. *Comput Security* (2023) 132:103323. doi:10.1016/j.cose.2023.103323

99. Kumar EP, Priyanka S. A comprehensive survey on hardware-assisted malware analysis and primitive techniques. *Comp Networks* (2023) 235:109967. doi:10.1016/j.comnet.2023.109967

100. Vashishtha LK, Chatterjee K, Rout SS. An Ensemble approach for advance malware memory analysis using Image classification techniques. *J Inf Security Appl* (2023) 77:103561. doi:10.1016/j.jisa.2023.103561

101. Lv M, Zeng H, Chen T, Zhu T. CTIMD: cyber threat intelligence enhanced malware detection using API call sequences with parameters. *Comput Security* (2023) 136:103518. doi:10.1016/j.cose.2023.103518

102. Khan SH, Alahmadi TJ, Ullah W, Iqbal J, Rahim A, Alkahtani HK, et al. A new deep boosted CNN and ensemble learning based IoT malware detection. *Comput Security* (2023) 133:103385. doi:10.1016/j.cose.2023.103385

103. Kara I. Fileless malware threats: recent advances, analysis approach through memory forensics and research challenges. *Expert Syst Appl* (2023) 214:119133. doi:10.1016/j.eswa.2022.119133

104. Liu C, Lu J, Feng W, Du E, Di L, Song Z. MOBIPCR: efficient, accurate, and strict ML-based mobile malware detection. *Future Generation Comp Syst* (2023) 144:140–50. doi:10.1016/j.future.2023.02.014

105. Kumar S, Panda K. SDIF-CNN: stacking deep image features using fine-tuned convolution neural network models for real-world malware detection and classification. *Appl Soft Comput* (2023) 146:110676. doi:10.1016/j.asoc.2023.110676

106. Zhu H, Wei H, Wang L, Xu Z, Sheng VS. An effective end-to-end android malware detection method. *Expert Syst Appl* (2023) 218:119593. doi:10.1016/j.eswa.2023.119593

107. Kishore P, Barisal SK, Mohapatra DP, Mall R. An efficient two-stage pipeline model with filtering algorithm for mislabeled malware detection. *Comput Security* (2023) 135:103499. doi:10.1016/j.cose.2023.103499

108. Bhat P, Behal S, Dutta K. A system call-based android malware detection approach with homogeneous and heterogeneous ensemble machine learning. *Comput Security* (2023) 130:103277. doi:10.1016/j.cose.2023.103277

109. Banik A, Singh JP. Android malware detection by correlated real permission couples using FP growth algorithm and neural networks. *IEEE Access* (2023) 11:124996–5010. doi:10.1109/access.2023.3323845

110. Perez AJ, Zeadally S, Tan DK. Detecting mobile malware associated with global pandemics. *IEEE Pervasive Comput* (2023) 22:45–54. doi:10.1109/mprv.2023.3321218

111. Chen YH, Lin SC, Huang SC, Lei CL, Huang CY. Guided malware sample analysis based on graph neural networks. *IEEE Trans Inf Forensics Security* (2023) 18:4128–43. doi:10.1109/tifs.2023.3283913

112. Lee H, Kim S, Baek D, Kim D, Hwang D. Robust IoT malware detection and classification using opcode category features on machine learning. *IEEE Access* (2023) 11:18855–67. doi:10.1109/access.2023.3247344

113. Al-Andoli MN, Sim KS, Tan SC, Goh PY, Lim CP. An ensemble-based parallel deep learning classifier with PSO-BP optimization for malware detection. *IEEE Access* (2023) 11:76330–46. doi:10.1109/access.2023.3296789

114. Manthena H, Kimmel JC, Abdelsalam M, Gupta M. Analyzing and explaining black-box models for online malware detection. *IEEE Access* (2023) 11:25237–52. doi:10.1109/access.2023.3255176

115. Abdelwahed MF, Kamal MM, Sayed SG. Detecting malware activities with MalpMiner: a dynamic analysis approach. *IEEE Access* (2023) 11:84772–84. doi:10.1109/access.2023.3266562

116. Lee S, Lee S, Park J, Kim K, Lee K. Hiding in the crowd: ransomware protection by adopting camouflage and hiding strategy with the link file. *IEEE Access* (2023) 11:92693–704. doi:10.1109/access.2023.3309879

117. Shin K, Lee Y, Lim J, Kang H, Lee S. System API vectorization for malware detection. *IEEE Access* (2023) 11:53788–805. doi:10.1109/access.2023.3276902

118. Niu W, Wang Y, Liu X, Yan R, Li X, Zhang X. GCDroid: android malware detection based on graph compression with reachability relationship extraction for IoT devices. *IEEE Internet Things J* (2023) 10:11343–56. doi:10.1109/jiot.2023.3241697

119. Yu Z, Li S, Bai Y, Han W, Wu X, Tian Z. REMSF: a robust ensemble model of malware detection based on semantic feature fusion. *IEEE Internet Things J* (2023) 10:16134–43. doi:10.1109/jiot.2023.3267337

120. Odat E, Yaseen QM. A novel machine learning approach for android malware detection based on the Co-existence of features. *IEEE Access* (2023) 11:15471–84. doi:10.1109/access.2023.3244656

121. Thummapudi K, Lama P, Boppana RV. Detection of ransomware attacks using processor and disk usage data. *IEEE Access* (2023) 11:51395–407. doi:10.1109/access.2023.3279819

122. Kim C, Chang SY, Kim J, Lee D, Kim J. Automated, reliable zero-day malware detection based on autoencoding architecture. *IEEE Trans Netw Serv Manag* (2023) 20:3900–14. doi:10.1109/tnsm.2023.3251282

123. Jin B, Choi J, Hong JB, Kim H. On the effectiveness of perturbations in generating evasive malware variants. *IEEE Access* (2023) 11:31062–74. doi:10.1109/access.2023.3262265

124. Kural OE, Kiliç E, Aksaç C. Apk2Audio4AndMal: audio based malware family detection framework. *IEEE Access* (2023) 11:27527–35. doi:10.1109/access.2023.3258377

125. Yonamine S, Taenaka Y, Kadobayashi Y, Miyamoto D. Design and implementation of a sandbox for facilitating and automating IoT malware analysis with techniques to elicit malicious behavior: case studies of functionalities for dissecting IoT malware. *J Comp Virol Hacking Tech* (2023) 19(2):149–63. doi:10.1007/s11416-023-00478-x

126. Masid AG, Higuera JB, Higuera JRB, Montalvo JAS. Application of the SAMA methodology to Ryuk malware. *J Comp Virol Hacking Tech* (2023) 19(2):165–98. doi:10.1007/s11416-022-00434-1

127. Singh AK, Taterh S, Mitra U. An efficient tactic for analysis and evaluation of malware dump file using the volatility tool. *SN Comp Sci* (2023) 4(5):457. doi:10.1007/s42979-023-01844-8

128. de Lima SM, Souza DM, Pinheiro RP, Silva SH, Lopes PG, de Lima RD, et al. Next-generation antivirus for JavaScript malware detection based on dynamic features. *Knowledge Inf Syst* (2023) 66:1337–70. doi:10.1007/s10115-023-01978-4

129. Sharma A, Gupta BB, Singh AK, Saraswat VK. A novel approach for detection of APT malware using multi-dimensional hybrid Bayesian belief network. *Int J Inf Security* (2023) 22(1):119–35. doi:10.1007/s10207-022-00631-5

130. Pereberina A, Kostyushko A, Tormasov A. An algorithm for scheduling of threads for system and application code split approach in dynamic malware analysis. *J Comp Virol Hacking Tech* (2023) 19:459–68. doi:10.1007/s11416-023-00473-2

131. Seyfari Y, Meimandi A. A new approach to android malware detection using fuzzy logic-based simulated annealing and feature selection. *Multimedia Tools Appl* (2023) 83:10525–49. doi:10.1007/s11042-023-16035-z

132. Alzubi OA, Alzubi JA, Alzubi TM, Singh A. Quantum Mayfly optimization with encoder-decoder driven LSTM networks for malware detection and classification model. *Mobile Networks Appl* (2023) 28:795–807. doi:10.1007/s11036-023-02105-x

133. Ullah F, Ullah S, Srivastava G, Lin JCW, Zhao Y. NMal-Droid: network-based android malware detection system using transfer learning and CNN-BiGRU ensemble. *Wireless Networks* (2023) 1–22. doi:10.1007/s11276-023-03414-5

134. Deng X, Cen M, Jiang M, Lu M. Ransomware early detection using deep reinforcement learning on portable executable header. *Cluster Comput* (2023) 1–15. doi:10.1007/s10586-023-04043-5

135. Balikcioglu PG, Sirlanci M, A. Kucuk O, Ulukapi B, Turkmen RK, Acarturk C. Malicious code detection in android: the role of sequence characteristics and disassembling methods. *Int J Inf Security* (2023) 22(1):107–18. doi:10.1007/s10207-022-00626-2

136. Gao C, Cai M, Yin S, Huang G, Li H, Yuan W, et al. Obfuscation-resilient android malware analysis based on complementary features. *IEEE Trans Inf Forensics Security* (2023) 18:5056–68. doi:10.1109/TIFS.2023.3302509

137. Gopinath M, Sethuraman SC. A comprehensive survey on deep learning based malware detection techniques. *Comp Sci Rev* (2023) 47:100529. doi:10.1016/j.cosrev.2022.100529

138. Zhu H-juan, Gu W, Wang L-min, Xu Z-cheng, Sheng VS. Android malware detection based on multi-head squeeze-and-excitation residual network. *Expert Syst Appl* (2023) 212:118705. doi:10.1016/j.eswa.2022.118705

139. Kumar R, Zhang X, Khan RU, Sharif A. Research on data mining of permission-induced risk for android IoT devices. *Appl Sci* (2019) 9:277. doi:10.3390/app9020277

140. Mustafa Majid A-A, Alshaibi AJ, Kostyuchenko E, Shelupanov A. A review of artificial intelligence based malware detection using deep learning. *Mater Today Proc* (2023) 80(3):2678–83. doi:10.1016/j.matpr.2021.07.012