International Journal on Advanced Science Engineering Information Technology

Vol.14 (2024) No. 3
ISSN: 2088-5334

# A Comprehensive Review of Machine Learning Approaches for Detecting Malicious Software

Liu Yuanming [a,*], Rodziah Latih [a]

[a] *Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Selangor, 43600, Malaysia*
*Corresponding author: [*]p103623@siswa.ukm.edu.my*

***Abstract*— With the continuous development of technology, the types of malware and their variants continue to increase, which has become an enormous challenge to network security. These malware use a variety of technical means to deceive or evade traditional detection methods, making traditional signature-based rule-based malware identification methods no longer applicable. Many machine algorithms have attracted widespread academic attention as powerful malware detection and classification methods in recent years. After an in-depth study of rich literature and a comprehensive survey of the latest scientific research results, feature extraction is used as the basis for classification. By extracting meaningful features from malware samples, such as behavioral patterns, code structures, and file attributes, researchers can discern unique characteristics that distinguish malicious software from benign ones. This process is the foundation for developing effective detection models and understanding the underlying mechanisms of malware behavior. We divide feature engineering and learning-based methods into two categories for investigation. Feature engineering involves selecting and extracting relevant features from raw data, while learning-based methods leverage machine learning algorithms to analyze and classify malware based on these features. Supervised, unsupervised, and deep learning techniques have shown promise in accurately detecting and classifying malware, even in the face of evolving threats. On this basis, we further look into the current problems and challenges malware identification research faces.**

***Keywords*—Malware; machine learning; deep learning; malware detection; malware classification.**

*Manuscript received 27 Mar. 2024; revised 25 Apr. 2024; accepted 9 May 2024. Date of publication 30 Jun. 2024.*


## I. INTRODUCTION

Malicious software runs on a user's computer or other terminal. It uses operating system or application loopholes to infringe on the user's legitimate rights and interests without their knowledge or approval. According to different characteristics and hazards, malware can be divided into adware, ransomware, backdoor, Trojan horse, and other categories at present, as shown in Table I. Malware poses serious safety threats to network users, businesses, industrial facilities, and web and information tools, and the number of new malwares has been increasing in recent years.

The continuous surge in malware threats has fueled extensive research in academia and industry, focusing on detecting and classifying malware families. Initially, signature matching dominated the industry's approach to malware identification [1], [2], [3], boasting high accuracy for known malware but falling short against unknown threats. Subsequently, heuristic-based methods emerged, defining static or dynamic rules to identify malware based on its characteristics [4], [5], [6]. While capable of detecting some unknown malicious codes, these methods suffer from a high false alarm rate. As malicious software evolved with advanced concealment techniques, traditional identification methods faced diminishing effectiveness.

TABLE I
MALWARE TYPE

| Malware type | Behavior |
|---|---|
| Adware | Create revenue for developers by increasing the exposure of commercial advertisements or collecting user information without user authorization. |
| Ransom ware | Encryption or file locking within the system limits user access privileges, compelling the target to pay a ransom to remove these restrictions. |
| Back door | They bypass the system's safety protection mechanism and install it on the system, which is convenient for attackers to access. |
| Trojan horse | A camouflaged virus with automatic renaming, self-hiding, and self-replicating. It tends to be a |

| Malware type | Behavior |
|---|---|
| | legitimate program to entice users to install or embed backdoor functions, allowing attackers to bypass security programs to enter the system, collect necessary information, and control it. |
| Spyware | Conducting covert espionage actions without the user's consent to gather confidential user information. |
| Virus | Exploit vulnerabilities within the operating system to harm the user system's functions and data, rendering it inoperable and propagating throughout the device. |
| Worm | Similar to a virus, it can cause huge damage to the system, but it can self-replicate and spread through the network. |
| Rootkits | The driver-level malicious program loaded into the system grain can hide the processes of other agendas and retain root privileges. Typically, it is employed with malware like Trojan horses and backdoors to obscure any traces of malicious activity. |
| Downloader | Various other types of malicious software are automatically installed without user permission. |
| Fileless malware [7] | It is highly concealable, achieves stealth by staying in memory for a long time, and prolongs the discovery time as much as possible. |

Machine learning-based malware identification has gained prominence in recent years, as depicted in Figure 1. Crucial to these algorithms is a robust feature representation. Early machine learning technologies relied on artificial feature engineering, where feature representations were manually extracted, and models like Support Vector Machine (SVM) and Random Forest (RF) were trained for malware detection (upper part of Figure 1). However, the high cost of feature engineering and the growing complexity of malicious code functions posed significant challenges.

Deep learning technology ushered in a new era for malware identification, leveraging models such as recurrent neural networks and convolutional neural networks (lower part of Figure 1). Deep learning models automatically learn discriminative feature representations, enabling end-to-end malware detection.

This article provides a comprehensive review of malware detection and classification and briefly describes malware identification through feature engineering, including static feature analysis and dynamic feature analysis. Malware identification based on feature learning is introduced, mainly from three perspectives: Recurrent Neural Networks, Convolutional Neural Networks, and Graph Neural Networks. The objective is to offer readers a comprehensive understanding of machine learning-based malware identification and its key advancements, serving as a technical reference to propel further development in the field.

## II. Materials and Method

Malware identification based on feature engineering extracts meaningful features from samples to represent them through artificially designed processing flow and then uses a machine learning model to identify malware. This section summarizes the existing research work from the malware feature representation. The related work is listed in Table II.

### *A. Static feature analysis*

Static feature analysis refers to extracting features without running the software, achieved by analyzing the content of files contained within the software. For malicious software targeting the Windows platform, numerous studies extract static feature representations from Portable Executable (PE) files such as EXE and DLL. Popoiu [8] utilized API and other strings reflective of attacker intent and objectives from PE files as features, constructing a malicious software recognition model based on a Support Vector Machine (SVM).

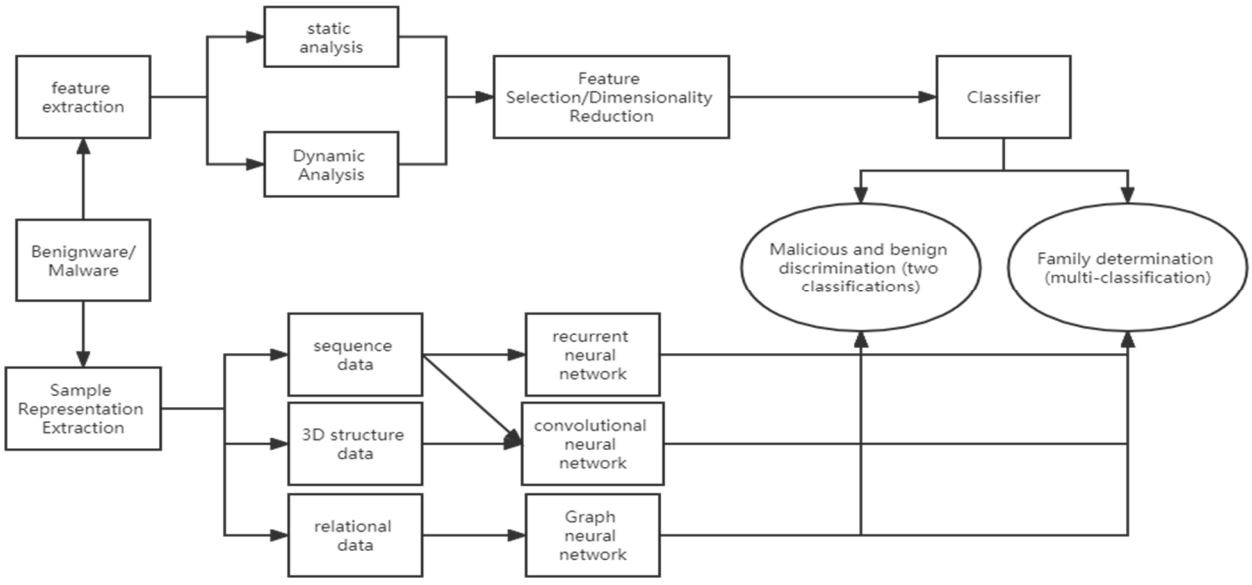


Fig. 1 Malware identification procedure

Shafiq et al. [9] extracted information from PE file headers, section tables, and sections, including dynamic link library references, symbol counts, linker versions, and import/export table sizes, as features. They built malicious software

recognition models using J48, Naive Bayes (NB), and other algorithms.

Some research focuses on extracting serialized static feature representations from PE files, using the frequency of n-grams as features. For instance, Tyagi et al. [10] selected the most frequent binary byte n-grams from PE files as features. Gülmez et al. [11] and McLaughlin et al. [12] used n-grams of assembly opcode sequences from PE files as features. Hoang et al. [13] compared the effectiveness of binary byte n-grams and assembly opcode n-grams in PE files, with experimental results indicating that opcode sequence n-grams are more effective.

The sequence above feature-based methods do not consider the actual control flow during program execution, i.e., the fundamental behavior of the software at runtime. Addressing this issue, McLaughlin et al. [12] transformed PE assembly files into execution trees, employing a tree traversal algorithm to generate opcode n-grams that encapsulate all conceivable execution paths.

For Android platform malware, Shao et al. [14] extracted permissions and API features from Android software manifest files and DEX disassembly files. They employed SVM and k-NN to build malware recognition models. Alazab et al. [15] further categorized API calls into fuzzy, risky, and destructive calls, evaluating the effectiveness of these features.

TABLE II
REVIEW OF MALWARE RECOGNITION RESEARCH BASED ON FEATURE ENGINEERING

| Literature | Operation object | Method | Data set | Accuracy (% ) |
|---|---|---|---|---|
| Huang et al. [16] | API calls and their parameters | FNN | 2.85 million malicious files and 3.65 million benign files provided by Microsoft analysts. | 99.64 |
| Shao et al. [14] | API calls + permissions | SVM & k-NN | 100 malware and 500 normal software. | 90.0 |
| Yang et al. [17] | Permission + Intent | RF | 1260 malicious software was obtained from the Android Malware [18], and 674 benign software was downloaded from Google's official and domestic third-party application markets. | 98.1 |
| Amenova et al. [19] | encryption + permissions, | CNN-LSTM | It covers more than 17,341 Android samples | 94.0 |
| Alazab et al. [15] | API calls + permissions | RF | 13,719 malicious applications from AndroZoo, Contagio, MalShare, VirusShare, and VirusTotal. | 94.3 |
| Purnama et al. [20] | API calls + n-gram | MLP | 245 malware and 245 benign samples were obtained from VX Heaven for internal review. | 98.5 |
| Su et al. [21] | Component + Permission + Intent etc. | DBN & SVM | 3986 Android malicious applications, 3986 Android benign applications, and 1515 unknown Android applications. | 97.5 |
| Gülmez et al. [11] | opcode sequence | SVM | 17,000 malicious programs were downloaded from the VxHeaven website, and 1,000 legitimate executable files were collected from computers. | 98.0 |
| McLaughlin et al. [12] | opcode sequence | DBN & SVM / k- NN / DT | 2550 malicious files, 2550 benign files, and 4000 unlabeled samples. | 96.8 |

Experiments indicated that the combination of destructive calls and risky calls yielded the best results. Yang et al. [17] extracted permission and intent features from the manifest file, proposing an improved random forest algorithm for malware identification. Amenova et al. [19] extracted six features: lock screen, encryption, permission, threat text, payment method, and network communication. They utilized the CNN-LSTM algorithm to detect ransomware.

### *B. Analysis of dynamic characteristics*

Dynamic feature analysis [22] extracts behavioral characteristics of malware at runtime by executing malware in a virtual machine or sandbox and monitoring its real-time behavior. For Windows platform malware, Huang et al. [16] utilized API calls, <API calls, parameters> combinations, and tri-grams of API call sequences as features. They trained a multi-task feedforward neural network for identifying malicious software. Balodi et al. [23] conducted a comprehensive feature extraction process, encompassing metrics such as the tally of accessed/deleted/modified files, registry keys, IP addresses, DNS queries, runtime API calls, and accessed links. This extensive feature set served as the foundation for constructing a sophisticated malware recognition model. However, some malware can detect sandbox environments and conceal their malicious behavior, leading to recognition model failures.

On the Android platform, Wu et al. [24] proposed a recognition model based on API clustering using the number of system API calls during runtime. Elalem et al. [25] designed 88 features capturing information in various aspects like runtime power, memory, network, touch screen, and keyboard. They used different classifiers, with J48 and NB performing better in experiments. Purnama et al. [20] integrated dynamic features such as API calls and actions/events with static features like permissions at runtime. They used a multi-layer perceptron (MLP) to build a recognition model. Singh et al. [26] enhanced the performance of the SVM-ML model through four different data preprocessing, transformation, outlier identification, filling and smoothing. Dynamic analysis can better identify malware's abnormal behavior compared to static analysis.

However, the feature extraction process is time-consuming, making it less suitable for time-sensitive detection tasks.

### C. Feature Selection and Dimensionality Reduction

To prevent the overfitting problem caused by high-dimensional features, many studies further perform feature selection or dimensionality reduction on the initial features. Among them, commonly used feature selection methods include information gain [27], [28] (IG), mutual information [14], etc. Commonly used feature dimensionality reduction methods include principal component analysis [9] (PCA), random mapping [29] etc. Huang et al. [16] successively used mutual information and random mapping to map the feature space from millions to thousands of dimensions. Shafiq et al. [9] used three different methods of redundant feature removal (RFR), PCA, and Haar wavelet transform (HWT) to reduce the dimensionality of the extracted features. Yang et al. [17] used the IG and ReliefF algorithms to select the extracted features, respectively. Experiments showed that the IG algorithm was more stable than the ReliefF as the number of features increased. Elalem et al. [25] compared Chi-square test [30], Fisher score [31] and IG three feature selection methods, and the experiments showed that none had an absolute advantage in all experimental settings.

TABLE III
OVERVIEW OF MALWARE RECOGNITION RESEARCH USING FEATURE LEARNING

| Literature | Operation object | Method | Data set | Accuracy (% ) |
|---|---|---|---|---|
| Kwan et al. [32] | opcode sequence | CNN | Including 10868 samples of 9 categories. | 98.7 |
| Priyanto et al. [33] | | Isolation Forest & LSTM | The data was collected from 2019 to 2020 from raw data from the land monitoring system.. | 95.72 |
| Patil et al. [34] | | | Uses 36 different malware types and families, with 9 malware variants from the Microsoft dataset. | 92.73 |
| Ke et al. [35] | bytecode | CNN | 2500 malicious apps from Android virus and 2500 benign apps collected from Xiaomi's official application store. | 98.7 |
| Migdady et al. [36] | | | Malming | 98.0 |
| Malani et al. [37] | | | A total of 138047 Android applications. | 98.02 |
| Alam et al. [38] | | | MalImg | 98.9 |
| Geremias et al. [39] | | | over 26 thousand Android apps | 98.4 |
| Kural et al. [40] | | | 24588 Android malware and 3000 benign applications | 94 |
| Gibert et al. [41] | | HWT & CNN | MMCC [42] | 98.96 |
| Rahul et al. [43] | | ANN & SVM | Benign | 93.0 |
| Kalash et al. [44] | bytecode | CNN | MalImg<br>MMCC | 98.52<br>99.97 |
| Yuan et al. [45] | bytecode | CNN | MMCC<br>4020 malwares from Drebin. | 99.26<br>97.36 |
| Kr cˇ ál et al. [46] | bytecode + handcrafted features | CNN & FNN | File size between 12 ~ 512kb. | 97.1 |
| Thosar et al. [47] | bytecode + API call + mnemonic sequence | CNN | 600000 labelled samples (50% labeled as malware and 50% benign) | 93.45 |
| Liu et al. [48] | API call sequence | Bi-LSTM | 13518 malicious samples and 7860 benign samples. | 97.85 |
| Lu et al. [49] | | Bi-Residual LSTM & RF | Malicious: 1430 malicious samples randomly extracted from VirusShare and VirusTotal;<br>Benign: 1352 normal samples. | 96.7 |
| Huang et al. [50] | | TCN | More than 6900 malicious PE files collected from CILPKU08 and Henchiri. | 98.60 |
| Tsunewaki et al. [51] | | LSTM | Mal-API-2019 [52] | 93.3 |
| Gui et al. [53] | | BGSA | Including 20000 samples of 5 categories. | 92.54 |
| Li et al. [54] | | GCN | There are 6686 malicious samples and 6938 benign samples. | 98.32 |

Some studies use Auto Encoder to achieve feature dimensionality reduction [55], but it is challenging to train encoders for tens of thousands of dimensional features. To solve this problem, the deep belief network (DBN) proposed by Chen et al. [56] can achieve a good dimensionality reduction effect through layer-by-layer pre-training and parameter fine-tuning. Su et al. [21] performed 0/1 encoding on features such as components, intents, and request permissions extracted from Android software, and used DBN for dimensionality reduction. After dimensionality reduction, the recognition accuracy rate reached 97.5%. McLaughlin et al. [12] extracted PE file opcode n-grams and used DBN for dimensionality reduction, improving performance.

## III. RESULT AND DISCUSSION

With the breakthrough of deep learning technology, malware identification using deep neural networks has become a new research trend. Since deep neural networks can learn discriminative features automatically, this paper presents this type of research as recognition methods based on feature learning. Different sample representation methods can be divided into recurrent neural network-based, convolutional

neural network-based, and graph neural network-based methods, as described in this section. Related models are listed in Table III.

### *A. Malware Identification based on Recurrent Neural Network*

Recurrent Neural Networks (RNN) excel at processing serialized data and find extensive applications in natural language processing and machine translation. In various studies, malware is first parsed into serialized forms like opcode and API call sequences through static or dynamic analysis. Following this, researchers construct models for malware recognition utilizing RNN.

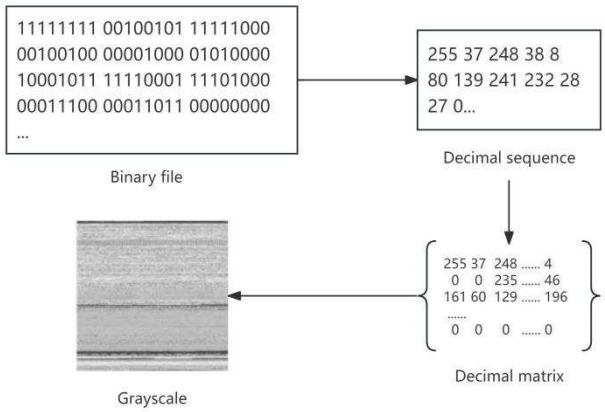


Fig. 2 Transformation process of gray-scale image

For instance, Tsunewaki et al. [51] dynamically extracted the API call sequence of software, utilized Embedding to obtain the tensor representation, and fed it into a Long Short-Term Memory network (LSTM) for malware family classification. Priyanto et al. [33] disassembled and extracted the opcode sequence, used a convolutional autoencoder to obtain a compressed vector representation, and inputted the vector into LSTM to calculate the malicious software's probability. Recognizing the variations in program implementation and execution processes among different malware variants, many researchers employ bidirectional RNNs to better capture dependencies from both forward and reverse directions to identify malicious sequence patterns. Gui et al. [53] utilized Bidirectional GRU with Self- Attention to classify software API call sequences, achieving significantly better results than CNN + Bidirectional LSTM. Liu et al. [48] employed BiLSTM to model API call sequences, comparing it with Gated Recurrent Unit (GRU), BiGRU, and LSTM, with BiLSTM performing the best in experiments. Lu et al. [49] selected key n-gram sequences of API calls via Information Gain (IG), proposed a bidirectional residual LSTM for further feature learning on the selected sequence, and ultimately used Random Forest (RF) for malware identification.

Despite the commendable performance of malware identification methods based on RNN, adversaries can obfuscate these models by introducing numerous non-critical sequence elements. Consequently, enhancing the resilience of the RNN recognition model against confusion caused by sequence elements remains an ongoing research challenge.

### *B. Malware Identification based on Convolutional Neural Network*

Convolutional Neural Networks (CNN) have demonstrated excellent performance in tasks involving spatial structure relationships, such as image processing and computer vision. Drawing inspiration from this, several studies adopt an approach where malware is transformed into data representations with spatial structure relationships, leveraging CNN for malware recognition [57], [58]. Migdady et al. [36] and Král et al. [46] proposed converting software binary files into hexadecimal sequences and employing 1-dimensional CNNs to construct malware recognition models. Gibert et al. [41] segmented the hexadecimal representation of malware into fixed-size blocks, calculated the entropy value of each block, used Haar Wavelet Transform (HWT) to obtain approximation and detail coefficients for the entropy sequence, and used these as inputs for CNN to build the recognition model. Huang et al. [50] utilized one-hot encoding to convert software API call sequences into digital representations, developing a malware recognition model with the Time Series Convolutional Network (TCN). The approach incorporated Cost-Sensitive cross-entropy loss to enhance model performance and facilitate the identification of malicious code fragments. Thosar et al. [47] introduced a multi-modal fusion recognition model HYDRA, utilizing CNN and Gradient Boosting to learn representations from assembly mnemonic sequences, byte sequences, and API calls. Representation fusion was achieved through a fully connected layer, leveraging the complementarity between different modes for enhanced performance.

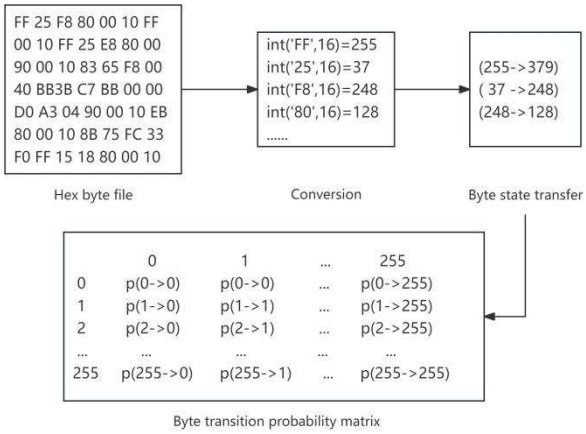


Fig. 3 Transformation process of byte transfer probability matrix

Some studies convert software into visual images and use methods and models in the field of images to build malware recognition models. Patil et al. [34] took every 8 bits in the software binary representation as a pixel, converted the file into a grayscale image with a channel number of 1 (the process is shown in Figure 2), and then used CNN for malware identification. Ke et al. [35] considered that the DEX file of Android software may contain a lot of noise, so only part of its data was converted into a grayscale image, and CNN was used for malware identification. Yuan et al. [45] regarded the integer value represented by the byte as a state, calculated the transition probability between states based on the byte sequence of the software, and then constructed a "Markov Image" (as shown in Figure 3) and input it into the

CNN model to identify malware. Kwan et al. [32] adopted a similar idea, taking the product of the transition probability of the software assembly operation code and its IG as a pixel to construct an image and further using methods such as histogram normalization, dilation, and erosion to enhance the image.

In general, deep convolutional networks can learn effective representations better than shallow network structures and have a more vital ability to fit data. A deeper network structure can deal with obfuscated samples more effectively for malware identification. However, as the depth increases and the network structure becomes more and more complex, it will cause the problem of gradient disappearance or gradient explosion during training so that the parameters cannot be effectively updated [59], [60], [61]. Sridhar et al. [62] proposed the ResNet model, which effectively alleviated the above problems through the residual structure so that the convolutional network structure could reach hundreds or even thousands of layers. However, due to the larger parameters scale, the deep convolutional network needs more data to train well. Given the limited scale of malware family annotation samples that can be obtained, many studies are based on transfer learning, first teaching a deep convolutional network model on large-scale image data and then transferring the knowledge learned by the model to the malware recognition task—for example, the IM CEC architecture proposed by Vasan et al. [63] first trains the VGG16 and ResNet50 models on the ImageNet dataset [64], and then transfer the parameters of the two models to the malware recognition model through transfer learning.

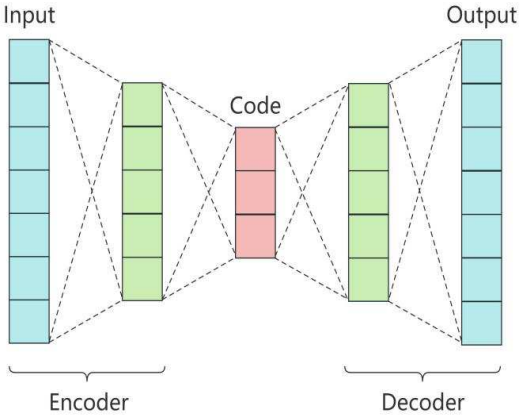


Fig. 4 Convolutional autoencoder

Training deep learning models on small datasets can easily lead to overfitting problems. Aiming at the training of malicious code models under minor sample conditions, Asaduzzaman et al. [65] introduced the tGAN model, where malware samples are transformed into images. Initially, a convolutional autoencoder is trained to capture the feature representation for each malware category (the process is shown in Figure 4). Subsequently, the decoder from the autoencoder is seamlessly transferred to the Generative Adversarial Network (GAN). The generator and discriminator components of the GAN are then employed for data augmentation and malware identification, respectively. Experiments show that the accuracy rate of the model is 90% under the condition of small samples.

Malware recognition models based on visualization and image processing technology can often achieve high accuracy. However, existing studies generally force-transform 1-dimensional sequences into 2-dimensional "images", so the second-dimensional structural relationship does not naturally exist in malware. Exploring the construction of a more rational visual representation to effectively leverage the strengths of the CNN model is a topic deserving further exploration. Furthermore, while recognition models based on architectures like VGG16 and ResNet50 can deliver favorable classification outcomes, the associated demands on training and inference time and the utilization of computing resources are often substantial. This limitation hinders the widespread application of such models.

### *C. Malware Identification based on Graph Neural Network*

Graph Neural Networks (GNN) incorporate graph analysis into deep learning frameworks, demonstrating proficiency in handling relational data. Numerous scholars are experimenting with representing software samples as graphs and subsequently employing GNN models to classify these graphs for effective malware identification. For example, Li et al. [54] revealed that the API call sequence dynamically extracted from the software sample and the Markov chain generated by the entire sample were represented as a directed cycle graph and a shared weight graph, respectively. The two were fused, and then a Graph Convolutional Network (GCN) was used for malware detection. Busch et al. [66] built a directed graph by monitoring the network traffic when the software was running and then used GNN to classify the software. The nodes in the graph correspond to the IP addresses in the network, and the edges represent the direction of network traffic and have the number of packets sent, packet Attributes such as the average length, the maximum and minimum value of the packet transmission interval, etc. To avoid the introduction of runtime overhead, Feng et al. [67] decompiled the Android application into a Smali file, extracted the function call relationship from it to construct a function call graph, used the semantic information, security level, and permissions of the function as node attributes, and used GNN Identify malware. Overall, there is still little research on applying GNN to malware identification, and the graph representation and representation learning of samples are still worthy of further discussion.

## IV. CONCLUSION

Although the research on malware identification based on machine learning has made significant progress, it still faces some problems and challenges, some of which are listed and analyzed in this chapter. Class imbalance poses a common challenge in malware recognition scenarios, where the abundance of benign samples often outweighs the availability of malicious samples. This imbalance is particularly evident in malware classification tasks, leading to skewed distributions among different malware categories. In the MMCC dataset, for instance, Lollipop and Kelihos ver3 boast 2478 and 2942 samples, respectively, whereas the Simda category comprises a mere 42 samples. The inherent class imbalance tends to bias machine learning methods toward more significant categories.

To mitigate this issue, various approaches have been proposed in the literature. Rahman et al. [68] introduced two integrated learning mechanisms involving feature extraction from large and small category samples. They advocate the joint training of multiple weak classifiers to produce outputs for both sample types, primarily suited for binary classification. However, in the context of malware identification, class imbalance is more pronounced in multi-classification tasks. Kim et al.'s tGAN model [69] addresses this by enhancing data across all categories through generation, employing an integrated method for multi-classification. Nonetheless, concerns about the stability of the GAN model during sample generation persist. Venkata et al. [70] applied weighted SoftMax loss to fine-tune CNN, assigning varied loss weights based on category sample numbers. While this approach partially alleviates the impact of class imbalance, effectively addressing this challenge in malware identification remains an area that warrants further exploration.

Various anti-identification technologies are also emerging with the continuous development of malware identification technology based on machine learning. Malware writers can design corresponding strategies according to the characteristics of the machine learning model so that malware can evade the detection of the model. Guo et al. [42] first synthesized three view visualization applications, including behavior view, operation view and bytecode view. We then fuse the comprehensive semantics at different levels to profile the maliciousness of the apps precisely based on multi-view learning. Abdelmonem et al. [52] only change features that do not affect malicious functionality to generate adversarial examples that retain their malicious functionality. In addition, supervised machine learning models are generally constructed based on training data, and their generalization to unknown malware is often not high. For the above two situations, it is usually necessary to rebuild the recognition model according to the anti-detection technology and the characteristics of new malware and deal with it passively. Establishing an active malware recognition model that can self-evolve based on theories and methods such as evolutionary learning, adversarial generation, and lifelong learning is also a direction worthy of further exploration.

Currently, machine learning models and techniques are extensively employed in tasks related to identifying malicious code. This application helps address the issues of inadequate generalization and elevated false favorable rates associated with traditional methods relying on signatures and heuristic rules. This paper sorts out and summarizes the research on malware identification based on machine learning in recent years, expounds the early recognition methods based on feature engineering from the aspects of feature representation, feature selection, and dimensionality reduction, and summarizes and analyzes the recent RNN, CNN, and GNN deep neural network feature learning class recognition method. By studying the research status in this field, this paper points out that in the future, problems and challenges such as category imbalance and active recognition methods. With the rapid development of machine learning and deep learning technology, the performance and efficiency of malware identification based on machine learning will be further improved.

## REFERENCES


[1] J. Acharya, A. Chuadhary, A. Chhabria, and S. Jangale, “Detecting malware, malicious URLs and virus using machine learning and signature matching,” in 2021 2nd International Conference for Emerging Technology, INCET 2021, 2021. doi:10.1109/INCET51464.2021.9456440.

[2] U. Garg, N. Sharma, M. Kumar and A. Singh, "Identification and Detection of Behavior Based Malware using Machine Learning," 2023 International Conference on Artificial Intelligence and Smart Communication (AISC), Greater Noida, India, 2023, pp. 915-918, doi:10.1109/AISC56616.2023.10085168.

[3] Srastika, N. Bhandary, R. S. Shalakha, P. Honnavalli, and E. Sivaraman, “An Enhanced Malware Detection Approach using Machine Learning and Feature Selection,” in 3rd International Conference on Electronics and Sustainable Communication Systems, ICESC 2022 - Proceedings, 2022. doi:10.1109/ICESC54411.2022.9885509.

[4] Q. Qiao, R. Feng, S. Chen, F. Zhang, and X. Li, “Multi-label Classification for Android Malware Based on Active Learning,” IEEE Trans Dependable Secure Comput, 2022, doi:10.1109/TDSC.2022.3213689.

[5] T. Lu and J. Wang, “DOMR: Toward Deep Open-World Malware Recognition,” IEEE Transactions on Information Forensics and Security, vol. 19, 2024, doi: 10.1109/TIFS.2023.3338469.

[6] M. A. Halim, A. Abdullah, and K. A. Z. Ariffin, “Recurrent neural network for malware detection,” International Journal of Advances in Soft Computing and its Applications, vol. 11, no. 1, 2019.

[7] P. Borana, V. Sihag, G. Choudhary, M. Vardhan, and P. Singh, “An assistive tool for fileless malware detection,” in World Automation Congress Proceedings, 2021. doi:10.23919/WAC50355.2021.9559449.

[8] G. Popoiu, “One side class SVM training methods for malware detection,” in Proceedings - 2022 24th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2022, 2022. doi: 10.1109/SYNASC57785.2022.00065.

[9] M. Z. Shafiq, S. M. Tabish, F. Mirza, and M. Farooq, “PE-miner: Mining structural information to detect malicious executables in realtime,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2009. doi: 10.1007/978-3-642-04342-0_7.

[10] S. Tyagi, A. Baghela, K. M. Dar, A. Patel, S. Kothari, and S. Bhosale, “Malware Detection in PE files using Machine Learning,” in 2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development, OTCON 2022, 2023. doi:10.1109/OTCON56053.2023.10113998.

[11] S. Gulmez and I. Sogukpinar, “Graph-Based Malware Detection Using Opcode Sequences,” in 9th International Symposium on Digital Forensics and Security, ISDFS 2021, 2021. doi:10.1109/ISDFS52919.2021.9486386.

[12] N. McLaughlin and J. M. Del Rincon, “Data Augmentation for Opcode Sequence Based Malware Detection,” in 2022 Cyber Research Conference - Ireland, Cyber-RCI 2022, 2022. doi: 10.1109/Cyber-RCI55324.2022.10032676.

[13] X. D. Hoang, B. C. Nguyen and T. T. Trang Ninh, "Detecting Malware Based on Statistics and Machine Learning Using Opcode N-Grams," 2023 RIVF International Conference on Computing and Communication Technologies (RIVF), Hanoi, Vietnam, 2023, pp. 118-123, doi: 10.1109/RIVF60135.2023.10471824.

[14] Y. H.-q. SHAO Shu-di, F. Gui-sheng, Detecting malware by combining api and permission features, Computer Science 44 (4) (2017) 135. doi:10.11896/j.issn.1002-137X.2017.04.029.

[15] M. Alazab, M. Alazab, A. Shalaginov, A. Mesleh, and A. Awajan, “Intelligent mobile malware detection using permission requests and API calls,” Future Generation Computer Systems, vol. 107, 2020, doi:10.1016/j.future.2020.02.002.

[16] W. Huang and J. W. Stokes, “MtNet: A multi-task neural network for dynamic malware classification,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016. doi: 10.1007/978-3-319-40667-1_20.

[17] H. Y. Yang and J. Xu, “Android malware detection based on improved random forest,” Tongxin Xuebao/Journal on Communications, vol. 38, no. 4, 2017, doi: 10.11959/j.issn.1000-436x.2017073.

[18] M. A. Khalifa, A. Elsayed, A. Hussien and A. S. Hussainy, "Android Malware Detection and Prevention Based on Deep Learning and Tweets Analysis," 2024 6th International Conference on Computing

and Informatics (ICCI), New Cairo - Cairo, Egypt, 2024, pp. 153-157, doi: 10.1109/ICCI61671.2024.10485022.
[19] S. Amenova, C. Turan, and D. Zharkynbek, "Android Malware Classification by CNN-LSTM," in SIST 2022 - 2022 International Conference on Smart Information Systems and Technologies, Proceedings, 2022. doi: 10.1109/SIST54437.2022.9945816.
[20] B. Purnama, D. Stiawan, D. Hanapi, E. A. Winanto, R. Budiarto, and M. Y. Bin Idris, "N-gram Effect in Malware Detection Using Multilayer Perceptron (MLP)," in International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), 2021. doi: 10.23919/EECSI53397.2021.9624273.
[21] X. Su, W. Shi, X. Qu, Y. Zheng, and X. Liu, "DroidDeep: using Deep Belief Network to characterize and detect android malware," Soft comput, vol. 24, no. 8, 2020, doi: 10.1007/s00500-019-04589-w.
[22] S. Khalid and F. B. Hussain, "Evaluating Dynamic Analysis Features for Android Malware Categorization," in 2022 International Wireless Communications and Mobile Computing, IWCMC 2022, 2022. doi:10.1109/IWCMC55113.2022.9824225.
[23] B. Balodi, S. Sharma, A. K. Shukla, and B. Singh, "Automated Static Malware Analysis Using Machine Learning," in Proceedings of the 10th International Conference on Signal Processing and Integrated Networks, SPIN 2023, 2023. doi:10.1109/SPIN57001.2023.10116580.
[24] Z. Wu, J. Zhang, and L. Kou, "A Model for Malware Detection Method based on API call Sequence Clustering," in Proceedings - 2022 9th International Conference on Dependable Systems and Their Applications, DSA 2022, 2022. doi: 10.1109/DSA56465.2022.00157.
[25] M. Elalem and T. Jabir, "Malware Analysis in Cyber Security based on Deep Learning; Recognition and Classification," in Proceeding - 2023 IEEE 3rd International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering, MI-STA 2023, 2023. doi: 10.1109/MI-STA57575.2023.10169310.
[26] P. Singh, S. K. Borgohain, and J. Kumar, "Performance Enhancement of SVM-based ML Malware Detection Model Using Data Preprocessing," in 2022 2nd International Conference on Emerging Frontiers in Electrical and Electronic Technologies, ICEFEET 2022, 2022. doi: 10.1109/ICEFEET51821.2022.9848192.
[27] D. Albashish, R. Al-Sayyed, A. Abdullah, M. H. Ryalat, and N. Ahmad Almansour, "Deep CNN Model based on VGG16 for Breast Cancer Classification," in 2021 International Conference on Information Technology, ICIT 2021 - Proceedings, 2021. doi:10.1109/ICIT52682.2021.9491631.
[28] M. I. Pavel, S. Y. Tan, and A. Abdullah, "Vision-Based Autonomous Vehicle Systems Based on Deep Learning: A Systematic Literature Review," Applied Sciences (Switzerland), vol. 12, no. 14. 2022. doi:10.3390/app12146831.
[29] J. GrasleyJ. Grasley and A. D. Alahmar, "Systematic Mapping of Machine Learning-Based Malware Detection Studies," in International Conference on Electrical, Computer, and Energy Technologies, ICECET 2022, 2022. doi: 10.1109/ICECET55527.2022.9872937.
[30] A. F. Rasheed, M. Zarkoosh, and S. S. Al-Azzawi, "The Impact of Feature Selection on Malware Classification Using Chi-Square and Machine Learning," in Proceedings of the 9th International Conference on Computer and Communication Engineering, ICCCE 2023, 2023. doi: 10.1109/ICCCE58854.2023.10246084.
[31] R. Kalakoti, S. Nomm, and H. Bahsi, "In-Depth Feature Selection for the Statistical Machine Learning-Based Botnet Detection in IoT Networks," IEEE Access, vol. 10, 2022, doi:10.1109/access.2022.3204001.
[32] L. M. Kwan, "Markov Image with Transfer Learning for Malware Detection and Classification," in IEEE Region 10 Annual International Conference, Proceedings/TENCON, 2022. doi:10.1109/tencon55691.2022.9977916.
[33] C. Y. Priyanto, Hendry, and H. D. Purnomo, "Combination of Isolation Forest and LSTM Autoencoder for Anomaly Detection," in 2021 2nd International Conference on Innovative and Creative Information Technology, ICITech 2021, 2021. doi:10.1109/ICITech50181.2021.9590143.
[34] V. Patil, S. Shetty, A. Tawte, and S. Wathare, "Deep Learning and Binary Representational Image Approach for Malware Detection," in 2023 International Conference on Power, Instrumentation, Control and Computing, PICC 2023, 2023. doi:10.1109/PICC57976.2023.10142644.
[35] X. Ke and Y. X. Hui, "Android Malware Detection Based on Image Analysis," in Proceedings of 2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence, ICIBA 2021, 2021. doi: 10.1109/ICIBA52610.2021.9688179.
[36] A. Migdady, L. Smadi, and Q. Yaseen, "A CNN and Image-Based Approach for Malware Analysis," in 2022 International Conference on Emerging Trends in Computing and Engineering Applications, ETCEA 2022 - Proceedings, 2022. doi:10.1109/ETCEA57049.2022.10009748.
[37] H. Malani, A. Bhat, S. Palriwala, J. Aditya, and A. Chaturvedi, "A Unique Approach to Malware Detection Using Deep Convolutional Neural Networks," in Proceedings, International Conference on Electrical, Control and Instrumentation Engineering, ICECIE, 2022. doi: 10.1109/ICECIE55199.2022.10000344.
[38] M. Alam, A. Akram, T. Saeed, and S. Arshad, "DeepMalware: A Deep Learning based Malware Images Classification," in 2021 International Conference on Cyber Warfare and Security, ICCWS 2021 - Proceedings, 2021. doi: 10.1109/ICCWS53234.2021.9703021.
[39] J. Geremias, E. K. Viegas, A. O. Santin, A. Britto, and P. Horchulhack, "Towards a Reliable Hierarchical Android Malware Detection Through Image-based CNN," in Proceedings - IEEE Consumer Communications and Networking Conference, CCNC, 2023. doi:10.1109/CCNC51644.2023.10060381.
[40] O. E. Kural, D. Ö. Şahin, S. Akleylek, E. Kiliç, and M. Ömüral, "Apk2Img4AndMal: Android Malware Detection Framework Based on Convolutional Neura Network," in Proceedings - 6th International Conference on Computer Science and Engineering, UBMK 2021, 2021. doi: 10.1109/UBMK52708.2021.9558983.
[41] D. Gibert, C. Mateu, J. Planes, and R. Vicens, "Classification of malware by using structural entropy on convolutional neural networks," in Proceedings of the 30th Innovative Applications of Artificial Intelligence Conference, IAAI 2018, 2018. doi:10.1609/aaai.v32i1.11409.
[42] J. Guo, Z. Meng, Q. Zhang, Y. Xiong, and W. Huang, "MVVDroid: Android Malware Detection based on Multi-View Visualization," in Proceedings - 2023 9th International Conference on Big Data Computing and Communications, BigCom 2023, 2023. doi:10.1109/bigcom61073.2023.00021.
[43] R. Rahul and L. Kumble, "Investigation of Malware & Threat Analysis on APKs Using SVM & ANN Algorithm. -A New Approach," in 2023 International Conference on Recent Advances in Information Technology for Sustainable Development, ICRAIS 2023 - Proceedings, 2023. doi: 10.1109/ICRAIS59684.2023.10367124.
[44] M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang, and F. Iqbal, "Malware Classification with Deep Convolutional Neural Networks," in 2018 9th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2018 - Proceedings, 2018. doi: 10.1109/NTMS.2018.8328749.
[45] B. Yuan, J. Wang, D. Liu, W. Guo, P. Wu, and X. Bao, "Byte-level malware classification based on markov images and deep learning," Comput Secur, vol. 92, 2020, doi: 10.1016/j.cose.2020.101740.
[46] M. Krčál, O. Švec, O. Jašek, and M. Bálek, "Deep convolutional malware classifiers can learn from raw executables and labels only," in 6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings, 2018.
[47] K. Thosar, P. Tiwari, R. Jyothula, and D. Ambawade, "Effective Malware Detection using Gradient Boosting and Convolutional Neural Network," in 2021 IEEE Bombay Section Signature Conference, IBSSC 2021, 2021. doi: 10.1109/IBSSC53889.2021.9673266.
[48] Y. Liu and Y. Wang, "A robust malware detection system using deep learning on API calls," in Proceedings of 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2019, 2019. doi: 10.1109/ITNEC.2019.8728992.
[49] L. Xiaofeng, J. Fangshuo, Z. Xiao, Y. Shengwei, S. Jing, and P. Lio, "ASSCA: API sequence and statistics features combined architecture for malware detection," Computer Networks, vol. 157, 2019, doi:10.1016/j.comnet.2019.04.007.
[50] J. Huang, C. Lu, G. Ping, L. Sun, and X. Ye, "TCN-ATT: A Non-recurrent Model for Sequence-Based Malware Detection," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2020. doi:10.1007/978-3-030-47436-2_14.
[51] K. Tsunewaki, T. Kimura, and J. Cheng, "LSTM-Based Ransomware Detection Using API Call Information," in Proceedings - 2022 IEEE International Conference on Consumer Electronics - Taiwan, ICCE-Taiwan 2022, 2022. doi: 10.1109/ICCE-Taiwan55306.2022.9869122.
[52] S. Abdelmonem, S. Seddik, R. El-Sayed, and A. S. Kaseb, "Enhancing Image-Based Malware Classification Using Semi-Supervised Learning," in NILES 2021 - 3rd Novel Intelligent and Leading Emerging Sciences Conference, Proceedings, 2021. doi:10.1109/NILES53778.2021.9600511.

[53] H. Gui, F. Liu, C. Zhang, and K. Tang, “A Malware Classification Method based on Attentive Bidirectional Model,” in 2022 7th International Conference on Intelligent Computing and Signal Processing, ICSP 2022, 2022. doi: 10.1109/ICSP54964.2022.9778322.

[54] S. Li, Q. Zhou, R. Zhou, and Q. Lv, “Intelligent malware detection based on graph convolutional network,” Journal of Supercomputing, vol. 78, no. 3, 2022, doi: 10.1007/s11227-021-04020-y.

[55] L. Zhang, J. Yin, J. Ning, Y. Wang, B. Adebisi, and J. Yang, “A Novel Unsupervised Malware Detection Method based on Adversarial Auto-encoder and Deep Clustering,” in Proceedings - 2022 9th International Conference on Dependable Systems and Their Applications, DSA 2022, 2022. doi: 10.1109/DSA56465.2022.00038.

[56] X. Chen, “Power System Malware Detection Based on Deep Belief Network Classifier,” in 2022 6th International Conference on Green Energy and Applications, ICGEA 2022, 2022. doi:10.1109/icgea54406.2022.9792083.

[57] I. S. Srinu and D. Vidyarthi, "Classification of Malware Using Deep Learning: A Study," 2023 IEEE International Carnahan Conference on Security Technology (ICCST), Pune, India, 2023, doi:10.1109/iccst59048.2023.10474230.

[58] A. Abdullah, R. C. Veltkamp, and M. A. Wiering, “Spatial pyramids and two-layer stacking SVM classifiers for image categorization: A comparative study,” in Proceedings of the International Joint Conference on Neural Networks, 2009. doi:10.1109/ijcnn.2009.5178743.

[59] U. Garg, S. S. Rana, D. S. Bisht, R. Rautela, and A. Garg, “A Comparative Analysis of IoT Malware Detection Using CNN and Deep Learning,” in Proceedings - International Conference on Technological Advancements in Computational Sciences, ICTACS 2023, 2023. doi: 10.1109/ICTACS59847.2023.10389976.

[60] L. Alsharafi, M. Asiri, S. Azzony, and A. Alqahtani, “Malware Detection Based on Deep Learning,” in 2023 3rd International Conference on Computing and Information Technology, ICCIT 2023, 2023. doi: 10.1109/ICCIT58132.2023.10273961.

[61] K. L. Lam, A. Abdullah, and D. Albashish, “Ensemble of Fully Convolutional Neural Networks with End-to-End Learning for Small Object Semantic Segmentation,” in Lecture Notes in Networks and Systems, 2023. doi: 10.1007/978-3-031-26889-2_12.

[62] S. Sridhar, R. Seetharaman, and S. Sanagavarapu, “Intelligent Vision-based Malware Classification using Quantised ResNets,” in 2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2021, 2021. doi:10.1109/iemcon53756.2021.9623219.

[63] D. Vasan, M. Alazab, S. Wassan, B. Safaei, and Q. Zheng, “Image-Based malware classification using ensemble of CNN architectures (IMCEC),” Comput Secur, vol. 92, 2020, doi:10.1016/j.cose.2020.101748.

[64] T. Deng, “A Survey of Convolutional Neural Networks for Image Classification: Models and Datasets,” in Proceedings - 2022 International Conference on Big Data, Information and Computer Network, BDICN 2022, 2022. doi: 10.1109/BDICN55575.2022.00145.

[65] MM. Asaduzzaman and M. M. Rahman, “An Adversarial Approach for Intrusion Detection Using Hybrid Deep Learning Model,” in 2022 International Conference on Information Technology Research and Innovation, ICITRI 2022, 2022. doi:10.1109/icitri56423.2022.9970221.

[66] J. Busch, A. Kocheturov, V. Tresp, and T. Seidl, “NF-GNN: Network Flow Graph Neural Networks for Malware Detection and Classification,” in ACM International Conference Proceeding Series, 2021. doi: 10.1145/3468791.3468814.

[67] P. Feng, J. Ma, T. Li, X. Ma, N. Xi, and D. Lu, “Android Malware Detection via Graph Representation Learning,” Mobile Information Systems, vol. 2021, 2021, doi: 10.1155/2021/5538841.

[68] M. M. Rahman et al., “CNN vs Transformer Variants: Malware Classification Using Binary Malware Images,” in Proceeding - COMNETSAT 2023: IEEE International Conference on Communication, Networks and Satellite, 2023. doi:10.1109/comnetsat59769.2023.10420585.

[69] J. Y. Kim, S. J. Bu, and S. B. Cho, “Malware detection using deep transferred generative adversarial networks,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2017. doi:10.1007/978-3-319-70087-8_58.

[70] E. Venkata Pawan Kalyan, A. Purushottam Adarsh, S. Sai Likith Reddy, and P. Renjith, “Detection of Malware Using CNN,” in 2022 2nd International Conference on Computer Science, Engineering and Applications, ICCSEA 2022, 2022. doi:10.1109/ICCSEA54677.2022.9936225.