JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 08 Issue 02
2025

Research Article
https://doi.org/10.56979/802/2025

MalvareVison: A Deep Learning-Driven Approach For Malware Classification

Aamir Ali[1], Malik Arslan Akram[2], Wajiha Farooq[3], Misbah Ali[4*], Moomna Nazir[5], Aown Muhammad[6], and Tehseen Mazhar[7]

[1]Department of Computer Science, Government College University Faisalabad, Sahiwal campus 57000, Pakistan.
[2]Department of Software Engineering, Southwest University of Science and Technology, Mianyang, Sichuan 621010, China.
[3]Department of Computer Science,Bahauddin Zakariya University, Multan 60800, Pakistan.
[4]Department of Computer Science, School Education Department, Government of Punjab, Jhang 40100, Pakistan.
[5]Department of Computer Science, Govt. Post Graduate College for Women, Sahiwal 57040, Pakistan.
[6]Department of Computer Science, University of Engineering and Technology, Lahore 54890, Pakistan.
[7]Department of Computer Science, School Education Department, Government of Punjab, Layyah 31200, Pakistan.
[*]Corresponding Author: Misbah Ali. Email: talktomisbah.ali@gmail.com

Received: January 03, 2025 **Accepted:** February 28, 2025

Abstract: The fast propagation of malware across the internet requires the development of advanced classification and detection techniques. Traditional signature-based detection malware methods often fail to identify new and obfuscated variants which demand advanced machine learning-based solutions. We propose MalwareVision, a framework based on deep learning for the classification of malware samples. The model was trained on the Malimg dataset comprising images of 9,339 malware images across 25 families and evaluated based on accuracy, precision, recall, and F1-score metrics. We observe that the model achieved an impressive accuracy of 95.09% in both the training and testing datasets and that the precision and recall values remained high for most malware families. The results highlight the effectiveness of deep learning-based Convolutional Neural Network (CNN) for malware classification. The proposed MalwareVision framework offers a scalable, automated solution for malware classification, contributing to the advancement of AI-driven cybersecurity defenses.

Keywords: Convolutional Neural Networks (CNNs); Cybersecurity; Deep Learning; Malware Classification

1. Introduction

With the growth of digital technology and networked appliances, the strike surface for cyber threats has been considerably expanded. Malware is a ubiquitous and evolving threat that poses an unending challenge to enterprise security staff and individuals (1). Recent industrial reports suggest that millions of malware variants are being developed consistently. Criminals are using more and more sophisticated methods to avoid detection such as obfuscation, polymorphism, and encryption (2). The lifecycle of a malware attack includes various stages as shown in Figure 1. Traditional heuristic-based systems rely on pre-defined signatures and are weak at finding advanced modifications of the same malicious code (3). Alarmingly, the spread of Internet of Things (IoT) devices has resulted in additional infiltrations of weak security. Malware attacks targeting the IOT create massive problems – ranging from individual identity thefts to large-scale DDoS attacks (4). This situation becomes more complicated, as many of these devices lack adequate malware security mechanisms. From stealing personal information to launching massive distributed DDoS attacks, malware attacks on IoT can lead to disastrous outcomes (5,6). This growing threat landscape necessitates more intelligent and smarter malware detection and classification mechanisms to be able to detect and classify the new emerging threats as their behaviors change and evolve. The malware can spread in multiple ways affecting IOT devices while stealing and damaging

critical data (7). The hierarchical model of cyber threats presenting infection sources, infection targets, propagation methods, and defense and mitigation strategies is presented in Figure 2.

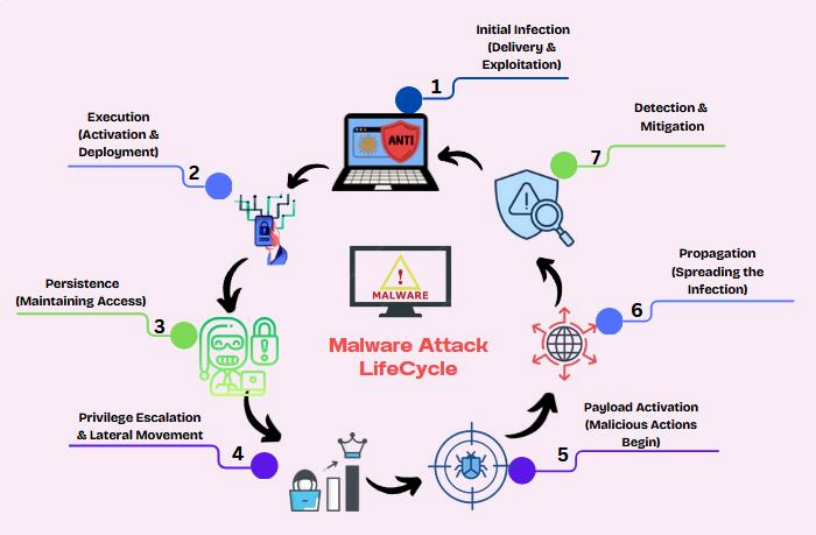

Figure 1. Malware attack lifecycle depicting the sequential phases involved in the attack

The classification of the malware variants into specific families is one of the primary problems that comes across during malware detection. As a result, malware developers are continuously introducing new variants of malware with minor alterations to bypass detection mechanisms. Antivirus solutions still struggle to handle this diversity which leads to increasing false negatives and compromised security (8).

To deal with these challenges and find optimized solutions for automated malware classification, various ML and Deep Learning (DL) techniques were examined by the researchers. However, conventional machine techniques require manual feature engineering and have a lesser generalization ability to classify malware variants into diverse families (9). Moreover, malware datasets have become more sophisticated and larger requiring scalable and efficient classification techniques.

The advanced form of ML i.e., DL has emerged as a transformative approach by offering automatic feature selection for effective malware classification. DL algorithms, specifically Convolutional Neural Networks (CNNs) exhibit efficient performance by learning hierarchical features directly on unseen data without executing feature extraction as a separate stage. The translation of malware binaries into grayscale images and the implementation of CNN architecture is a promising solution for malware classification (10).

CNNs have produced remarkable results on a range of image classification tasks making them suitable choices for malware binary images. The generalization ability of CNN can be enhanced by optimizing the model through hyperparameter optimization (11).

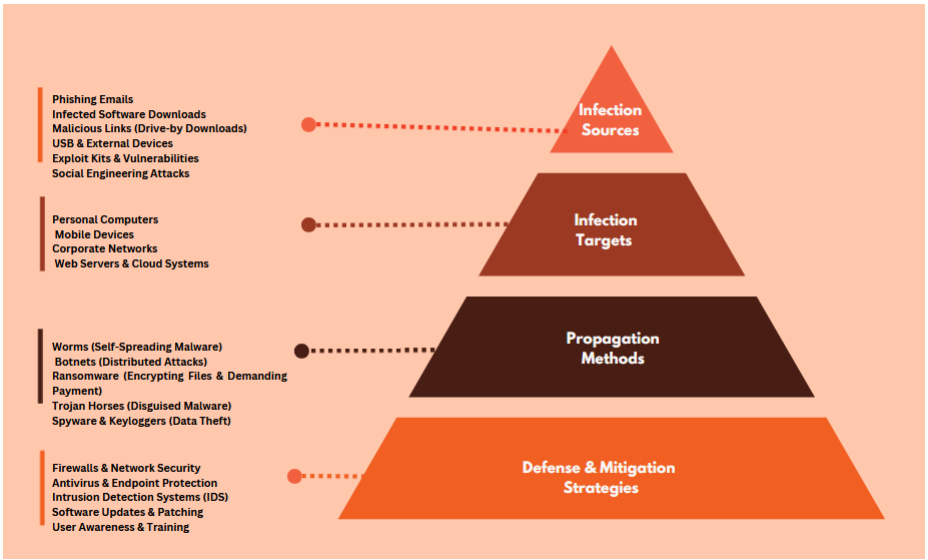

Figure 2. The Cyber Threat Pyramid representation

This research aims to develop a robust malware classification framework leveraging a DL-based CNN model to improve accuracy and adaptability. The key objectives of this study include: the development of an efficient and lightweight malware detection framework suitable for real-time deployment in resource-constrained environments. Moreover, we compared the proposed DL framework against modern ML approaches focusing on accuracy. By addressing the challenges of present malware detection techniques and leveraging DL advancements, this research designs a robust solution for automated malware classification in dynamic cybersecurity landscapes.

The subsequent sections of this study are organized as follows: Section 2 offers a review of previous malware classification techniques. Section 3 illustrates the methodology comprising dataset acquisition and preparation, feature extraction, classification, and evaluation metrics. Section 4 details the performance results along with the comparison of the proposed framework with Straight-Of-The-Art techniques. Section 5 concludes the study along with the future directions.

2. Related work

This section will explore various malware classification methods including static, Machine learning (ML), and DL techniques for improved security against malware. Cybersecurity specialists continuously research malware detection techniques to improve security and prevent infections.

2.1. Static Analysis for Malware Detection

The study (12) performed a static analysis of various Android executables supporting collaborative malware detection. The authors proposed an approach that worked by extracting features from Android executables without executing them and produced an efficient method for malicious software detection. The researchers emphasized the advantages of static analysis in mobile security, particularly in resource-constrained environments. However, the method's high false positive rate and inefficiency in classifying malware using APIs were also highlighted by the researchers.

Another research (13) focuses on malware classification using extracted API sequences obtained via static analysis. The study highlighted that through the examination of API calls in executable programs, malware samples can be divided among benign and malicious software. The limitations found in static analysis were addressed by the interdiction of obfuscation techniques that further demonstrated the shortcomings of static analysis. Recently, the study (14) presented Eureka, a framework based on static malware analysis. It analyzed malware samples to find meaningful features and improve on the existing techniques of static detection. The target was to improve the accuracy of malware classification while studying structural properties without running code thereby reducing the chances for malicious code execution when you're analyzing it statically.

2.2. Machine learning-based malware classification

With the rapid growth of AI-based models, researchers have been researchers have examined various ML technologies for more effective malware classification across families of viruses. A recent study (15), examined a number of ML techniques for effective classification and detection of malicious software while highlighting multiple developments and challenges in the field of cybersecurity. The researchers grouped ML techniques into three types - static, dynamic, and hybrid. The survey also emphasized feature extraction from a given dataset, including frequency analysis of opcode and monitoring system calls. The study discussed many problems encountered, such as adversarial attacks, evasion techniques, and imbalanced datasets. The findings emphasized that ML models play an efficient role in improving malware classifying accuracy. A different study (16) investigated the ensemble methods based on ML for detecting Windows Portable Executable (PE) malware. To determine the best strategy for Windows PE-based malware samples, the study investigated several ensemble techniques, including random forest, gradient boosting, and stacking generalization. The efficiency of manual feature engineering in enhancing the effectiveness of ML-based malware categorization was highlighted by the authors. According to the study, combining many base models to create a more potent ensemble model can increase classification accuracy and solve the issue of class bias in unbalanced datasets.

Researchers explored Self-Organizing Feature Maps (SOFM) and system behavior data for malware classification (17). The authors examine the unsupervised ML techniques to identify malware variants without relying on predefined class labels. They suggested that by the clustering of system activity logs, the SOFM can identify anomalies indicating malicious behavior. They also highlighted that the

combination of SOFM with ML models can enhance predictive accuracy while reducing false positives in malware detection systems.

A Novel Multi-View Fuzzy Consensus Clustering Model (MVFCC) was proposed by (18) to detect and analyze malware threats. The research introduced a fuzzy logic-based clustering technique that combines various malware samples depending on their behavioral similarities. The experiments revealed that the proposed MVFCC model offered interpretable clustering results while effectively classifying malware families and enhancing cyber threat detection accuracy. Multi-view heuristic analysis was explored by researchers in (19). The researchers focused on cyber threat identification by applying heuristic-based feature selection, adversarial analysis, and meta-learning techniques. The study revealed that multi-view analysis is capable of offering improved threat profiling and advanced persistent threats.

2.3. Deep learning-based malware classification

Deep learning techniques, particularly CNNs and Recurrent Neural Networks (RNNs) have been increasingly applied for malware categorization. CNNs are particularly effective for pattern recognition and image classification, making them well-suited for malware image detection tasks (20). Studies have shown that modern CNN architectures can achieve high accuracy in large-scale image classification tasks (21).

The study (22) introduced a multi-view feature fusion approach for malware identification based on DL while integrating multiple feature extraction techniques to improve classification accuracy. The proposed model leverages CNNs, RNNs, and ensemble learning techniques to analyze different feature sets, such as opcode sequences, API call traces, and binary images. By combining these multiple perspectives, the model enhances detection rates and robustness against obfuscation techniques. The paper highlights that DL models outperform traditional classifiers when properly trained on diverse feature representations. Similarly, the work in (23) investigated the impact of automatic feature selection and highlighted that DL models can outperform traditional ML models by learning hierarchical feature representations directly from raw data. The findings suggest that DL models, particularly CNNs and deep neural networks, can significantly improve classification tasks by reducing the reliance on manual feature engineering. This is highly relevant to malware classification, where traditional feature extraction (such as API calls or opcode sequences) can be time-consuming and prone to evasion techniques. Another research (24) introduced an improved CNN model for malware classification. By leveraging CNNs, the study demonstrated how malware binaries can be transformed into image representations and classified effectively. The authors proposed architectural enhancements to traditional CNN, optimizing its ability to capture discriminative patterns in malware samples. The study highlighted that DL-based approaches outperform conventional signature-based detection methods, offering better generalization to novel malware variants.

The study (25) surveyed recent developments in malware classification, focusing on behavioral analysis, static code inspection, and hybrid approaches. It provided an in-depth examination of malware families, their evolving structures, and how modern classifiers differentiate among variants. The study also discussed feature engineering techniques, including API call graphs, opcode sequences, and binary visualization along with the evaluation of automated feature extraction using DL. The study highlighted the effectiveness of DL techniques such as CNNs, autoencoders, and graph neural networks (GNNs) for extracting high-level malware signatures. A comparison between static analysis and intelligent classification based on advanced ML techniques for effective malware classification is depicted in Figure 3.

Despite advancements in DL for malware classification, certain limitations exist. Many existing models struggle with class imbalance, leading to poor recall scores for underrepresented malware families (26). Additionally, DL-based malware detection systems require significant computational power, limiting their feasibility in environments with restricted resources (22). This research focuses on optimizing lightweight DL architectures specifically CNNs, optimizing the model architecture for enhanced malware classification performance. The summary of existing literature for malware classification is shown in Table 1.

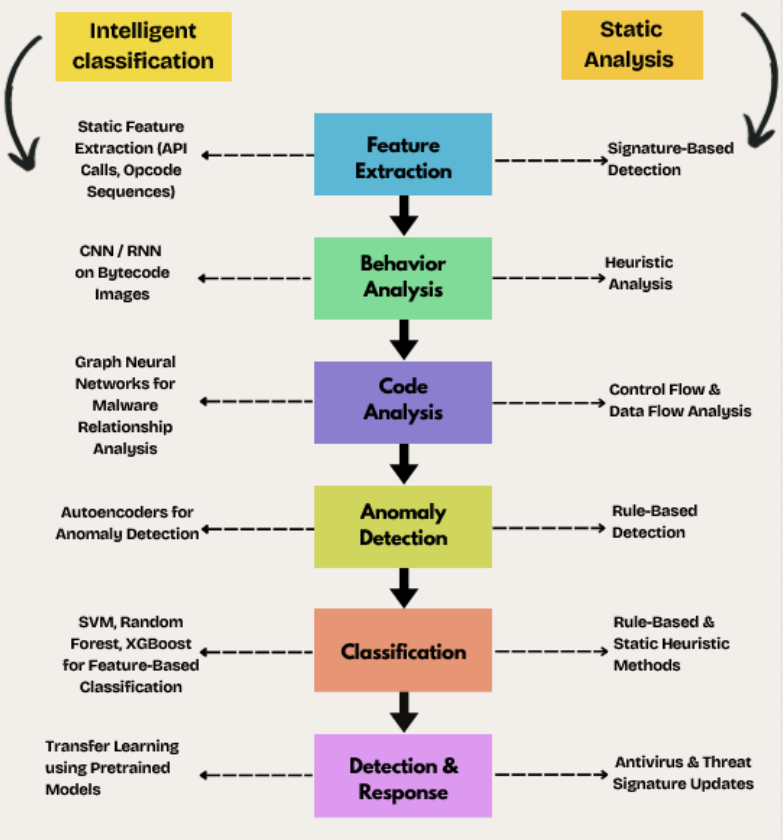

Figure 3. Comparison between static and advanced ML approaches for malware classification

Table 1. Summary of Related Work

Study	Approach	Key Findings
	Static Analysis-Based Malware Detection	
[1]	Static analysis of Android executables	Effective for malware detection but has a high false positive rate.
[2]	API sequence analysis via static analysis	API call patterns distinguish malware; obfuscation techniques improve detection.
[3]	Eureka framework for static malware analysis	Extracts structural features for improved malware classification.
	ML-Based Malware Classification	
[4]	Review of ML techniques in malware detection	Categorizes ML techniques; highlights DL's growing role.
[5]	Application of ensemble learning for Windows PE malware detection	The ensemble technique improves accuracy, particularly for imbalanced datasets
[6]	SOFM and system behavior data for malware classification	Unsupervised learning detects malware variants without labels.
[7]	MVFCC: Multi-View Fuzzy Consensus Clustering	Groups malware based on behavior; enhances threat intelligence.
[8]	Cyber threat attribution using Multi-View Heuristic Analysis	Improves threat profiling for advanced persistent threats (APTs).
	DL-Based Malware Classification	
[9]	Multi-view feature fusion with CNNs, RNNs, and ensemble learning	Enhances detection rates and resists obfuscation techniques.
[10]	Automatic feature extraction in DL	DL reduces reliance on manual feature engineering.
[11]	CNN-based malware classification with architectural enhancements	Converts malware binaries to images; CNNs outperform traditional methods.

[12]	Survey on malware classification and feature extraction	Discusses automated feature extraction via CNNs, autoencoders, and GNNs.

In this study, we analyze state-of-the-art malware classification techniques with DL and emphasize their role in cybersecurity, advocating for further research opportunities in this area to build robust and efficient malware classifiers. While integrating advanced DL techniques, this research aims to contribute to cyber security by developing an advanced solution with improved predictive accuracy for threat detection.

3. Materials and Method

We have implemented DL techniques for the classification of malware across diverse families. We have introduced a new framework called MalwareVision that takes pre-translated grayscale images as input and applies optimized CNN for effective classification. The proposed methodology is divided into 4 key stages: 1) Data acquisition, 2) Data preprocessing, 3) Feature Extraction & Classification, and 4) Performance evaluation. Utilizing the proposed framework, we can work through each stage in a systematic way that allows classifying malware accurately and efficiently to provide a solution to modern cybersecurity attacks. The following section describes each stage of MalwareVision in detail.

Stage 1: Data Acquisition

Model performance is directly related to data acquisition, which is an essential step in the process of malware classification. The study utilized a Malimg dataset of 9,339 images comprising malware of 25 different families. It was first proposed by Nataraj et al. in (27), and it also has been used in many DL-based malware analysis studies. An overview of the applied Malimg malware dataset is outlined below in Table 2.

Table 2. Overview of the Malimg Dataset

Attribute	Description
Dataset	Malimg Malware Dataset
Total Samples	9,339
Number of Malware Families	25

In contrast to the normal raw malware datasets, Malimg offers preprocessed grayscale images, where malware binaries have been transformed into images. These images correspond to the binary structure of malware executables where pixel intensities map to byte values. The pre-transformed grayscale images provide pattern identification and feature representation enabling deep-learning models to process raw executable files without manual feature engineering. These images are then investigated to identify data similarities across distinct malware families while enhancing model robustness and offering alternate representations of malware samples. Moreover, this approach prevents malware detection from relying solely on static analysis which may evaded by advanced malware variants. A batch of these grayscale images along with their corresponding class labels presenting the inherent patterns and textures of diverse malware classes is visualized in Figure 4.

The Malimg dataset presenting the malware class names, malware family, number of malware samples per class, and number of malware samples per malware is summarized in Table 3. A pie chart presenting the distribution of malware samples across 25 diverse families is shown in Figure 5.

Stage 2: Data preprocessing

To ensure optimal model performance and reliable classification, we performed a series of preprocessing steps on the malware image dataset. The images were loaded using the Keras ImageDataGenerator function, which facilitated the organization of data into batches. To create a standardized data set, all malicious images were resized to 64×64 pixels. This not only keeps computational costs down but also preserves key structural features of the images. The pixel intensity values of these images are scaled between the range 0 and 1. By scaling all pixel values by 1/255, the model's convergence rate can be accelerated in training. The dataset is split into training and testing portions, with 70% of the data used as training data and 30% for testing. This method helps to avoid overfitting the model on the training data so that it will work equally well for unseen malicious samples. Proper data splitting can prevent overfitting and make it possible for the model to generalize to unseen malware samples. The malware categories corresponding to each data point were then one-hot encoded, a method of setting each

malware category to a vector. This encoding scheme allows the model to learn malware patterns effectively and to distinguish among unseen samples.

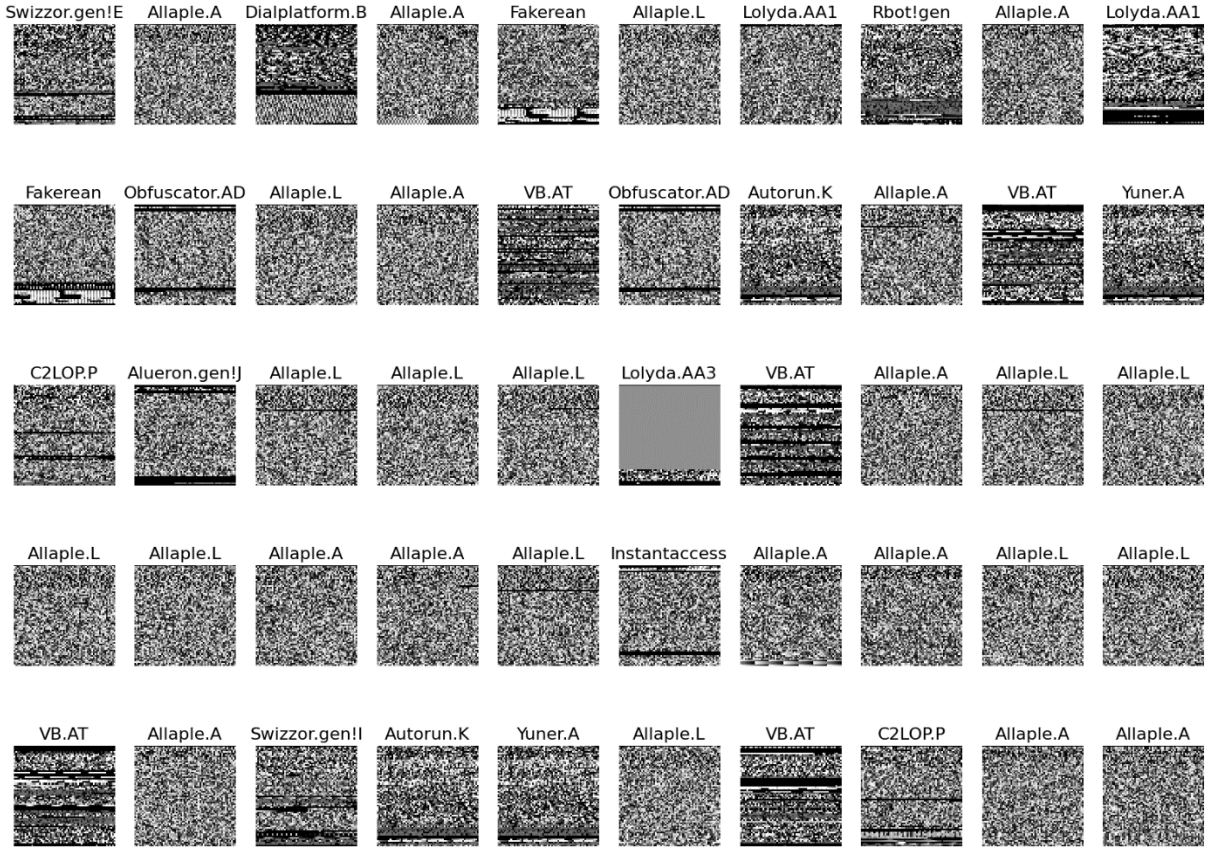

Figure 4. Visualization of malware variants of the Malimg dataset

Table 3. Statistical summary of Malimg dataset

Malware class name	Malware family	Number of malware samples per class	Number of malware samples per family
Agent.FYI	Backdoor	116	274
Rbot!gen	Backdoor	158	
Adialer. C	Dialer	122	
Dialplatform. B	Dialer	177	730
Instantaccess	Dialer	431	
Lolyda.AA1	Password stealer (PWS)	213	
Lolyda.AA2	PWS	184	679
Lolyda.AA3	PWS	123	
Lolyda. AT	PWS	159	
Fakerean	Rogue	381	381
Alueron.gen!J	Trojan	198	
C2LOP.P	Trojan	146	
C2LOP.gen!g	Trojan	200	760
Malex.gen!J	Trojan	136	
Skintrim. N	Trojan	80	
Dontovo. A	Trojan Downloader	162	
Obfuscator. AD	Trojan Downloader	142	
Swizzor.gen!E	Trojan Downloader	128	661
Swizzor.gen!I	Trojan Downloader	132	
Wintrim. BX	Trojan Downloader	97	

Allaple. A	Worm	2949	
Allaple. L	Worm	1591	5748
VB.AT	Worm	408	
Yuner. A	Worm	800	
Autorun.K	Worm.AutoIT	106	106
Total		9339	

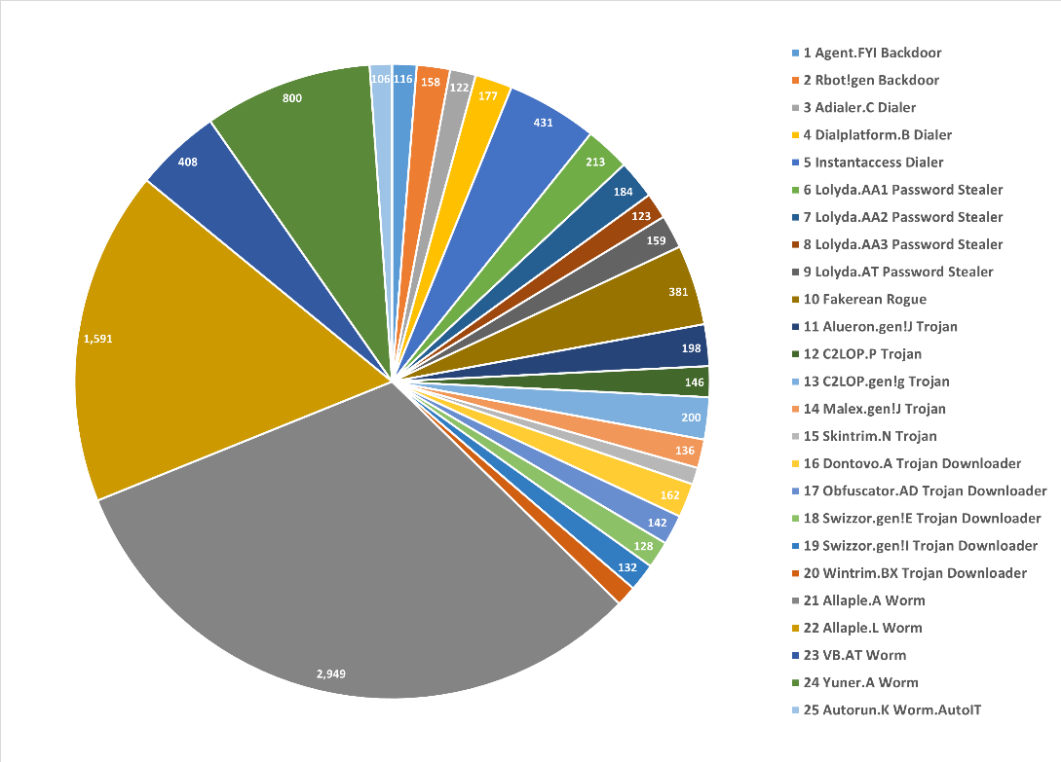

Figure 5. Overview of the MalimG dataset

The purpose of preprocessing techniques is to prepare the dataset, reduce computational load, and optimize the input for malware classification. These steps converted the transformed images into a format that allows for structured datasets to be formed; this is the prerequisite for effective training and evaluation.

In the end, the preprocessed dataset is found to contain no inconsistencies or errors. This final stage ensured that all the grayscale images and their associated malware labels were matched. Such qualitative checks have confirmed that a correct conversion has been made between image forms and label placement, while at the same time revealing the characteristic texture patterns.

During preprocessing, we have applied data security measures by using a Virtual Machine specifically VirtualBox with snapshots and no internet access to isolate malware samples hence preventing contamination from real malware threats. The systematic approach followed to preprocess the dataset is depicted in Figure 6.

Stage 3: Feature Extraction and Classification

The proposed framework, named MalwareVision, leverages a convolutional neural network for automated feature extraction and classification of malware images. The architecture implemented to produce an optimized version of CNN for effective malware classification is depicted in Figure 7. The model takes 64×64×3 sized input images and passes them through a series of layers. The architecture is composed of two convolutional layers; comprising 30 (3 × 3) filters along with the ReLU activation function followed by a 2 × 2 max pooling layer. It is subsequently followed by another conv layer with 15 filters with a kernel size of 3×3 and also using ReLU. After feature extraction through convolution and pooling, the feature maps are flattened into a one-dimensional vector. This single vector is fed into two fully connected layers, each having 128 neurons and a ReLU activation function. To prevent overfitting, a dropout layer has also been applied. The final layer has multiple output neurons, each corresponding to a

different malware class, where classification is achieved through the application of the softmax activation function. The final layer has several output neurons, each representing a different malware family (such as Adialer. C, Allaple. A, Allaple. L, etc.).

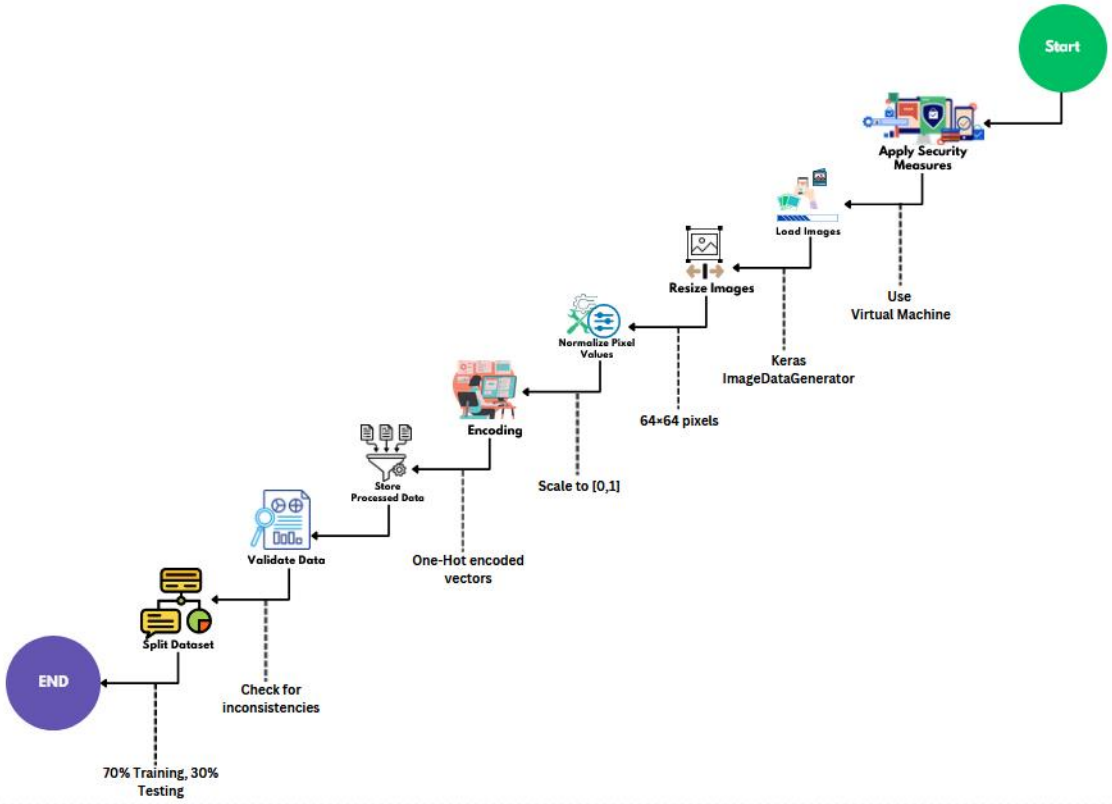

Figure 6. Systematic preprocessing of the Malimg dataset

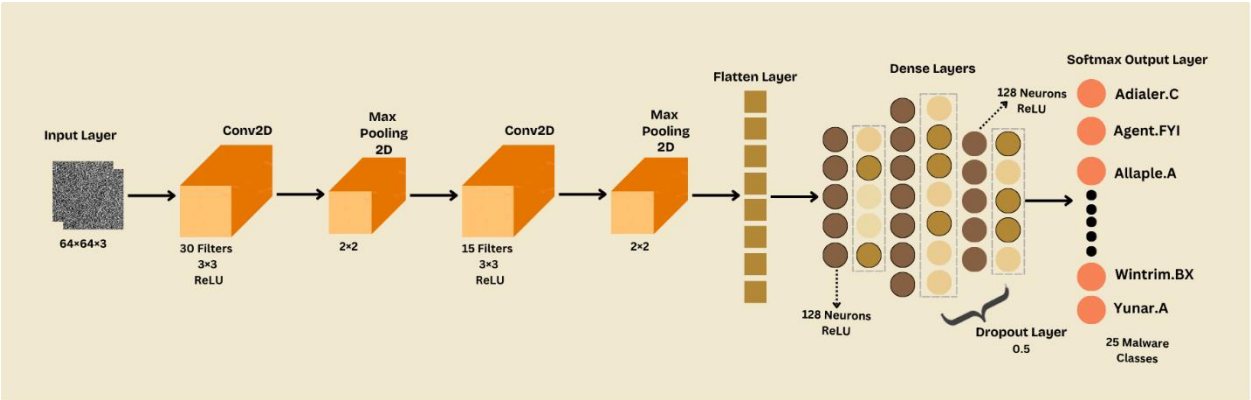

Figure 7. Optimized architecture of CNN

Unlike traditional approaches that rely on manual features, CNNs learn spatial and structural patterns directly from grayscale malware images, represented as 64×64×3 matrices. The convolutional layers in the model capture local texture variations and malware-specific structural characteristics. Additionally, a MaxPooling 2-dimensional layer reduces the spatial dimensions while preserving crucial features. The other convolutional layer, further refines the extracted representations, followed by another MaxPooling2D layer that downsamples the feature maps to 14×14×15. A Dropout layer (0.25) is incorporated at this stage while avoiding overfitting and enhancing generalization.

After feature extraction, the proposed framework proceeds with classification. The softmax output layer comprising 25 neurons, corresponding to 25 malware families assigns probability scores to each class. The model has numerous trainable parameters that have been optimized using the Adam optimizer, which

adaptively adjusts the learning rate for stable convergence. Categorical cross-entropy loss is used as the objective function, given the multi-class classification nature of the problem.

To optimize classification performance, the CNN was trained using the **Adam optimizer**, which adaptively adjusts the learning rate for stable convergence. The model was trained for 22 epochs with continuous monitoring of training and testing accuracy. The final trained model achieved reliable classification performance, and a confusion matrix was generated to analyze misclassifications across different malware families. With a structured DL architecture, MalwareVision effectively captures hierarchical features from malware images, reducing manual feature engineering efforts and improving detection accuracy. The trained model was saved as malware.h5, enabling future deployment for real-world malware detection tasks. The proposed malware classification framework is shown in Figure 8.

Stage 4: Performance assessment

Performance assessment is an essential task to determine the efficiency and generalizability of the proposed approach (28). Appropriate performance measures ensure the correct predictions while categorizing malicious software into diverse families. This evaluation attempts to anticipate the future evolution of malware by devising methods for efficient and reliable predictions. We have applied widely used performance measures such as accuracy, precision, recall, F1-score, and loss to guarantee performance robustness for cybersecurity applications (29,30).

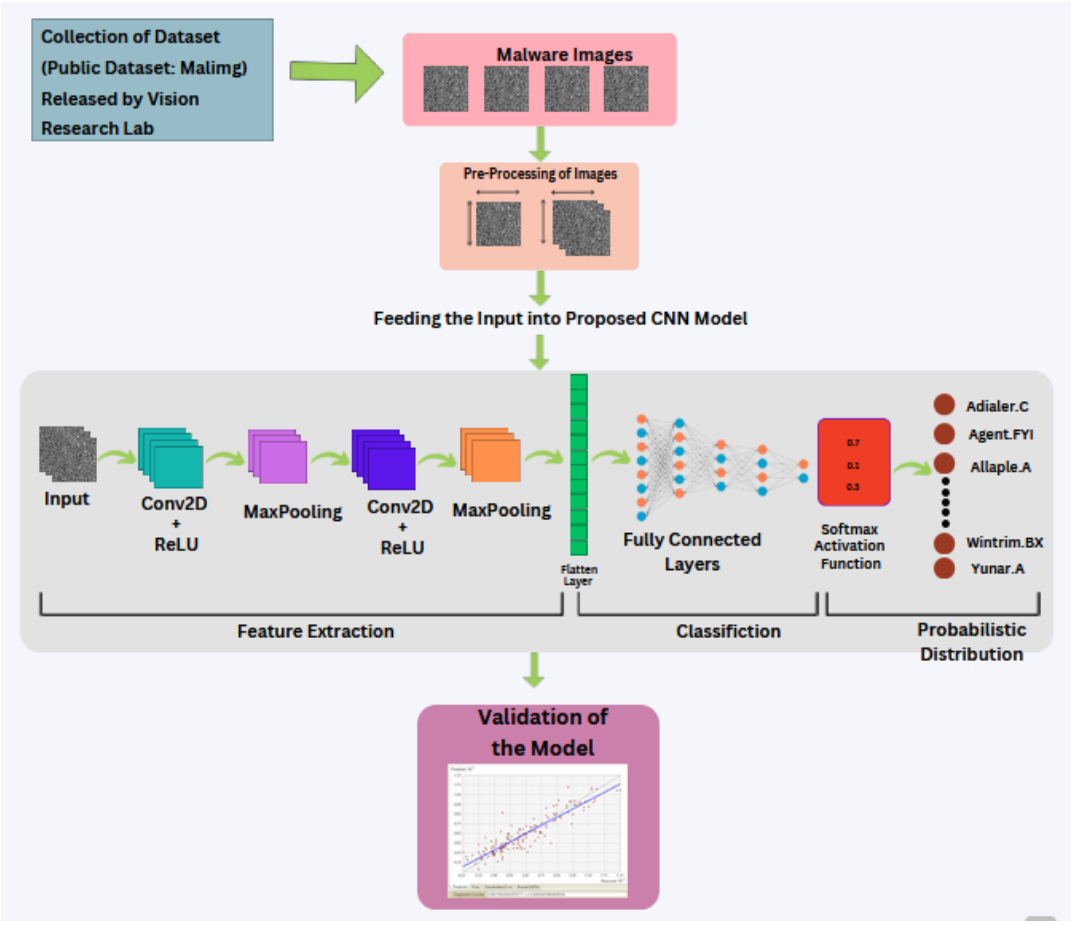

Figure 8. MalwareVsion: Malware classification framework

To evaluate the classification performance of MalwareVision, a number of common evaluation indicators have been used, including accuracy, precision, recall, F-score, and loss. The accuracy measure is defined as the rate of how many malware samples were classified correctly. Precision quantifies the ratio of correctly predicted positive instances to all predicted positives, to ensure that the model reduces false positives effectively. In addition, recall checks whether a model can identify true positives among all available cases; it tests how likely each sample is to be truly a good match (31). The F1-Score is the harmonic mean of precision and recall and combines both measures into a single metric (32). These performance measures have been given below mathematically by the formula:

$$\text{Precison} = \frac{TP}{TP+FP} \quad (1)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (2)$$

$$\text{F1} - \text{Score} = \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

where **TP** (True Positives) represents correctly classified malware samples, **TN** (True Negatives) represents correctly classified benign samples, **FP** (False Positives) are benign files incorrectly classified as malware, and **FN** (False Negatives) are malware files misclassified as benign. Additionally, the loss function used in this study is categorical cross-entropy, which measures the divergence between the predicted probability distribution and the actual distribution. It is calculated as follows:

$$\text{Loss} = -\sum_{j=1}^{N} y_j \log(y\hat{}_j)$$

where y_j represents the actual class label, $y_{\hat{}j}$ is the predicted probability for class j, and N is the total number of classes. A decrease in training and testing loss indicates improved learning and better generalization of new data.

4. Results

The training accuracy increased steadily throughout the epochs, beginning at 47.87% in the first epoch and ultimately reaching 96.82% in the last. The accuracy improved, indicating that the model learned the recurring patterns and distinguishing characteristics in the training data. Throughout the epochs, the testing accuracy was also a discernible rise. In the first epoch, it had an accuracy of 66.63%, but by the time it got to the last epoch, it had reached a high of 95.90%. The ability of the model to generalize successfully to date it has not encountered before is demonstrated by the steady improvement in testing accuracy. Figure 9 shows the training and testing accuracy of the proposed framework.

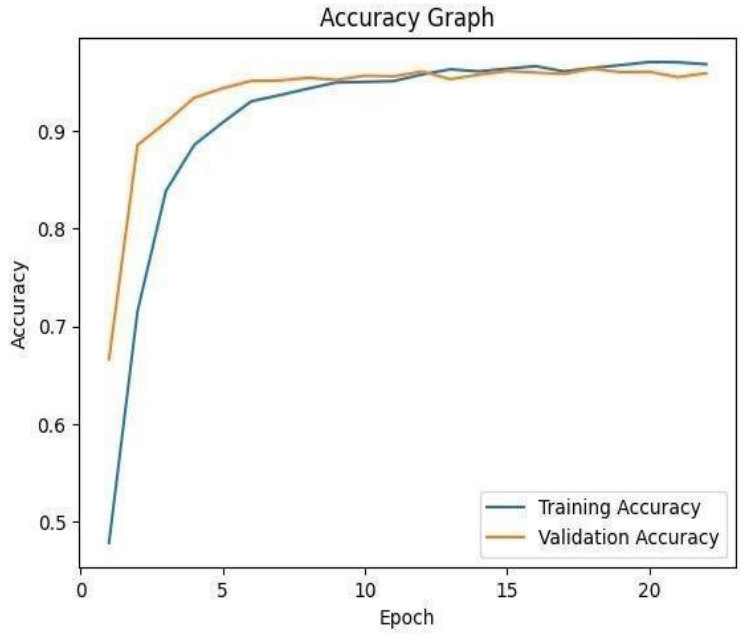

Figure 9. Training and testing accuracy

As the epochs continued, the training loss slowed until it reached its lowest point. The value began at 1.7587 and dropped down to 0.0940 in the final period. The declining trend suggests that the model successfully minimized the training loss and improved its ability to make accurate predictions by improving its ability to make accurate forecasts. Analogous to the training loss, the testing loss improved as more epochs passed. In the first epoch, it was 1.0130, and by the time the last epoch rolled around, it had dropped to 0.1636. The fact that the testing loss has decreased suggests that the model generalized effectively and produced accurate predictions based on testing data it had not seen before. Figure 10 shows an epoch-wise loss graph for the proposed framework.

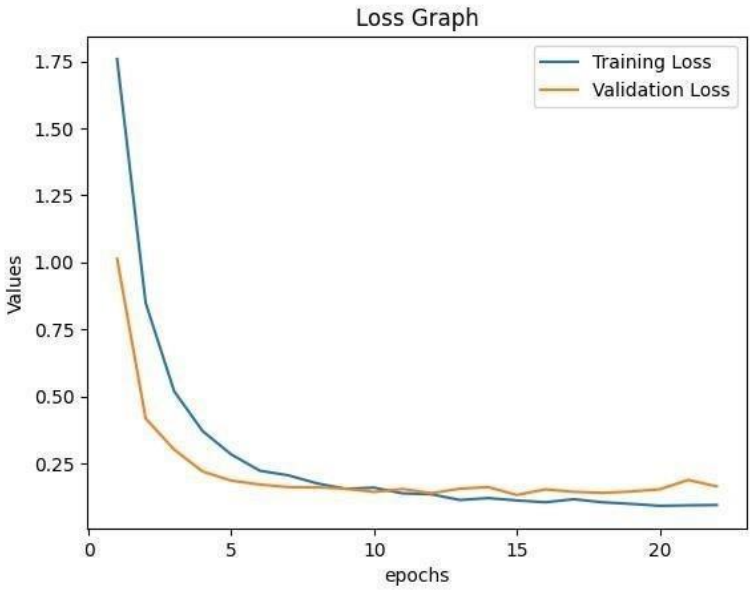

Figure 10. Epoch-wise loss graph

The results achieved from both the training and testing phases reveal that our customized CNN is efficient at learning and identifying intricate data patterns. The model effectively differentiated among diverse malware classes indicating its efficiency in multi-classification. Although it suffered from a relatively low loss, yet the optimized model showed sophisticated representations based on the training data which ultimately enabled it to produce high predictive accuracy in each epoch. Similarly, the steady increase in testing accuracy presented that the model has strong generalization ability for unseen data. In the same way, the gradual decrease in training and testing losses indicates that the model is capable of minimizing training and testing errors while capturing the underlying complex data patterns. This proficiency of the proposed optimizations in the CNN model presents that it can effectively classify malware families while demonstrating epoch-wise enhanced accuracy and decreased loss.

4.1. Class-wise testing results

This section demonstrates the performance evaluations for each malware class by reflecting precision, recall, and F1-score for the proposed Malware Vision framework. The level of precision refers to how accurate the model is when it comes to making positive predictions. In this testing situation, the precision numbers fall between 0.00 and 1.00. Classes 0, 4, 14, 17, 19, and 22 all achieved a precision score of 1, which indicates that the model did not provide any erroneous positive predictions for any of these categories. The precision ratings for the other classes ranged from 0.69 to 0.99, showing a minimal number of false positives overall. The class-wise precision score is shown in Figure 11.

The capacity of a model to correctly detect positive cases is measured by a statistic called recall, also referred to as sensitivity or the true positive rate. It is the proportion of correct diagnoses to the aggregate of correct diagnoses and erroneous negative results. Except for classes 5 and 21, the recall values for all other classes are either exactly 1.00 or very close to 1.00. This suggests that the model could accurately identify most of the positive examples of these classifications. On the other hand, the recall is 0.00 for class 5 and 0.10 for class 21, indicating that the model had difficulty accurately identifying positive cases for these classes. The class-wise recall score is presented in Figure 12.

The F1-score is calculated by taking the harmonic mean of both precision and recall, which provides a level of equilibrium between the two metrics. It is a metric that determines how accurate a model is. The F1 scores for the majority of the classes fall in the range of 0.96 to 1.00, indicating a high level of accuracy in classifying the examples. However, classes 5, 21, and 20 have considerably lower F1 scores, with values of 0.00, 0.17, and 0.64, respectively. This indicates that the model needed help correctly categorizing examples for these classes. The support indicates the total number of instances contained within each class. It varies from class to class, with possible values spanning from 18 to 900. The support values allow a better comprehension of the distribution of instances within the dataset. The class-wise F1-score is graphically presented in Figure 13.

The results of the tests indicate that the model did well in successfully classifying the majority of the categories. This is demonstrated by the high precision, recall, and F1-score values, which all point to this conclusion. Despite this, there are a few categories in which the model performed poorly, which led to a lower recall percentage and F1 score. The relevance of analyzing the model's performance on a per-class basis to identify areas that need additional improvement is brought into focus by these findings. The confusion matrix for the proposed MalwareVision framework is shown in Figure 14.

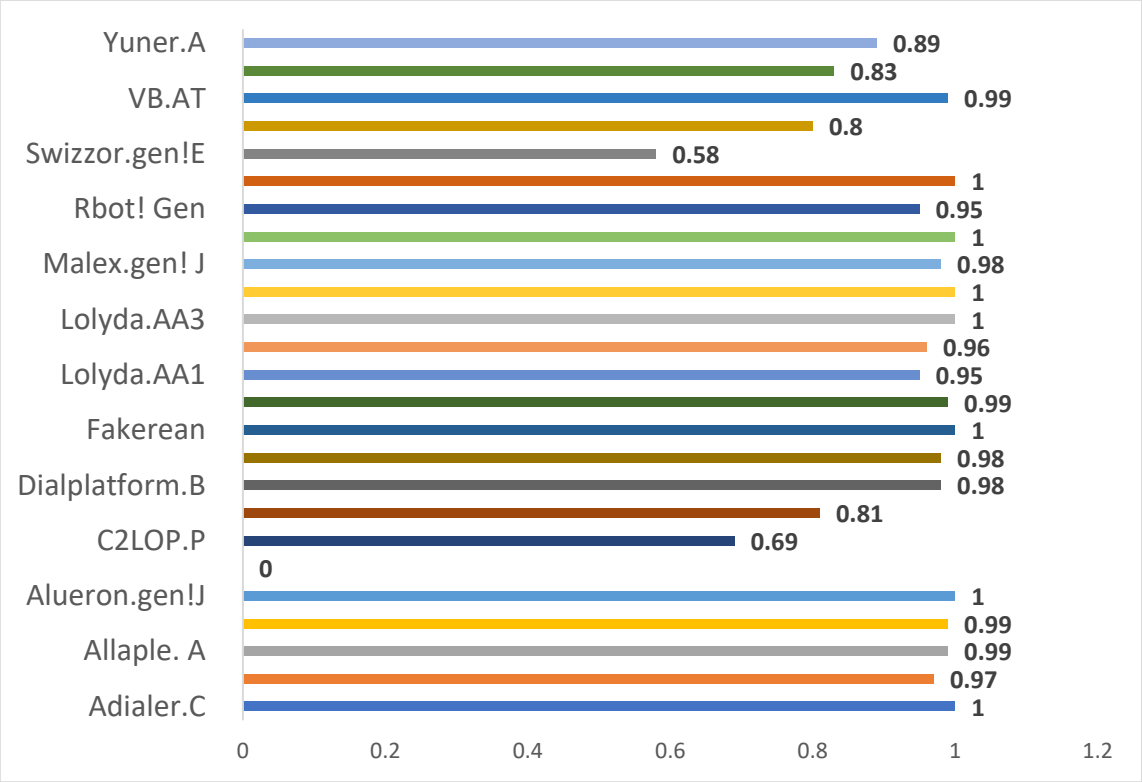

Figure 11. Class-wise precision score

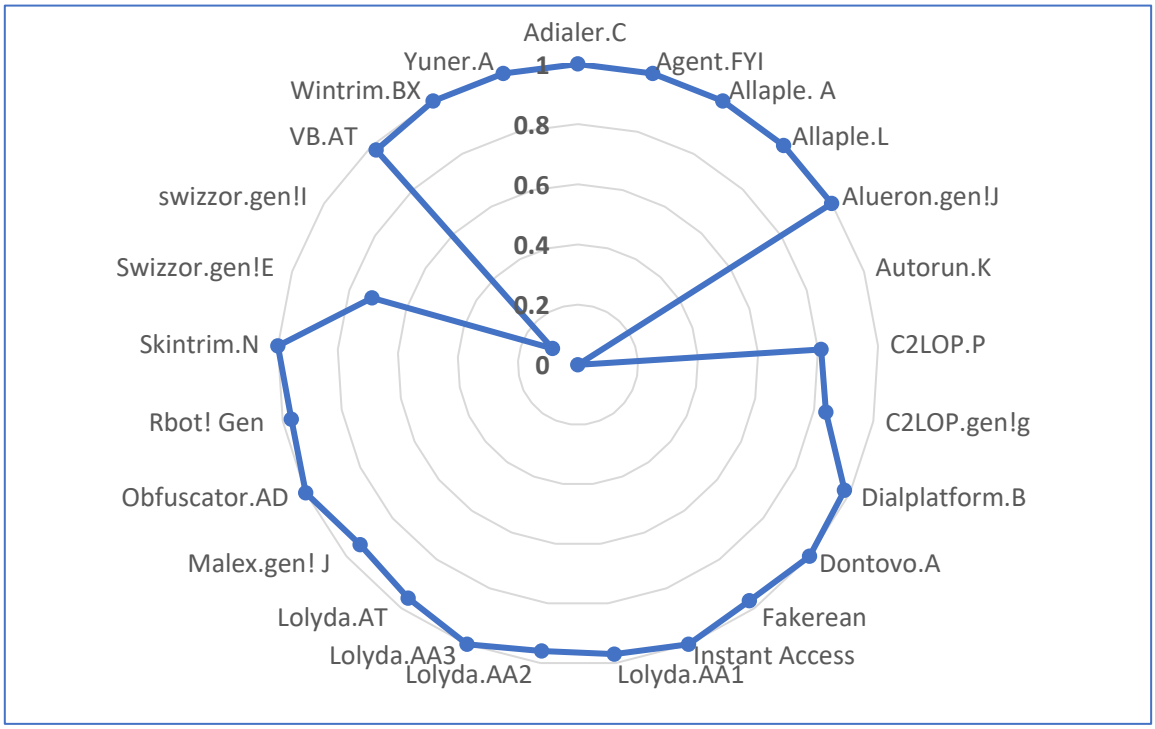

Figure 12. Class-wise recall score

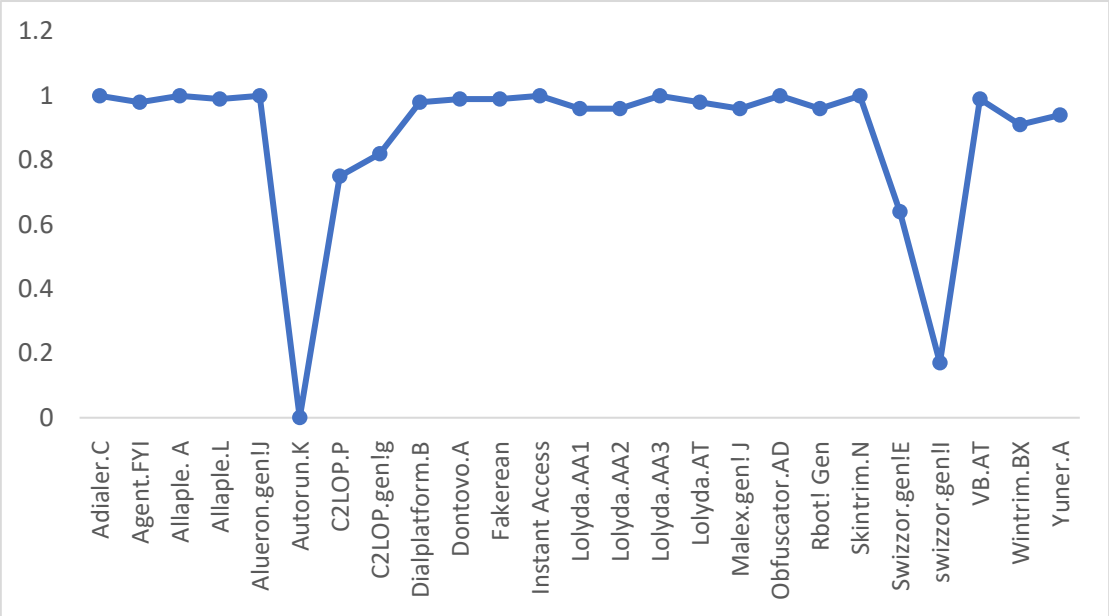

Figure 13. Class-wise F1-score

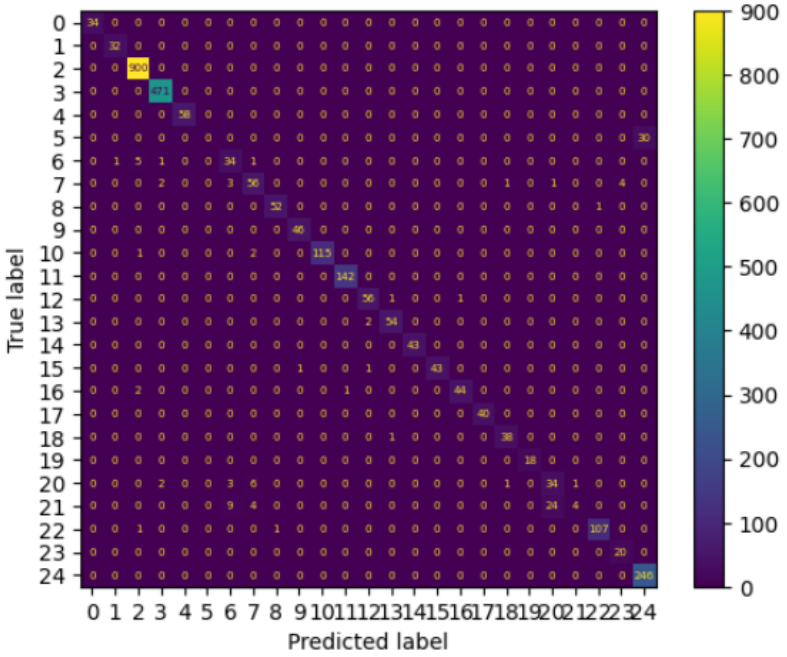

Figure 14. Confusion matrix of MalwareVision framework

The Class-wise performance of the proposed framework is shown in Table 4.

Table 4. Class-wise performance of the proposed framework

Class	precision	recall	f1- score	support
Adialer.C	1	1	1	34
Agent.FYI	0.97	1	0.98	32
Allaple. A	0.99	1	1	900
Allaple.L	0.99	1	0.99	471
Alueron.gen!J	1	1	1	58
Autorun.K	0	0	0	30
C2LOP.P	0.69	0.81	0.75	42
C2LOP.gen!g	0.81	0.84	0.82	67
Dialplatform.B	0.98	0.98	0.98	53
Dontovo.A	0.98	1	0.99	46

Fakerean	1	0.97	0.99	118
Instant Access	0.99	1	1	142
Lolyda.AA1	0.95	0.97	0.96	58
Lolyda.AA2	0.96	0.96	0.96	56
Lolyda.AA3	1	1	1	43
Lolyda.AT	1	0.96	0.98	45
Malex.gen! J	0.98	0.94	0.96	47
Obfuscator.AD	1	1	1	40
Robot! Gen	0.95	0.97	0.96	39
Skintrim.N	1	1	1	18
Swizzor.gen!E	0.58	0.72	0.64	47
swizzor.gen!I	0.8	0.1	0.17	41
VB.AT	0.99	0.98	0.99	109
Wintrim.BX	0.83	1	0.91	20
Yuner.A	0.89	1	0.94	246

The diagonal dominance in the matrix indicates that the model correctly classifies most malware families, as the highest values (highlighted in yellow and green) are concentrated along the diagonal. This suggests that the CNN-based model is effective in distinguishing different malware families with a high degree of accuracy. However, some off-diagonal values indicate misclassifications, where certain malware families are incorrectly classified as others. These misclassifications could be due to similarities between certain malware families, leading to feature overlap suggesting that further refinement in feature extraction may be necessary to enhance classification accuracy.

4.2. Comparative analysis

To prove the effectiveness of the proposed framework, we have conducted a comparative analysis that presents improved predictive accuracy of MalwareVision against state-of-the-art techniques. We have considered the latest articles published from 2020 to 2024 with a focus on malware classification. Table 5 reflects the enhanced accuracy of advanced deep learning-based MalwareVision compared to modern techniques as shown in Table 5.

Table 5. Comparative analysis of MalwareVision with modern techniques

Reference	Technique	Highest accuracy %	Dataset	Malware families
(33)	Complex CNN Model (Multiple Layers)	88	*Malware Behavioral Dataset* (Custom)	2
(34)	Code-Aware Data Generation with CNN	90	VirusTotal malware dataset	**3**
(35)	Random Forest	87.30	CIC-MalMem-2022	**4**
(36)	LightGBM	94	CICMalDroid 2020	**5**
(37)	Neural Networks	94.6	Custom	**10**
(38)	Random Forest	87.30	Curated Memory Analysis	**4**
(39)	Extra Tree	83.05	CIC-MalMem-2022	**4**
(40)	dilated convolutional network	83.53	CIC-MalMem-2022	**2**
This study	Optimized CNN	95.9	Malimg	**25**

The accuracy-based comparison of MalwareVision against modern techniques is presented in Figure 15.

4.3. Limitations

While the proposed DL-based malware classification framework achieved high accuracy, it has several limitations. The model's performance is affected by class imbalance, as certain malware families had significantly fewer training samples, leading to overfitting along with lower recall and F1-scores for those categories (41). Additionally, the reliance on grayscale image transformation may not capture all behavioral aspects of malware, limiting its effectiveness against highly obfuscated or polymorphic

malware (42). Moreover, the computational complexity of DL models, particularly CNNs, requires substantial processing power and memory, making real-time deployment in resource-constrained environments challenging (43). Future research should focus on integrating hybrid approaches that combine static and dynamic analysis while also exploring methods to improve class balance and generalization across diverse malware datasets.

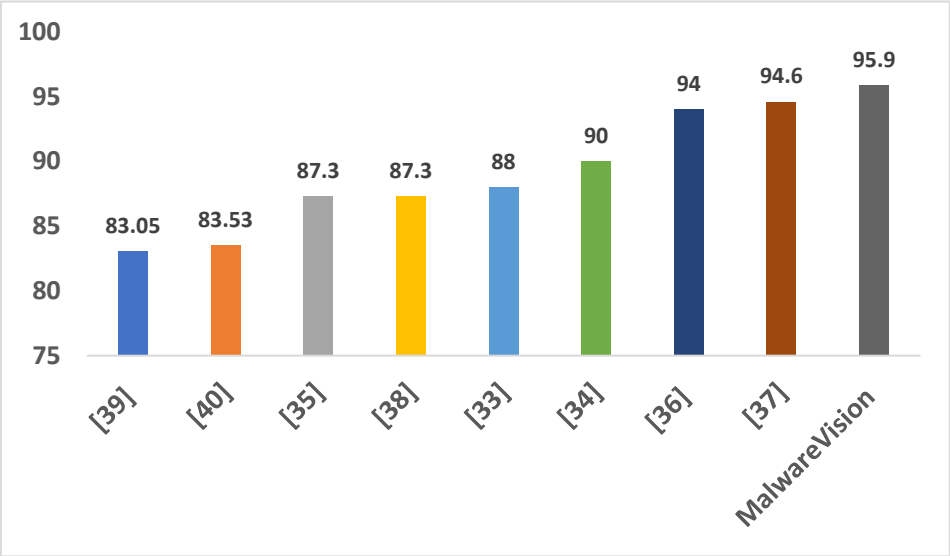

Figure 15. Accuracy-based comparison of MalwareVision against modern technique

5. Conclusions

This research harnessed advanced deep-learning methods to detect malware, yielding promising results. By extensively training and testing on a large collection of malware samples, we gained valuable insights. Our custom deep-learning model displayed impressive growth in both the training and testing phases. During training, accuracy steadily improved from 47.87% to an impressive 96.82%, and testing accuracy also rose from 66.63% to 95.90%. This steady progress shows that the model can perform generalization well to new or unseen data. Notably, training and testing losses steadily fell over time, reflecting the successful detection of specific patterns and precise predictions. Class-specific testing showed that the model worked well for most malware categories. These findings emphasize the importance of class-wise framework evaluation while highlighting areas needing attention in the future. Overall, our proposed DL-based MalwaereVision framework demonstrated promising results while making effective predictions. Future research should focus on developing cross-platform models that are able to detect and classify malware in different environments such as Windows, Linux, and Android, thereby ensuring a more robust security framework.

Funding: This research received no external funding. The APC was funded privately by one of the authors.
Data Availability Statement: L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: visualization and automatic classification," in *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, Pittsburgh Pennsylvania USA: ACM, Jul. 2011, pp. 1–7. doi: 10.1145/2016904.2016908.
Acknowledgments: Conceptualization, Aamir Ali and Malik Arslan Akram; Methodology, Wajiha Farooq and Aown Muhammad; Software and Implementation, Moomna Nazir and Tehseen Mazhar; Validation and Formal Analysis, Malik Arslan Akram and Aown Muhammad; Investigation and Data Curation, Wajiha Farooq and Moomna Nazir; Writing—Original Draft Preparation, Aamir Ali and Tehseen Mazhar; Writing—Review and Editing, Misbah Ali and Malik Arslan Akram; Visualization, Moomna Nazir; Supervision, Misbah Ali; Project Administration, Misbah Ali; Funding Acquisition, Misbah Ali.
All authors have read and agreed to the published version of the manuscript.
Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gaber MG, Ahmed M, Janicke H. Malware Detection with Artificial Intelligence: A Systematic Literature Review. ACM Comput Surv. 2024 Jun 30;56(6):1–33.
2. Smmarwar SK, Gupta GP, Kumar S. Android malware detection and identification frameworks by leveraging the machine and deep learning techniques: A comprehensive review. Telemat Inform Rep. 2024;100130.
3. Kalla D, Mohammed AS, Boddapati VN, Jiwani N, Kiruthiga T. Investigating the Impact of Heuristic Algorithms on Cyberthreat Detection. In: 2024 2nd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT) [Internet]. IEEE; 2024 [cited 2025 Mar 20]. p. 450–5. Available from: https://www.researchgate.net/profile/Abdul-Sajid-Mohammed/publication/389819871_Investigating_the_Impact_of_Heuristic_Algorithms_on_Cyberthreat_Detection/links/67d371462719090652b97396/Investigating-the-Impact-of-Heuristic-Algorithms-on-Cyberthreat-Detection.pdf
4. Abiola AM, Marhusin MF. Signature-Based Malware Detection Using Sequences of N-grams. Int J Eng Technol. 2018 Oct 7;7(4.15):120–5.
5. Al Sadawi A, Hassan MS, Ndiaye M. A survey on the integration of blockchain with IoT to enhance performance and eliminate challenges. IEEE Access. 2021;9:54478–97.
6. Pradhan B, Bhattacharyya S, Pal K. IoT-Based Applications in Healthcare Devices. Fareed MMS, editor. J Healthc Eng. 2021 Mar 18;2021:1–18.
7. Bui HT, Aboutorab H, Mahboubi A, Gao Y, Sultan NH, Chauhan A, et al. Agriculture 4.0 and Beyond: Evaluating cyber threat intelligence sources and techniques in smart farming ecosystems. Comput Secur. 2024;140:103754.
8. 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 Tutor. 2023;25(3):1748–74.
9. Sarker IH. Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning Perspective. SN Comput Sci. 2021 May;2(3):154.
10. Akhtar MS, Feng T. Detection of malware by deep learning as CNN-LSTM machine learning techniques in real-time. Symmetry. 2022;14(11):2308.
11. Ullah F, Ullah S, Naeem MR, Mostarda L, Rho S, Cheng X. Cyber-threat detection system using a hybrid approach of transfer learning and multi-model image representation. Sensors. 2022;22(15):5883.
12. Schmidt AD, Bye R, Schmidt HG, Clausen J, Kiraz O, Yuksel KA, et al. Static Analysis of Executables for Collaborative Malware Detection on Android. In: 2009 IEEE International Conference on Communications [Internet]. 2009 [cited 2025 Mar 3]. p. 1–5. Available from: https://ieeexplore.ieee.org/abstract/document/5199486
13. Iwamoto K, Wasaki K. Malware classification based on extracted API sequences using static analysis. In: Proceedings of the 8th Asian Internet Engineering Conference [Internet]. New York, NY, USA: Association for Computing Machinery; 2012 [cited 2025 Mar 2]. p. 31–8. (AINTEC '12). Available from: https://doi.org/10.1145/2402599.2402604
14. Sharif M, Yegneswaran V, Saidi H, Porras P, Lee W. Eureka: A Framework for Enabling Static Malware Analysis. In: Jajodia S, Lopez J, editors. Computer Security - ESORICS 2008. Berlin, Heidelberg: Springer; 2008. p. 481–500.
15. Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. J Netw Comput Appl. 2020 Mar 1;153:102526.
16. Azeez NA, Odufuwa OE, Misra S, Oluranti J, Damaševičius R. Windows PE Malware Detection Using Ensemble Learning. Informatics. 2021 Mar;8(1):10.
17. Burnap P, French R, Turner F, Jones K. Malware classification using self-organising feature maps and machine activity data. Comput Secur. 2018 Mar 1;73:399–410.

18. Haddadpajouh H, Azmoodeh A, Dehghantanha A, Parizi RM. MVFCC: A Multi-View Fuzzy Consensus Clustering Model for Malware Threat Attribution. IEEE Access. 2020;8:139188–98.
19. Sahoo D. Cyber Threat Attribution with Multi-View Heuristic Analysis. In: Choo KKR, Dehghantanha A, editors. Handbook of Big Data Analytics and Forensics [Internet]. Cham: Springer International Publishing; 2022 [cited 2025 Mar 3]. p. 53–73. Available from: https://doi.org/10.1007/978-3-030-74753-4_4
20. Wasif MS, Miah MP, Hossain MS, Alenazi MJ, Atiquzzaman M. CNN-ViT synergy: An efficient Android malware detection approach through deep learning. Comput Electr Eng. 2025;123:110039.
21. Bakır H. A new method for tuning the CNN pre-trained models as a feature extractor for malware detection. Pattern Anal Appl. 2025 Mar;28(1):26.
22. Chaganti R, Ravi V, Pham TD. A multi-view feature fusion approach for effective malware classification using Deep Learning. J Inf Secur Appl. 2023 Feb 1;72:103402.
23. Shaheen F, Verma B, Asafuddoula M. Impact of automatic feature extraction in deep learning architecture [Internet]. CQUniversity; 2016 [cited 2025 Mar 3]. Available from: https://acquire.cqu.edu.au/articles/conference_contribution/Impact_of_automatic_feature_extraction_in_deep_learning_architecture/13440848/1
24. Dept. of CSE, Rajarambapu Institute of Technology, Rajaramnagar, Sangli, Maharashtra, India, Lad SS, Adamuthe AC. Malware Classification with Improved Convolutional Neural Network Model. Int J Comput Netw Inf Secur. 2021 Dec 8;12(6):30–43.
25. Abusitta A, Li MQ, Fung BCM. Malware classification and composition analysis: A survey of recent developments. J Inf Secur Appl. 2021 Jun 1;59:102828.
26. A Comprehensive Review on Malware Detection Approaches [Internet]. [cited 2025 Mar 3]. Available from: https://ieeexplore.ieee.org/abstract/document/8949524
27. Nataraj L, Karthikeyan S, Jacob G, Manjunath BS. Malware images: visualization and automatic classification. In: Proceedings of the 8th International Symposium on Visualization for Cyber Security [Internet]. Pittsburgh Pennsylvania USA: ACM; 2011 [cited 2025 Mar 20]. p. 1–7. Available from: https://dl.acm.org/doi/10.1145/2016904.2016908
28. Ali M, Mazhar T, Arif Y, Al-Otaibi S, Ghadi YY, Shahzad T, et al. Software defect prediction using an intelligent ensemble-based model. IEEE Access [Internet]. 2024 [cited 2025 Jan 18]; Available from: https://ieeexplore.ieee.org/abstract/document/10413365/
29. Ali M, Mazhar T, Shahzad T, Ghadi YY, Mohsin SM, Akber SMA, et al. Analysis of feature selection methods in software defect prediction models. IEEE Access [Internet]. 2023 [cited 2025 Jan 18]; Available from: https://ieeexplore.ieee.org/abstract/document/10359521/
30. Ali M, Mazhar T, Al-Rasheed A, Shahzad T, Ghadi YY, Khan MA. Enhancing software defect prediction: a framework with improved feature selection and ensemble machine learning. PeerJ Comput Sci. 2024;10:e1860.
31. Ali M. Optimizing Software Defect Prediction: A Genetic Algorithm Based Comparative Analysis. In: International. p. 56–73.
32. Ali M, Azam MS, Shahzad T. Random Search-Based Parameter Optimization on Binary Classifiers for Software Defect Prediction. J Ilm Tek Elektro Komput Dan Inform JITEKI. 2024;10(2):476–88.
33. Alhamedi N, Dongshik K. Detecting Malware on Windows OS Using AI Classification of Extracted Behavioral Features from Images. Int J Adv Comput Sci Appl Ijacsa [Internet]. 2024 Jul 30 [cited 2025 Mar 17];15(8). Available from: https://thesai.org/Publications/ViewPaper?Volume=15&Issue=8&Code=ijacsa&SerialNo=129
34. Athira, Baburaj D, Gupta D. A Comprehensive Exploration of Machine Learning and Explainable AI Techniques for Malware Classification. In: 2024 2nd World Conference on Communication & Computing (WCONF) [Internet]. 2024 [cited 2025 Mar 17]. p. 1–7. Available from: https://ieeexplore.ieee.org/document/10692299

35. Hafiz MdFB, Khan NA, Kamal Z, Hossain S, Barman S. A Robust Malware Classification Approach Leveraging Explainable AI. In: 2024 International Conference on Intelligent Systems for Cybersecurity (ISCS) [Internet]. 2024 [cited 2025 Mar 17]. p. 1–6. Available from: https://ieeexplore.ieee.org/document/10581382
36. Baghirov E. A comprehensive investigation into robust malware detection with explainable AI. Cyber Security Appl. 2025 Dec;3:100072.
37. Zhao Q, Wei X, Dong C, Jin B, Yu Z, Gao F, et al. Malware Detection and Analysis based on AI Algorithm. In: 2024 International Conference on Distributed Computing and Optimization Techniques (ICDCOT) [Internet]. 2024 [cited 2025 Mar 17]. p. 1–6. Available from: https://ieeexplore.ieee.org/document/10515944
38. Ch R, Manoranjini J, Pallavi S, Naresh U, Telang S, Kiran S. Advancing Malware Detection Using Memory Analysis and Explainable AI Approach. In: 2024 Second International Conference on Intelligent Cyber-Physical Systems and Internet of Things (ICoICI) [Internet]. 2024 [cited 2025 Mar 17]. p. 518–23. Available from: https://ieeexplore.ieee.org/document/10696406
39. Roy KS, Ahmed T, Udas PB, Karim MdE, Majumdar S. MalHyStack: A hybrid stacked ensemble learning framework with feature engineering schemes for obfuscated malware analysis. Intell Syst Appl. 2023 Nov;20:200283.
40. Mezina A, Burget R. Obfuscated malware detection using dilated convolutional network. In: 2022 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT) [Internet]. Valencia, Spain: IEEE; 2022 [cited 2025 Mar 19]. p. 110–5. Available from: https://ieeexplore.ieee.org/document/9943443/
41. Almajed H, Alsaqer A, Frikha M. Imbalance Datasets in Malware Detection: A Review of Current Solutions and Future Directions. Int J Adv Comput Sci Appl [Internet]. 2025 [cited 2025 Mar 21];16(1). Available from: https://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=2158107X&AN=182970604&h=16BtsMzdAvH1j7TnfbLp0YMhlNCEeFy0m5NokCtui64ucd3yGpj9zxi%2FAksQuxNjhxgQQ4wlsqe6ZfxtiIg7zA%3D%3D&crl=c
42. Johny JA, Asmitha KA, Vinod P, Radhamani G, Rehiman KAR, Conti M. Deep learning fusion for effective malware detection: leveraging visual features. Clust Comput. 2025 Apr;28(2):135.
43. Karat G, Kannimoola JM, Nair N, Vazhayil A, VG S, Poornachandran P. CNN-LSTM hybrid model for enhanced malware analysis and detection. Procedia Comput Sci. 2024;233:492–503.