Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

Research Article

Image-Based Malware Detection Using Deep CNN Models

[1]Hawraa.O. Musa
Information Institute for Postgraduate Studies
Iraqi Commission for Computers and Informatics
Baghdad, Iraq
ms202310749@iips.edu.iq

[2]Muhanad.T. Younis
Department of Computer Science
College of Science
Mustansiriya University
Baghdad, Iraq
mty@uomustansiriyah.edu.iq

ARTICLE INFO
Article History
Received: 06/01/2025
Accepted: 20/02/2025
Published: 05/6/2025
This is an open-access article under the CC BY 4.0 license:
http://creativecommons.org/licenses/by/4.0/

ABSTRACT
Malware or malicious software represents one of the most remarkable threats to cybersecurity, as it compromises the integrity, confidentiality, and availability of computer systems and networks. Traditional malware detection methodologies frequently prove inadequate in identifying innovative and sophisticated malware variants. Deep learning (DL) presents a promising strategy for malware detection by utilizing advanced algorithms that are capable of discerning intricate patterns from extensive datasets. This study presents a model based on deep learning with Convolutional Neural Network (CNN) for malware classification. This research utilized the Malimg dataset, which includes 9,339 malware samples from 25 distinct families. The approach requires resizing malware images to a resolution of 64 x 64 pixels and normalizing these images for model training. The selection of a 64 × 64 size frame reduces network complexity while speeding up training without sacrificing important information. The architecture of the proposed CNN primarily consists of more than one convolutional layer, max-pooling, dropout to mitigate overfitting problem, fully connected layers for achieving better classification results. The proposed model established an impressive accuracy of 96%. For model evaluation, the following measures of accuracy were used: precision, recall, F1-score, and accuracy. This research shows that CNN-based methods can have a high level of effectiveness in detecting obfuscated malware.

Keywords: *CNN; malware detection; Cyber security; Malimg dataset; Deep Learning.*

1. INTRODUCTION

Malware is one of the most powerful threats to computer security. Given the rapid development of malware, methods using signatures to identify malware become ineffective. Consequently, a focus has been observed regarding organizing and utilizing machine learning, especially deep learning for malware detection [1]. Research concerning the classification of malware indicates that malware samples generally belong to a family characterized by shared behavioral traits; that is, a majority of novel malware are derivatives of previously identified variants. Therefore, the potential to develop a mechanism for categorizing malware by its family, regardless of whether it is a variant, seems especially beneficial and is a way to address the growth of the malware population [2]. Many of the conventional structural antimalware programs require sign-based method and cannot easily handle complex and unique malwares. These limitations call for novel and other efficient approaches to identify patterns, thereby prompting our proposal to use deep learning. Previous detection schemes characteristic of antivirus software that performs recognition based on signatures experience difficulty detecting morphed or new types of malwares [3].

Several works distinguish the malware samples from the benign ones by extracting the features from the datasets before using them as input for certain machine learning or deep learning models, with quite excellent accuracy in their Classification results. However, the machine learning and deep learning models require a large amount of diversified data with the exact labels to assure reliability [4]. This research targets the relationship that exists

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

between data sample volume and processing operation speed. Model performance benefits from larger datasets because it provides better data representations that lead to improved generalization ability. Such improvements require additional training duration. Through such limitations, the use of machine learning (ML) and deep learning (DL) approaches have been deemed competent in the analysis of pattern and data generalization [5]. Deep learning models have shown a huge potential in numerous areas such as picture recognition text and speech analysis, Deep learning particularly Convolutional Neural Networks (CNNs). CNNs are effective in image and signal processing. In recent years, several CNN-based models for malware detection have been proposed. Recent advancements in deep learning techniques added effectiveness to text recognition and language translation process alongside sentiment examination and made possible spoken language understanding. Smart assistant technologies along with human-machine interaction have undergone profound enhancements owing to this advancement [6]. Deep learning occurs through effective pattern detection in large datasets because multi-layer artificial neural networks enable this capability. CNNs handle image work and signal processing, while Recurrent Neural Networks work with text and speech processing in combination with Transformers.

This work considers the application of CNNs to classify samples of malware belonging to the Malimg dataset, which is used in many studies and includes grayscale images of malware binaries of 25 families. The proposed approach will employ deep learning to enhance the classification of malware samples and overcome the issue of obfuscation used in modern malware. The methodology followed in this study entails the preprocessing of the dataset to centralize the image sizes and normalize pixel intensity. Meanwhile, the CNN model is trained with a sequential training methodology. The main contributions of this study are as follows:

- This work presents an efficient framework of Machine Learning based on CNNs to analyze Malimg dataset malware samples while sorting 25 different malware families based on their binary grayscale images.
- The use of methods: to optimize the model, the methodology includes two preprocessing steps that resize malware images to 64 × 64 pixels resolution and normalize pixel intensity values.
- The study conducts a CNN structure and incorporates numerous convolutional layers combined with max-pooling mechanisms together with dropout layers to fight overfitting patterns and full-connection layers for labeling improvement.
- The model demonstrates successful malware classification by reaching an accuracy level of 96%. This finding can be used as reference when using deep learning in the analysis of malware images in the future.

This article is arranged as follows: Section 2 reviews related works; while Section 3 details the proposed methodology, involving data preprocessing, model architecture, and evaluation metrics. Section 4 discusses the experimental results, and Section 5 concludes the paper with future research directions.

2. RELATED WORK

Mesut G. (2024) [7] focused on static analysis, which is performed without executing code that would transform the binaries into image formats and is examined with Deep Learning models. Different approaches were used including CNNs, Autoencoder models, and transfer learning models such as VGG16, ResNet50, and MobileNet. These models showed high capabilities of malware detection with the help of one of the largest datasets in this field. Hence, real testing was performed across various datasets, namely, Trapmine, Malavi's, and Malimg to provide strong validation to the models. The CNN models had an overall accuracy of approximately 80%; as such, it is good for detecting general patterns of malware. The autoencoder model was presented to be superior with an accuracy of 90%. While comparing classification capability, VGG16 was seen to be the most accurate in both groups which achieved 91.25% accuracy. Thus, the models based on transfer learning (VGG16 and MobileNet) had high accuracy. This finding implies the power of transferred learning, indicating the importance of balancing data and developing better performance techniques.

Moreover, MD. HARIS et al. (2023) [8] focused on the classification of malware, employing image-based deep learning techniques. This study identifies deficiencies such as the absence of high-caliber datasets and encounters

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

obstacles encompassing data scarcity and quality, overfitting, and computational intricacy. This study uses the Malimg dataset for their analysis. Their approach comprised preprocessing the images through resizing and normalization, subsequently followed by the training of diverse deep learning architectures. The architectures encompassed CNNs, Caps-Net, VGG16, ResNet50, and InceptionV3. The investigation assessed the efficacy of each individual model and proposed an ensemble architecture that amalgamated the predictions from these distinct models. The findings indicated the ensemble model's superior efficacy, attaining an approximate accuracy of 92.30% when compared with the individual models.

Hammad et al. (2022) [9] focused on issues such as research gaps and challenges, including data imbalance, feature extraction complexity, inconsistent classifier performance, and unbalanced datasets. The proposed methodology employs a five-step protocol: transforming malware binaries into two-dimensional grayscale images, normalizing them for input into CNNs, deriving features through both handcrafted techniques (Tamura) and deep learning architectures (Google Net), and executing classification utilizing K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Extreme Learning Machine (ELM). The methodology is evaluated on the Malimg dataset, demonstrating high accuracy in the direct handling of unbalanced data, surpassing alternative techniques with classification performance rates of 95.42% (Tamura + ELM) and 96.84% (Google Net + KNN).

Zhong et al. (2023) [10] dealt with the issues related to the classification of malware with an innovative tool VisMal, which solves the limitations of the static and dynamic approaches. The VisMal framework utilized for the classification of malware encounters constraints that include knowledge deficiencies among security experts, substantial computational demands, and intricate feature extraction processes. The proposed model transforms malware binaries into grayscale images and applies a Contrast-Limited Adaptive Histogram Equalization (CLAHE) algorithm to enhance local contrast and improve feature recognition. Classification is executed utilizing CNN. The methodology was evaluated on the Malimg dataset. The proposed framework demonstrated high accuracy and outperformed existing methods in both precision and processing efficiency.

Abien M. (2019) [11] performed research which utilized hybrid deep learning paradigms combined with SVMs to classify malware families. Researchers focused on malware classification by studying three essential obstacles involving computational speed and model designs alongside requirements for advanced research on deep learning methods and training approaches. The study utilized the Malimg dataset. The researchers devised the following three hybrid architectures: CNN-SVM: T, GRU-SVM, and MLP-SV. Among the proposed models, GRU-SVM attained the highest test accuracy of 84.92%. This research elucidates the potential of amalgamating deep learning methodologies with SVM techniques, particularly in fields necessitating elevated generalization capabilities, such as malware classification.

Al-Fawa'reh et al. (2021) [12] introduced an innovative methodology for the detection of Android malware by transforming APK binaries into grayscale images, subsequently classifying them using CNN. Similar to methods focusing on specific sections of APK files, this approach analyzes the entire APK, ensuring comprehensive detection. This research focuses on mobile malware detection difficulties due to restricted datasets and extraction challenges. The proposed methodology underwent evaluation on the CICMalDroid 2020 dataset, encompassing both balanced (10,878 benign and 10,878 malicious files) and unbalanced datasets (578 benign and 10,878 malicious files). This proposed CNN model was evaluated by comparing the results from testing the balanced and the unbalanced datasets. It returned 74% accuracy on the balanced data and 95.9% on the unbalanced testing data. Moreover, through the use of transfer learning, the efficiency of training improved further owing to the avoidance of first impressions.

Similarly, Tran et al. (2024) [5] examined profound issues that CNNs encounter when deployed in malware detection such as dataset imbalance, Dying ReLU problem, model convergence problems, and the necessity of diverse constantly updated datasets. They proposed a CNN-based methodology for malware detection and classification using the MaleVis dataset. It involved transforming the malware binaries into gray-scale and RGB images which enlarge an all-encompassing picture-based analysis. The study utilized three CNN architectures: parts of the models that we will be using include AlexNet, VGG-16, and ResNet-50; we will ensure that the models apply various techniques such as Batch Normalization to overcome problems such as Dying ReLU in handling Class Balance Loss to handle imbalanced datasets. The MaleVis dataset included 11,266 malware samples from 25 families and 1,482 goodware samples, ensuring a balanced evaluation. The experiment showed that AlexNet is better at grayscale images with an accuracy of 89.17%, while ResNet-50 worked stably and efficiently with RGB images with an accuracy of 86.12%. The result of the study shows that CNNs based on the images improve the efficiency as well as accuracy in detecting a malware.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

The issues in malware identification stem from signature-based system restrictions and expensive deep learning model operation with the requirement for effective algorithms. However, Kim et al. (2022) [13] proposed MAPAS, a practical Android malware detection system. MAPAS utilizes CNNs to identify high-weight features of API call graphs, which represent malicious behaviors. These features are then used with a lightweight classifier based on Jaccard Similarity for malware detection. MAPAS was evaluated using datasets from Virus Share, Android Malware Dataset (AMD), and Google Play Store, consisting of 18,000 samples for training and 25,692 for testing. The system achieved an accuracy of 93.2% in malware detection and 91% in detecting unknown malware.

3. METHODOLOGY

This section discusses the methodology through the following (Dataset, Pre-processing, Model Architecture, and Evaluation Metrics). Fig 1 illustrates the block diagram of the proposed system in which the first dataset is obtained from the Malimg dataset repository, which includes over 25 family types of malwares. Initially, the data are fed into the pre-processing stage where the images are rescaled to 64*64 resolution. Then, the images are normalized to enhance the learning rate and increase the running efficiency. Afterwards, the pre-processed image will again go to the CNN classifier for proper classification of the family.

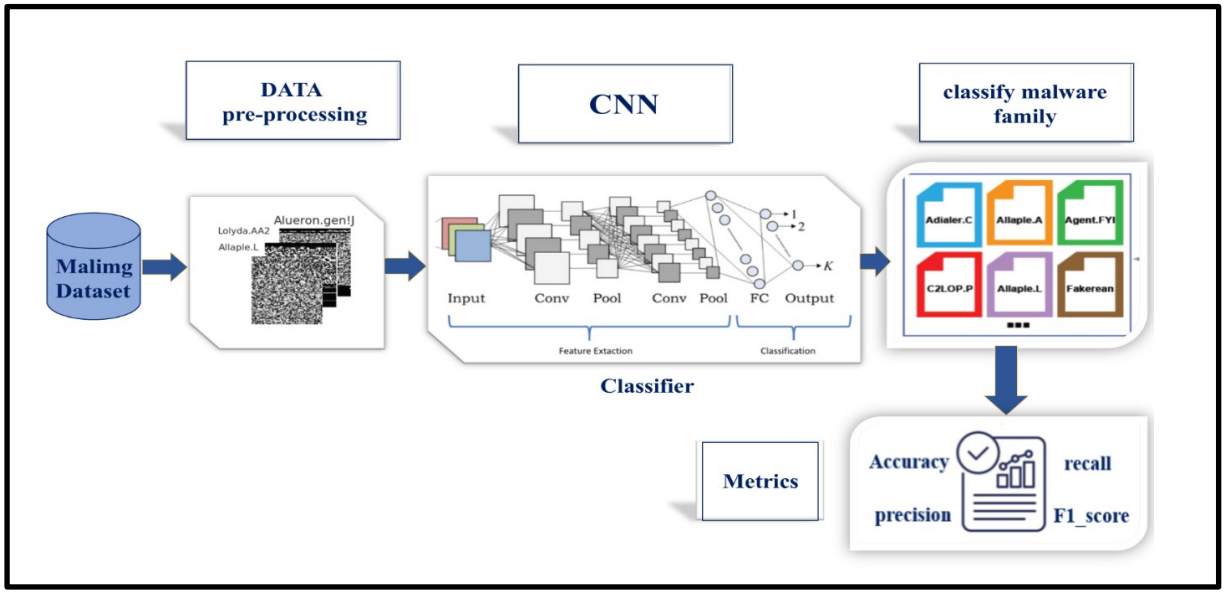

Fig 1. Block Diagram of the Proposed System

3.1 Dataset

The models in this study were assessed using the Malimg dataset containing 9,339 malware samples belonging to 25 families of malware. The dataset size is relatively suitable for training a CNN because such networks typically need extensive and diverse datasets to reach their optimal generalization capabilities. TABLE I displays the number of occurrences of each family and its variants in the Malimg dataset [14]. The distribution of classes within the dataset is imbalanced, the most populous class (“Allaple. A”) comprises 2,949 instances; whereas the least populous class (“Allaple. B”) constitutes merely 80 instances. The Malimg collection encapsulates malware samples in the form of grayscale imagery, with pixel intensity values that extend from 0 (representing black) to 255 (indicating white).

TABLE I. Malware families found in the Malimg Dataset

#	Class	Family	#
1	Worm	Allaple.L	1591
2	Worm	Allaple.A	2949
3	Worm	Yuner.A	800

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

4	PWS	Lolyda.AA 1	213
5	PWS	Lolyda.AA 2	184
6	PWS	Lolyda.AA 3	123
7	Trojan	C2Lop.P	146
8	Trojan	C2Lop.gen! g	200
9	Dialer	Instantaccess	431
10	TDownloader	Swizzot.gen! I	132
11	TDownloader	Swizzor.gen! E	128
12	Worm	VB.AT	408
13	Rogue	Fakerean	381
14	Trojan	Alueron.gen! J	198
15	Trojan	Malex.gen! J	136
16	PWS	Lolyda.AT	159
17	Dialer	Adialer.C	125
18	TDownloader	Wintrim.BX	97
19	Dialer	Dialplatform.B	177
20	TDownloader	Dontovo.A	162
21	TDownloader	Obfuscator.AD	142
22	Backdoor	Agent.FYI	116
23	Worm: AutoIT	Autorun. K	106
24	Backdoor	Rbot! gen	158
25	Trojan	Skintrim. N	80

3.2 Data pre-processing

The preprocessing of data represents an essential technique which strengthens data quality while improving model performance and optimizing training when the input data and structure are normalized.

1. Data Normalization and resizing

In Malimg dataset, image preprocessing starts with normalization that scales pixel values between 0 to 1 and is followed by resizing to 64 × 64 pixels before converting all images into 7,459 files across 25 classes. Fig 2 shows the image in size 64 x 64. Normalization was used as part of the data preparation for image processing before inputting them into the deep learning model. This process was achieved using rescale = 1./255 within the ImageDataGenerator object. The data normalization utilized in the current work is (**Min-Max Normalization**) using the following equation:

$$\frac{x - x_{min}}{x_{max} - x_{min}} = x', \quad (1)$$

where

x' is normalized pixel value, and the value ranges from 0 to 1;

x denotes the original pixel values spanning from 0 to 255 pixels;

$x\ min$ is the **minimum value** in the dataset; and

$x\ max$ is the **maximum value** in the dataset.

2. Data Extraction and Shape

Images and labels are extracted using the dataset which has a shape of (7,459, 64, 64, 3) for images (RGB) and (7,459, 25) for labels, which are encoded using one-hot encoding.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

When CNN performs a picture classification task, it uses picture information of the same dimensions as the information. Picture information must have, in most cases, a similar size, that is, with equal measurement of length and width. Given that chief records come in various sizes, grayscale picture sizes radically differ; therefore, all grayscale photographs should be normalized to ensure consistent pixel intensity distribution and improve model stability and overall classification accuracy.

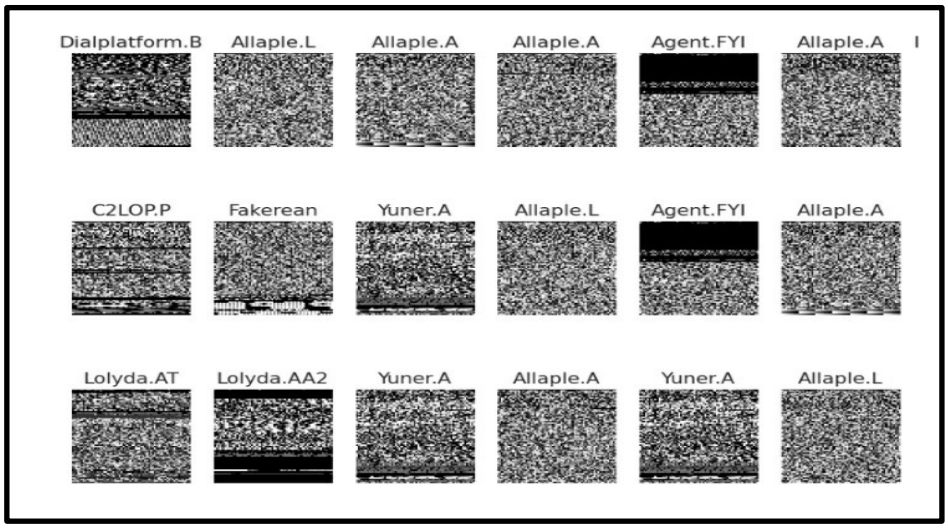

Fig 2. image in size 64 x 64

3.3 Model Architecture

In this subsection, we present a comprehensive examination of the structural framework of our initial model. The model follows a sequential configuration, and Fig 3 presents its architecture.

The initial layer is convolutional layer with 30 filters and (3 x 3) kernel size, a ReLU activation function is used for non-linearity to extract the complex patterns from the input features. All images are given an input shape of 64 x 64 (pixels and color mode of RGB to match the data provided in the dataset. Thereafter, a MaxPooling2D layer is applied to reduce the spatial dimensions of the feature maps, thereby decreasing computational complexity while preserving the most important features. The second convolution layer contains 15 filters of size 3 x 3 (and activation function of ReLU to extract higher-level features. Next, a second MaxPooling2D layer was implemented as in the first pooling layer to minimize the dimensionality of the feature map.

Additionally, to avoid overfitting of the model, a Dropout layer with a rate of 0.25 is used; during the training phase, 25% of neurons are ignored to reduce over-reliance among neural units and improve the model's performance on unseen data. The implementation of a Dropout layer with a rate of 0.25 decreases inter-neuron dependencies and increases both generalizability and performance on new observations. The output of the convolutional layers is flattened to one dimension with the help of Flatten layer so that it can now feed into the fully connected layers. The first Dense layer has 128 neurons, and all layer's use ReLU activation function which is capable of learning different relationships between features. A second Dropout layer with the dropout rate fixed at 50% is inserted with an aim of generalization by avoiding overfitting. The next Dense layer contains 50 neurons and applies again the ReLU Function, which adds further non-linearity. Finally, the output layer is a Dense layer cross entropy with the number of units equal to the number of classes (num_classes) and uses the SoftMax activation function to produce a probability distribution across all classes. The model is compiled using the categorical cross entropy loss function owing to its compatibility with multi-class classification tasks. Additionally, the Adam optimizer is selected because it adapts learning rates and efficient convergence to handle complex malware patterns.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

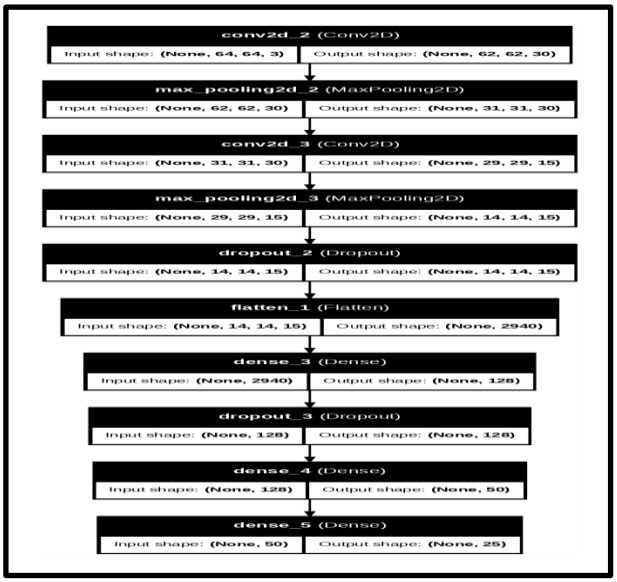

Fig 3. Architecture of the basic CNN

The CNN architecture was determined from multiple trials using multiple hyperparameter combinations to determine the best achievable performance. The research evaluated numerous CNN arrangements through tests using various convolutional layer amounts and filter sizes and dropout rates and fully connected layer dimensions. The decision was made for the final configuration by evaluating model accuracy as well as training time and overfitting prevention metrics. TABLE II shows the different configurations tested during experimentation parameters:

TABLE II: Comparison of Different CNN Configurations

Model Configuration	**Conv Layers**	**Filters**	**Dropout Rate**	**Dense Layers**	**Accuracy (%)**	**Training Time**
Model 1	2	(32, 64)	0.25	(128, 50)	92.5	Medium
Model 2	**2**	**(30, 15)**	**0.25**	**(128, 50)**	**96.0**	**Optimal**
Model 3	3	(64, 32, 16)	0.3	(256, 128, 50)	95.8	High

3.4 Evaluation Metrics

Different evaluation techniques were employed to assess the trained CNN model results to compare their efficiency. These evaluation measures are aimed to generate a numerical value of the model in as much as the model performance is concerned through the use of numerous equations and methods. The Metrics of accuracy, precision, recall, and F1-score have been selected as the evaluation measures for this research [8]. These metrics are defined as follows:

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

$$\text{Precision} = \frac{TP}{TP + FP}, \quad (2)$$

$$Recall = \frac{TP}{TP + FN}, \quad (3)$$

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

$$F1\text{-}score = \frac{2 * Precision + Recall}{Precision \times Recall}. \quad (5)$$

In this context, True Positive (TP) and False Positive (FP) denote the quantities of file samples that have been accurately and inaccurately categorized as malicious, correspondingly. In parallel, True Negative (TN) and False Negative (FN) represent the quantities of file samples that have been correctly and incorrectly identified as benign [15].

4. EXPREMENTAL RESULT

The model is executed utilizing the Python programming, The Google Collab stage is utilized to carry out the model. The performance of the models was evaluated in terms of accuracy, precision, F1_Score, and sensitivity (recall). We evaluated our model on CNN techniques based on the Malimg dataset.

The validation accuracy of several approaches can be observed in Table 3 using the Malimg dataset. Among all the earlier methods, the accuracy of our model was the highest at 96.0%.

TABLE III demonstrates the evaluation between different previous studies along with the proposed model on the Malimg dataset for accuracy and precision, recall, and F1-score measurement. The conclusions from the results show that the CNN model proposed here performs better than the previous techniques for Malimg dataset malware categorization.

TABLE III. Comparison of existing works with the proposed method for the Malimg dataset

Author	Dataset	Use Model	Acc (%)	P (%)	R (%)	F1 (%)
Mesut G. (2024) [7]	Malimg	VGG16 MobileNet	90,65% 90,65%	-	-	-
MD. HARIS et al. (2023) [8]	Malimg	Ensemble Model (custom CNN, the VGG16, ResNet50, and InceptionV3)	92.30 %	92.05 %	92.15 %	92.25 %
Hammad et al. (2022) [9]	Malimg	Proposed (Tamura + ELM)	95.42%	-	-	-
Abien M. (2019) [11]	Malimg	GRU-SVM	84.92%	0.85%	0.85%	0.85%
Proposed model	**Malimg**	**CNN**	**96.0%**	**95.0%**	**96.0%**	**95.0%**

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

TABLE IV displays the classification report which thoroughly describes the results of the model for the different classes of malware. The accuracy of the model was rated at 96%, implying that the model is extremely powerful in the classification of the samples. The average for precision and recall across classes with an F1-score shows equal or even stronger values at 95%, 96%, and 96%, respectively.

TABLE IV: Classification Report

Class	Precision	Recall	F1-Score	Support
Adialer.C	1.00	1.00	1.00	34
Agent.FYI	1.00	0.97	0.98	30
Allaple.A	1.00	1.00	1.00	674
Allaple.L	1.00	1.00	1.00	387
Alueron.gen! J	0.97	1.00	0.99	36
Autorun.K	0	0	0	25
C2LOP.P	0.9	0.63	0.74	41
C2LOP.gen! g	0.88	0.77	0.82	56
Dialplatform.B	1.00	0.98	0.99	43
Dontovo.A	1.00	1.00	1.00	47
Fakerean	0.99	0.98	0.98	90
Instantaccess	1.00	1.00	1.00	112
Lolyda.AA1	0.95	1.00	0.97	57
Lolyda.AA2	1.00	0.97	0.99	35
Lolyda.AA3	1.00	0.97	0.99	37
Lolyda.AT	1.00	0.97	0.98	32
Malex.gen! J	0.93	0.97	0.95	29
Obfuscator.AD	1.00	1.00	1.00	28
Rbot! gen	0.89	1.00	0.94	42
Skintrim.N	0.9	1.00	0.95	18
Swizzor.gen! E	0.39	0.25	0.3	28
Swizzor.gen! I	0.46	0.78	0.58	32
VB.AT	0.97	0.99	0.98	97
Wintrim.BX	0.91	0.88	0.89	24
Yuner.A	0.89	1.00	0.94	204
accuracy	0.96	0.96	0.96	2,238
macro avg	0.88	0.88	0.88	2,238
weighted avg	0.95	0.96	0.96	2,238

In TABLE IV, a value of **1.00** appears multiple times under Precision, Recall, and F1-Score. Sections in Table 3 indicate the perfect classification abilities of the model for these specific classes. A model achieves this result by making flawless classifications of each class without generating any misleading positive or negative results.

Some of these classes include Adialer.C, Allaple.A, Allaple.L, and Dialplatform.B, Instantaccess, and Obfuscator.AD, among others, which exhibit near perfect performance in terms of accuracy, recall, and F1-scores. Thus, the model can easily distinguish these types of malwares without frequent incorrect identifications of new types.

Fig 4 displays the plots of the training and testing accuracy as well as loss over the number of epochs for the proposed model using the Malimg datasets. From Fig 4, we see that the accuracy increases for other epochs; whereas the loss decreases as epochs increase.

The accuracy and loss performance across training and testing phases can be found in Fig 4 from 21 different epochs. This section presents an in-depth analysis of the obtained results.

1. Accuracy Analysis

An initial steep increase appears in the training accuracy blue line as well as testing accuracy orange line during the first several epochs. The model achieves an accuracy of over 90% between Epoch 5 and onward which

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

shows that effective learning has occurred. The latter stages of training lead to stabilized accuracy levels in which training and testing curves demonstrate virtually no difference, indicating low overfitting together with good generalization capabilities. The classification performance achieves approximately 96% accuracy in its final stage.

2. Loss Analysis

The training loss (blue line) shows a fast reduction during initial epochs; thus, it drops from above 2.0 to near 0.1 which indicates an effective learning process. After a few epochs, the testing loss (dashed orange line) demonstrates stability by staying at a low and consistent value, thus indicating good performance on new input data. The model demonstrates effective generalization due to the absence of any noticeable difference between training and testing loss values.

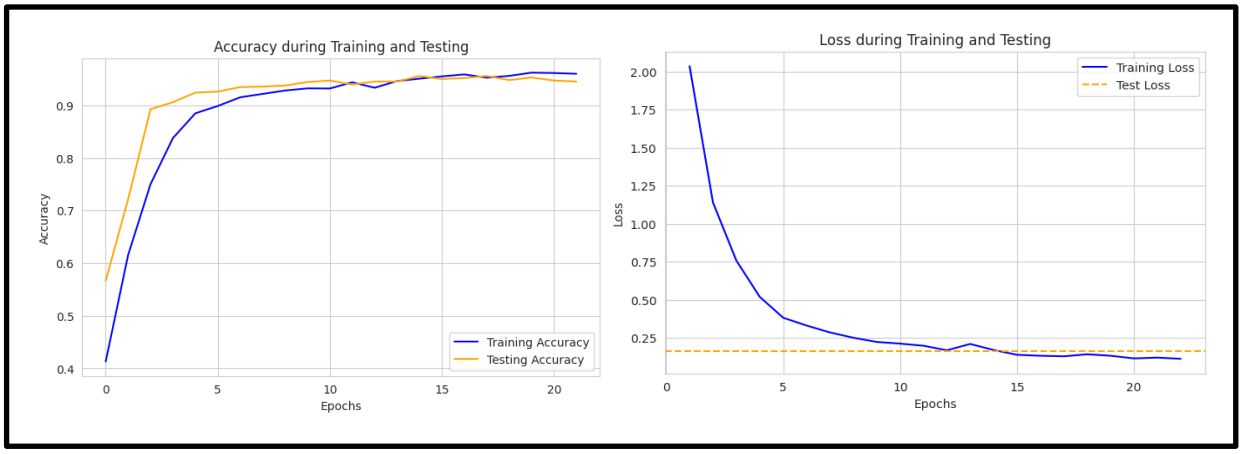

Fig 4. Training and test accuracy and loss for the Malimg dataset

5. CONCLUSION AND FUTURE WORK

This paper introduces various conclusions from the results obtained after implementing the proposed method as well as analyzing these results. The study results conclude as follows:

1. For the Malimg dataset, this proposed model led to better classification. Most of the obfuscated malware samples were detected by the proposed model, and its effectiveness was proven against the use of malware mitigation methods.
2. This paper affirms CNN's exceptional capability to identify broad patterns of malware. A basic model made up of different CNN layers succeeded in achieving solid detection results.
3. Traditional malware detection currently experiences challenges because their detection accuracy is unstable alongside their inconsistent abilities to detect new types of malwares. This research solves the identified limitation through the use of deep learning techniques to detect malware.
4. Finally, the model was classified by CNN; for comparison with other models, it was checked over different measures. It gained an accuracy of 96% higher than the other models.
5. Future research can focus on involving either making adjustments and changes to the CNN architectures identified in this work or by combining CNN models with other machine learning models by using other techniques that can further enhance its performance.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 51, No. 1, 2025, pp. 64–74
DOI: https://doi.org/10.25195/ijci.v51i1.542
Print ISSN: 2313-190X, Online ISSN: 2520-4912

References

[1] I. Jemal, M. A. Haddar, O. Cheikhrouhou, and A. Mahfoudhi "*Performance evaluation of Convolutional Neural Network for web security*," Computer Communications, vol. 175, pp. 58–67, 2021. https://doi.org/10.1016/j.comcom.2021.04.029

[2] M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang, and F. Iqbal*," Malware Classification with Deep Convolutional Neural Networks*," IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–5,2018. https://doi.org/10.1109/NTMS.2018.8328749

[3] N. Bhojani," *Malware Analysis*," Proceedings of the Ethical Hacking Conference, pp. 1–5,2014. http://dx.doi.org/10.13140/2.1.4750.6889

[4] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath," *Malware Images: Visualization and Automatic Classification,*" International Symposium on Visualization for Cyber Security (VizSec'11), pp. 1–8, 2011. http://dx.doi.org/10.1145/2016904.2016908

[5] T. H.Hai, D.M. Quang, and N.H.Hoa,"*An Application of CNN-based Models with Fine-tuning Techniques on MaleVis Malware Dataset*," Proceedings of the 13th International Conference on Information Technology and Its Applications (CITA), vol. 2, pp. 210–222, 2024.https://elib.vku.udn.vn/handle/123456789/4034

[6] A. Mihoub, O. B. Fredj, O. Cheikhrouhou, A. Derhab, and M. Krichen, "*Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques*," Computers &Electrical Engineering, vol. 98, p. 107716, 2022. https://doi.org/10.1016/j.compeleceng.2022.107716

[7] M. Guven," *Leveraging deep learning and image conversion of executable files for effective malware detection: A static malware analysis approach*," AIMS Mathematics, Volume 9, pp. 15223–15245, 2024.https://doi.org/10.3934/math.2024739

[8] M. H. U. Sharif, N. Jiwani, K. Gupta, M. A. Mohammed, and M. F. Ansari," *A deep learning-based technique for the classification of malware images*," Journal of Theoretical and Applied Information Technology, Vol.101, pp. 135–149,2023.https://www.researchgate.net/publication/367272342

[9] B. T. Hammad, N. Jamil, I. T. Ahmed, Z. M. Zain, and S. Basheer, "*Robust malware family classification using effective features and classifiers*," *Applied Sciences*, vol. 12, no. 15, pp.7877, 2022. https://doi.org/10.3390/app12157877

[10] F. Zhong, Z. Chen, M. Xu, G. Zhang, D. Yu, and X. Cheng," *Malware-on-the-Brain: Illuminating Malware Byte Codes with Images for Malware Classification*," IEEE Transactions on Computers, Vol 71, pp.438 - 451,2022. https://doi.org/10.1109/TC.2022.3160357

[11] A. F. M. Agarap, "*Towards building an intelligent anti-malware system: A deep learning approach using Support Vector Machine (SVM) for malware classification*," vol. 1801, pp. 1–8, 2019. https://doi.org/10.48550/arXiv.1801.00318

[12] M. Al-Fawa'reh, A. Saif, M. T. Jafar, and A. Elhassan," *Malware Detection by Eating a Whole APK,"* International Conference for Internet Technology and Secured Transactions (ICITST), 2021. https://doi.org/10.23919/ICITST51030.2020.9351333

[13] J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, "*MAPAS: A practical deep learning-based android malware detection system,"* International Journal of Information Security, vol. 21, pp. 725–738, 2022.https://doi.org/10.1007/s10207-022-00579-6

[14] Malimg https://www.kaggle.com/datasets/manmandes/malimg/data

[15] Y. Ye, T. Li, D. Adjeroh, and S. S. Iyengar, "*Detection of malicious code variants based on deep learning,"* IEEE Transactions on Industrial Informatics, vol. 14 pp. 3187-3196, 2018.https://doi.org/10.1109/TII.2018.2822680