Journal of Telecommunication, Electronic and Computer Engineering
ISSN: 2180 – 1843 e-ISSN: 2289-8131 Vol. 15 No. 3
jtec.utem.edu.my

A Deep Learning Model for Malware Multi-Class Classification based on Colored Malware Images

B. Yadav[1], S. Tokekar[2]
[1]*Computer Engineering, Institute of Engineering and Technology, DAVV Indore, India.*
[2]*Electronics and Telecommunication Engineering, Institute of Engineering and Technology, DAVV, Indore, India.*
balram.dreamsworld@gmail.com

Article Info

Article history:
Received Apr 30th, 2023
Revised July 1st, 2023
Accepted July 11th, 2023

Index Terms:
CNN
Color Malware Images
Deep Learning
Image Processing
Malware Classification
Malware Visualization

Abstract

A malicious computer program and its unique attacks have been a source of concern for decades and a major threat to the people of the cyber world. There has been a dramatic increase in malware attacks, their exploration, and the complexity of code and types, which has made malware classification very difficult. With the advent of automated strategies and tools for producing malware, a newly developed malicious program evades detection strategies. Deep Learning (DL) has gained a lot of attention, popularity, and performance in malware analysis. Although DL models reach high-performance levels, they require extensive training samples, high-resolution images, and deep DL structures. This study investigates and highlights the performance of the Convolutional Neural Network (CNN) based malware classifier on colored malware images. To test the CNN model, a well-known malimg dataset was used with a classification speed of fewer than three milliseconds per step, achieving 98.394% test accuracy. The test results encourage the usage of color image processing compared to grayscale images and demonstrate good efficiency and accuracy. It emphasizes that even with basic DL structures, remarkable performance can be attained when dealing with low-dimensional images.

I. INTRODUCTION

Malware, widely regarded as a malicious program, is a broad term for one of the largest significant issues in the computing world. Malware carries out actions to cause harm to computer resources, data, servers, and networks. Users are always concerned about the system and information security. Malware is developed using automated tools and processes and the types and attack types are constantly modified to elude detection systems. As a result, identifying and classifying malware has become an ongoing and complicated process.

The rapid increase in malware [1-2], its variants, and malicious attacks create severe problems and challenges for nations, economies, societies, users, and antivirus companies. Malware classification is regarded as one of the most important measures in the fight against malware [3].

As evidenced by annual statistical studies from renowned entities such as AV-TEST institute and McAfee Antivirus Corporation, the surge in malware is alarming. Figures 1 and 2 show the AV-TEST Institute reported over 450000 new malicious programs (malware) and undesirable apps. There were 9.95 million new malware reported in one month up until January 27, 2022, and 1322.58 million malware reported at the start of the year 2022 [2].

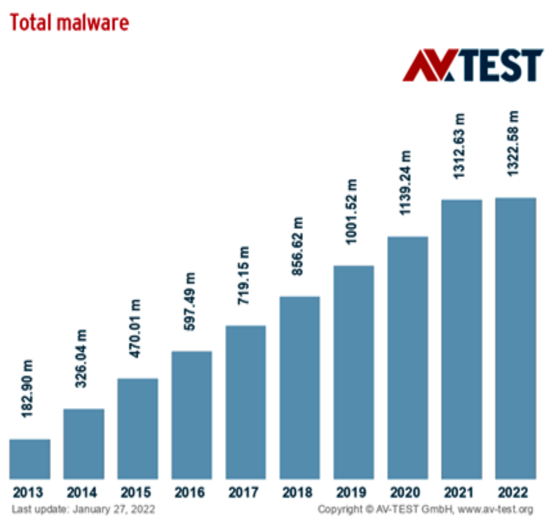

Figure 1: Malware statistics

Malware accounted for 92.14% of all unwanted application deployments in the last year. In the second and third quarters of 2021, McAfee statistics reveal an increase in ransomware, malware, cloud threats, and other unwelcome apps.

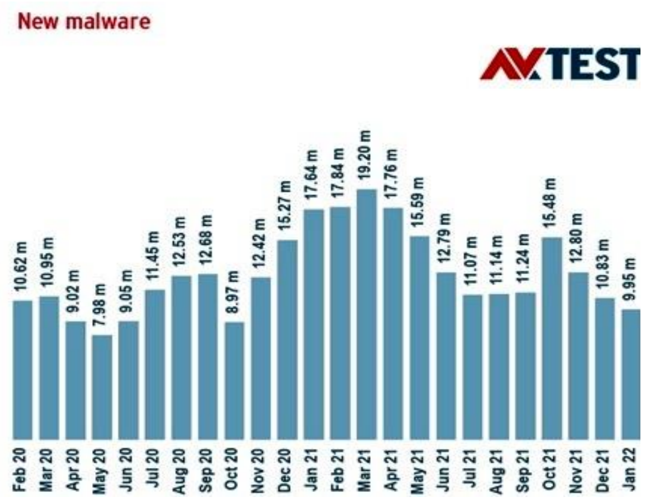

Figure 2: New Malware statistics

Traditionally, malware analysis has been conducted using a combination of static, dynamic, and hybrid analysis; approaches based on data mining and Machine Learning (ML). While effective to some extent, these methods come with substantial downsides, such as time-consuming, requiring manual feature extraction, demanding more resources, and other disadvantages.

Deep Learning (DL)-based models have recently made a breakthrough in malware analysis, particularly malware classification. DL is part of a much larger ML technology family. The DL architecture has proven to excel in a variety of areas, including natural language processing, computer vision, speech recognition, and medicine, making great strides in a wide range of areas. In terms of image classification and associated image-processing tasks, CNN is the most widely used and well evaluated DL model [36]. We compared CNN with other DL models for the same task in the study [38], highlighting both its benefits and drawbacks. CNN has demonstrated a high level of accuracy in classifying images in literature. This research provides a DL-based classification model that classifies malware with color images using CNN architecture and achieves high testing accuracy.

The main objective of this paper is to examine the effectiveness of malware classification using colored malware images versus grayscale images. This paper aims to present a DL model, a simplest solution (no deep/complex DL architecture) for image-based malware classification, with no transformation of the malware images required, no data augmentation techniques, and does not address class imbalance issue.

The objective of classifying malware into a malware family is to aid investigators in identifying the attack's possible source and comprehending the nature of malware sample, its destructive behavior, and the consequences of attack. By defining the specific malware class of malicious code, system administrators, researchers, and security analysts are better positioned to devise effective remediation measures and proactive defense mechanisms against such threats.

The primary contribution of this paper is to test and examine the performance of a previously created CNN model when applied to color malware images instead of grayscale malware images. The main goal is to determine whether the utilization of color malware images as opposed to grayscale ones, influences the model's effectiveness by either increasing or decreasing its accuracy. This inquiry is grounded in the understanding that color malware images contain more nuanced information than their grayscale counterparts, even though employing them may extend the training and testing durations and introduce additional complexity.

The paper is structured as follows. Section II is devoted to a comprehensive literature review. A detailed methodology, with a full description of the dataset, and the proposed DL architecture are included in Section III. Section IV presents the experimental design. The results and discussions of the study are covered in Section V, and the future scope and conclusion of the papers are covered in Section VI.

II. RELATED WORKS

This section contains relevant work on the classification of malware through the visualization of color malware images and the application of DL techniques. The visualization of malware constitutes a substantial advancement in the field of analysis tasks.

The approach provided by Nataraj et al. [5-6] is acknowledged as the first malware visualization technique, where they innovatively transformed malware binary code into grayscale images. By leveraging GIST characteristics to compare the texture information within these images, they utilized the K-nearest neighbors (KNN) algorithm to classify them. In a different approach, Han et al. [7] conceptualized the visualization of malware binaries as color image matrices. By extracting binary information (using opcodes as binary data) through static analysis and dividing it into blocks, they applied two hash functions to generate RGB-colored pixels data. A selective area-matching technique was employed to assess the similarity of image matrices. The results showed that classification was effective, with a similarity calculation time of around 2.4 milliseconds.

Pal and Sudeep [8] advanced the field by highlighting the importance of pre-processing techniques in malware classification. By implementing a CNN model to color malware images, they emphasized and demonstrated that utilizing raw data with a Deep Neural Network (DNN) does not give satisfactory results, whereas pre-processed data increases classification accuracy. They conducted experiments with three different normalization techniques (mean normalization, standardization, and zero component analysis), illustrating how accuracy fluctuated accordingly. Kornish et al. [9] further implemented a CNN model for malware classification using color images. The dataset contains assembly code and raw binary samples, employing the latter to save time required for visualization assembly codes. Hexadecimal values, easily converted into decimal values, were represented as image features using the hamming distance. They utilized three pre-trained CNN models including AlexNet, Visual Geometry Group (VGG) 16, and VGG19 to classify malware images.

Kim et al. [10] introduced and implemented a new DL architecture transferred Generative Adversarial Network (GAN) for the detection and classification of unknown malware within color images. By pre-training the GAN with an Auto Encoder (AE), they mitigated the fundamental training limitation inherent in GAN architecture. Their approach outperformed other traditional models and facilitated rapid malware detection. In a 2019 study, Singh et al. [3] explored a new technique to portray malware binaries as color images, favoring RGB representation over grayscale images. They employed deep NN architecture such as Residual Neural Network (ResNet)-50 including a dense CNN to classify the images. The authors also described a novel approach for converting binary files (a string of zeros and ones) into RGB values.

Naeem [11] addressed the malware detection problem on the Internet of Things (IoT) networks. The authors proposed a naive method that converts the malware binaries into high-dimensional color images. To detect malware within IoT networks, they implemented a fast CNN model, comprising four deep layers with large numbers of filters. Yin et al. [12] implemented a hybrid DL model tailored for color malware image classification. The approach was multi-pronged: Initially, features were optimized using CNN. Subsequently, these extracted features were subject to both CNN and recurrent neural network (RNN) networks processes. While this layered approach surpassed the original CNN model in performance, the results were not promising. In a study by Naeem et al. [13] in 2020, raw Android files underwent a transformation into color images, which were then submitted into a deep CNN model for malware detection. Their deep CNN model, crafted to counteract IoT malware, displayed impressive accuracy rates. Kumar and Bagane [14] in the same year, implemented a hybrid DL model for malware classification, combining CNN with bi-directional long short-term memory (LSTM). CNN was utilized initially for automatic feature extraction, after which the flattened output was classified by the LSTM layer. In 2020, Vasan et al. [15] proposed a new approach based on the ensemble CNN architecture for malware detection and classification. By harnessing the power of pre-trained models such as VGG16 and ResNet50, they extracted features, combined them, and accurately classified malware into their corresponding families, underlining the efficacy of the proposed method. Kabanga and Kim [16] implemented a 3-layer CNN model to classify color malware images. The author used CNN because of its reliability, its ability to process an entire image at once, and its prowess in automatic feature extraction to achieve high accuracy rates. Lu and Li [17] implemented a GAN, one of the latest DL models for malware classification, to address data imbalance issues. Although their 18-layer deep CNN within the GAN structure was a bold initiative, the obtained results were not promising as anticipated. Even with the deployment of the most recent DL model and the generation of additional training samples via GAN, the results remained suboptimal.

Azab et al. [18] implemented a CNN-based malware classification framework that uses spectrogram images. Unlike the grayscale classification, the proposed method first converts the raw bits of the malware binaries into spectrogram images (by signals fundamentals: applying Fourier Transform visualizes malware as spectrogram images) and then feeds them to CNN. Fu et al. [19] visualized malware as RGB color images to reduce the complexity of the DL-based training model. Low-dimensional global features from the images were extracted, and code and data segments were also extracted as local features. Their method was a combination of global and local features to achieve effective malware classification. For classification, three different types of classifiers Random Forest (RF), K-NN, and Support Vector Machine (SVM) were selected to classify malware and achieved the highest 97.47% accuracy with RF. Mustafa et al. [20] proposed a multi-model, feature fusion-based CNN (VGG19) framework to classify malware families and achieved remarkable performance. The author applied a multi-model by combining various features, one feature matrix from color malware images and three different matrices from grayscale malware images and feeding these into distinct CNN models then output from each is combined into a single vector and then feed into multilayer perceptron neural network and achieved high accuracy. Awan et al. [21] implemented a CNN-based framework that extracts spatial information (dynamic spatial convolution) from color malware images to classify 25 different malware families without addressing the class imbalance issue. The authors applied spatial convolution and proved that some regions in the images are more informative as compared to the whole area and suited for classification.

In summary, previous research shows that malware visualization is effective and useful in raising classification performance metrics. Previous research has mostly used transfer learning-based models to categorize malware utilizing a pre-trained very advanced and complicated DL architecture and high-dimensional gray-scale malware image processing, which makes the classification incredibly challenging and time-consuming. Towards this undertaking, this work suggests and focuses on malware color image representation (as color images contain more information than grayscale images and aid in malware classification as malware from the same family display similar visual representations and differ from other family patterns), low dimensional image processing, based DL model that improves classification accuracy and other performance measures.

III. PROPOSED METHODOLOGY

The proposed approach introduces a distinctive methodology wherein CNN is used for learning discriminative patterns automatically from the color malware images and classifying them. CNN is a feed-forward neural network and includes a variety of layers (convolutional, pooling, and drop-out layers) that can be stacked to build a CNN model. In the proposed approach, the malware classification task is converted into an image classification task by converting malware binaries into images. The classification module is divided into two subtasks. The first one is referred to as the training phase where the CNN classifier is created from a set of labeled malware images and families. The second step is the testing phase where the classifier tested on a set of unlabeled and unseen malware images that were not part of the training phase.

Once the model is designed and the two phases are operational, it is used to classify malware samples into their corresponding malware family. The architecture of the CNN network was utilized to investigate and test the CNN model on color malware images.

A. Pre-processing

Pre-processing is a necessary step before building the model. It involves resize operation as samples of malware images in the utilized dataset differ in size and dimension and the CNN model requires images of the same size for processing that is why resizing images is essential.

B. *Malimg Data set*

The malware dataset used in this study is highly unbalanced and complicated, with 25 different malware classes (Class 1 to 25) and 9339 malware samples that differ in image dimensions and file size. Table 1 shows the different family classes and their variations. The following is a breakdown of the data: In each class, 80% of the images (7564 malware samples) were utilized for training, 10% (841 samples) for validation, and the remaining 10% (841 samples) for testing. The dataset images receive no extra data augmentation beyond the color map and scaling.

Table 1
Malimg malware dataset description

Malware class	Malware Type	Class	Malware images
Class 1	Worm	Allaple.A	2949
Class 2	Worm	Allaple.L	1591
Class 3	Worm	Yuner.A	800
Class 4	Dialer	Instantaccess	431
Class 5	Worm	VB.AT	408
Class 6	Rogue	Fakerean	381
Class 7	PWS	Lolyda.AA1	213
Class 8	Trojan	C2Lop.gen!G	200
Class 9	Trojan	Alueron.gen!J	198
Class 10	PWS	Lolyda.AA2	184
Class 11	Dialer	Dialplatform.B	177
Class 12	Trojan Downloader	Dontovo.A	162
Class 13	PWS	Lolyda.AT	159
Class 14	Backdoor	Rbot!gen	158
Class 15	Trojan	C2Lop.P	146
Class 16	Trojan Downloader	Obfuscator.AD	142
Class 17	Trojan	Malex.gen!J	136
Class 18	Trojan Downloader	Swizzor.gen!I	132
Class 19	Trojan Downloader	Swizzor.gen!E	128
Class 20	PWS	Lolyda.AA3	123
Class 21	Dialer	Adialer.C	122
Class 22	Backdoor	Agent.FYI	116
Class 23	Worm:AutoIT	Autorun.K	106
Class 24	Trojan Downloader	Wintrim.BX	97
Class 25	Trojan	Skintrim.N	80

C. *Resizing*

The dataset contains all images of different sizes, resizing of images is done and malware images are resized into 32×32 pixels.

D. *Coloring*

The dataset contains 9339 grayscale malware images. To convert grayscale images into color images *OpenCV* library is utilized and *Python* programming language is used. OpenCV provides 13 different color maps in Table 2.

To strengthen the visualization capabilities of computer vision applications and color image processing, these color maps were applied to the grayscale images of the dataset and generated 13 different color maps of the whole dataset shown in Figure 3. Besides grayscale to color malware image translations, no other transformation was applied to the dataset.

The transformed Jet color maps malware images of *ADialer.C, Agent.FYI, Rbot!gen, Apple.B* malware family is shown in Figure 4.

E. *Proposed CNN architecture*

CNN requires three-dimensional inputs in the form of image height, width, and depth. The image of dimension 32×32×3 is provided to the CNN model for processing malware images.

Table 2
OpenCV color maps details

S. No.	Color_maps	The constant value applies to the color maps
1	Autumn	0
2	Bone	1
3	Jet	2
4	Winter	3
5	Rainbow	4
6	Ocean	5
7	Summer	6
8	Spring	7
9	Cool	8
10	Hsv	9
11	Pink	10
12	Hot	11
13	Parula	12

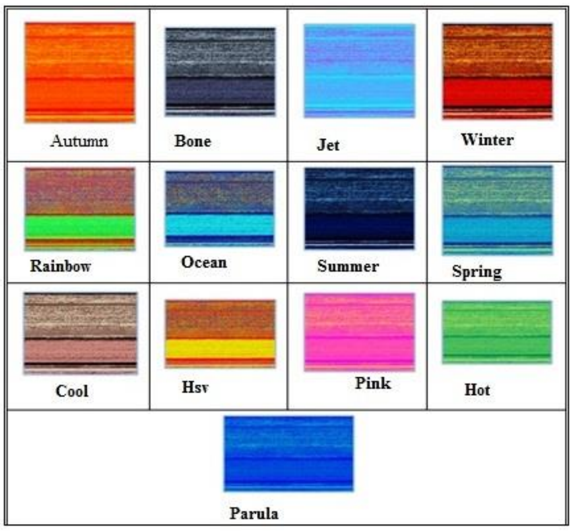

Figure 3: Thirteen different Color maps of malware images

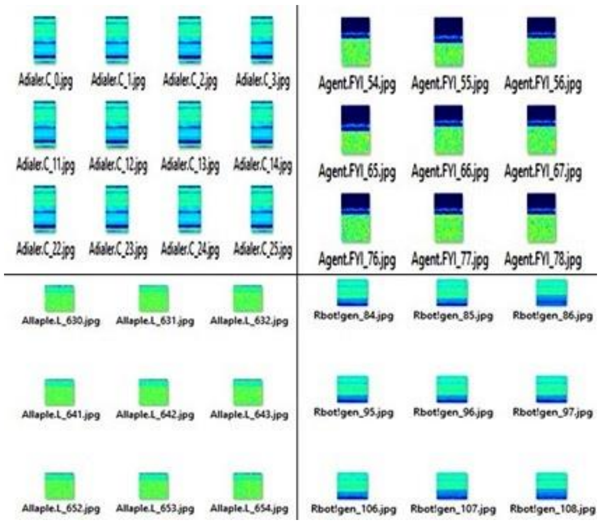

Figure 4: Sample images of the Jet color maps

A two-layer CNN was designed for the classification task; the model contains two convolutional (Conv) layers followed by two Max-pooling layers and then two Fully Connected (FC) layers as shown in Figure 5.

The CNN model architecture is described by the following steps:

- Dataset images are reshaped into a size of 32×32×3, where 32×32 is an image dimension and three is the color image channel.
- Input = 32×32×3
- The first convolutional layer (Conv1) has 32 filters of size 4×4. The output of the convolutional layer on the input image is

$$G\,(m,n) = (f \times h)\,(m,n) = \sum_j \sum_k h\,(j,k) \times f([m-j, n-k]) \quad (1)$$

where,
f represents a 2-dimensional input image of (width× height× channel) and h represents a 2-dimensional kernel/filter size.
ReLU is an activation function (to add nonlinearity in the system) and it is defined as ReLU(x) = (max (0, x)).

- followed by a max-pooling layer of size 2×2
- the second convolutional layer (Conv2) has 64 filters of size 3×3
- ReLU as an activation function
- followed by a max-pooling layer of 2×2
- Flatten Layer
- FC Layer 1 = 1024 neurons
- Final FC Layer 2 = 25 neurons (corresponding to 25 different malware families of the dataset) with *Softmax* activation function and it is defined as

$$Softmax(x_i) = \frac{e^{(x_i)}}{\sum_{j=1}^{n} e^{(x_j)}} \quad (2)$$

where x is an input vector for the pre-activation value and e represents the base of the natural logarithm system.

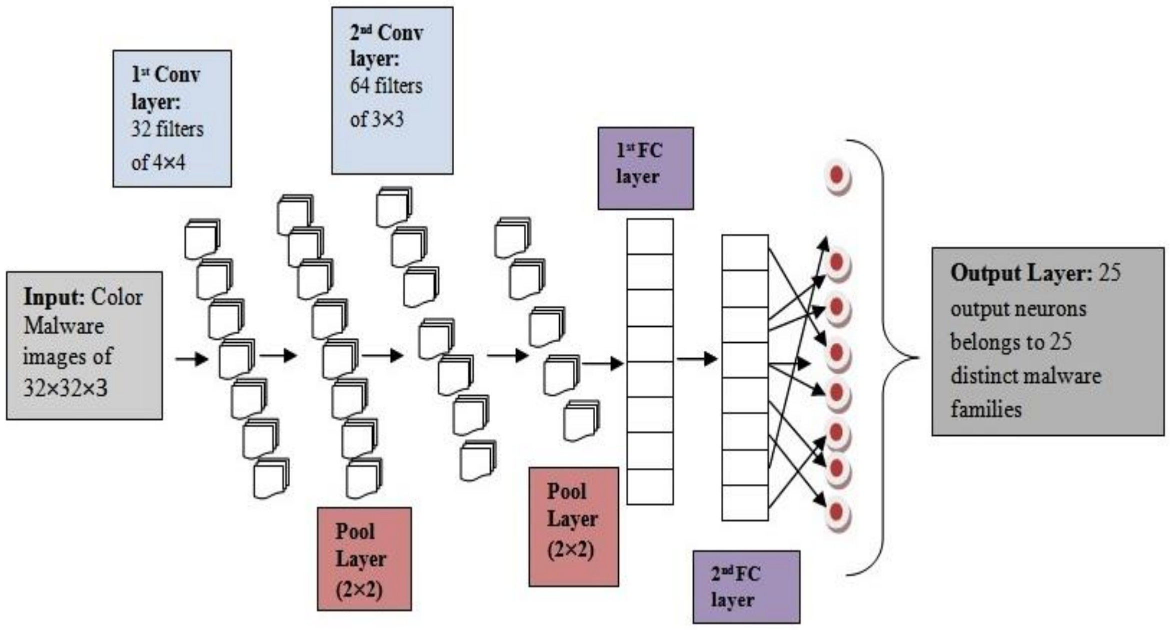
1st Conv layer: 32 filters of 4×4
2nd Conv layer: 64 filters of 3×3
1st FC layer
Input: Color Malware images of 32×32×3
Output Layer: 25 output neurons belongs to 25 distinct malware families
Pool Layer (2×2)
Pool Layer (2×2)
2nd FC layer

Figure 5: Layered Architecture of the Proposed CNN Model

Table 3
CNN model summary

Layer (type)	Output Shape	Number of Parameters
conv2d_3 (Conv2D)	(None, 29, 29, 32)	1568
max_pooling2d_3(MaxPooling2)	(None, 14, 14, 32)	0
conv2d_4 (Conv2D)	(None, 12, 12, 64)	18496
max_pooling2d_4(MaxPooling2)	(None, 6, 6, 64)	0
dropout_3 (Dropout)	(None, 6, 6, 64)	0
flatten_2 (Flatten)	(None, 2304)	0
dense_3 (Dense)	(None, 1024)	2360320
dropout_4 (Dropout)	(None, 1024)	0
dense_4 (Dense)	(None, 25)	25625
Total params: 2,406,009		
Trainable params: 2,406,009		
Non-trainable params: 0		

IV. EXPERIMENTAL DETAILS

The proposed architecture is simple, requires no pre-processing, and is trained, and tested on the benchmark, complexes, and highly imbalanced malware dataset of grayscale images. The proposed methodology requires no special transformation of the images in the dataset and with the 32×32 image size, it outperforms almost all other state-of-the-art DL models that are far deeper in architecture, complex, and take more time as well as hard to train. All experiments were run on a personal laptop with an Intel Core i3 2.40 GHz processor and 4 GB of RAM that was running Windows 10, 64-bit. Python was used; along with Python support tools, packages, and libraries such as *Keras*, *Numpy*, *Scipy*, *Pandas*, and others. The *Spyder* editor provided by *Anaconda Navigator* was employed for all the coding and implementations of the CNN model.

V. RESULTS AND DISCUSSION

The results show that using the proposed CNN architecture, employing color malware images leads to a modest improvement in accuracy when compared to grayscale malware images. Precision is defined as the ratio of correctly predicted positive samples to all positive predictions and represents how accurately the model places the samples in a true class. The recall is defined as the percentage of correctly predicted samples in the total number of correct samples. Accuracy is a simple performance metric that refers to the number of samples that are correctly estimated from the total number of samples in the dataset. The F1 score is an important performance measure when the dataset is unbalanced because it indicates the harmonic mean of precision and recall.

Performance metrics precision as in (4), recall as in (5), F1-score as in (6), and accuracy as in (3) were used to effectively evaluate the performance of the developed CNN model because the dataset was significantly imbalanced.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \quad (3)$$

$$Precision = \frac{TP}{(TP + FP)} \quad (4)$$

$$Recall = \frac{TP}{(TP + FN)} \quad (5)$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (6)$$

The proposed classifier's performance is evaluated through accuracy, precision, recall, and F1-score performance measures. Table 4 represents the hyperparameter values.

Table 4
Hyperparameters values for the malware classification

Parameters	Values
Image size	32×32 color images
Train-Validation-Test ratio	80%-10%-10 %
Loss function	Categorical Cross entropy
Optimizer	Adam optimizer
Learning rate	0.001
Batch size	50
Epochs	30

Table 5 shows the different performance measures of the classification model.

Table 5
Experiment results of the CNN model

Parameters	Values
Model	CNN
Accuracy	98.394%
Precision	97.56%
Recall	98.39%
F1-score	97.94%
Test loss	0.068
Test time	3 seconds (3 ms/step)

The training and validation losses and accuracy of the implemented CNN model are shown in Figure 6 and Figure 7 respectively.

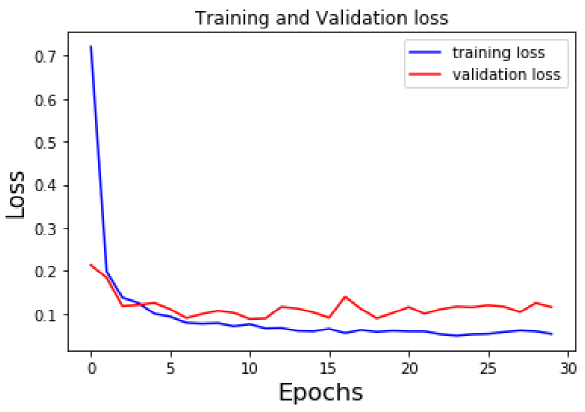

Figure 6: Training and validation losses for the CNN model

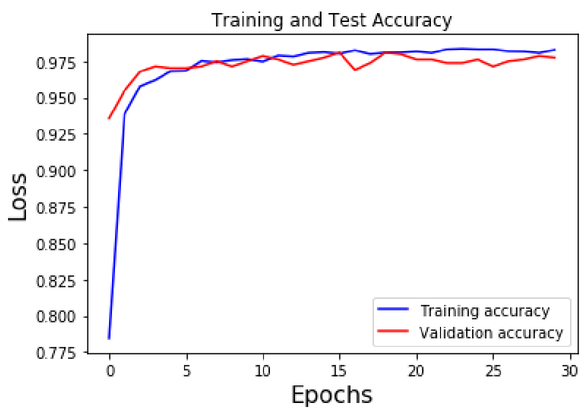

Figure 7: Training and validation accuracy for the CNN model

Table 6 represents the classification report of the individual malware family; the report clearly illustrates the different performance measure parameters (precision, recall, etc.) of each family/class.

A confusion matrix (also known as an error matrix) serves as a tool to evaluate the accuracy of a classifier. It thoroughly visualizes and expands the results of the classifier; it not only shows the correct classification but also provides accurate details on the classification errors (misclassification) and the type of misclassification done. Figure 8 depicts the confusion matrix (A confusion matrix was utilized to assess the performance of the implemented CNN model) and it clearly illustrates the variants that are correctly classified and variants that are misclassified in different malware families.

The malware families in Figure 8 show how some malware can impede the training of DL models and degrade performance. These malware families share comparable visual traits and family structures. Some families have an odd likeness, as in the case of all *Autorun.K* malware samples were erroneously labeled as *Yuner.A*, and many more are wrongly classified. A total of 15 malware variants with erroneous labels were found among 934 malware test samples.

Table 6
Classification Report of CNN model for each dataset class

Malware class	Precision	Recall	F1-score	Support
Class 1	1.00	1.00	1.00	11
Class 2	1.00	1.00	1.00	9
Class 3	1.00	1.00	1.00	296
Class 4	1.00	1.00	1.00	173
Class 5	1.00	1.00	1.00	20
Class 6	0.00	0.00	0.00	9
Class 7	0.93	1.00	0.97	14
Class 8	1.00	0.90	0.95	21
Class 9	1.00	1.00	1.00	16
Class 10	1.00	1.00	1.00	20
Class 11	1.00	1.00	1.00	37
Class 12	1.00	1.00	1.00	27
Class 13	1.00	1.00	1.00	21
Class 14	1.00	1.00	1.00	21
Class 15	1.00	1.00	1.00	12
Class 16	1.00	1.00	1.00	16
Class 17	1.00	1.00	1.00	11
Class 18	1.00	1.00	1.00	7
Class 19	1.00	1.00	1.00	19
Class 20	1.00	1.00	1.00	7
Class 21	0.92	0.85	0.88	13
Class 22	0.77	0.91	0.83	11
Class 23	1.00	1.00	1.00	36
Class 24	1.00	1.00	1.00	11
Class 25	0.91	1.00	0.96	96

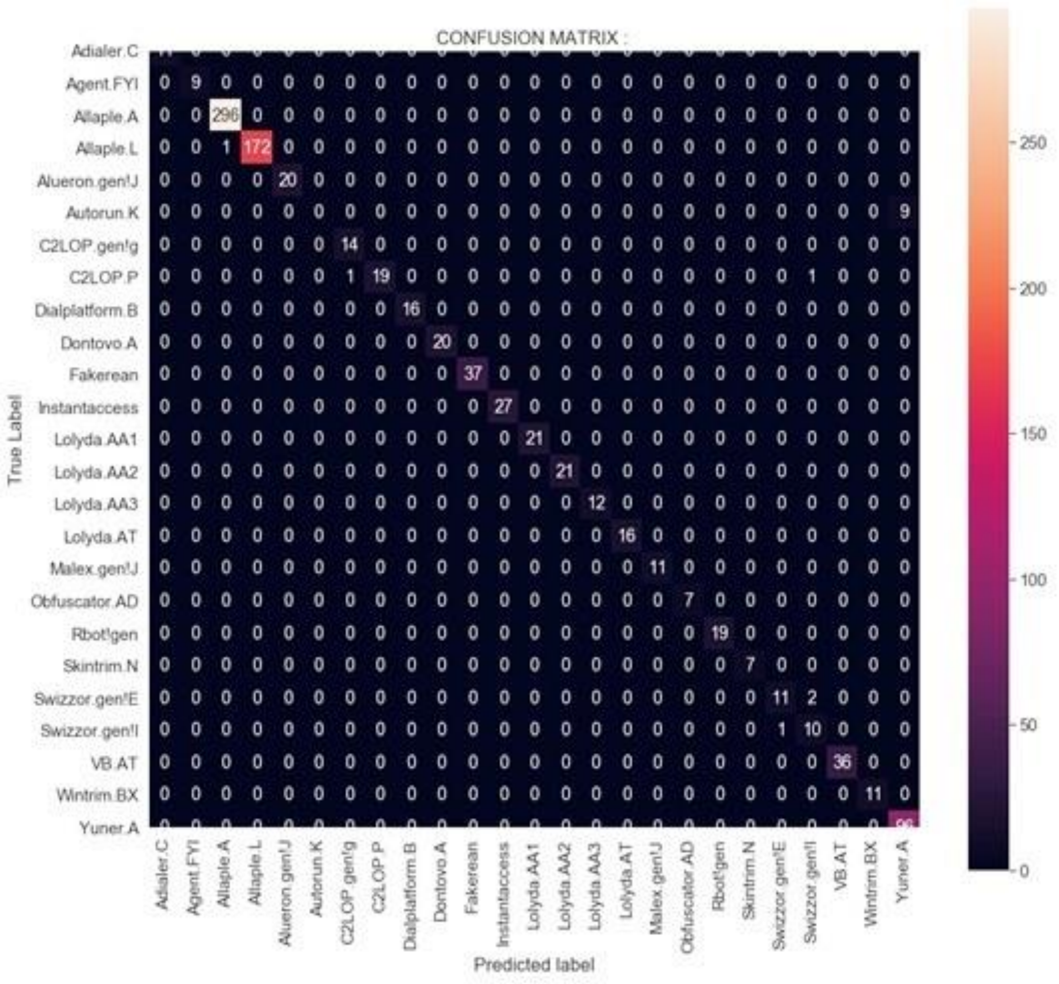

Figure 8: The Confusion Matrix for classification of the 25-malware family

Table 7 compares the results of the proposed model to some benchmark traditional methods of malware classification using DL and visualization.

The proposed model has some advantages over other models proposed by other authors for the same task. Most of the authors use high-dimensional color images with very deep and complex DL architecture and some use image transformation techniques with data augmentation, some address class imbalance issues. The proposed model is a simple, lightweight model that classifies malware more effectively and obtains high accuracy and no complexity in its processing.

The accuracy results are shown by [13], [25], which is higher than the proposed method. However, their techniques need additional time due to the disassembling process, which is not suitable to meet the users' requirements of the IoT network, while the currently proposed technique eliminates this additional processing because the features are extracted directly from a raw binary file. Besides, their results do not reflect real accuracy due to the small dataset that they used. As shown in Table 7, employing color malware visualization rather than grayscale malware visualization has an advantageous impact on model performance, with a modest improvement in accuracy gained compared to grayscale malware image processing on the same CNN architecture [37].

Table 7
Performance comparison on accuracy with the same dataset

Reference	Method	Image details	Accuracy (%)
[12]	CNN, RNN	Color image (32×32)	80
[17]	GAN	Color image (32×32)	84
[18]	CNN	Color image (200×200)	92.8
[19]	Random Forest	RGB image	97.47
	K-NN		96.23
	SVM		95.23
[21]	VGG16(without class balance)	Color image (224×224)	97.62
	VGG16(with class balance)		97.42
	VGG19 (with class balance)		97.38
[24]	CNN + LSTM	Color image (224×224)	NA
[11]	Deep CNN	Color image (192×192)	98.18
Proposed approach	CNN	Color image (32×32)	98.394
[13]	Deep CNN	Color image (224×224)	97.81
[25]	VGG16,	Color image (224×224)	98.47
[20]	ResNeT50	Color image (224×224)	99.50
	VGG-19		99.72

The suggested approach is distinctive in that it uses a 4×4 filter/kernel for classification for the first time and builds a lightweight CNN model without using any pre-trained DL models. The results and proposed model presented here have limitations as well. The first arises from the utilized dataset itself for training and performance evaluation. The dataset is highly imbalanced in the number of malware variants per malware family/class, where some classes are majority class (have a thousand variants), others are minority classes (have variants in double-digits), and due to this, it suffers from misclassification rates. This work also lacks utilization and exploration in the data augmentation, image transformation, and feature engineering domains.

VI. CONCLUSION

This paper presented a CNN model to improve the classification of malware variants through the application of color malware visualization. First, the method transformed the malicious samples into color images, next the malware images were classified by a CNN model that automatically extracts optimized features from the images. The results (achieving an accuracy of 98.394%) proved the effectiveness and efficiency of the proposed model and the speed of the CNN model significantly faster, the results are competitive, and the architecture is simpler as compared to other state-of-the-art models in the literature. The primary purpose of this paper is to investigate and enhance malware classification accuracy by implementing a DL model based on color malware images. This study is significant in the sense that the proposed CNN model is light weighted, enhances the performance of the classifier, and achieves better classification accuracy and test time, for more challenging and highly imbalanced datasets. In the future, the proposed model should also be tested on other malware datasets, especially in IoT networks because of its lightweight architecture. This work is extended by hybridizing this model with other DL architectures to improve the time efficiency and accuracy of classification.

REFERENCES

[1] Malware statistics and Trends Report [online] by AV-test institute January 2022. Available at: https://www.av-test.org/en/statistics/malware/ [Accessed 28 January 2022].

[2] McAfee labs Advanced Threat Research Report [online] October 2021. Available at: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-threats-oct-2021.pdf. [Accessed 27 January 2022].

[3] A. Singh, A. Handa, N. Kumar, S.K. Shukla, “Malware Classification Using Image Representation,” in *Dolev S., Hendler D., Lodha S., Yung M. (eds) Cyber Security Cryptography and Machine Learning, CSCML 2019, LNCS 11527*, pp. 75-92 (2019), https://doi.org/10.1007/978-3-030-20951-3_6.

[4] M. Kalash, M. Rochan, N. Mohammed, N.D.B. Bruce, Y. Wang, F. Iqbal, “Malware Classification with Deep Convolutional Neural Networks,” *9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, Paris, pp. 1-5 (2018), https://doi.org/10.1109/NTMS.2018.8328749.

[5] L. Nataraj, S. Karthikeyan, G. Jacob, B.S. Manjunath, “Malware images: visualization and automatic classification,” in *Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec ’11), Association for Computing Machinery*, New York, NY, USA, Article 4, pp. 1-7 (2011), https://doi.org/10.1145/2016904.2016908.

[6] L. Nataraj, V. Yegneswaran, P. Porras, J. Zhang, “A comparative assessment of malware classification using binary texture analysis and dynamic analysis,” in *Proceedings of the 4th ACM workshop on Security and artificial intelligence (AISec ’11), Association for Computing Machinery*, New York, NY, USA, pp. 21-30 (2011), https://doi.org/10.1145/2046684.2046689.

[7] K. Han, J. H. Lim, and E. G. Im, ”Malware analysis method using visualization of binary files,” in *Proceedings of the 2013 Research in Adaptive and Convergent Systems (RACS ’13), Association for Computing Machinery*, New York, NY, USA, pp. 317-321, 2013, DOI:https://doi.org/10.1145/2513228.2513294.

[8] K.K. Pal and K.S. Sudeep, ”Preprocessing for image classification by convolutional neural networks,” *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, Bangalore, pp. 1778-1781, 2016, DOI:https://doi.org/10.1109/RTEICT.2016.7808140.

[9] D. Kornish, J. Geary, V. Sansing, S. Ezekiel, L. Pearlstein and L. Njilla, ”Malware Classification using Deep Convolutional Neural Networks,” *2018 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, Washington, DC, USA, pp. 1-6, 2018, DOI:https://doi.org/10.1109/AIPR.2018.8707429.

[10] JY. Kim, SJ. Bu, SB. Cho,” Malware Detection Using Deep Transferred Generative Adversarial Networks,” in *Liu D., Xie S., Li Y., Zhao D., El-Alfy ES. (eds) Neural Information Processing, - 24th International Conference, ICONIP 2017*, vol. 10634, pp. 556-564, 2017, DOI:https://doi.org/10.1007/978-3- 319-70087-8 58.

[11] H. Naeem, “Detection of Malicious Activities in Internet of Things Environment Based on Binary Visualization and Machine Intelligence,” *Wireless Pers Communication* 108, pp. 2609-2629, 2019, https://doi.org/10.1007/s11277-019-06540-6.

[12] Y. Qiwei, Z. Ruixun and S. XiuLi, ”CNN and RNN mixed model for image classification,” *MATEC Web of Conferences*, 277, 2019, DOI:https://doi.org/10.1051/matecconf/201927702001.

[13] H. Naeem, U. Farhan, N. M. Rashid, K. Shehzad, V. Danish, J. Sohail and S. Saqib, ”Malware Detection in Industrial Internet of Things based on Hybrid Image Visualization and Deep Learning Model,” *Ad Hoc Networks* 105, 2020. DOI:https://doi.org/10.1016/j.adhoc.2020.102154.

[14] G.S. Kumar, P. Bagane,” Detection of Malware Using Deep Learning Techniques,” *International journal of scientific & technology research*, vol. 9, issue 01, pp. 1688-1691, January 2020.

[15] D. Vasan, M. Alazab, S. Wassan, B. Safaei, Q. Zheng, “Image-Based malware classification using ensemble of CNN architectures (IMCEC),” *Computers & Security*, vol. 92, 2020, https://doi.org/10.1016/j.cose.2020.101748.

[16] K. Espoir, K. Chang, “Malware Images Classification Using Convolutional Neural Network,” *Journal of Computer and Communications*, vol. 6, pp. 153-158, 2018, https://doi.org/10.4236/jcc.2018.61016.

[17] Y. Lu, J. Li, “Generative Adversarial Network for Improving Deep Learning Based Malware Classification,” *2019 Winter Simulation Conference*, National Harbor, MD, USA, pp. 584-593, 2019, https://doi.org/10.1109/WSC40007.2019.9004932.

[18] A. Azab, M. Khasawneh, "MSIC: Malware Spectrogram Image Classification," in *IEEE Access*, Vol. 8, pp. 102007-102021, 2020, https://doi.org/10.1109/ACCESS.2020.2999320.

[19] J. Fu, J. Xue, Y. Wang, Z. Liu, & C. Shan,"Malware Visualization for Fine-Grained Classification," in *IEEE Access*, vol. 6, pp. 14510-14523, 2018, https://doi.org/10.1109/ACCESS.2018.2805301.

[20] M.U. Demirezen, “Image Based Malware Classification with Multimodal Deep Learning,” *International journal of information security science (IJISS)*, Vol.10, No.2, pp.42-59, 2021.

[21] Awan, M.J.; Masood, O.A.; Mohammed, M.A.; Yasin, A.; Zain, A.M.; Damaševˇcius, R.; Abdulkareem, K.H.,”Image-Based Malware Classification Using VGG19 Network and Spatial Convolutional Attention,” *Electronics 2021*, Vol. 10, No. 19, p. 2444.https:// doi.org/10.3390/electronics10192444.

[22] Malimg Dataset Based on grayscale images. Available at: https://www.kaggle.com/afagarap/malimg-dataset.

[23] K. Kosmidis and C. Kalloniatis,” Machine Learning and Images for Malware Detection and Classification,” in *Proceedings of the 21st Pan-Hellenic Conference on Informatics (PCI 2017), Association for Computing Machinery*, NY, USA, Article 5, pp. 1–6, 2017, https://doi.org/10.1145/3139367.3139400.

[24] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint, arXiv: 1409.1556, 2014.

[25] R. Kumar, Z. Xiaosong, R.U. Khan, I. Ahad, J. Kumar, “Malicious Code Detection based on Image Processing Using Deep Learning,” in *Proceedings of the 2018 International Conference on Computing and Artificial Intelligence (ICCAI 2018), Association for Computing Machinery*, NY, USA, pp. 81-85, 2018, https://doi.org/10.1145/3194452.3194459.

[26] D. Gavrilu, M. Cimpoeu, D. Anton, L. Ciortuz, “Malware detection using machine learning,” 2009 *International Multiconference on*

Computer Science and Information Technology, Mragowo, pp. 735-741, 2009. https://doi.org/10.1109/IMCSIT.2009.5352759.

[27] M. Jain, W. Andreopoulos, M. Stamp, "Convolutional neural networks and extreme learning machines for malware classification," *J ComputVirol Hack Tech 16*, pp. 229-244, 2020, https://doi.org/10.1007/s11416-020-00354-y.

[28] A.F. Agarap, "Towards building an intelligent anti-malware system: a deep learning approach using support vector machine (SVM) for malware classification," arXiv preprint, arXiv: 1801.00318.

[29] Sharma G.A., Singh K.J., Singh M.D., "A Deep Learning Approach to Image-Based Malware Analysis," in *Das H., Pattnaik P., Rautaray S., Li KC. (eds) Progress in Computing, Analytics and Networking. Advances in Intelligent Systems and Computing*, Vol. 1119, pp. 327-339, 2020, https://doi.org/10.1007/978-981-15-2414-1_33.

[30] S.S. Lad, A. C. Adamuthe, "Malware Classification with Improved Convolutional Neural Network Model," *International Journal of Computer Network and Information Security (IJCNIS)*, Vol.12, No.6, pp.30-43, 2020,https://doi.org/10.5815%2Fijcnis.2020.06.03.

[31] Y. Sravani, V. Raja, S. Selvin, F.D. Troia, and M. Stamp, "Deep Learning versus Gist Descriptors for Image-based Malware Classification," In *ICISSP*, pp. 553-561. 2018, https://doi.org/10.5220/0006685805530561.

[32] B. Niket, P. Prajapati, F. D. Troia, and M. Stamp, "Transfer learning for image-based malware classification," arXiv preprint arXiv: 1903.11551, 2019.

[33] Y. Songqing, "Imbalanced malware images classification: a CNN based approach," arXiv preprint, arXiv: 1708.08042 (2017).

[34] R. Mitsuhashi, and T. Shinagawa, "High-accuracy malware classification with a malware-optimized deep learning model," arXiv preprint, arXiv: 2004.05258, 2020.

[35] R. Vinayakumar, M. Alazab, K. Soman, P. Prabaharan, V. Sitalakshmi, "Robust Intelligent Malware Detection Using Deep Learning," *IEEE Access*, vol.7, pp. 46717-46738, 2019, https://doi.org/10.1109/ACCESS.2019.2906934.

[36] D. Indrawal, A. Sharma, "Multi-Module Convolutional Neural Network Based Optimal Face Recognition with Minibatch Optimization", *International Journal of Image, Graphics and Signal Processing (IJIGSP)*, Vol.14, No.3, pp. 32-46, 2022, DOI: 10.5815/ijigsp.2022.03.04

[37] B. Yadav, and S. Tokekar, "Malware Multi-Class Classification based on Malware Visualization using a Convolutional Neural Network Model", *International Journal of Information Engineering and Electronic Business(IJIEEB),* Vol.15, No.2, pp. 20-29, 2023. DOI:10.5815/ijieeb.2023.02.03.

[38] B. Yadav, and S. Tokekar, "Recent innovations and comparison of deep learning techniques in malware classification: a review," *International Journal of Information Security Science (IJISS),* Vol. *9, No.* 4, pp. 230-247, 2021.