



Enhanced Malware Family Classification via Image-Based Analysis Utilizing a Balance-Augmented VGG16 Model

Nagababu Pachhala^{1*}, Subbaiyan Jothilakshmi¹, Bhanu Prakash Battula²

¹ Department of Information Technology, Faculty of Engineering and Technology, Annamalai University, Annamalainagar 608002, Tamil Nadu, India

² Department of CSE, KKR & KSR Institute of Technology and Sciences, Guntur 522017, Andhra Pradesh, India

Corresponding Author Email: nagababupachhala2024@gmail.com

<https://doi.org/10.18280/ts.400534>

ABSTRACT

Received: 2 February 2023

Revised: 12 July 2023

Accepted: 9 August 2023

Available online: 30 October 2023

Keywords:

malware, deep learning, malicious software, malware detection, VGG16, malware families

The escalating sophistication and automation of malware generation techniques have led to an unprecedented proliferation of diverse and potent malicious software, thereby posing a considerable threat to individual, commercial, and digital security. Traditional detection systems often fall short in identifying these evolving threats, underscoring the critical necessity for advanced detection and classification strategies. Herein, we present an innovative deep learning approach for malware image analysis, employing a multi-layer VGG16 model—termed as MLVGGNET. We meticulously assess the performance of our proposed model using a representative dataset, embodying twenty-five distinct malware species, commonly known as the "Maling" dataset. Our proposed model is evaluated with state-of-the-art techniques. The model's performance metrics include recall, specificity, accuracy, and the F1 score. Our investigations reveal that the MLVGGNET model, particularly when enhanced with class balancing techniques, demonstrates superior performance over existing methodologies. Remarkably, the incorporation of class balancing in the benign class results in highly promising outcomes. Despite its relative simplicity, our proposed MLVGGNET model exhibits robust efficacy in photograph intrusion detection systems, as substantiated by our empirical results. This study thus underscores the potential of our model as an efficient tool for the precise detection and classification of malware, outpacing current approaches.

1. INTRODUCTION

Malware (also called malicious code) refers to all programmers developed to carry out some illicit [1] action. Because practically every household has at most one electrical gadget which a cyberattack may penetrate, the number of occurrences of such malicious applications has rapidly expanded over the previous several years. It is consequently vital to identify speedy and reliable strategies to discover and battle developing diseases. Malware samples developed in recent times frequently contain effective protections versus reverse engineering and various other forms of software inspection [2]. Prevarications, for instance, are employed rather regularly in changing malware [3, 4]. These are modification of syntactic language that took a programming like source as well as build a new programming that is difficult to studies [5-9] while keeping the original project's performance. The challenge of decrypting infection was added to the difficulty when prevarications are employed in concert with the computer optimization algorithms that are often incorporated in compilers. As a result, numerous steps in the disassembly procedure are typically decelerated or halted [10]. Due to such a, it is of the greatest necessity to hunt for strategies that act directly with raw combinations rather than depending on higher-level attributes that result through attempts at data manipulation. Obfuscations are widely exploited in ransomware [11-17], and their agreement is

established it more complicated to categorize freshly identified viruses into the groups into which it belongs. This approach, which is characterized as malware detection, is commonly performed with the assistance of machine learning programs.

These may range between realm [18], which involves human feature extraction so before preprocessing step, to computational modeling [19-25] models, which can sometimes function directly on the raw data and thus do not demand any antecedent feature engineering. The drawback of uncomplicated terms is that they demand domain expertise, which suggests both and resources are essential to examine the instances in the dataset prior continuing to the training step. This is one causes why superficial systems are not nearly as powerful as deeper solutions. The incorporation of human-engineered components, on the other hand, often helps it be easier for a person to comprehend both the model and the outcomes. The investment of this physical endeavor is high since new malicious software is developing frighteningly swiftly. Deep-learning algorithms can result in enhanced qualities from the information samples, minimizing the necessity for minute feature engineering or specific domain expertise. Because of its usefulness, deep learning has emerged as the dominant paradigm for identifying malware.

When compared to deeper models, deep learning algorithms have a stronger inclination to overfit when trained with fewer datasets [26], which is one of techniques' downsides. In fields such as programming analysis, particularly in categorizing

malware, this may be a worry since it demands a lot of resources and substantially longer time to collect enough samples with the proper ground truth. This problem often appears in other domains, such as photograph image recognition and computer vision categorization [27]. The problem of a shortage of training phase is straightforward to solve in vision since current data items may be made from traditional facilities by performing different semantics-preserving alterations to the photos, such as rotation, transformations in spatially, or chosen cropping. This provides to produce more data points without affecting the data's original relevance. Data augmentation comprises creating new knowledge by augmenting already obtained data, being a fundamental feature of deep learning. This crucial strategy component is originally offered in case number [28]. Reusing a section of an already training phase, typically the segment devoted with edge detection is another approach for decreasing the issues that models may identify with a tiny number of data points. These models could be built utilizing massive amounts of data, and thus may then be utilized for a new issue setup by the head of something like a modelling (the dense layering) and subsequently retraining the true leader while "keeping" it's the rest of the network. This technique is referred to as "repurposing." Most of the values are "frozen," resulting in the result, they do not display as free factors when employing this approach. This makes the operation less time-consuming and avoids the issue of requiring additional data for training. Learning is the term given to this approach [29].

This research employs a multi-layer variant of the VGG16 model to categorize multi-class malware data. In this paper, we present a method for classifying malware families that takes use of the Deep Neural Network (D.N.N.) developed by Visual Geometry Group using 16 layers (VGG16) [30]. The VGG16 is just a Deep Neural Network that is being offered by Visual Geometry Group and contains a total of 16 layers. Malware samples are initially encoded in pictures, for each byte of data represented by one shade of color inside a grayscale image. Transfer learning is employed to derive intermediate filter feature map (also known as bottleneck features) from the ImageNet dataset by employing the convolutional layers of VGG16 well before on the ImageNet dataset. This helps the convolution layer of VGG16 to become comfortable with the ImageNet dataset. The objectives of the paper as follows:

1. Develop a VGG16-based deep learning model for accurate classification of malware families.
2. Investigate the effectiveness of transfer learning with pre-trained VGG16 on ImageNet for improved malware detection.
3. Address obfuscation challenges in malware analysis to enhance classification accuracy.
4. Evaluate the proposed model's performance and compare it with existing methods for malware detection.

The remaining of the paper is arranged as follows section 2 covers the available literature, section 3 depicts the suggested study, section 4 discusses the experimental data, and section 5 ends the paper.

2. LITERATURE SURVEY

In the present situation, classic letter, and heuristic approaches for identifying computer viruses cannot deliver a

suitable high degree of detection for freshly and previously undisclosed kinds of malware. This is a situation as these techniques depend on whatever was before behavior patterns were established. The response to this issue will disclose if machine learning approaches can be applied to tackle the scenario at hand. It is achievable to boost the accuracy and resilience of malware detection by employing attempting to reduce deep learning methods blended with transfer learning approaches. This may be accomplished alone without expertise or help of a security specialist.

In their research, Rezende et al. [29] adopted a neural network design that they described dubbed transfer based on ResNet-50. This design was utilized in their presentation. Their RGB pictures had a size of 224×224 pixels and then were folded 10 times. The Glorot uniform approach was utilized for lot and gained and Adam optimization. After completing the training procedure, the model's efficiency has risen to 98.62 percent, strengthened by 750 periods of training. They also employed a procedure known as GIST characteristics with K.N.N. and set the iterations number to 4, which lead to a reliability of 97.48 percent. In addition, they applied bottleneck characteristics, which raised the accuracy results up to 98.0 percent.

As an element of both the data preparation step and the super model, Khan et al. [30] carried out a thorough investigation on domain adaptation for malware classification. During their investigation, they employed ResNet and GoogleNet to classify malicious software. The accuracy frequencies for Alexnet 18, 34, 50, 101, or 152 included 83 percent, 86.51%, 86.62%, 85.94 percent, and adaptive capacity percent, correspondingly. The accuracy scores that Resnet 18 got varied from 83% to 86.51%, 86.62%, 85.94%, and 87.98 percent, accordingly. The GoogleNet statistics revealed that the exactness was 84 percent.

An ensemble model was created by Vasani et al. [31], and both of its elements, VGG16 and ResNet-50, were employed in it. Both systems' performance has been tuned to the best potential level. We were able to decrease 80% of the total of the features in the dataset by applying P.C.A. and then feed them it in to a one-vs-all, multi-class SVM algorithm. This enabled us to perform a better job of categorising the data. They trained the CNN model for 100 to 200 periods, which result in an efficiency of 99.50% generally after they are perfectly alright their system for 50 epochs.

Yosinski et al. [32] showed a model with Fifteen categories and 7087 samples utilising multiple feature extraction approaches. The authors observed that their most accurate approaches led to the highest reliability, which was 97.47 percent.

To attain a reliability rate of 97% in their results, Nataraj et al. [33] employed feature extraction methodologies including GIST descriptor and machine learning approaches like K.N.N. Their strategy incorporates previously defined bi-gram averages as well as periodic feature categorizations. When it comes to this approach, the most essential thing to bear in mind is that if the enemy is aware of its qualities, there is a potential that they'll be able to create countermeasures and totally avoid detection.

Agarap [34] provided CNN or LSTM hybrid vehicles networks integrating SVM and several SVM composite architecture including deep-learning methods, each of which was applied in their tests. Convolution had a productivity of 77.22%, with GRU-SVM methods had a stability of achieve particular %, and thier MLP-SVM hybrid technique had a

productivity of 80.46%.

A CNN-LSTM optimization model with a special modification of the images was created in Akarsh et al. [35]. Some authors of this paper referred to the method as the CNN-LSTM control scheme. Specifically, they deployed a convolutional network (CNN) that consisted of two layers. They were connected to an Activation map with 60 memory chips and an F.C.N. stack of 20 tons. They used to have a trained model and classified pass in their networking. In the last experiment, the dependability spanned at 96.64% and 96.68%, relying on which splits were applied.

In a various study, Akarsh et al. [36] utilised 2 layers of a 1D CNN using just an LSTM for feature extraction, with 0.1 degradation and 70 secondary storages of an LSTM, in added to a premium strategy to their model. The outcomes of this research were positive. In their experiment, Kumar [37] upgraded the ResNet50 network by replacing a wholly connected thick network again after photographer's bottom layer that was previously educated on ImageNet. This was accomplished as part of their effort to increase Electronics 2021,10, and 24444 from 19. They discovered that this technique offered the greatest amount of precision. The entirely connected dense layer discoveries are given to the Softmax algorithm, which tackles categorizing risky software.

Vinayakumar et al. [38] presented a framework that they termed Ember. This approach utilised domain-level knowledge, differentiating attributes from processed transportable execution (P.E.) files and genre elements such as raw byte histogram.

According to Xiao et al. [39]'s work, an approach for categorizing malware named MalFCS was suggested. Within this technique, malware programs were displayed as probability chains utilizing structural entropy. Afterwards, deep convolution networks, usually known as CNNs, became applied to capture similarities from multiple networks that even a family shared. In the end, SVM was implemented to categorize viruses in agreement with both the features obtained from it.

The 'Bat Algorithm' was advocated for usage by Cui et al. [40], whereas the topic of dynamic picture resampling was discussed. Their purpose was to repair the imbalance in the given dataset, which they effectively achieved. This method, paired with data augmentation, was utilized to generate a CNN with an accuracy of 94.5%, and the conclusions of this experiment were reported.

A approach for data stability relying on an NSGA-evolutionary algorithms without equalization was suggested by Cui et al. [41]. This approach provided an accuracy of 92.1 percent; when paired with a single method, this accuracy rose to 96.1%; when combined with an inter algorithm, this recognition accuracy to 97.1 percent, which is the greatest accuracy conceivable.

Artificial neural machines, also known as E.L.M.s, and convolutional neural networks, popularly known as CNNs, were brought together by Jain et al. [42], who subsequently presented an ensemble model. When compared to the accuracy acquired by 2 CNN layers, which was 95.7 percent, the accuracy reached by a one CNN layer was 96.30 percent.

The hybrid visualization tool Naeem et al. [43] designed is based on deep intelligence and the Internet of Things. They could construct models with just an efficiency of up to 98.47% and 98.79% by experimenting with various picture ratios; nevertheless, to attain these conclusions, they were reliant on dynamic image features.

As part of a hybrid energy storage system, an identity system was suggested by Venkatraman et al. [44] to train the recommended hybrid CNNBiLSTM and CNN BiGRU models, expense and cost-insensitive learning strategies were applied. These models were then utilised to create the suggested hybrid CNNBiLSTM and CNN BiGRU models. Accuracy may range anywhere between 94.48 percent to 96.3 percent among all types, each giving a different variety of options and settings for the user to play with.

Vu et al. [45] built a Fully convolutional structure with improvements on the source photos, consisting of byte category, inclination angle, Hilbert, and entropy adjustments, and a hybrid scene utilising GIST and CNN-based designs. Likewise, they intended a CNN-based architecture with a reinforced photograph transition utilising CNN-based models. The correctness for their own GIST, because once implemented to grayscale images, was 94.27%, and then when especially in comparison the with contest, CNN's achievement utilising the hybrid vehicles image transition (H.I.T.) method is superior.

El-Shafai et al. [46] recommended a spyware inter structure which it utilises which was previously and perfection okay CNN concepts with transfer learning. VGG16 managed to earn the successfully complete again for virus acknowledgement job among all of the methodologies that had been analysed up to this time.

It recommended a 2 artificial neural network (ANN) antimalware strategic plan that utilised both file as well as picture features presented by Moussas and Andreatos [47]. Mousas and Andreatos developed this approach. The categories of spamware unpredictability were classified by the 2nd floor of ANNs leveraging functionalities of malware pictures. In contrast, the classifications of spamware ambiguity were resolved by the original level of ANNs of using file characteristics.

Roseline et al. [48] applied a night after work gown deep forest technique for rootkit detection and identification. The findings of that sort of work were presented in Virus Survey. The suggested methodology is data-independent and creates the ethnic divisions from the information it is provided as input. This is proffered as an alternative path to the standard methodology of obviously it depends on physically supplied feature descriptions. When recognising malware in a given situation, the recommended approach surpasses deeper neural networks due mainly to its use and of wide outfit layering and its comparatively small degree of complexity and high.

Verma et al. [49] recommended using a mix of the both first-order numerically extraction features and dark founder column (GLCM)-based second-data collection contour characteristic, both of which were categorised using ensemble techniques. This was selected to offer inside the structure of our inquiry. The modules based E.L.M. classification was applied to classify malware, attaining a result of 94.25% merely on Maling dataset.

Çayır et al. [50] established their approach to highlight just on different classifiers of capsule network (CapsNet). The CapsNet approach employs easy architectural engineering rather than just reliant on advanced CNN structures and procedures that are problematic when it comes to generating features. In addition, CapsNet does not demand deep learning framework, and the structure may be taught from either the bottom up with a reasonable amount of complexity.

Woźniak et al. [51] advised merging the recently created RNN-LSTM classifier with both the present NAdam

optimization approach. This concerns Android malware classification. Based on the results, the accuracy of the performance evaluation on 2 data sets was 99 percent.

According to Nisa et al. [52], showcase fusion is a strategy that combines features taken from pictures representing malware code blending parallelization spectral texture analysis with characteristics acquired from which was before neural network. along with AlexNet and Inception-v3 (SFTA). Classifiers include such support vector machines (SVM), kernel neural networks (K.N.N.), decisions trees (D.T.), but others are employed in the process of categorising the characteristics of malware.

Hemalatha et al. [53] linked the DenseNet framework with a repeating math course polynomial function to get impressive performance gains in recognising malware photos in the actual world. This was done by dealing with the problems posed by imbalanced data and applying the DenseNet model.

The researchers [54] applied a layered deep learning picture recognition technology in system threat detection as the ultimate way of recognising network breaches. This process was really the experts' last strategy for spotting network intrusions. The attributes of said network are reinterpreted the visuals with four independent streams. After that, the images are employed that will further educate and analyse the deep neural algorithm is defined is ResNet50, who has been previously offered training. The approach is proven using the dataset collections UNSW-NB15 and BOUNDdos, which may be obtained without price on the internet.

Pektas and Acarman [55] applied machine learning for detect virus using the application reference implementation (API) call graphs, that were then translated into a numerical set of features to reflect the virus's execution pathways.

The sequences the the Application Programming Interface (API) calls that have been called by programmes were kept as empty image-like matrices by D'Angelo et al. [56]. (API images). After that, autoencoders retrieved the most informative elements from these photographs. These attributes were then put into such an ANN-based classification to recognize malicious software.

Naeem et al. [57] took a raw Smartphone data and turned into a colour picture. They then placed this snapshot it into DCNN [58-60] system which attained a reliability of 97.81% on either a Leopard Mobile malware collection and 98.47 percent of total on a Skylight dataset.

Finally, applying supervised learning algorithms to pinpoint risky network and software incursions based on attributes that have transferred into visuals is gaining traction. Additionally, various machine learning algorithms and configurations are always being explored, updated, and applied. Considering this, as there is such huge range of frameworks for pattern recognition and a significant number of hyperparameters, extra investigation is important to discover whose solutions are the finest suited for cybersecurity.

3. PROPOSED WORK

In the proposed work illustrates the malware image data analysis using VGG16 architecture. In this section initially pre-processing has been made, data augmentation is done to expand the dataset to train the model and finally malware classification using VGG16 is presented the architecture is mentioned in Figure 1.

Pre-processing

Pre-processing is required to eliminate contaminants and noise from the images. Since not all the photographs in the collection are the same size, scaling the images is required. During this experimental effort, the pictures are resized to numerous dimensions, including 32, 64, 128, and 256 pixels in size.

Data augmentation

An approach known as “data augmentation” is used when a dataset does not include sufficient information to be of any value.

The Maling data, used to categorise malware, has 9339 examples from 25 distinct families. Unfortunately, the dispersion of samples among these 25 categories is totally skewed. Several procedures, including such measurement errors and up sampling, are employed to overcome this issue so when input data is delivered to a CNN model. Using this methodology, a whole new dataset is fabricated out of the already existing one. An image dataset is used as input for developing a new dataset, referred to as “picture data augmentation.” This process involves applying transformations to the dataset in question. To make images more attractive, you may change them by performing operations such as rotation, shearing, zooming, flipping them horizontally or vertically, and adjusting the degrees of brightness.

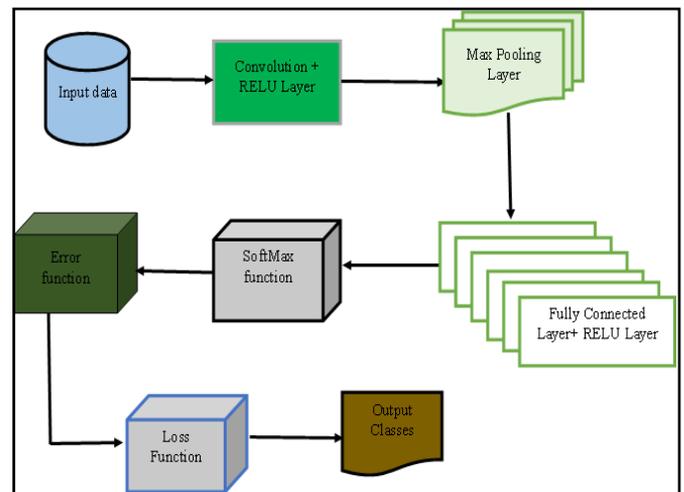


Figure 1. VGG16-based malware classification architecture

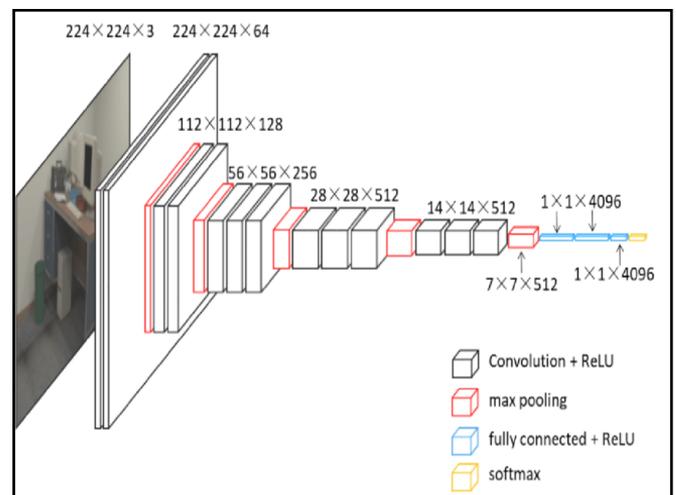


Figure 2. VGG-16 architecture

VGG16 must receive information in the form of three-dimensional (3-D) models. All three dimensions are measured in inches, including height, width, and depth. To be more exact, the VGG16 model is provided with a 32×32×1 input picture to process. This is followed by three convolution layers with filters of 50, 70, and 70, respectively before the user receives the final output image. The ReLU and Max Pooling layers come after each convolution layer. These layers are used to reduce the signal levels of the input signals, and they come after each convolution layer. When the flatten layer is applied, the input is first normalized and then split into 256 neurons of the F.C. layer. These 256 nerves will then be classed with 25 neurons using the activation function in SoftMax after deploying the flattened layer.

Algorithm 1: Malware Classification

Input: Maling Image dataset

Output: Classification of different malware

Step 1: Convolution Layer+RELU Layer

In the Convolution layer, the filters are applied to the original image. The RELU layer contributes to the reduction of the exponential increase in the amount of computing that is necessary to run the neural network.

$$N = [(a - x + 2g)/S] + 1$$

Step 2: The Max Pooling algorithm chooses the most significant feature from the area of the feature map covered by the filter.

$$(b_h - y + 1)/s \times (b_a - y + 1)/s \times b_c$$

Step 3: Step 1 and Step 2 are repeated three times.

Step 4: Fully Connected Layer with RELU Support.

In the Fully Connected Layer, the neuron will perform a linear mapping on the input vector using a weight matrix as the intermediate step. The number of weights equals (n*m) when there are n inputs and m outputs.

For the output node, (n+1)*m parameters are present.

Step 5: Step 4 is repeated three times.

Step 6: A vector of K absolute values may be converted into a matrix of K actual values that all add up to 1 using a SoftMax function.

$$P(y = j|\theta^{(i)}) = \frac{e^{\theta^{(i)}}}{\sum_{k=0}^K e^{\theta_k^{(i)}}}$$

Step 7: The Error function is used to minimize the errors.

$$E = \frac{1}{n} \sum_k \min_i d(C_i, G_k)$$

d=0 if $C_i = G_k$ else $d = 1$.

Step 8: The Loss function is the calculation that determines the gap that exists between both the output that the algorithm is now producing and the result that is anticipated.

$$E = \frac{1}{3} (\min_i d(C_i, G_1) + \min_i d(C_i, G_2) + \min_i d(C_i, G_3))$$

Step 9: Output Classes

Here Figure 2 illustrates the VGG-16 architecture. The fundamental ideas behind VGG-Nets are identical to those

behind regular CNNs; the distinguishing feature of this kind of approach is that it increases depth by employing an architecture with tiny (3x3) convolution filters. Roseline et al. [48] presented six varieties of VGG-Nets, ranging in the number of layers they contained from 16 to 24. In the scheme that we have proposed, we make use of a 16-layer VGG-Net called VGG16.

4. EXPERIMENTAL RESULTS

Dataset

The Maling [51] data includes pictures of virus generated from malware binaries. We applied deep learning algorithms for teaching them to categorise each kind of virus in turn, as well as the dataset is dubbed Maling [51]. The collection comprises viruses from 25 distinct families and consists of 9348 black and white photographs. The Maling dataset is uneven due to each family having a different number of samples to contribute, which is why the dataset itself is unbalanced. There is an imbalance in the distribution of classes within the dataset. For instance, 2949 pictures represent the Allapple malware family, but there are only 80 pictures that represent the Skintrim malware family. This illustrates a mismatch in the arrangement of classes only within dataset. It is responsible for keeping the same amount of training sets to basically the competence of knowledge trials, and it achieves this using picture enhancement techniques. The problem of uneven data distribution has been resolved thanks to the use of the data augmentation strategy, and each family of malicious software now contains a total of one thousand malware samples.

Table 1. VGG16 parameters

Parameter	Description
Input size	224×224
Color channels	3 (ROB)
Filter size	3×3
Activation function	Rectification Linear Unit (ReLU)
Pooling layer	Max Pooling
Max pooling size	2×2
Stride	2
Final layer	Softmax
Dropout	0
Total number of layers	16
Frozen layers	First 8 layers
Total Parameters	13,43,62,969
Trainable Parameters	13,43,62,969
Non-Trainable Parameters	0

Table 1 represents the VGG-16 parameters required for Training on the Maling dataset. The model uses input size 224×224. Size of the colour channel is 3, filter size is 3, uses the Activation function as RELU and uses Softmax as output layer. Total number of parameters and trainable parameters are 13, 43, 62, 969.

Figure 3 shows the accuracy of the proposed VGG16 malware model and existing DCNN, Alexnet-inceptionV3, and Autoencoder-ANN models. The proposed model produces more than 98% accuracy for classifying multiple malwares in the Maling dataset. Because the VGG16 model has undergone deep training, it can extract the most relevant features for the classification. Whereas existing DCNN produces 92% accuracy because it failed to grab all the required elements for classification, another model autoencoder based ANN model

gives the accuracy of 86% because it is not fit for classification of image data. Alex net and inception-v3 model offers 95% only. The proposed VGG16 model outperformed all existing models.

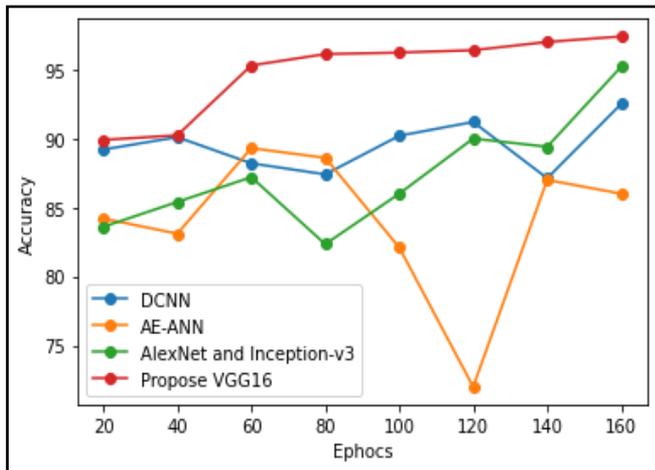


Figure 3. Accuracy

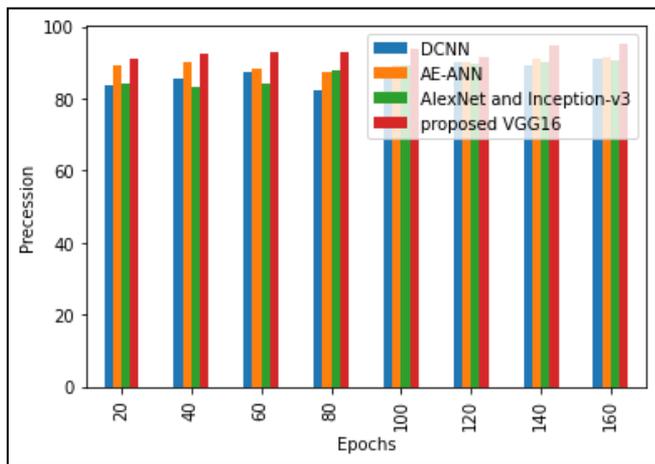


Figure 4. Precision

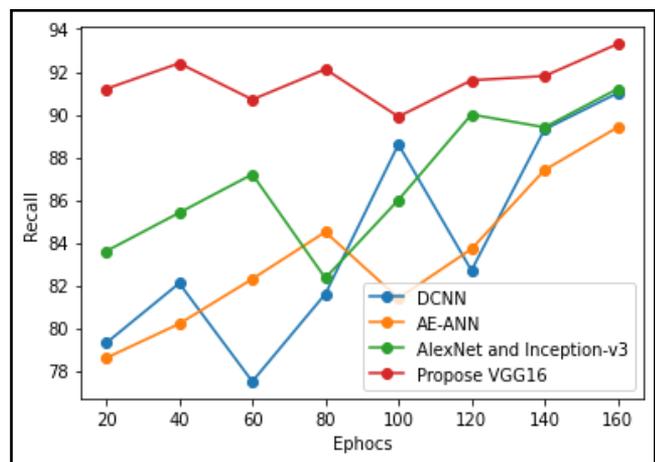


Figure 5. Recall

Here Figure 4 represents precision for existing DCNN, ANN, AlexNet, and proposed VGG16 models. DCNN is between 84% to 92% due to DCNN's inability to be spatially invariant to the input data. ANN model is unsuitable for image data but combined with the auto-encoder model, and it cannot

produce better results; it only gives precision between 91% and 86%. AlexNet is also not. It also provides 92% precision. Whereas the number of epochs increases, the model precision is decreased to 88%. On the other hand, the proposed VGG16 model outperformed all existing models. It gives a precision value of 94%. Initially, it starts at 90% while increasing the number of epochs. The precision of VGG16 also increases.

Figure 5 shows the recall of existing DCNN, ANN, and AlexNet and the proposed VGG6 model. The recall of DCNN varies between 79% to 90% because it fails to handle imbalanced data in Maling dataset. Some of the classes are imbalanced DCNN is not able to handle imbalanced data. Whereas ANN is also unable to handle imbalanced data, it only produces 88% of recall. AlexNet is not making good recall 88% only due to. Because the model is not very deep, it has difficulty scanning for all the characteristics, resulting in the production of models with poor performance. The proposed VGG16 model outperforms all existing models; it gives 93% recall because it can handle imbalanced data.

Figure 6 shows the performance of existing DCNN, ANN, AlexNet, and proposed VGG16 models concerning F-score. Owing to the DCNN model's inability to be spatially robust to the raw data, it received a score of 93% for accuracy. ANN is not apt for the image kind of data classification; AlexNet is not performing because it is not too deep to be adequately trained. At the same time, the Proposed VGG16 model outperformed than existing models. It gives a 96% F-score.

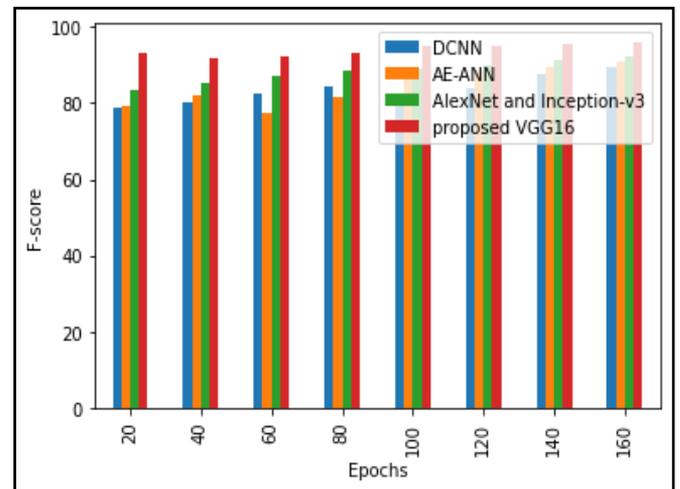


Figure 6. F-Score

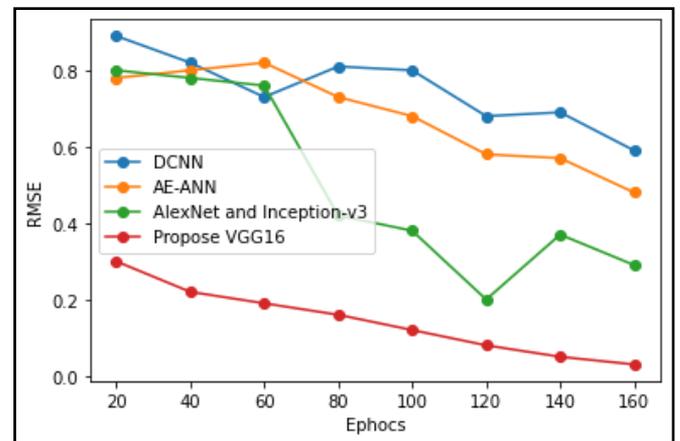


Figure 7. RMSE

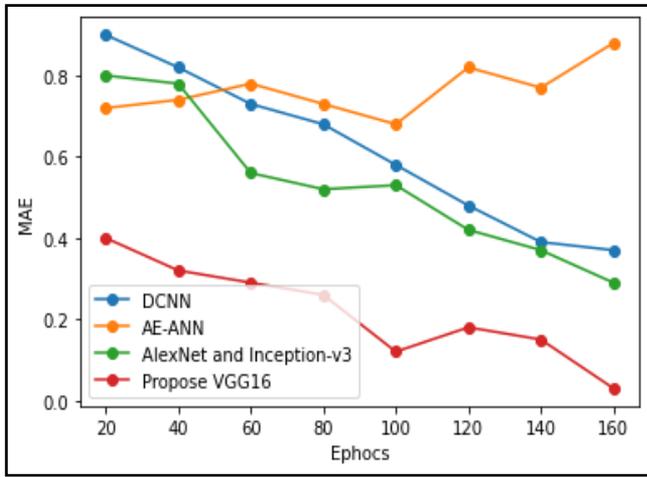


Figure 8. MAE

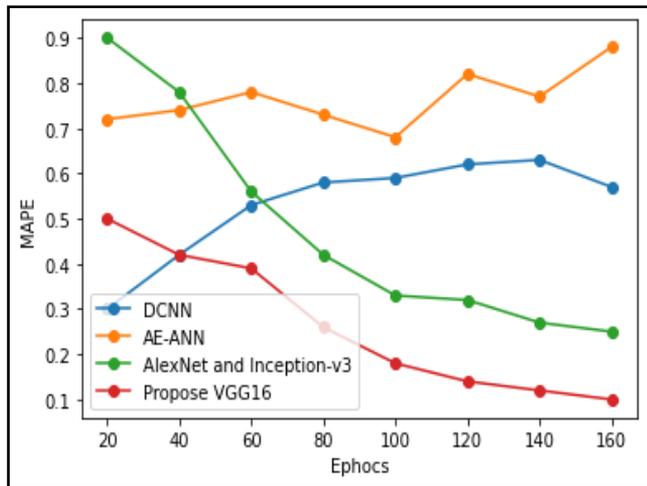


Figure 9. MAPE

The root-mean-square error, often abbreviated as RMSE, is indeed the standard deviation of a mistakes that emerge because when a prediction is made using a dataset. This is similar to M.S.E., that standing as “Mean Squared Error,” however when determining the validity of a model, the roots of the integer is considered. Figure 7 shows the RMSE comparison of proposed VGG16 and existing models in classifying multi-class malware data. The proposed model gives lower RMSE reaches from 0.3 to zero by a varying number of epochs. At the same time, the existing model RMSE is a bit higher than the proposed model.

MAE is an error that can be defined as the sum of all the deviations between the actual values and the projected values. Since this is an absolute difference, all negative numbers in the final tally are disregarded. Figure 8 compares the proposed VGG16 model and other models already used for classifying multi-class malware data. The suggested model achieves low M.A.E. levels beginning at 0.4 and getting closer and closer to zero as the number of epochs increases. On the other hand, era, the M.A.E. of such present model is somewhat greater than those of the recommended model.

The mean absolute percentage error (MAPE) normalizes the fundamental error over the data so that it may be compared across data sets of varying sizes has been shown in Figure 9. MAPE is the average of the absolute errors (errors) over the actual values (errors). For the categorization of data from multiple classes of malware. The proposed model can produce

low MAPE values, commencing at a value of 0.5 and growing ever closer to zero as the number of epochs increases. In contrast, the MAPE of the existing model has been shown to have a little higher value than that of the model that has been suggested.

Table 2. Analysis of existing and proposed models on Maling dataset

Epochs	Model	Data	Training Time (Mins.)	F-Score
160	DCNN	Raw Data	70	0.89
160	ANN	Raw Data	76	0.91
160	AlexNet	Raw Data	81	0.92
160	VGG16	Raw Data	93	0.96

Table 2 shows existing DCNN, ANN, and AlexNet models with training time and Fscore value. The proposed VGG16 model takes huge time for training but it gives better F-Score value. But other existing model gives lower F-score in addition to the suggested VGG16 model. The DCNN model was given a score of 93% for accuracy despite its failure to be geographically robust to the raw data. This was because it could not properly process the input. The artificial neural network (ANN) is not suited for the categorization of picture data; AlexNet is not performing well since it does not have enough depth to be trained correctly. At the same time, the newly proposed VGG16 model performed better than any of the models that were already in use. It results in an F-score of 96%.

Table 3. Analysis of proposed VGG16 and existing models on Maling dataset

Model	Accuracy	Precession	Recall	RMSE
DCNN	92	91	91	0.37
ANN	86	91	89	0.88
AlexNet	95	90	91	0.29
VGG16	97	95	93	0.03

Table 3 shows existing DCNN, ANN, and AlexNet models and proposed VGG16 model on Maling dataset with different performance parameters (accuracy, precession, recall and F-score) the proposed VGG16 model perform better compared to existing models.

5. CONCLUSIONS

Malware is among the most prevalent malware deployed by cyberattacks to conduct out assaults. Security specialists and antimalware software organizations are continually creating new methods to stop criminals form conducting assault, data theft, and causing harm to computer systems to keep control. This research says that the most successful way of classifying spyware is built on deep learning. In the existing, various researchers' strategies for categorizing malware were assessed by using Maling dataset, which has 9339 pieces of ransomware from 25 distinct malware families. We suggested the Multi-layered VGG16 model by integrating data pre-processing and supplementation approaches about just this dataset and considered existing models' computing resources and time needs. The recommended model is superior for malware categorization. The suggested model's 97.03% efficiency was notably superior to those produced by several current architecture such as DCNN, ANN, and AlexNet.

REFERENCES

- [1] Rieck, K., Trinius, P., Willems, C., Holz, T. (2011). Automatic analysis of malware behavior using machine learning. *Journal of Computer Security*, 19(4): 639-668. <https://doi.org/10.3233/JCS-2010-0410>
- [2] Awan, M.J., Farooq, U., Babar, H.M.A., Yasin, A., Nobanee, H., Hussain, M., Hakeem, O., Zain, A.M. (2021). Real-time DDoS attack detection system using big data approach. *Sustainability*, 13: 10743. <https://doi.org/10.3390/su131910743>
- [3] Ferooz, F., Hassan, M.T., Awan, M.J., Nobanee, H., Kamal, M., Yasin, A., Zain, A.M. (2021). Suicide bomb attack identification and analytics through data mining techniques. *Electronics*, 10: 2398. <https://doi.org/10.3390/electronics10192398>
- [4] Belbus, N.V., Yeo, S.S., Cho, E.S., Kim, J.A. (2008). Malware and antivirus deployment for enterprise I.T. security. In *Proceedings of the 2008 International Symposium on Ubiquitous Multimedia Computing*, Hobart, Australia, 13-15 October 2008. <https://doi.org/10.1109/UMC.2008.58>
- [5] Azeez, N.A., Salaudeen, B.B., Misra, S., Damaševičius, R., Maskeliūnas, R. (2020). Identifying phishing attacks in communication networks using URL consistency features. *International Journal of Electronic Security and Digital Forensics*, 12(2): 200-213. <https://doi.org/10.1504/IJESDF.2020.106318>
- [6] Yong, B., Wei, W., Li, K.C., Shen, J., Zhou, Q., Wozniak, M., Połap, D., Damaševičius, R. (2022). Ensemble machine learning approaches for webshell detection in Internet of things environments. *Transactions on Emerging Telecommunications Technologies*, 33(6): e4085. <https://doi.org/10.1002/ett.4085>
- [7] Mohammed, M.A., Ibrahim, D.A., Salman, A.O. (2021). Adaptive intelligent learning approach based on visual anti-spam email model for multi-natural language. *Journal of Intelligent Systems*, 30(1): 774-792. <https://doi.org/10.1515/jisys-2021-0045>
- [8] Rehman, A.A., Awan, M.J., Butt, I. (2018). Comparison and evaluation of information retrieval models. *VFAST Transactions on Software Engineering*, 6(1): 7-14. <https://doi.org/10.21015/vtse.v13i1.496>
- [9] Rhode, M., Burnap, P., Jones, K. (2018). Early-stage malware prediction using recurrent neural networks. *Computers & Security*, 77: 578-594. <https://doi.org/10.1016/j.cose.2018.05.010>
- [10] Alam, T.M., Awan, M.J. (2018). Domain analysis of information extraction techniques. *International Journal of Multidisciplinary Sciences and Engineering*, 9(6).
- [11] Adebayo, O.S., Abdul Aziz, N. (2019). Improved malware detection model with apriori association rule and particle swarm optimization. *Security and Communication Networks*, 2019. <https://doi.org/10.1155/2019/2850932>
- [12] Ali, Y., Farooq, A., Alam, T.M., Farooq, M.S., Awan, M.J., Baig, T.I. (2019). Detection of schistosomiasis factors using association rule mining. *IEEE Access*, 7: 186108-186114. <https://doi.org/10.1109/ACCESS.2019.2956020>
- [13] Akram, R.N., Chen, H.H., Lopez, J., Sauveron, D., Yang, L.T. (2018). Security, privacy and trust of user-centric solutions. *Future Generation Computer Systems*, 80: 417-420. <https://doi.org/10.1016/j.future.2017.11.026>
- [14] Anderson, H.S., Kharkar, A., Filar, B., Roth, P. (2017). Evading machine learning malware detection. In *Proceedings of the Black Hat, Las Vegas, NV, U.S.A.*, 22-27 July 2017, pp. 1-6.
- [15] Azeez, N.A., Odufuwa, O.E., Misra, S., Oluranti, J., Damaševičius, R. (2021). Windows PE malware detection using ensemble learning. In *Informatics*, 8(1): 10. <https://doi.org/10.3390/informatics8010010>
- [16] Khalaf, B.A., Mostafa, S.A., Mustapha, A., Mohammed, M.A., Mahmoud, M.A., Al-Rimy, B.A.S., Razak, S.A., Elhoseny, M., Marks, A. (2021). An adaptive protection of flooding attacks model for complex network environments. *Security and Communication Networks*, 2021: 1-17. <https://doi.org/10.1155/2021/5542919>
- [17] Anam, M., Hussain, M., Nadeem, M.W., Javed Awan, M., Goh, H.G., Qadeer, S. (2021). Osteoporosis prediction for trabecular bone using machine learning: A review. *Computers, Materials & Continua (CMC)*, 67(1). <https://doi.org/10.32604/cmc.2021.013159>
- [18] Azizan, A.H., Mostafa, S.A., Mustapha, A., Foozy, C.F.M., Wahab, M.H.A., Mohammed, M.A., Khalaf, B.A. (2021). A machine learning approach for improving the performance of network intrusion detection systems. *Annals of Emerging Technologies in Computing (AETiC)*, 5(5): 201-208. <https://doi.org/10.33166/AETiC.2021.05.025>
- [19] Gupta, M., Jain, R., Arora, S., Gupta, A., Javed Awann, M.J., Chaudhary, G., Nobanee, H. (2021). AI-Enabled COVID-19 outbreak analysis and prediction. *Indian States Vs. Union Territories, Computers, Materials & Continua*, 67(1): 933-950. <https://doi.org/10.32604/cmc.2021.014221>
- [20] Damaševičius, R., Venčkauskas, A., Toldinas, J., Grigaliūnas, Š. (2021). Ensemble-based classification using neural networks and machine learning models for windows pe malware detection. *Electronics*, 10(4): 485. <https://doi.org/10.3390/electronics10040485>
- [21] Awan, M.J., Yasin, A., Nobanee, H., Ali, A.A., Shahzad, Z., Nabeel, M., Zain, A.M., Shahzad, H.M.F. (2021). Fake news data exploration and analytics. *Electronics*, 10(19): 2326. <https://doi.org/10.3390/electronics10192326>
- [22] Lal, S., Rehman, S.U., Shah, J.H., Meraj, T., Rauf, H.T., Damaševičius, R., Mohammed, M.A., Abdulkareem, K.H. (2021). Adversarial attack and defence through adversarial training and feature fusion for diabetic retinopathy recognition. *Sensors*, 21(11): 3922. <https://doi.org/10.3390/s21113922>
- [23] Alharbi, A., Alosaimi, W., Alyami, H., Rauf, H.T., Damaševičius, R. (2021). Botnet attack detection using local global best bat algorithm for industrial internet of things. *Electronics*, 10(11): 1341. <https://doi.org/10.3390/electronics10111341>
- [24] Mahdavarfar, S., Ghorbani, A.A. (2019). Application of deep learning to cybersecurity: A survey. *Neurocomputing*, 347: 149-176. <https://doi.org/10.1016/j.neucom.2019.02.056>
- [25] Conti, G., Dean, E., Sinda, M., Sangster, B. (2008). Visual reverse engineering of binary and data files. In *Proceedings of the International Workshop on Visualization for Computer Security*, Cambridge, MA, U.S.A., 15 September 2008, pp. 1-17. https://doi.org/10.1007/978-3-540-85933-8_1
- [26] Nagi, A.T., Awan, M.J., Javed, R., Ayesha, N. (2021). A

- Comparison of two-stage classifier algorithm with ensemble techniques on detection of diabetic retinopathy. In Proceedings of the 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA), Riyadh, Saudi Arabia, 6-7 April 2021, pp. 212-215. <https://doi.org/10.1109/CAIDA51941.2021.9425129>
- [27] Abdullah, A., Awan, M., Shehzad, M., Ashraf, M. (2020). Fake news classification bimodal using convolutional neural network and long short-term memory. *International Journal of Emerging Technologies in Learning (IJET)*, 11(2): 209-212.
- [28] Mujahid, A., Awan, M.J., Yasin, A., Mohammed, M.A., Damaševičius, R., Maskeliūnas, R., Abdulkareem, K.H. (2021). Real-time hand gesture recognition based on deep learning YOLOv3 model. *Applied Sciences*, 11(9): 4164. <https://doi.org/10.3390/app11094164>
- [29] Rezende, E., Ruppert, G., Carvalho, T., Ramos, F., De Geus, P. (2017). Malicious software classification using transfer learning of resnet-50 deep neural network. In Proceedings of the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, 18-21 December 2017, pp. 1011-1014. <https://doi.org/10.1109/ICMLA.2017.00-19>
- [30] Khan, R.U., Zhang, X., Kumar, R. (2019). Analysis of ResNet and GoogleNet models for malware detection. *Journal of Computer Virology and Hacking Techniques*, 15: 29-37. <https://doi.org/10.1007/s11416-018-0324-z>
- [31] Vasan, D., Alazab, M., Wassan, S., Safaei, B., Zheng, Q. (2020). Image-based malware classification using ensemble of CNN architectures (IMCEC). *Computers & Security*, 92: 101748. <https://doi.org/10.1016/j.cose.2020.101748>
- [32] Yosinski, J., Clune, J., Bengio, Y., Lipson, H. (2014). How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems*, 27.
- [33] Nataraj, L., Karthikeyan, S., Jacob, G., Manjunath, B.S. (2011). Malware images: Visualization and automatic classification. In Proceedings of the 8th International Symposium on Visualization for Cyber Security, Pittsburgh, PA, U.S.A., 20 June 2011, pp. 1-7. <https://doi.org/10.1145/2016904.2016908>
- [34] Agarap, A.F. (2017). Towards building an intelligent anti-malware system: A deep learning approach using support vector machine (SVM) for malware classification. *arXiv Preprint arXiv:1801.00318*. <https://doi.org/10.48550/arXiv.1801.00318>
- [35] Akarsh, S., Poornachandran, P., Menon, V.K., Soman, K. (2019). A detailed investigation and analysis of deep learning architectures and visualization techniques for malware family identification. In *Cybersecurity and Secure Information Systems*. Springer: New York, NY, U.S.A., 2019, pp. 241-286. https://doi.org/10.1007/978-3-030-16837-7_1
- [36] Akarsh, S., Simran, K., Poornachandran, P., Menon, V.K., Soman, K. (2019). Deep learning framework and visualization for malware classification. In Proceedings of the 2019 5th International Conference on Advanced Computing Communication Systems (ICACCS), Coimbatore, India, 15-16 March 2019, pp. 1059-1063. <https://doi.org/10.1109/ICACCS.2019.8728471>
- [37] Kumar, S. (2021). MCFT-CNN: Malware classification with fine-tune convolution neural networks using traditional and transfer learning in Internet of Things. *Future Generation Computer Systems*, 125: 334-351. <https://doi.org/10.1016/j.future.2021.06.029>
- [38] Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. *IEEE Access*, 7: 46717-46738. <https://doi.org/10.1109/ACCESS.2019.2906934>
- [39] Xiao, G., Li, J., Chen, Y., Li, K. (2020). MalFCS: An effective malware classification framework with automated feature extraction based on deep convolutional neural networks. *Journal of Parallel and Distributed Computing*, 141: 49-58. <https://doi.org/10.1016/j.jpdc.2020.03.012>
- [40] Cui, Z., Xue, F., Cai, X., Cao, Y., Wang, G.G., Chen, J. (2018). Detection of malicious code variants based on deep learning. *IEEE Transactions on Industrial Informatics*, 14(7): 3187-3196. <https://doi.org/10.1109/TII.2018.2822680>
- [41] Cui, Z., Du, L., Wang, P., Cai, X., Zhang, W. (2019). Malicious code detection based on CNNs and multi-objective algorithm. *Journal of Parallel and Distributed Computing*, 129: 50-58. <https://doi.org/10.1016/j.jpdc.2019.03.010>
- [42] Jain, M., Andreopoulos, W., Stamp, M. (2021). CNN vs ELM for image-based malware classification. *arXiv Preprint arXiv:2103.13820*. <https://doi.org/10.48550/arXiv.2103.13820>
- [43] Naem, H., Ullah, F., Naem, M.R., Khalid, S., Vasan, D., Jabbar, S., Saeed, S. (2020). Malware detection in industrial internet of things based on hybrid image visualization and deep learning model. *Ad Hoc Networks*, 105: 102154. <https://doi.org/10.1016/j.adhoc.2020.102154>
- [44] Venkatraman, S., Alazab, M., Vinayakumar, R. (2019). A hybrid deep learning image-based analysis for effective malware detection. *Journal of Information Security and Applications*, 47: 377-389. <https://doi.org/10.1016/j.jisa.2019.06.006>
- [45] Vu, D.L., Nguyen, T.K., Nguyen, T.V., Nguyen, T.N., Massacci, F., Phung, P.H. (2019). A convolutional transformation network for malware classification. In 2019 6th NAFOSTED Conference on Information and Computer Science (NICS), pp. 234-239. <https://doi.org/10.1109/NICS48868.2019.9023876>
- [46] El-Shafai, W., Almomani, I., AlKhayer, A. (2021). Visualized malware multi-classification framework using fine-tuned CNN-based transfer learning models. *Applied Sciences*, 11(14): 6446. <https://doi.org/10.3390/app11146446>
- [47] Moussas, V., Andreatos, A. (2021). Malware detection based on code visualization and two-level classification. *Information*, 12(3): 118. <https://doi.org/10.3390/info12030118>
- [48] Roseline, S.A., Geetha, S., Kadry, S., Nam, Y. (2020). Intelligent vision-based malware detection and classification using deep random forest paradigm. *IEEE Access*, 8: 206303-206324. <https://doi.org/10.1109/ACCESS.2020.3036491>
- [49] Verma, V., Muttoo, S.K., Singh, V.B. (2020). Multiclass malware classification via first-and second-order texture statistics. *Computers & Security*, 97: 101895. <https://doi.org/10.1016/j.cose.2020.101895>
- [50] Çayır, A., Ünal, U., Dağ, H. (2021). Random CapsNet forest model for imbalanced malware type classification task. *Computers & Security*, 102: 102133.

- <https://doi.org/10.1016/j.cose.2020.102133>
- [51] Woźniak, M., Siłka, J., Wiczorek, M., Alrashoud, M. (2020). Recurrent neural network model for IoT and networking malware threat detection. *IEEE Transactions on Industrial Informatics*, 17(8): 5583-5594. <https://doi.org/10.1109/TII.2020.3021689>
- [52] Nisa, M., Shah, J.H., Kanwal, S., Raza, M., Khan, M.A., Damaševičius, R., Blažauskas, T. (2020). Hybrid malware classification method using segmentation-based fractal texture analysis and deep convolution neural network features. *Applied Sciences*, 10(14): 4966. <https://doi.org/10.3390/app10144966>
- [53] Hemalatha, J., Roseline, S.A., Geetha, S., Kadry, S., Damaševičius, R. (2021). An efficient densenet-based deep learning model for malware detection. *Entropy*, 23(3): 344. <https://doi.org/10.3390/e23030344>
- [54] Toldinas, J., Venčkauskas, A., Damaševičius, R., Grigaliūnas, Š., Morkevičius, N., Baranauskas, E. (2021). A novel approach for network intrusion detection using multistage deep learning image recognition. *Electronics*, 10(15): 1854. <https://doi.org/10.3390/electronics10151854>
- [55] Pektaş, A., Acarman, T. (2020). Deep learning for effective Android malware detection using API call graph embeddings. *Soft Computing*, 24: 1027-1043. <https://doi.org/10.1007/s00500-019-03940-5>
- [56] D'Angelo, G., Ficco, M., Palmieri, F. (2020). Malware detection in mobile environments based on Autoencoders and API-images. *Journal of Parallel and Distributed Computing*, 137: 26-33. <https://doi.org/10.1016/j.jpdc.2019.11.001>
- [57] Naeem, H., Ullah, F., Naeem, M.R., Khalid, S., Vasan, D., Jabbar, S., Saeed, S. (2020). Malware detection in industrial internet of things based on hybrid image visualization and deep learning model. *Ad Hoc Networks*, 105: 102154. <https://doi.org/10.1016/j.adhoc.2020.102154>
- [58] Nisa, M., Shah, J.H., Kanwal, S., Raza, M., Khan, M.A., Damaševičius, R., Blažauskas, T. (2020). Hybrid malware classification method using segmentation-based fractal texture analysis and deep convolution neural network features. *Applied Sciences*, 10(14): 4966. <https://doi.org/10.3390/app10144966>
- [59] Arepalli, P.G., Khetavath, J.N. (2023). An IoT framework for quality analysis of aquatic water data using time-series convolutional neural network. *Environmental Science and Pollution Research*, 1-20. <https://doi.org/10.1007/s11356-023-27922-1>
- [60] Arepalli, P.G., Naik, K.J. (2023). A deep learning-enabled IoT framework for early hypoxia detection in aqua water using light weight spatially shared attention-LSTM network. *The Journal of Supercomputing*, 1-30. <http://doi.org/10.1007/s11227-023-05580-x>