Attention-based convolutional neural network deep learning approach for robust malware classification

Ravi, Vinayakumar; Alazab, Mamoun

Published in:
Computational Intelligence

DOI:
10.1111/coin.12551

E-pub ahead of print: 01/01/2022

Document Version
Peer reviewed version

Noname manuscript No.
(will be inserted by the editor)

Attention-based Convolutional Neural Network Deep learning Approach for Robust Malware Classification

Vinayakumar Ravi · **Mamoun Alazab**

the date of receipt and acceptance should be inserted later

Abstract Recently, transforming windows files into images and its analysis using machine learning and deep learning have been considered as a state-of-the art works for malware detection and classification. This is mainly due to the fact that image-based malware detection and classification is platform independent, and the recent surge of success of deep learning model performance in image classification. Literature survey shows that convolutional neural network (CNN) deep learning methods are successfully employed for image-based windows malware classification. However, the malwares were embedded in a tiny portion in the overall image representation. Identifying and locating these affected tiny portions is important to achieve a good malware classification accuracy. In this work, a multi-headed attention based approach is integrated to a CNN to locate and identify the tiny infected regions in the overall image. A detailed investigation and analysis of the proposed method was done on a malware image dataset. The performance of the proposed multi-headed attention-based CNN approach was compared with various non-attention-CNN-based approaches on various data splits of training and testing malware image benchmark dataset. In all the data-splits, the attention-based CNN method outperformed non-attention-based CNN methods while ensuring computational efficiency. Most importantly, most of the methods show consistent performance on all the data splits of training and testing and that illuminates multi-headed attention with CNN model's generalizability to perform on the diverse datasets. With less number of trainable parameters, the proposed method has achieved an accuracy of 99% to classify the 25 malware families and performed better than the existing non-attention based methods. The proposed method can be applied on any operating system and it has the capability to detect packed malware, metamorphic malware, obfuscated malware, malware family variants, and polymorphic malware. In addition, the proposed method is malware file agnostic and avoids usual methods such as disassembly, de-compiling, de-obfuscation or execution of the malware binary in a virtual environment in detecting malware and classifying malware into their malware family.

Vinayakumar Ravi
Center for Artificial Intelligence, Prince Mohammad Bin Fahd University,
Khobar, Saudi Arabia.
E-mail: vravi@pmu.edu.sa

Mamoun Alazab
College of Engineering, IT and Environment, Charles Darwin University, NT, Australia.
E-mail: alazab.m@ieee.org

Keywords Cybersecurity, Cybercrime, Malware classification, Malware image representation, Malware image analysis, Deep learning, Convolutional neural network, Attention.

1 Introduction

With the internet growing exponentially, cyberspace and its users are facing tremendous threats from cybercriminals. Cybercriminals use malware as a tool to infect a computer. Malware, short for malicious software, refers to any program or file designed with an intent to damage or exploit a computer, server, or network. Cybercriminals typically use it to steal sensitive information like personal data such as passwords, emails, location or browsing history, financial data, and health-related data. Since its inception more than 30 years ago, malware has found various methods of attack. Malware related activities can mainly be performed for financial or political gain and also to overcome the fear of competition. Novel malware creation and hiding approaches are emerging to constantly evade the malware detection and successfully install the malware in victim machines. The AV-TEST Institute has registered an increasing number of malicious programs over the past years[1]. The working from home situation due to COVID-19 may even pose more risk to malware infection, as the devices are exposed to the public internet with few security controls in a home network environment. These report numbers are quite alarming that the effective and advanced malware detection and identification techniques are required to combat the ever-growing malware.

There are different types of malware and some are Backdoors, Spyware, Adware, Worm, Virus, Trojans and more. Based on the operating system platform such as Windows, Linux, Android, and others, the malwares can also be classified. Further, malwares are classified into families depending on their functionality. These families include many variants. These variants are self-similar in nature and perform almost the same function.

Initial days, signature-based malware detection was popular and most of the available anti-virus systems in the industries are signature-based systems. In signature-based malware detection, raw binaries were analyzed and built signature based on n-grams [1]. This type of approach is accurate and simple. However, it requires the signature database to be updated continuously to handle new malware or variants of the existing malware. Static analysis, dynamic analysis, hybrid analysis and heuristic based techniques were the most commonly employed standard approaches for malware analysis [2]. Overall, all these methods involve scanning systems with the aim to identify malicious or illegitimate activities. Next, the security admin looks into the files that are under suspicion and these files are quarantined or the vulnerable system is patched with an update. All the methods are often slow to react to new attacks and threats. Static analysis is an approach that analyzes the programming codes and builds control flow graphs without program execution. However, static analysis fails to handle code obfuscation. Dynamic analysis executes the malware in a virtual environment or simulators or sandbox and analyzes its execution trace (behavior analysis). It is a promising approach to handle code obfuscation but complex, time consuming, and computationally expensive. Recent days, to take advantage of both static analysis and dynamic analysis, both the approaches were combined and called hybrid analysis. Recent literature survey shows a rapid increase in the number of malware due to the advancement in technologies and hackers wanting to get into the networks using malicious programs with the aim to compromise the connected devices and use them to infect others also [3]. Also, hackers are finding new ways and advancing their malicious programs to evade defense mechanisms. Most importantly, the existing studies show that these malwares are variants of existing malware family [4]. Mostly, the variants are generated by making changes to the malware code or by using executable packers. Also, complex techniques such as encryption and decryption, compression and hybrid of both compression and encryption

[1] https://www.av-test.org/en/statistics/malware/

and decryption techniques are also employed to create variants of the existing malware family. The variants of the same malware family exhibit structural and visual similarity.

Security research straggling towards developing a robust malware detection and classification approach that can handle strong encryption, code obfuscation, polymorphic, and metamorphic [5]. Malware classification is the task of identifying the family for unknown malware. It is considered to be an important task as there is a rapid increase in malware families in recent days and to understand the specific function of malware, a malware family name is required. Most of the available malware detection and classification was focused on a single operating system for example either for Windows or Linux. However, these malwares are quickly expanding towards other environments such as Android, OS X, etc. Thus, there is a requirement of a single detection system that can be used for all operating systems. One such approach is to transform the malware into images and employ data-driven approaches for image analysis [6]. This type of approach is platform independent and it works really well on the data samples that are similar in structure. Also, the image-based malware detection is fast because it does not require disassembly or execution in a virtual environment. This method is faster than both static analysis and dynamic analysis. Most importantly, the image-based malware detection approaches can handle packed malware. Though the malware is packed, there is still a visual similarity among the packed malware variants. Some of the variants of the malware family are shown in Figure 1.

Initially, the malwares were transformed into grayscale images and GIST feature with k-nearest neighbour classifier was employed for malware classification by [4]. The detailed investigation and analysis shown by the author and also the Malimg dataset was made publicly available for research. Due to the recent surge of deep learning [7], [8], mainly the success of CNN, there are many methods proposed by researchers and evaluated for malware classification using Malimg dataset. In addition, deep learning-based approaches performed better than the other methods for various applications [9], [10], [11], [12]. The detailed literature survey included in Section 2. Most of the existing methods are based on CNN or feature extraction with machine learning classifiers for malware detection and classification. Recent days, deep learning-based methods are effective for malware detection and classification based on transforming the malwares into images. This is mainly due to the reason that these methods completely avoid manual feature engineering approaches. As the manual feature engineering approaches require domain experts and do not generalize well for unseen and complex data [1]. However, these methods are not effective in identifying the tiny part of the infected regions in the malware image. Mostly, the malware authors make only small changes to the existing files. Though CNN is able to learn better feature representation on image by passing the filters across the image, the performance towards identifying the infected regions is very limited. To improve the performance of the CNN on malware image-based tasks, security researchers followed increasing the width and depth of the CNN network [13, 14]. These types of methodologies result in higher trainable parameters and then increase the computational cost. Also, these models require high hardware resources for training a model. In addition, there may be the possibility that these types of models end up in overfitting and convergence problems. In this work, we propose an end-to-end and lightweight multi-headed attention deep learning-based approach for malware image classification. Limited research has been done to integrate attention to CNN in malare image classification. This enhances the representation power of convolutional features and directs the learning towards only the important region of the malware. The proposed approach transforms malware into grayscale images and later these were passed into deep learning for automatic feature extraction and classification. Integrating attention to CNN helps in learning relevant features for effective feature representation and identification of smaller parts of the malware regions in the overall image. The proposed method has performed better than the non-attention based methods and achieves 99% accuracy with 80% training and 20% testing of Malimg malware image dataset. The contributions of this work are:

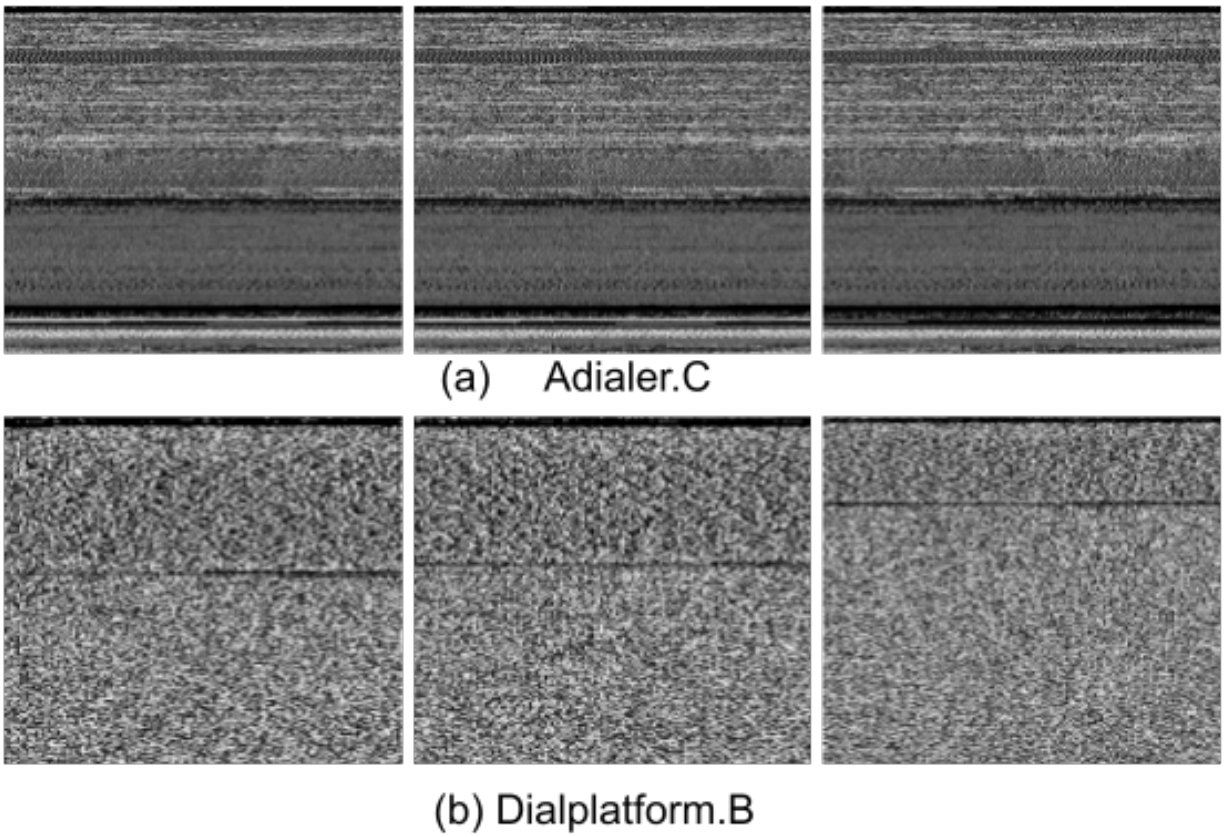

Fig. 1: Adialer.C and Dialplatform.B packed malware variants.

1. The work proposes deep learning image-based architecture and demonstrates the efficiency of using attention with CNN in malware family classification.
2. Detailed investigation and analysis of the proposed approach and non-attention-based method performances were shown on various data splits in training and testing of Malimg malware image dataset.
3. The performance of the proposed method is compared with non-attention CNN and detailed results are reported for all the data splits of training and testing of Malimg malware image dataset.
4. A detailed comparative study is shown for the proposed method with various existing methods and all the existing methods are based on non-attention.
5. The performance of the proposed method is compared with the existing approach such as spatial pyramid pooling network (SPP) and CNN-LSTM on all the data-splits of training and testing of Malimg malware image dataset.

The rest of the paper is organized as follows: Section 2 includes the literature survey of image-based malware classification using machine learning and deep learning. The proposed work details are included in Section 3. Details of the image-based malware dataset i.e. Malimg is included in Section 4. The performance

metrics information is included in Section 5. Results and discussion in Section 6 and conclusion is in Section 7.

2 Literature Survey

This section discusses some of the existing works on transforming the malware into images and following analysis and classification of images using machine learning and deep learning [6]. Most of the existing studies reported that deep learning based approaches were better than machine learning. This is mainly due to the reason that deep learning does not require any feature engineering and most importantly achieves better performance compared to traditional machine learning approaches.

Visualization is important in Cybersecurity and there are many tools such as text editors and binary editors available to visualize and manipulate binary data. But there is less work in malware visualization. Though there were many methods proposed by security researchers for malware visualization, none of the methods were robust. These methods are not platform independent and don't work really well in detecting and classifying the malware variants. To overcome these limitations, Nataraj et al. [4] has proposed an end-to-end malware classification approach. This contains malware visualization by transforming the malware bytes into images and followed by GIST feature extraction with KNN classifier for classification. Also, the dataset Malimg has been made publicly available for research purposes. Later there were many studies on malware classification with different feature engineering and different classifiers using Malimg dataset to enhance the performance. These methods were not efficient and not generalizable. Most importantly, feature engineering is a daunting task as the number of malware variants rapidly increases every day and to cope with the new malware, the trained model has to be retrained continuously. To overcome feature engineering, in recent days deep learning approaches were employed and these methods are robust compared to machine learning and most importantly have the capability to learn optimal feature representation and achieve better performances.

The malware images were fed into CNN for classifying the malware. Training of the deep learning algorithm was done with different kernel and data size whereas the evaluation of this work was done using ROC AUC [15]. The classification of malware images achieved an higher accuracy and AUC equal to 0.9973. Deep learning framework to classify malware is proposed in [16]. Initially, malware was converted into images. This framework uses deep CNN for malware classification and has achieved 91.7% in this work. Deep learning based intelligent anti-malware system was proposed in [17]. Malimg dataset which is a collection of malware images was used in this work. A L2SVM was used along with deep learning model for multi-class classification. CNN-SVM, GRU-SVM, and MLP-SVM were used in this work for classification and in which GRU-SVM model achieved the highest accuracy of around 84.92%. A machine learning model was proposed in [18] to analysis and detect malwares. This model consists of three different modules which are processing the data, making decision, and detecting new malwares. The first module basically extracts features from malware by working on different types of data such as Opcode n-gram, grayscale images, and import functions. Utilizing the features created by data processing, the second module classifies the malware and also at the same time identify any suspicious malware. The final module discovers any new malware families utilizing shared nearest neighbor (SNN) algorithm. More than 20000 malware dataset was utilized by this model. 98.9% was the best accuracy achieved by this model to successfully classify unknown malwares whereas 86.7% is the success detection rate of the new malwares. To mitigate the issue of imbalance in classifying malware images utilizing CNN, a simple and effective method was proposed in [19]. In this methodology, the last layer of the CNN employs a weighted softmax loss. This makes the error of classification for different classes to have different weights which makes the classifier to treat the error differently. This proposed method is fearile on any existing working CNN models.

SimHash along with the CNN was utilized for malware classification in [20]. Conversion of disassembled malware code into a grayscale image is done by SimHash for visualization whereas CNN uses these images to identify the malware family. For improving the performance methods such as major block selection, multi-hash, and bilinear interpolation were utilized. 10805 malware dataset samples were used in this work and achieved an average accuracy of 98.862%. To recognize a new sample, only 1.41 seconds are needed by this model. Score based GIST descriptors were utilized for implementing, testing, and analyzing malwares in [21]. Robustness of this proposed method along with feature reduction for determining minimum number of GIST features required were also analysed in this work. The effectiveness of the proposed approach with respect to deep learning techniques is also evaluated. DDoS malware detection using a lightweight CNN for IoT environment in [22]. Binary malwares are converted to one channel grayscale images to be fed into the CNN which classifies the malware. The accuracy for classification of DDoS malware and benign was 94.0%. In [23], a deep learning approach was proposed to classify malwares using malware images. The proposed method is a combination of recurrent neural networks and CNN. RNN was utilized to reduce the training dependencies with respect to categorical labels. This predictive code was combined with the original code to generate image features by minhash. CNN took this newly combined data to classify the malware. For classifying malware, a deep learning approach was proposed in [24]. Malimg and Microsoft malware classification challenge (MMCC) benchmark data sets were utilized in this work. The dataset was first converted to grayscale images to be fed into CNN. 98.52% accuracy was achieved for Malimg dataset whereas for MMCC dataset the accuracy was 99.97%. GoogleNet and ResNet models were analysed for malware detection in [25]. Two datasets were utilized, one dataset MMCC and another dataset which contains 3000 benign files. The dataset is first converted into opcode from .exe files and later to images using opcode. GoogleNet has an accuracy of 74.5% whereas ResNet has a Precision of 88.36%. Transfer learning approach was applied to classify malwares in [26]. The deep neural network trained with computer vision data was used to classify static malwares. Three experiments were performed in this work and the results showed that the proposed approach has better accuracy, true positive rate, false positive rate, and F1-score than other machine learning approaches. Biggest advantage is that this model accelerates the training time of the network and at the same time has high performance. BAT algorithm for data equilibrium was utilized in this work to reduce the data imbalance issue. The efficiency and effectiveness of the CNN approach was shown in the experiments conducted in this work.

Various features were extracted from colour images and byte sequences. Further machine learning algorithms were used for classification [27]. Detailed investigation and analysis of various machine learning algorithms were shown with various features for malware image classification. The proposed methods are ineffective at identifying the packed malwares. VGG-16-based approach was employed for malware classification and detailed performance of learned features was shown using t-SNE [28]. Though the VGG-16 performed better than the GIST, performance reported by VGG-16 further can be improved by using the recently released pretrained model. Further, ResNet-50 model-based approach was employed by [29]. Random capsule neural network-based ensemble learning approach is proposed for malware classification and its performance shown on both Malimg and MMCC [30]. This method is more effective for an imbalanced dataset and shows a better generalization performance. However, the proposed approach has relied on adam optimizer for better convergence rate. This approach is computationally expensive and takes more time to train a model. Also, the proposed method works well only to 50*50 100*100 and 120*120 image resolution. A hybrid of CNN and LSTM-based approach is proposed for malware classification [2]. The performance of the proposed approach is evaluated for static analysis features, dynamic analysis features, and image-based malwares. The detailed investigation and analysis of the proposed method is shown on various data sets in static analysis, dynamic analysis, and image-based malware detection. In all three methods, the proposed method performed better than the existing methods. However, all these models are not combined in a

single framework for malware classification. Various image transformation approaches were employed for converting malware into images and the performance of deep CNN shown on all the transformations. Performance of grayscale image transformation outperformed other transformation [31]. This study indicates that the malware image transformation scheme proposed by [4] is robust. A deep CNN based approach is proposed for malware image classification [32]. The performance of the proposed approach is shown on both grayscale and colour image malware dataset and the proposed method performed better than the existing methods on both the datasets. Though the model proposed for Industrial Internet of Things (IIoT) malware detection, the proposed model cannot be deployed. Because, the developed model is not lightweight and cannot be directly deployed on IIoT devices. The transformed malware images have various dimensions and the dimension of the image has to be changed to fixed size. To avoid fixed size image transformation, SPP model was proposed by [33]. However, the SPP model completely fails to detect the 'Autorun.K' malware family. RCNN-based approach is proposed for malware classification by transforming malware into images on MMCC dataset [34]. The proposed method performed better than the existing methods, however the proposed approach is computationally expensive and no experiments have been shown to show that the proposed method can detect the packed malware and the robustness of the proposed method on malware variants detection and classification. CNN with bidirectional LSTM based approach is proposed for malware classification by transforming them into images [35]. The data samples were increased by following various data augmentation approaches. In addition, authors pointed out that attention-based models are effective at identifying important features when there are different obfuscations. Various image visualization was shown for malware and a hybrid framework based on deep learning was proposed by [1]. Though the authors have shown a detailed investigation and analysis performance of the proposed method, still the performance reported by the proposed method can be enhanced by using deep CNN models. Also, no experimental study was shown for packed malware analysis. Various experiments of CNN-based pretrained model performance were shown by [13, 14]. In these studies, a detailed experimental study was shown for packed malware. However, still the performance can be enhanced by using recently released CNN-based pretrained models.

The aforementioned literature survey shows that various machine learning and deep learning-based approaches were employed for malware image classification. Most importantly, CNN-based approaches were largely used methods. Most of the existing studies used Malimg and MMCC dataset. Both of these datasets are publicly available. Malimg database has malware samples in the form of images but the corresponding binaries are not available. MMCC dataset available in the form of .asm and bytes file. Though the MMCC dataset can be considered as robust, in this work Malimg dataset is used. Becuase, Malimg contains data samples of packed malware, metamorphic and polymorphic malware, encrypted malware, malware variants of various malware families. Malimg is preferred over MMCC for Image-based malware detection. Also, to show the proposed method is generalizable, the performances of the methods were evaluated on various data splits of training and testing of Malimg dataset.

3 Multi-headed attention CNN for malware image classification

The proposed multi-headed attention CNN approach for malware image classification is shown in Figure 2. The proposed method is feature agnostic and does not rely on binary code analysis or reverse engineering technique. It contains the following

1. **Transformation of malware into image:** Hex editors are the most commonly used tool for viewing and editing the bytes in hexadecimal representation of the binaries. Figure 3 shows the transformation of malware binaries into images. All the images are basically a matrix of pixel values. The pixel is encoded as 8-bits. The malware picture or the image is interpreted as (0 (black), 255 (white)) in binary

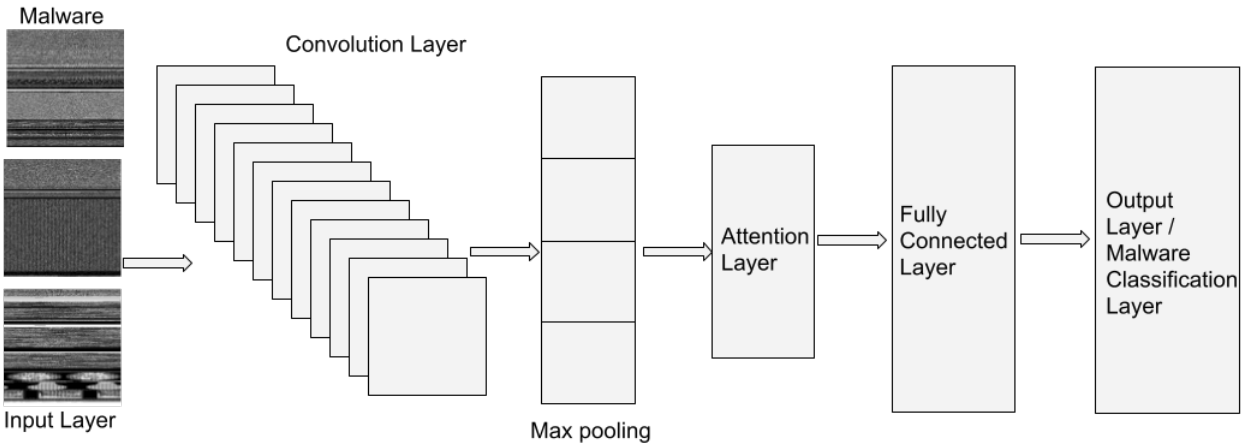

Fig. 2: Schematic diagram of the malware classification system

language. The height of the image depends on the file size and it can vary whereas the width of the image is fixed. Most of the unpacked malwares are developed using simple mutation techniques and these variants differ in only 7 bytes. There are many packers available and some are commercial whereas some are open source and publicly available to everyone. This allows hackers to use packers easily and most of the recent malware are packed and even detecting whether the malware is packed or not is a challenging problem.

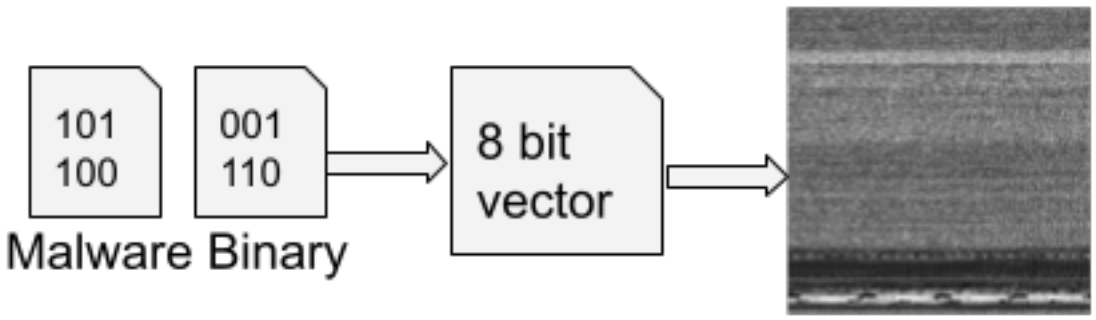

Fig. 3: Transforming malware binary into an image representation

2. **Convolutional layer:** Convolution is an important operation in CNN. It consists of a set of learnable filters which are convolved across the width and the height of the input features during the forward pass producing a 2-dimensional activation map of the filter. Convolution layer uses convolution and filter operation to learn high-level features from the input data. The features from the convolution layer are meaningful, low-dimensional, and invariant.

3. **Maxpooling:** It is a form of non-linear down-sampling operation and it partitions the image into a set of non-overlapping rectangles and for each sub-region, maxpooling outputs the maximum value. Maxpooling reduce the computational complexity of the CNN model and preserves the translation invariance.
4. **Attention layer:** In CNN, the images are divided into multiple channels and each channel extracts certain parts of the information from the images. The information extracted from different channels are not unique and each represents certain parts of the image features. The convolution operation aims at extracting edges, textures, parts of the object, and finally, complete the object sequentially. Though the CNN has kernels that help to learn important parts of the image, the importance for the tiny important portion of the image given by CNN is limited. This is important because the malware codes are not present in the entire image instead the codes are available in the tiny portion of the image. To extract and give more importance to these tiny portions of the malware image, attention layer is added after the convolution layer. There are many attention approaches available [36] and in this work, we employ multi-headed attention. Attention approach uses the three components and they are queries (Qa), the keys (Ka), and the values (Va). Scaled dot product attention is an operation that computes a dot product for each query (Qa), with all of the keys (Ka). Following, each of the results are divided by $\sqrt{d_k}$ and next, softmax function was employed on the final result. This process learns the weights that are used to scale the values (Va). Mathematically,

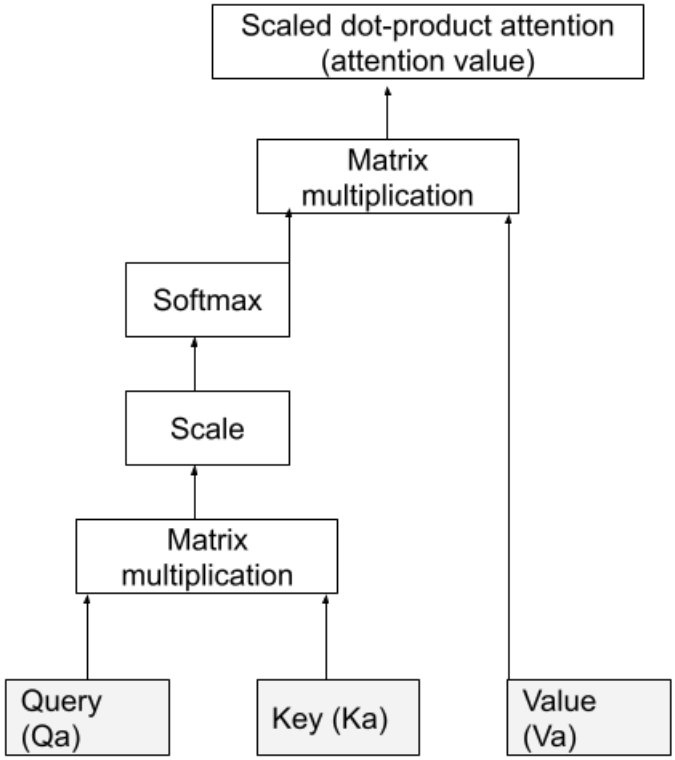

Fig. 4: Components of Scaled dot-product attention

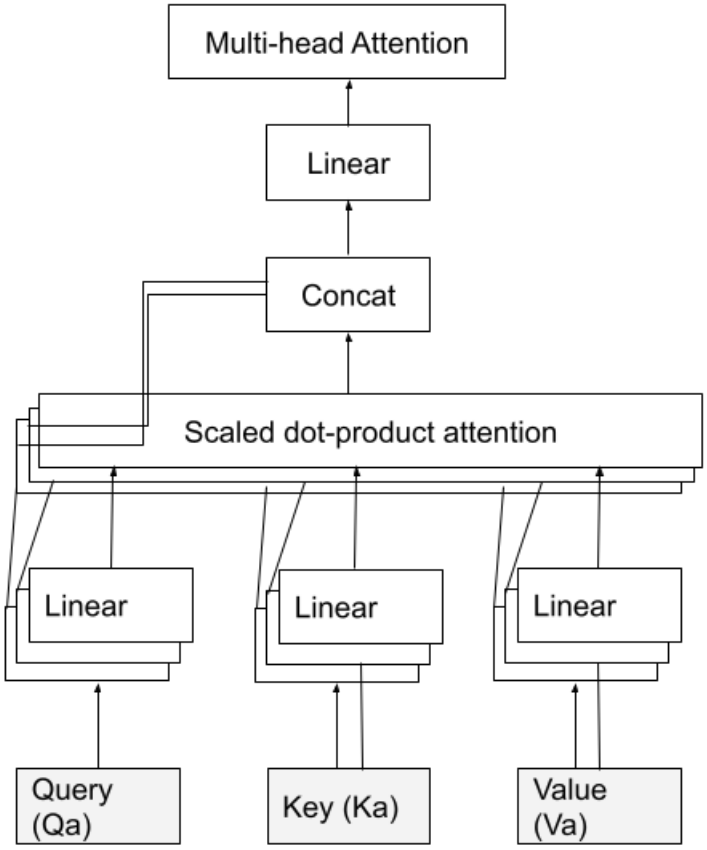

Fig. 5: The architecture of Multi-head Attention approach

$$Attention(Qa, Ka, Va) = soft\max\left(\frac{QaKa^T}{\sqrt{d_k}}\right) \tag{1}$$

The idea behind scaled dot product attention is to attend over a sequence and it limits linear views of the same sequence. To overcome this, attention approach is extended to multiple heads and each head outputs are concatenated and multiplied by a linear layer to match the desired output dimension. The correlation information that exists in different subsequences are effectively learned by multi-head attention models while ensuring computational efficiency.

$$Multiheadd_{Attention}(Qa, Ka, Va) = Concat(h_1, \cdots, h_h)w^o \tag{2}$$

Where $h_i = Attention(QaW_i^{Qa}, KaW_i^{Ka}, VaW_i^{Va}$
Above h defines head and W are learnable parameters. The computation graph of Scaled dot product attention and multi-head attention is shown in Figure 4 and Figure 5 respectively.

5. **Fully connected layer:** In a fully connected layer, the neurons have connections to all the neurons in the previous layer. To learn the non-linear combination of the features extracted by the convolution layer, a

fully connected layer is employed between the output layer and convolution layer. Generally, the fully-connected layer aims to learn non-linear in the invariant feature space formed by the convolution layer.

6. **Output layer or malware classification layer:** The output layer or malware classification layer is also called a fully-connected layer. It contains 25 neurons with softmax activation function and categorical cross-entropy loss function.

4 Description of Malware Image Dataset

In this work, the experimental studies of the proposed method was done on Malimg malware image dataset [4]. The image sample of Malimg dataset can be directly used for image classification. Malimg has both packed and unpacked malwares. However, the binaries corresponding to images of Malimg are not readily publicly available. The detailed statistics of the dataset is shown in Figure 6. The dataset contains 9342 grayscale images of 25 malware families and it is imbalanced. In Malimg, Allaple.A contains the maximum number of malware image samples (2949) and the Skintrim.N family has the least number of malware image samples (80 images). From Malimg dataset, we prepared various train and test data splits such as 90%-10%, 80%-20%, 70%-30%, 60%-40%, and 50%-50%. In all the data splits, the train and test datasets are disjoint.

5 Performance metrics

To evaluate the performances of the deep learning-based models for image-based Android malware detection, the following metrics are considered in this study.

Accuracy is the classifier's ability to classify all positive samples as positive and all negative samples as negative.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{3}$$

Precision is a measure of the classifier's ability to not mark a negative sample as positive.

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

Recall is a measure of the classifier's ability to mark all positive samples as positive.

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

F1-score is the weighted average of Precision and Recall.

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6}$$

where TP, FP, TN, and FN are true positive, false positive, true negative and false negative respectively. The values for TP, FP, TN, and FN are obtained from the confusion matrix. It is a matrix that displays and compares actual values with the predicted values.

To know the performance of the model at class level for image-based malware detection, Precision, Recall, and F1-score were considered in this work. Precision, Recall, and F1-score were included for both macro average and weighted average. Macro metric computes the Precision, Recall, and F1-score for each class and returns the average without considering the proportion for each class in the image-based malware dataset. The weighted metric computes the Precision, Recall, and F1-score for each class and returns the average by considering the proportion for each class in the image-based malware dataset.

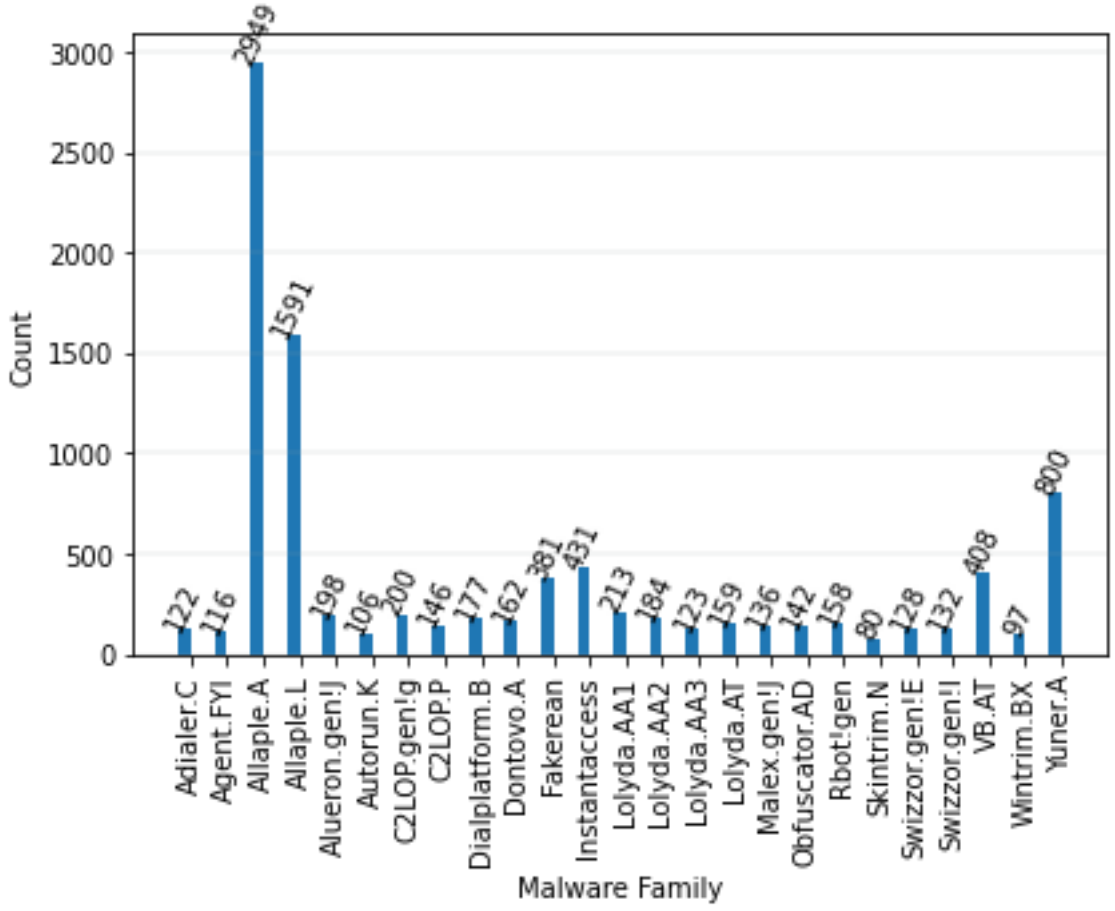

Fig. 6: Number of samples in each malware family of Malimg Image malware dataset

6 Experiments, Results and Discussions

All the experiments were run on Kaggle NVidia K80 GPUs[2] and the models were implemented using Keras[3] with TensorFlow[4] as backend. The performance of the models were done using scikit-learn[5]. All the models were trained in batch-sizes of 32 using adam optimizer.

CNN has a list of parameters and the optimal performance of CNN depends on these parameters. Various trials of experiments were done to identify the optimal parameters of CNN during training. The best parameters of CNN are learning rate 0.001, batch size 32, and optimizer adam. Next, a few trials of experiments

[2]https://www.kaggle.com/
[3]https://keras.io/
[4]https://www.tensorflow.org/
[5]https://www.tensorflow.org/

were run with a single convolutional layer followed by maxpooling and a fully connected layer to identify the best parameter for filters in convolution layers with 16, 32, 48, and 64. The experiments with 32 performed better than 16 and the training accuracy remained the same across epochs with filters 48 and 64. Thus the filter value is set to 32. Next, we ran several trials of experiments to identify the CNN network structure and we found the best performed model that is shown in Table 1 for CNN without attention and Table 2 for CNN with attention. The total parameter details for CNN without attention and CNN with attention is 11123897 and 2025369 respectively. 11123897 and 2025305 are trainable parameter details for CNN without attention and CNN with attention respectively. The non-trainable parameter details for CNN without attention and CNN with attention are 0 and 64 respectively. The attention-based proposed approach has lesser parameters, computationally inexpensive compared to the non-attention based model and achieves higher accuracy than the non-attention CNN.

Table 1: CNN model architecture details

Layer (type)	**Output Shape**	**Param #**
input_1 (InputLayer)	[(None, 64, 64, 3)]	0
conv2d	(None, 64, 64, 32)	416
max_pooling2d	(None, 32, 32, 32)	0
conv2d_1	(None, 31, 31, 64)	8256
max_pooling2d_1	(None, 16, 16, 64)	0
conv2d_2	(None, 15, 15, 192)	49344
flatten (Flatten)	(None, 43200)	0
dense (Dense)	(None, 256)	11059456
dense_1 (Dense)	(None, 25)	6425

Table 2: CNN with attention model architecture details

Layer (type)	**Output Shape**	**Param #**	**Connected to**
input_1 (InputLayer)	[(None, 64, 64, 3)]	0	
conv2d	(None, 64, 64, 32)	416	input_1[0][0]
max_pooling2d	(None, 32, 32, 32)	0	conv2d[0][0]
conv2d_1	(None, 31, 31, 64)	8256	max_pooling2d[0][0]
max_pooling2d_1	(None, 16, 16, 64)	0	conv2d_1[0][0]
conv2d_2	(None, 15, 15, 192)	49344	max_pooling2d_1[0][0]
reshape	(None, 225, 192)	0	conv2d_2[0][0]
functional_1	(None, 225, 32)	117344	reshape[0][0] reshape[0][0] reshape[0][0]
reshape_5	(None, 15, 15, 32)	0	functional_1[0][0]
batch_normalization	(None, 15, 15, 32)	128	reshape_5[0][0]
flatten	(None, 7200)	0	batch_normalization[0][0]
dense_4	(None, 256)	1843456	flatten[0][0]
dense_5	(None, 25)	6425	dense_4[0][0]

There are no separate training and testing datasets in Malimg. Thus we decided to experiment with all possible combinations of the training and testing of Malimg to show that the methods are generalizable. We randomly divided the Malimg dataset into training and testing with 90%-10%, 80%-20%, 70%-30%, 60%-40%, and 50%-50%. The best performed one is 80%-20% and training accuracy and loss is shown in

Figure 7 and Figure 8 respectively. The plot shows a gradual increase and decrease in the training accuracy and training loss for the attention-based model whereas the non-attention-based model has not followed the same. This may be due to overfitting and other learning issues of the non-attention based model. Overall, the attention-based model has shown better performances during training. These plots helped us to know the convergence and learning capability of the model.

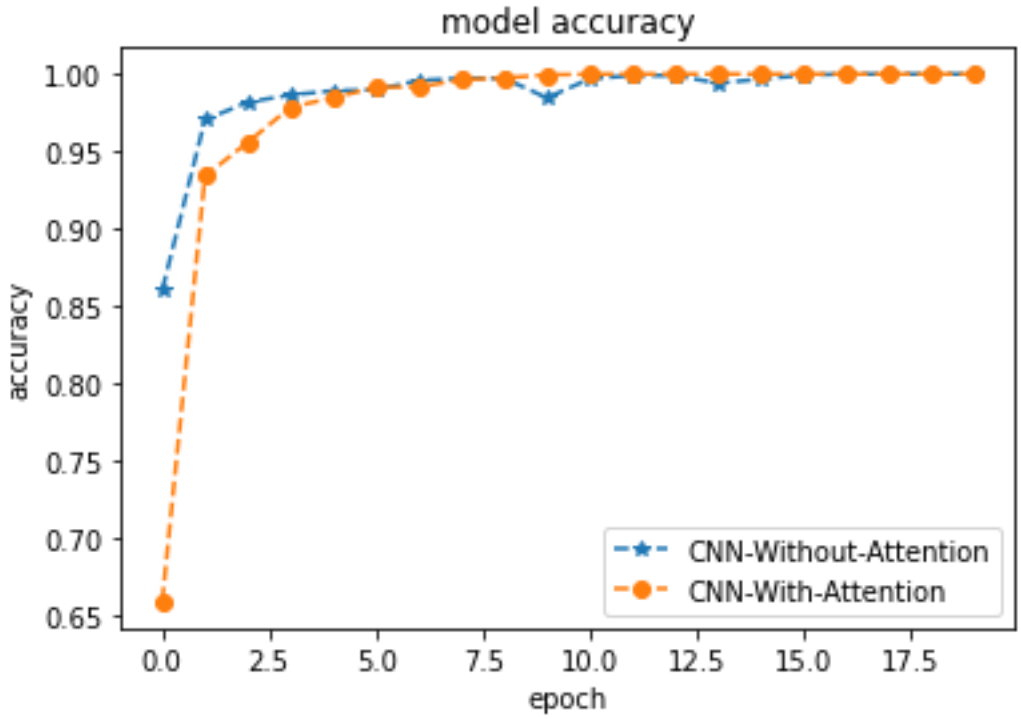

Fig. 7: Train accuracy of CNN without attention and CNN with attention model for Malimg dataset (80% training and 20% testing)

After the training, the model performances were evaluated on the test dataset and the performances were reported in Table 4 for attention-based approach and Table 3 for non-attention based approach. Attention-based approach accuracy is 98% for all the data splits except the best performed with 99% accuracy for 80%-20% whereas the non-attention-based model has shown similar performance i.e. 98% accuracy on all the data splits except 80%-20%. The performance obtained by non-attention-based models on 80%-20% is 93% accuracy that is lesser compared to all the data splits. This indicates that the performance of the attention-based approach is more consistent across any of the data-splits and the method is generalizable compared to the non-attention based approach. Apart from accuracy, the performance metrics such as Precision, Recall, and F1-score are also reported with weighted and macro. With 80%-20% training-testing datasets, the attention based model has shown 0.99 for all weighted Precision, weighted Recall, and weighted F1-score and 0.97% for all macro Precision, macro Recall, and macro F1-score. Non-attention-based model

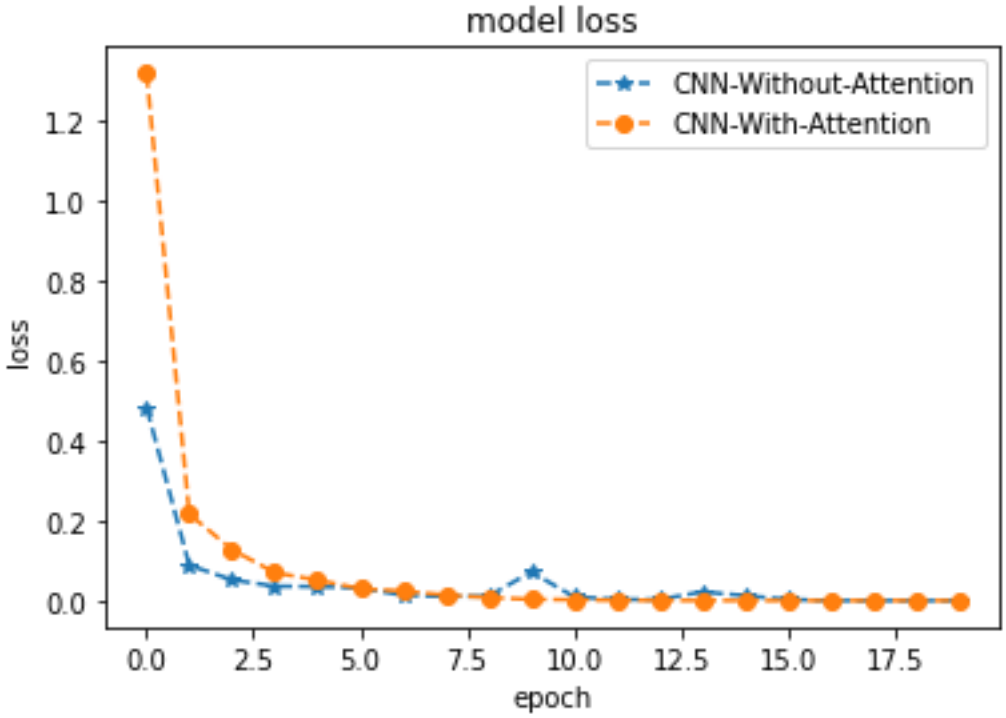

Fig. 8: Train loss curve of CNN without attention and CNN with attetnion model for Malimg dataset (80% training and 20% testing)

Table 3: Detailed results for CNN Without Attention for Malimg dataset

Dataset Split (Train-Test)	Accuracy	Type	Precision	Recall	F1-score
50%-50%	0.98	Macro	0.94	0.94	0.94
		Weighted	0.97	0.98	0.97
60%-40%	0.98	Macro	0.94	0.95	0.95
		Weighted	0.98	0.98	0.98
70%-30%	0.98	Macro	0.94	0.95	0.94
		Weighted	0.98	0.98	0.98
80%-20%	0.93	Macro	0.87	0.83	0.83
		Weighted	0.92	0.93	0.92
90%-10%	0.98	Macro	0.95	0.95	0.95
		Weighted	0.98	0.98	0.98

has shown 0.87 macro Precision, 0.83 macro Recall, and 0.83 F1-score and 0.92 weighted Precision, 0.93 weighted Recall, and 0.92 weighted F1-score. Both the attention and non-attention-based methods showed better performances for Precision, Recall, and F1-score in terms of weighted compared to macro. However, the difference in performances of macro and weighted are larger for non-attention compared to attention approaches. Overall, still, the learning capability can be enhanced to handle macro performances of Precision, Recall, and F1-score. Thus it can avoid misclassification in highly imbalanced datasets.

Table 4: Detailed results for CNN With Attention for Malimg dataset

Dataset Split (Train-Test)	Accuracy	Type	Precision	Recall	F1-score
50%-50%	0.98	Macro	0.95	0.95	0.95
		Weighted	0.98	0.98	0.98
60%-40%	0.98	Macro	0.95	0.95	0.95
		Weighted	0.98	0.98	0.98
70%-30%	0.98	Macro	0.95	0.95	0.95
		Weighted	0.98	0.98	0.98
80%-20%	0.99	Macro	0.97	0.97	0.97
		Weighted	0.99	0.99	0.99
90%-10%	0.98	Macro	0.95	0.95	0.95
		Weighted	0.98	0.98	0.98

Table 5: Class-wise performance of CNN without attention model for Malimg dataset (80% training and 20% testing)

Malware Family	Precision	Recall	F1-score
Adialer.C	1.00	1.00	1.00
Agent.FYI	1.00	1.00	1.00
Allaple.A	0.93	0.99	0.96
Allaple.L	0.98	0.95	0.97
Alueron.gen!J	0.98	1.00	0.99
Autorun.K	0.00	0.00	0.00
C2LOP.gen!g	0.83	0.44	0.58
C2LOP.P	0.42	0.59	0.49
Dialplatform.B	1.00	1.00	1.00
Dontovo.A	1.00	1.00	1.00
Fakerean	0.96	0.99	0.97
Instantaccess	1.00	1.00	1.00
Lolyda.AA1	0.75	0.97	0.85
Lolyda.AA2	1.00	0.83	0.91
Lolyda.AA3	1.00	1.00	1.00
Lolyda.AT	0.93	1.00	0.96
Malex.gen!J	1.00	0.33	0.50
Obfuscator.AD	1.00	1.00	1.00
Rbot!gen	0.89	1.00	0.94
Skintrim.N	1.00	1.00	1.00
Swizzor.gen!E	0.44	0.63	0.52
Swizzor.gen!I	0.69	0.31	0.43
VB.AT	1.00	0.93	0.96
Wintrim.BX	1.00	0.74	0.85
Yuner.A	0.90	1.00	0.95
macro avg	0.87	0.83	0.83
weighted avg	0.92	0.93	0.92

The detailed performance across each malware family for the best performing model on 80%-20% training-testing dataset is shown in Table 5 for non-attention based approach and Table 6 for attention-based approach. Non-attention-based CNN has shown less than 0.50 Precision for C2LOP.P, and Swizzor.gen!E malware families whereas the attention with CNN has shown 0.79 and 0.70 Precision for C2LOP.P, and Swizzor.gen!E malware families. Most importantly, non-attention-based CNN completely failed to detect the Autorun.K malware family and the attention with CNN has zero misclassification for Autorun.K malware family. F1-score of non-attention CNN is less than 0.50 for Autorun.K, C2LOP.P, and Swizzor.gen!I mal-

Table 6: Class-wise performance of CNN with attention model for Malimg dataset (80% training and 20% testing)

Malware Family	Precision	Recall	F1-score
Adialer.C	1.00	1.00	1.00
Agent.FYI	1.00	1.00	1.00
Allaple.A	0.99	1.00	1.00
Allaple.L	1.00	0.99	1.00
Alueron.gen!J	1.00	1.00	1.00
Autorun.K	1.00	1.00	1.00
C2LOP.gen!g	0.91	0.91	0.91
C2LOP.P	0.79	0.85	0.82
Dialplatform.B	1.00	1.00	1.00
Dontovo.A	1.00	1.00	1.00
Fakerean	1.00	0.99	0.99
Instantaccess	1.00	1.00	1.00
Lolyda.AA1	0.97	1.00	0.99
Lolyda.AA2	1.00	0.98	0.99
Lolyda.AA3	1.00	1.00	1.00
Lolyda.AT	1.00	1.00	1.00
Malex.gen!J	0.96	0.96	0.96
Obfuscator.AD	1.00	1.00	1.00
Rbot!gen	0.97	1.00	0.98
Skintrim.N	1.00	1.00	1.00
Swizzor.gen!E	0.70	0.84	0.76
Swizzor.gen!I	0.83	0.66	0.73
VB.AT	1.00	0.99	0.99
Wintrim.BX	1.00	1.00	1.00
Yuner.A	1.00	1.00	1.00
macro avg	0.97	0.97	0.97
weighted avg	0.99	0.99	0.99

ware families whereas attention with CNN has shown F1-score 1.00, 0.82, and 0.73 for Autorun.K, C2LOP.P, and Swizzor.gen!I malware families respectively. Recall is lesser than 0.50 for C2LOP.gen!g, Malex.gen!J, and Swizzor.gen!I malware families and attention with CNN has shown better Recall compared to non-attention with CNN for all C2LOP.gen!g, Malex.gen!J, and Swizzor.gen!I malware families. Attention with CNN has shown more than 0.99 Precision, Recall, and F1-score for 17 malware families such as Adialer.C, Agent.FYI, Allaple.A, Allaple.L, Alueron.gen!J, Autorun.K, Dialplatform.B, Dontovo.A, Fakerean, Instantaccess, Lolyda.AA3, Lolyda.AT, Obfuscator.AD, Skintrim.N, VB.AT, Wintrim.BX, and Yuner.A whereas non-attention with CNN has shown more than 0.99 Precision, Recall, and F1-score for 8 malware families such as Adialer.C, Agent.FYI, Dialplatform.B Dontovo.A, Instantaccess, Lolyda.AA3, Obfuscator.AD, and Skintrim.N. Overall, attention with CNN has shown better Precision, Recall, and F1-score for all the malware families compared to the non-attention with CNN approach.

The confusion matrix for the proposed method on 80%-20% data split for training and testing is shown in Figure 9. The proposed model has shown accuracy of 0.9855 with misclassification rate 0.0145. The proposed approach classifies all the samples of Adialer.C, Agent.FYI, Alueron.gen!J, Autorun.K, Dialplatform.B, Dontovo.A, Instantaccess, Lolyda.AA1, Lolyda.AA3, Lolyda.AT, Obfuscator.AD, Rbot!gen, Skintrim.N, Wintrim.BX, and Yuner.A malware families. The misclassified malware families and the number data samples misclassified are Allaple.A : 1 (misclassified to Malex.gen!J), Allaple.L : 2 (misclassified to Allaple.A), C2LOP.gen!g : 3 (2 samples misclassified to C2LOP.P and 1 sample misclassified to Swizzor.gen!E), C2LOP.P:

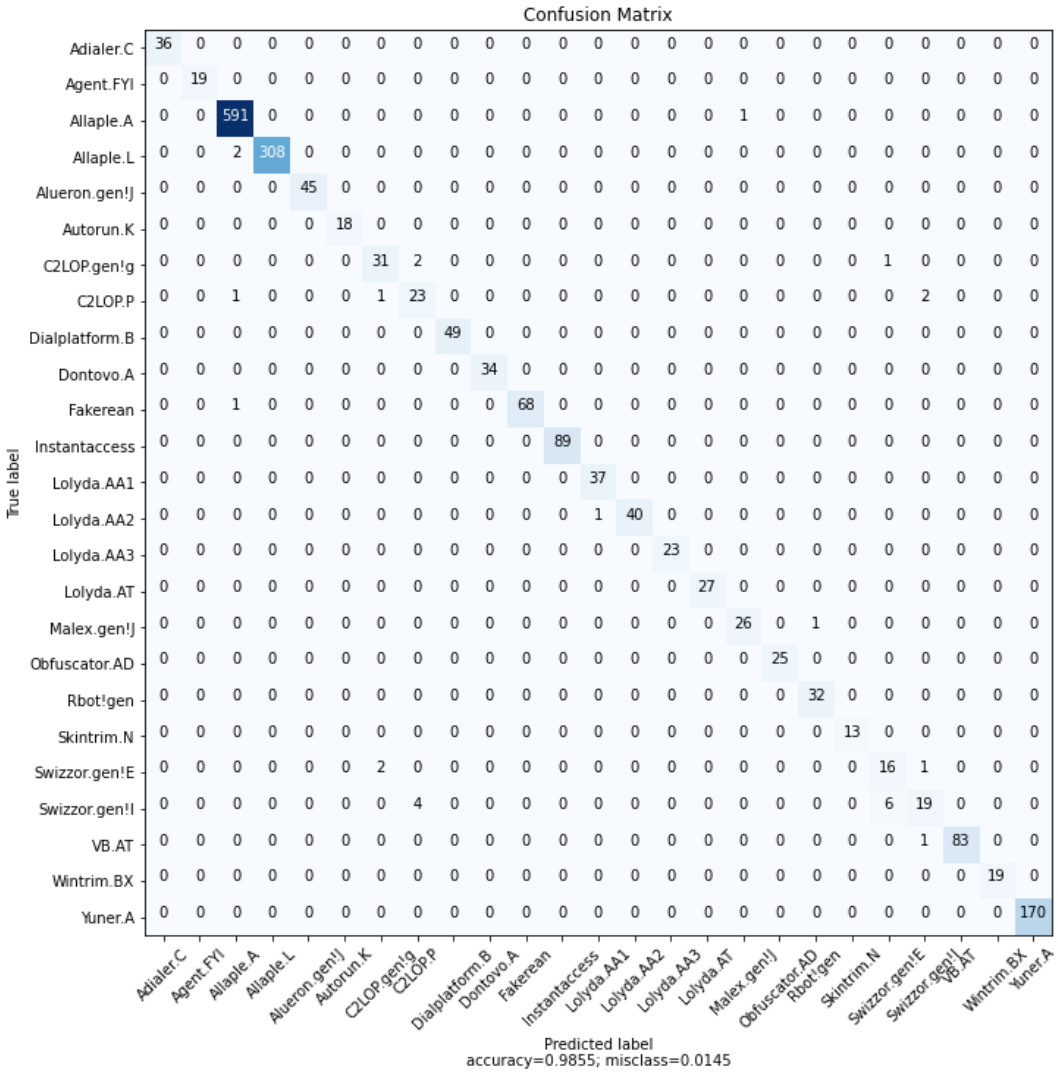

Fig. 9: Proposed Multi-headed attention-based CNN approach Confusion matrix on 80% training and 20% testing Malimg dataset

4 (1 sample misclassified to C2LOP.gen!g and Allaple.A respectively, 2 samples misclassified to Swizzor.gen!E), Fakerean: 1 (misclassified to Allaple.A), Lolyda.AA2: 1 (misclassified to Lolyda.AA1), Malex.gen!J: 1 (misclassified to Rbot!gen), Swizzor.gen!E: 3 (1 sample misclassified to Swizzor.gen!I and 2 samples misclassified to C2LOP.gen!g), Swizzor.gen!I: 10 (4 samples are misclassified to C2LOP.P and 6 samples are misclassified to Swizzor.gen!I), and VB.AT: 1 (misclassified to Swizzor.gen!I). Overall, Swizzor.gen!I has the highest number of misclassified data samples with 10 and most importantly 6 are misclassified into nearby malware families such as Swizzor.gen!I. In addition, most of the single misclassified malware fami-

lies have misclassification to the nearby malware family. This indicates that the proposed model is robust but still the learning capability has to be improved by including more data samples to the misclassified malware family. This is one of the limitations of the proposed model and increasing the number of data samples can overcome this limitation.

We compare the performance of the proposed method with the existing methods in Tables 7, 8. As accuracy is used as a common metric in all the existing methods, the performance of the proposed method and existing method accuracy is included in tables. The tables shows that the proposed method performed better than all the existing methods and all these methods are not based on attention even though the methods are based on deep learning specifically, CNN. The main difference between the existing CNN-based methods and the proposed method is, the proposed method includes attention in CNN that helps to learn the tiny infected regions of the malware images. These differences lead to a superior performance for malware image classification with 99% accuracy for 80-20% train and test data splits of Malimg dataset.

Table 7: Comparison of results of the proposed method with the existing works for Malimg dataset

Reference	**Year**	**Dataset**	**Visualization Approach**	**Approach**	**Accuracy**
Vinayakumar et al. [2]	2018	Malimg	Grayscale	Deep learning	90.0
					93.6
					95.0
					96.3
Bhodia et al. [37]	2019	Malimg	Grayscale	Deep learning	94.8
Akarsh et al. [38]	2019	Malimg	Grayscale	Deep learning	95.5
Venkatraman et al. [1]	2019	Malimg	Grayscale	Deep learning	96.3
Agarap, A. F. et al. [17]	2019	Malimg	Grayscale	Deep learning	84.9
Nataraj et al. [4]	2011	Malimg	Grayscale	Machine learning	97.18
Yue et al. [19]	2017	Malimg	Grayscale	Deep learning	97.32
Zhihua et al. [30]	2018	Malimg	Grayscale	Deep learning	97.60
Yajamanam et al. [21]	2018	Malimg	Grayscale	Machine learning	97
Cui et al. [39]	2018	Malimg	Grayscale	Machine learning	92.2
Cui et al. [39]	2018	Malimg	Grayscale	Machine learning	91.90
Cui et al. [39]	2018	Malimg	Grayscale	Machine learning	93.20
Cui et al. [39]	2018	Malimg	Grayscale	Machine learning	92.50
Cui et al. [39]	2018	Malimg	Grayscale	Deep learning	94.50
Cui et al. [30]	2019	Malimg	Grayscale	Deep learning	97.6
Danish et al. [14]	2020	Malimg	color	Deep learning	98.82
Su et al. [22]	2018	Malimg	Grayscale	Deep learning	94
Rezende et al. [28, 29]	2018	Malimg	Grayscale	Machine learning	89.43
Rezende et al. [28, 29]	2018	Malimg	Grayscale	Deep learning	83.68
Rezende et al. [28, 29]	2018	Malimg	Grayscale	Deep learning	90.77
Rezende et al. [28, 29]	2018	Malimg	Grayscale	Deep learning	92.29
Rezende et al. [28, 29]	2018	Malimg	Grayscale	Deep learning	98.62
Khan et al. [25]	2019	Malimg	Grascale	Deep learning	98
Proposed	**Proposed**	Malimg	Grayscale	Deep learning	**0.98 (Without attention and CNN)**
					0.99 (With attention and CNN)

Image-based malware detection approach is malware file agnostic and avoids usual methods such as disassembly, de-compiling, de-obfuscation or execution of the malware binary in a virtual environment in

Table 8: Comparison of results of the proposed method on various data splits of Malimg dataset with the existing works

Reference	Architecture	Data Split	Accuracy
Sriram et al. [33]	SPPNet 1	50-50	96.9
	SPPNet 2	60-40	97.4
	SPPNet 3	70-30	97.6
	SPPNet 4	80-20	98.1
	SPPNet 5	90-10	99.3
Akarsh et al. [40]	CNN1-LSTM	50-50	96.1
	CNN2-LSTM	60-40	96.7
	CNN2-LSTM	70-30	96.4
	CNN1-LSTM	80-20	96.5
	CNN2-LSTM	90-10	96.8
Proposed	CNN with Attention	50-50	98
	CNN without Attention		98
	CNN with Attention	60-40	98
	CNN without Attention		98
	CNN with Attention	70-30	98
	CNN without Attention		98
	CNN with Attention	80-20	99
	CNN without Attention		93
	CNN with Attention	90-10	98
	CNN without Attention		98

detecting malware and classifying malware into their malware family. Hence, image-based malware detection is faster than static, dynamic, and hybrid analysis. Though the signature based methods are accurate in detecting and classifying the malware, this cannot be a good approach because continuously the signature database has to be updated by malware domain experts. This is time consuming, more expensive, and most importantly completely fails to detect the new malware or even variants of the existing malware. As the recent malwares are variants of existing malware families, they are self-similar in nature. Image-based approaches are effective in handling similar malware images and can give more information about the structure of the malware. Image-based malware detection and classification approaches are data-driven and these methods analysis based on existing malware. Thus, image-based malware detection doesn't meet a zero day attack and the proposal of a novel approach to deal with a zero day attack can be considered as a significant direction towards future work. Also, image-based malware detection does not give much information about the actual behavior of the malware other than the label given by AV software. There can be another subsystem that can give actual behaviour of the malware and this subsystem can be integrated to the image-based malware classification. This is another future direction of the proposed work.

7 Conclusion and Future works

In this work, we propose an end-to-end multi-headed attention CNN-based approach for malware image classification. The proposed approach converts the malware binaries into images and later the automatic feature extraction and classification done by deep learning. Integrating attention later to the CNN network facilitates locating and identifying the tiny infected regions of the malware image. The efficacy of the proposed method is shown for malware classification by conducting extensive experiments. The proposed attention-based approach performed better than the non-attention-based CNN and other existing methods in all the experiments on Malimg dataset while ensuring computational efficiency. This indicates that the proposed method is ro-

bust, generalizable, and it has the capability to detect the variants of the existing malware or completely new malware itself. The structure of packed malware variants remain same after packing and the image-based malware detection and classification approach is able to identify similarity among packed malware variants. However, static analysis approaches completely fail to handle packed malware variants. Also, similar obfuscation approaches used by malware authors for creating malware variants to attack non-windows operating systems such as Linux, Android, and OS X. The proposed approach can be used on any operating system and it is platform independent. Overall, the proposed image-based malware classification approach works really well on malware variants that have similar structure. Detailed investigation and analysis of the proposed method has to be done on the big malware image dataset that includes code obfuscation, packed malware, metamorphic, and polymorphic. This can be considered as future work. Various visualizations can be included in the malware analysis using image-based approaches and these methods can help the malware analyst to see the features and overall, visualization will be more effective towards malware detection and classification. This can be another direction towards future works of the proposed work.

Funding

Not applicable.

Conflicts of interest/Competing interests

The authors declare no conflict of interest.

Availability of data and material

Data used in this study are available from the first author upon request.

Code availability

Programming codes implemented in this study are available from the first author upon request.

References

1. Sitalakshmi Venkatraman, Mamoun Alazab, and R Vinayakumar. A hybrid deep learning image-based analysis for effective malware detection. *Journal of Information Security and Applications*, 47:377–389, 2019.
2. R Vinayakumar, Mamoun Alazab, KP Soman, Prabaharan Poornachandran, and Sitalakshmi Venkatraman. Robust intelligent malware detection using deep learning. *IEEE Access*, 7:46717–46738, 2019.
3. Mamoun Alazab. Profiling and classifying the behavior of malicious codes. *Journal of Systems and Software*, 100:91–102, 2015.
4. Lakshmanan Nataraj, Sreejith Karthikeyan, Gregoire Jacob, and Bangalore S Manjunath. Malware images: visualization and automatic classification. In *Proceedings of the 8th international symposium on visualization for cyber security*, pages 1–7, 2011.
5. Jagsir Singh and Jaswinder Singh. A survey on machine learning-based malware detection in executable files. *Journal of Systems Architecture*, 112:101861, 2021.
6. Jiawei Zhao, Rahat Masood, and Suranga Seneviratne. A review of computer vision methods in network security. *IEEE Communications Surveys & Tutorials*, 2021.
7. Mahmoud Hassaballah and Ali Ismail Awad. *Deep learning in computer vision: principles and applications*. CRC Press, 2020.

8. M Hassaballah, Mohamed Abdel Hameed, Ali Ismail Awad, and Khan Muhammad. A novel image steganography method for industrial internet of things security. *IEEE Transactions on Industrial Informatics*, 17(11):7743–7751, 2021.
9. Hong Xiong and Ying Fan. How to better identify venture capital network communities: Exploration of a semi-supervised community detection method. *Journal of Social Computing*, 2(1):27–42, 2021.
10. Jiantao Zheng, Cuixiang Lin, Zhenpeng Wu, and Hong-Dong Li. A comparison of computational approaches for intron retention detection. *Big Data Mining and Analytics*, 5(1):15–31, 2022.
11. Wei Zhong, Ning Yu, and Chunyu Ai. Applying big data based deep learning system to intrusion detection. *Big Data Mining and Analytics*, 3(3):181–195, 2020.
12. Weiping Wang, Zhaorong Wang, Zhanfan Zhou, Haixia Deng, Weiliang Zhao, Chunyang Wang, and Yongzhen Guo. Anomaly detection of industrial control systems based on transfer learning. *Tsinghua Science and Technology*, 26(6):821–832, 2021.
13. Danish Vasan, Mamoun Alazab, Sobia Wassan, Babak Safaei, and Qin Zheng. Image-based malware classification using ensemble of cnn architectures (imcec). *Computers & Security*, 92:101748, 2020.
14. Danish Vasan, Mamoun Alazab, Sobia Wassan, Hamad Naeem, Babak Safaei, and Qin Zheng. Imcfn: Image-based malware classification using fine-tuned convolutional neural network architecture. *Computer Networks*, 171:107138, 2020.
15. Wei Guo, Tenghai Wang, and Jizeng Wei. Malware detection with convolutional neural network using hardware events. In *CCF National Conference on Computer Engineering and Technology*, pages 104–115. Springer, 2017.
16. Hae-Jung Kim. Image-based malware classification using convolutional neural network. In *Advances in computer science and ubiquitous computing*, pages 1352–1357. Springer, 2017.
17. Abien Fred 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*, 2017.
18. Liu Liu, Bao-sheng Wang, Bo Yu, and Qiu-xi Zhong. Automatic malware classification and new malware detection using machine learning. *Frontiers of Information Technology & Electronic Engineering*, 18(9):1336–1347, 2017.
19. Songqing Yue. Imbalanced malware images classification: a cnn based approach. *arXiv preprint arXiv:1708.08042*, 2017.
20. Sang Ni, Quan Qian, and Rui Zhang. Malware identification using visualization images and deep learning. *Computers & Security*, 77:871–885, 2018.
21. Sravani Yajamanam, Vikash Raja Samuel Selvin, Fabio Di Troia, and Mark Stamp. Deep learning versus gist descriptors for image-based malware classification. In *Icissp*, pages 553–561, 2018.
22. Jiawei Su, Danilo Vargas Vasconcellos, Sanjiva Prasad, Daniele Sgandurra, Yaokai Feng, and Kouichi Sakurai. Lightweight classification of iot malware based on image recognition. In *2018 IEEE 42Nd annual computer software and applications conference (COMPSAC)*, volume 2, pages 664–669. IEEE, 2018.
23. Guosong Sun and Quan Qian. Deep learning and visualization for identifying malware families. *IEEE Transactions on Dependable and Secure Computing*, 2018.
24. Mahmoud Kalash, Mrigank Rochan, Noman Mohammed, Neil DB Bruce, Yang Wang, and Farkhund Iqbal. Malware classification with deep convolutional neural networks. In *2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5. IEEE, 2018.
25. Riaz Ullah Khan, Xiaosong Zhang, and Rajesh Kumar. Analysis of resnet and googlenet models for malware detection. *Journal of Computer Virology and Hacking Techniques*, 15(1):29–37, 2019.
26. Li Chen. Deep transfer learning for static malware classification. *arXiv preprint arXiv:1812.07606*, 2018.
27. Jianwen Fu, Jingfeng Xue, Yong Wang, Zhenyan Liu, and Chun Shan. Malware visualization for fine-grained classification. *IEEE Access*, 6:14510–14523, 2018.
28. Edmar Rezende, Guilherme Ruppert, Tiago Carvalho, Antonio Theophilo, Fabio Ramos, and Paulo de Geus. Malicious software classification using vgg16 deep neural network's bottleneck features. In *Information Technology-New Generations*, pages 51–59. Springer, 2018.
29. Edmar Rezende, Guilherme Ruppert, Tiago Carvalho, Fabio Ramos, and Paulo De Geus. Malicious software classification using transfer learning of resnet-50 deep neural network. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1011–1014. IEEE, 2017.
30. Zhihua Cui, Lei Du, Penghong Wang, Xingjuan Cai, and Wensheng Zhang. Malicious code detection based on cnns and multi-objective algorithm. *Journal of Parallel and Distributed Computing*, 129:50–58, 2019.
31. Duc-Ly Vu, Trong-Kha Nguyen, Tam V Nguyen, Tu N Nguyen, Fabio Massacci, and Phu H Phung. A convolutional transformation network for malware classification. In *2019 6th NAFOSTED conference on information and computer science (NICS)*, pages 234–239. IEEE, 2019.
32. Hamad Naeem, Farhan Ullah, Muhammad Rashid Naeem, Shehzad Khalid, Danish Vasan, Sohail Jabbar, and Saqib Saeed. Malware detection in industrial internet of things based on hybrid image visualization and deep learning model. *Ad Hoc Networks*, 105:102154, 2020.
33. S Sriram, R Vinayakumar, V Sowmya, Mamoun Alazab, and KP Soman. Multi-scale learning based malware variant detection using spatial pyramid pooling network. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 740–745. IEEE, 2020.
34. Yuntao Zhao, Wenjie Cui, Shengnan Geng, Bo Bo, Yongxin Feng, and Wenbo Zhang. A malware detection method of code texture visualization based on an improved faster rcnn combining transfer learning. *IEEE Access*, 8:166630–166641, 2020.

35. Niccolò Marastoni, Roberto Giacobazzi, and Mila Dalla Preda. Data augmentation and transfer learning to classify malware images in a deep learning context. *Journal of Computer Virology and Hacking Techniques*, pages 1–19, 2021.
36. Vinayakumar Ravi, Vasundhara Acharya, and Tuan D Pham. Attention deep learning-based large-scale learning classifier for cassava leaf disease classification. *Expert Systems*, page e12862.
37. Niket Bhodia, Pratikkumar Prajapati, Fabio Di Troia, and Mark Stamp. Transfer learning for image-based malware classification. *arXiv preprint arXiv:1903.11551*, 2019.
38. S Akarsh, K Simran, Prabaharan Poornachandran, Vijay Krishna Menon, and KP Soman. Deep learning framework and visualization for malware classification. In *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, pages 1059–1063. IEEE, 2019.
39. Zhihua Cui, Fei Xue, Xingjuan Cai, Yang Cao, Gai-ge Wang, and Jinjun Chen. Detection of malicious code variants based on deep learning. *IEEE Transactions on Industrial Informatics*, 14(7):3187–3196, 2018.
40. S Akarsh, Prabaharan Poornachandran, Vijay Krishna Menon, and KP Soman. A detailed investigation and analysis of deep learning architectures and visualization techniques for malware family identification. In *Cybersecurity and Secure Information Systems*, pages 241–286. Springer, 2019.