Visual Profiling and Automated Classification of Malware Samples using Deep Learning.

P. Subhash[1*], Y. Sri Varsha[2], K. Saketh Reddy[3], B. Akshaya[4], S. Kalyani[5]

[1,2,3,4,5]VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad.
subhash.parimalla@gmail.com

Abstract. Information security is facing a significant issue due to the proliferation of malware programs. Malware analysis refers to the process of interpreting malicious software to determine its functionality and intent and assist in detection. Conventional methods, which rely on both static and dynamic analyses for malware identification and categorization, often strive to keep up with the ever-rising evolution of malware. Therefore, our proposal presents a thorough deep learning powered malware analysis system that is divided into three essential modules: data processing, feature extraction, detection, and classification. The data processing module handles converting binary data into grayscale photos specifically, includes an import feature, and skillfully extracts essential virus information. This module makes effective use of these extracted attributes to identify potentially suspicious samples and classify malware cases. The Detection and classification module, which completed the architecture, uses deep learning algorithms to identify malware and classify into respected families, resulting in a strong and proactive approach to cybersecurity. This paper contributes to the realm of enhanced cybersecurity by providing a method that not only enhances accuracy but also has the potential to adapt to emerging malware threats.

Keywords: Malware, Visualization, Detection, Classification, Deep Learning.

1 Introduction

Malware is malicious software intended to harm or allow unauthorized access to computer networks and systems. It includes Trojan horses, worms, viruses, and other harmful software, causing a serious threat to information security. According to Kaspersky Labs, corporate systems are routinely targeted and a sizable portion of them are attacked. Despite the ongoing flood of new malware, many of them descend from families that already exist, emphasizing the significance of accurately classifying and identifying them. Thousands of new malware samples are found every day however a sizable number of these samples can be linked to known malware families (Egele et al., 2012).

Malware classification improves proactive cybersecurity by empowering organizations to modify their defenses, spot possible threats before they become problems, and support educated incident management, fortifying the distribution network against

C. Kiran Mai et al. (eds.), *Proceedings of the Fourth International Conference on Advances in Computer Engineering and Communication Systems (ICACECS 2023)*, Atlantis Highlights in Computer Sciences 18,
https://doi.org/10.2991/978-94-6463-314-6_25

developing cyberattacks. These evasion techniques are now widely utilized to avoid the detection mechanisms of antivirus software (Musale et al., 2015). Antivirus makers often rely on signatures approaches in their malware detection and classification efforts. This signature method [1], [2] asserts to be extremely accurate, but it has trouble detecting new malicious programs, demanding manual real-time feature library updates (Yan et al., 2013). The difficulty of effectively categorizing various malware in cybersecurity is highlighted by anomaly detection's tendency to produce significant false alarm rates while also identifying new hazards.

Majority of the current methods for classifying malware extract information through static or dynamic analysis [3]. Traditional malware analysis techniques have issues with reverse analysis, dynamic insights, and manual feature development, which limits their ability to effectively combat the emergence of several malware variants. To improve malware classification accuracy, this study proposes a novel method that uses pictures and deep learning algorithms at the byte level. This method successfully addresses changing threats and enhances detection systems.

The paper is structured as follows: Section 2 provides an overview of previous work on malware classification. In Section 3, the working process of detecting and classifying malware images method. The experimental evaluation is demonstrated in Section 4 The work is concluded; section 5 explores potential future research areas.

2 Literature Survey

This section gives a general overview of common existing malware classification techniques, prioritise if they use unit testing or frequency analysis.

Igor Santos et al. [1] focuses on static malware detection and identifying malware using function length methods. They compared function length with factors like opcode frequency and function calls by examining function length with various malwares. Function length was successful in classification and complemented other analysis methods. Future work should improve calculating techniques and investigate how they relate to other code aspects.

The study used byte-level n-grams [4] that were compressed using random projections to condense the feature space while maintaining inter-sample distances. When a neural network classifier was trained using the generated feature vectors, it was able to classify data with over 98 percent accuracy. The method was found to be effective for large feature vectors, adaptable to different features, and providing a versatile tool for malware analysis.

Decision trees, random forests, Naive Bayes, and k-NN algorithms were used in conjunction with a variety of features, such as byte frequency, opcode n-grams, and DDL calls. K-NN generated about 97 percent [3], whereas Random Forest and Naive Bayes [5] produced accuracy levels of above 98 percent. The importance of feature selection and SVM optimization for reliable malware classification is highlighted by the 97.18 percent accuracy achieved by combining PCA and RFE.

Low-level malware elements, such as system calls, network activity, and file actions, were retrieved using RNN and ESN models [6]. High-level classifiers like SVMs and

random forests were then fed these features. With a 98.68 percent accuracy and speedier training and testing, the RNN-based technique surpassed SVMs and Random Forests. The explanation of malware classification using deep convolutional neural networks is presented in the publication [7,11,12]. It concentrates on using deep learning methods to classify malware. Convolutional neural networks are used in this method to extract features from malware data.

3 Proposed Methodology

3.1 System overview

Fig. 1 depicts our system's tripartite structure. The work is divided into various components: data processing, feature extraction, and categorization. Data processing model involves extracting the required features, employing grayscale images, and importing functions for malware feature extraction and selection. Feature selection focuses on solving dimensionality issues and enhances the performance of detecting and classifying. An image array measuring 64x64 pixels is produced after being processed by feature extraction, and it is then combined with the label for species of attributable malware. Next, a training set and a testing set are initiated by casually adjusting the synthetic data. A large set of classifiers are trained with a few examples to forecast unknown samples and jointly detect suspect ones. In categorizing, suspicious samples are assigned to respective malware families. Samples without confirmed malware links are retained for forthcoming database classifying.

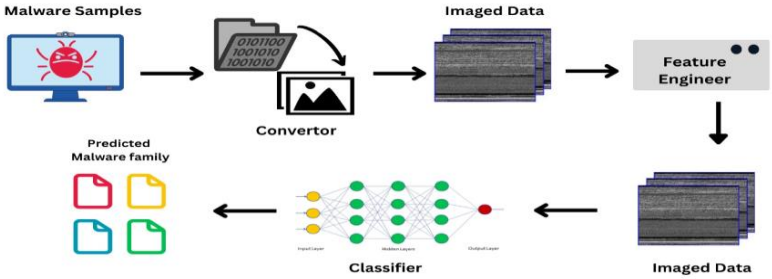

Fig. 1. Overview of the process.

3.2 Modules

3.2.1 Data processing

Fig. 2 illustrates the conversion process from binary data into images. Each byte of binary data is converted into a decimal integer in the [0-255] range, which is then saved in a 1D array and used to represent the values of the grayscale pixels. This approach arranges the data into a 2D grayscale image with fixed width and variable height to fit

different file sizes. The resulting image is then stored in PNG format using libraries like cv2 or PIL.

Fig. 2. Conversion of malware binary data into image.

3.2.2 Feature Extraction

As mentioned in [8], [9], and [10], malware variants frequently derive from recycled core codes, indicating the potential for using instruction sequence similarity for categorization. To recognize similar pixel regions in images, it converts instructions into pixel values. This is an important feature since identical sequences may appear in different places within the same malware family image contrast is essential for spotting analogous pixel patches because it draws attention to similar instruction sequences.

To improve local and global image contrast, contrast-limited adaptive levelling is used during the feature extraction process. Four crucial processes are included in this procedure: segmentation into small fragments, cumulative frequency analysis, checking that the clip limit threshold is fulfilled, and pixel value transformation based on index value, like signal transformation in neural networks. The conversion process is

$$y = h_R(x) = round\left(\frac{cdf(x) - cdf_{min}}{cdf_{\max} - cdf_{min}} X (L-1)\right) \quad (1)$$

L, that implies the estimated range of image pixels, Cdfmax strives for the peak score and the cumulative rate function's minimal no zero limit, notated via cdfmin, was predicted since the Cumulation stage. These changes improve contrast and capture relationships between pixel values, ultimately strengthening the feature extraction capabilities of the Classifier. After being altered, the image is scaled and fed into Classifier for more investigation.

3.2.3 Categorization

To determine the likelihood of categorizing malware samples into families, we use the CNN algorithm. This CNN-based approach benefits from CNN's strength in image processing, enabling effective differentiation between various malware families. The initial convolutional layer's deliverables, input, which was originally a 2D vector (w, h), is transformed into a 3D vector (w, h, 1).

Zero-padding filters are used in convolutional layers, and regularization prevents overfitting. Layers with sequential max-pooling keep settings consistent. After convolution and max pooling, a dropout layer strengthens generalization. Through completely coupled feedforward layers, the output is flattened. The malware's likelihood of being a

member of families is determine a SoftMax activation mode is characterized by the next network of connected stack. The classifier analyses a malware image during testing and determines the appropriate family.

$$L_{088} = -\sum_{i=1}^{f3} y_{i \log y\hat{}_i} \quad (2)$$

The true label (yi) and predicted label (y^i) for malware samples are included in the classifier's loss function. Effectively differentiating between various malware types in the dataset depends on minimizing this loss. This minimization improves the classifier's capacity to identify the proper malware family when given a malware image as input during testing.

4 Evaluation

4.1 Experiment Setup

The Malimg dataset, which consists of 9,342 malware pictures grouped into 25 malware families, is one of the malware datasets that are available for academic research. These images are from a variety of families, including Worm, Dialer, Backdoor, PWS, Rouge, Trojan, Trojan downloader. To make the images compatible with CNN models, the images were first transformed into 8-bit vector binary and then back into images. The fact that PE files are the source of all malware pictures is noteworthy.

4.2 Evaluation Results

A Malimg dataset evaluation is carried out, along with comparisons to a version of the dataset even with the transition stage and couple alternative ways utilizing distinct tactics. The accuracy is evaluated using a 10-fold layered cross-validation, which guarantees balanced family representation across folds. Table 1 lists the computed averages for accuracy, precision, recall, and F1 scores. According to the results, our system achieves average F1 scores of 95.2 percent, 95.3 percent, 96.0 percent, and 96.0 percent, respectively. With an accuracy close to or above 98 percent, the majority of the 25 malware families are accurately described. A few noteworthy categorization recall rates are 100% for Yuner.A, 97.9% for VB.AT, 97.8% for Malex.gen!J, and 94.5 for Rbot!gen. Due to their striking similarities, samples of Autorun.K that were incorrectly labelled as Yuner.A. The confusion matrix in Fig.3. which displays the precise classification outcomes, indicates that allYunr.A is the category for samples of the malware Autorun.K. The resulting mutual claifcaion of the respective malware samples into each other families result in a somewhat low level of accuracy.

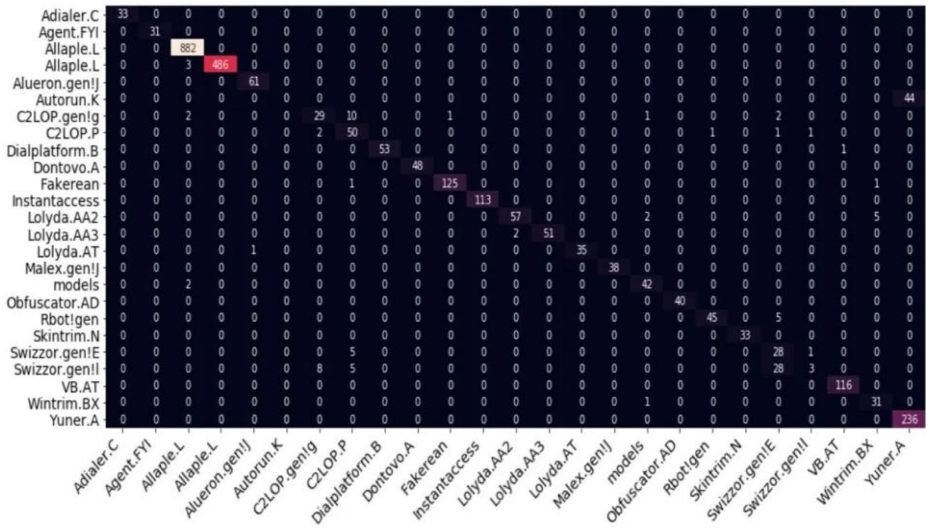

Fig. 3. Confusion Matrix.

The comparative results between several classification techniques used on the Malimg dataset are shown in Fig. 4. While LR exhibits good recall with moderate accuracy, the deep learning-based technique achieves an accuracy of 91% with a BAT algorithm to balance the sample numbers, DRBA+CNN achieves an accuracy of 94.5 percent. Each malware grayscale image is subjected to Gabor filtering to extract statistical features, and GIST finally chooses 320 of these features as input for the k-nearest neighbour algorithm in the final classification step. By counting the occurrence of neighbouring byte pairs and utilising a deep convolution-al neural network for categorization, MDMC, in contrast, converts malware binaries into Markov probability matrices. In terms of malware classification, our model performs noticeably better than GIST and MDMC, with GIST scoring somewhat better but being less effective.

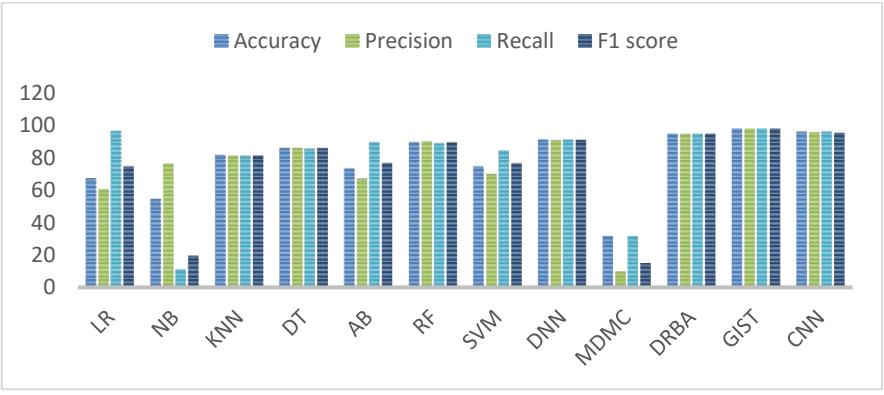

Fig. 4. Comparison Results

Table 1. Classification Report

Family of Malwares	Accuracy	Precision	Recall	F1-score
Adialer.C	100	98.9	99.9	97
Agent.FYI	99.9	97.6	99.9	98.9
Allaple.A	100	99.5	100	99.7
Allaple.L	99.8	100	99.8	99.9
Alueron.gen!J	98.2	99.6	98.1	98.9
Autorun.K	93	94	93	92
C2LOP.gen! g	80.7	84.4	80.4	78.5
C2LOP.P	76.7	76.2	76.7	75.4
Dialplatform.B	100	99.6	100	99.9
Dontovo.A	99.9	99.3	99.9	99.6
Fakerean	98.6	99.4	98.2	98.6
Instantaccess	100	99.9	99.9	100
Lolyda.AA1	96.5	97.7	96.8	97.3
Lolyda.AA2	97	96.9	97.2	96.9
Lolyda.AA3	97.8	99.3	97.8	98.1
Lolyda.AT	98	98.3	98.4	98.4
Malex.gen!J	97	96	97.9	95
Obfuscator.AD	99.9	98	99.9	99.9
Rbot!gen	94	93.6	92.7	93
Skintrim.N	90.1	87.8	90.2	88.5
Swizzor.gen!E	47.6	52.6	47.4	46.3
Swizzor.gen!I	43.4	51.2	43.3	36.8
VB.AT	97.7	99.2	97.7	98.3
Wintrim.BX	97.8	99.2	97.9	98.6
Yuner.A	100	89.3	99.9	94.4

5 Conclusion

An innovative and effective malware categorization model is introduced in this paper. This technique improves classification accuracy through dynamic analysis while addressing the difficulty of comprehending the complex architectures of various malware samples. By focusing on texture representation and moving away from traditional binary interpretation, it turns malware samples into images. By using encoding to identify commonalities within and differences between malware families, the model reveals byte code encoding patterns. It uses a contrast-limited adaptive equalization technique to improve classification accuracy by increasing local contrast and focusing on similar image regions rather than similar codes. Additionally, it offers a robust method for security engineers to analyze classification rates and pinpoint variables affecting Classifiers, assisting in performance improvement.

Future studies will investigate the relationship between the byte-code sequences and the malware picture information learned by neural networks. This knowledge might result in signature extraction for malware families, possibly handled by blockchain for wider malware detection. The model will also go into detail about finding software weaknesses using visualization approaches, giving developers insights to improve their scripts. Continued work will also improve the model's capacity to differentiate between malicious and good software.

References

1. R. Tian, L. M. Batten "Function length as a tool for malware classification," 2008.
2. G. E. Dahl, J. W. Stokes, L. Deng, D. Yu, "Large-scale malware classification using random projections and neural networks," 2013.
3. N. Udayakumar, V. J. Saglani, A. V. Cupta and T. Subbulakshmi, "Malware Classification Using Machine Learning Algorithms," 2018.
4. Nari, Saeed, and Ali A. Ghorbani. "Automated malware classification based on network behavior." 2013.
5. R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu and A. Thomas, "Malware classification with recurrent networks," 2015.
6. F. Zhong, Z. Chen, M. Xu, G. Zhang, D. Yu and X. Cheng, "Malware-on-the-Brain: Illuminating Malware Byte Codes with Images for Malware Classification," 2023.
7. Yuan, Baoguo, et al. "Byte-level malware classification based on markov images and deep learning." 2020.
8. A. Calleja, J. Tapiador, and J. Caballero, “The malsource dataset: Quantifying complexity and code reuse in malware development,” 2019.
9. D. Korczynski and H. Yin, “Capturing malware propagations with code injections and code-reuse attacks,” 2017.
10. N. Marastoni, A. Continella, D. Quarta, S. Zanero, and M. D. Preda, “GroupDroid: Automatically grouping mobile malware by extracting code similarities,” 2017.
11. M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang and F. Iqbal, "Malware Classification with Deep Convolutional Neural Networks," 2018.
12. T. Shibahara, T. Yagi, M. Akiyama, D. Chiba and T. Yada, "Efficient Dynamic Malware Analysis Based on Network Behavior Using Deep Learning," *2016*