Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.038639
Article

VMCTE: Visualization-Based Malware Classification Using Transfer and Ensemble Learning

Zhiguo Chen[1,2,*] and Jiabing Cao[1,2]

[1]Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology, Nanjing, 210044, China
[2]School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, 210044, China
*Corresponding Author: Zhiguo Chen. Email: chenzhiguo@nuist.edu.cn
Received: 22 December 2022; Accepted: 17 February 2023

Abstract: The Corona Virus Disease 2019 (COVID-19) effect has made telecommuting and remote learning the norm. The growing number of Internet-connected devices provides cyber attackers with more attack vectors. The development of malware by criminals also incorporates a number of sophisticated obfuscation techniques, making it difficult to classify and detect malware using conventional approaches. Therefore, this paper proposes a novel visualization-based malware classification system using transfer and ensemble learning (VMCTE). VMCTE has a strong anti-interference ability. Even if malware uses obfuscation, fuzzing, encryption, and other techniques to evade detection, it can be accurately classified into its corresponding malware family. Unlike traditional dynamic and static analysis techniques, VMCTE does not require either reverse engineering or the aid of domain expert knowledge. The proposed classification system combines three strong deep convolutional neural networks (ResNet50, MobilenetV1, and MobilenetV2) as feature extractors, lessens the dimension of the extracted features using principal component analysis, and employs a support vector machine to establish the classification model. The semantic representations of malware images can be extracted using various convolutional neural network (CNN) architectures, obtaining higher-quality features than traditional methods. Integrating fine-tuned and non-fine-tuned classification models based on transfer learning can greatly enhance the capacity to classify various families of malware. The experimental findings on the Malimg dataset demonstrate that VMCTE can attain 99.64%, 99.64%, 99.66%, and 99.64% accuracy, F1-score, precision, and recall, respectively.

Keywords: Malware classification; ensemble learning; visualization; transfer learning

1 Introduction

Malware is malicious software designed to damage or disrupt the normal use of an endpoint device. Based on its functionalities and characteristics, it can be categorized into many types; examples include Trojan horses, spyware, viruses, worms, adware, ransomware, and rootkits. Malware is used by cybercriminals to target computers and networks to sabotage or control computer resources, obtain money, and steal business intelligence or state secrets. The McAfee Labs Threats Report (June 2021) [1] revealed that the total volume of malware threats averaged 688 per minute, an increase of 40 threats per minute in the first quarter of 2021. The economic losses caused by malicious attacks should not be underestimated. According to a survey by Check Point [2], the average ransom demand brought on by ransomware rose to $233,817 in the third quarter of 2020, a 30% rise from the second quarter. Overall, the rapid growth and evolution of malware pose considerable security threats to businesses and individuals. Therefore, there is a pressing need to enhance the performance of systems that automatically classify and detect malware.

Many researchers have put forth research plans meant to boost malware detection and classification efficiency [3,4]. Analyzing static data and analyzing dynamic data are the two primary approaches. Static analysis can extract a variety of static features, such as portable executable (PE) headers, n-grams, opcode sequences, and other features for code-level classification and detection, through classic static analysis tools (e.g., PEview, Yara, IDA Pro, Accesschk), without executing the malware. However, cybercriminals can evade static analysis using methods such as fuzzing, obfuscation, and encryption. Conversely, dynamic analysis requires the loading and running of malware samples and relies more on interaction with the computer for analysis. It usually involves building a secure, controlled environment that runs sample files to monitor activities, including registry key changes, application programming interface (API) calls, or system file deletions. Dynamic analysis might potentially infect endpoint systems and is time-consuming, although being effective against obfuscation tactics. Furthermore, most static analysis tools construct classification systems through the fusion of various features of different malware classes. Dynamic analyses involve the construction of isolated environments and the capture of behavioral features. Both methods require the researcher to have adequate domain expertise and experience.

Malware identification and classification have both benefited greatly from the use of machine learning techniques [5–8]. Conventional machine learning methods are superior to deep learning methods when the sample data size is limited. However, with the exponential growth of malicious programs and their increasing complexity, it is challenging to adapt machine learning techniques to tasks in complex scenarios (e.g., insufficient learning in the face of high-dimensional features, tedious feature engineering, and weak adaptability). As a result, classical machine learning techniques alone cannot completely meet the needs for malware detection and classification in today's increasingly severe computer security environment. Deep learning-related technologies have advantages in dealing with high-dimensional data. To reduce computational resources and time consumption, there must not be over-reliance on domain expertise. The utilization of image visualization techniques for malware classification and detection has been advocated in recent studies with encouraging outcomes [9,10]. The public dataset used for this research suffers from an imbalance in the number of malware family samples. Transfer learning is used in this research to lessen the effect of data imbalance on classification performance. To further improve the performance of classification systems, researchers have used ensemble learning to fuse the advantages of multiple classification models to achieve accurate classification of malware families [11,12].

This paper introduces a novel visualization-based malware classification system (VMCTE), which fuses the strengths of three classical neural networks (ResNet50, MobilenetV1, and MobilenetV2) based on ensemble learning. All three neural networks transfer weights pre-trained on the ImageNet dataset [13]. Transfer learning effectively alleviates the impact of data imbalance on classification accuracy, and also makes the network unnecessary to train from scratch. The system achieves a promising classification performance on the Malimg dataset [14], which contains obfuscated and encrypted malware samples. The following are the work's main contributions:

- A new malware classification system is proposed, which reduces the reliance on feature engineering techniques and domain expertise. Furthermore, it has high classification accuracy, a straightforward deployment procedure, and a novel structure.
- This paper fuses fine-tuned and non-fine-tuned deep convolutional neural network (CNN) classification structures based on transfer learning. Compared with sub-classification systems, experimental results show that the fused model can improve malware classification.
- On the Malimg dataset, which contains obfuscated and encrypted samples, the suggested system's capacity to categorize malware samples is assessed. The experiment demonstrates that the system is capable of classifying malware samples with accuracy and F1-score of 99.64% and 99.64%, respectively.

The remainder is organized as follows: The discussion and review of malware researchers' work can be found in Section 2. Section 3 provides a comprehensive explanation of VMCTE. In Section 4, the experimental findings are assessed and displayed. The research is finally summarized and a future work plan is presented in Section 5.

2 Related Work

In this section, related works on anomaly-based malware analysis are reviewed, including static analysis, dynamic analysis, and visualization analysis. Some of the techniques employed in the proposed malware classification system are then introduced, including transfer and ensemble learning.

2.1 Anomaly-Based Malware Analysis

The anomaly-based malware analysis method, which is primarily introduced from three aspects: static analysis, dynamic analysis, and visualization analysis, is introduced in this part.

2.1.1 Static Analysis

Anti-malware systems based on static analysis usually rely on extracting and analyzing static features of PE files, such as hash values, n-grams, PE header information, opcode sequences, string information, and their combinations to identify malware. Static analysis-based anti-malware solutions often rely on extracting and analyzing static aspects of PE files, including PE header information, n-grams, hash values, opcode sequences, string information, and their combinations. Moskovitch et al. [15] utilized opcodes generated by disassembling executables as classification features and used machine learning algorithms to classify malware. The approach achieved an accuracy of over 99% on the dataset, in which less than 15% of the files were malicious. For the first time, Schultz et al. [16] utilized data mining technologies for the classification of malware. They attained a classification accuracy of 97.11% utilizing static features extracted from string sequences, byte sequences, and PE headers, demonstrating the feasibility of data mining techniques in malware classification. Narayanan et al. [17] proposed using principal component analysis (PCA) to extract malware features. On a public dataset (BIG2015) offered by the Microsoft Malware Classification Challenge, the approach used the k-nearest neighbors (KNN) algorithm and achieved 96.6% classification accuracy. David et al. [18] found that a large portion of malware did not adhere to Microsoft specifications for the MZ-PE format. Their experimental results showed that structural analysis tests on PE headers can accurately identify malware. These results could greatly reduce the workload of analyzing malware and are useful for future research.

Static malware analysis techniques are fast and can be used for large-scale real-time classification. However, their classification efficiency is lacking in the face of malware that uses techniques such as obfuscation, encryption, and fuzzing. Currently, malware classification schemes are primarily built based on the researchers' expertise or the features suggested by the literature to which they refer, which may result in the absence of some crucial aspects and limit the classification performance. In addition, static analysis usually requires researchers to obtain a level of domain expertise and knowledge of feature engineering techniques.

2.1.2 Dynamic Analysis

It is common practice to identify and classify newly discovered malware samples using dynamic analysis-based malware analysis techniques. Researchers execute software samples in a secure environment to distinguish between malicious and benign sample files by observing system resource consumption and activities such as API calls, registry key changes, new file creation, and web access. Pektaş et al. [19] introduced a malware classification approach for Windows computers utilizing runtime behavior. The system achieved 98% classification accuracy on a dataset of 17,400 malware by examining features such as network usage, file access, registry activity, and API calls. Dash et al. [20] presented a malware classification system that combines conformal prediction and support vector machine (SVM). The system uses system calls and virtual machine introspection to reconstruct inter-process communication, which can alleviate the issue of data sparsity. It achieved 94% classification accuracy on datasets collected in the real world. Cai et al. [21] introduced a malware classification model (DroidCat) that utilizes method calls and inter-component communication. The system can fully handle reflective calls without using features based on application resources and permissions. The approach earned an F1-score of 97.8% on the Malgenome dataset. Afonso et al. [4] presented a classification system based on machine learning and API calls that can effectively classify malware. The classification accuracy of the system was 96.66% on a dataset containing 7,520 malicious samples.

Dynamic analysis can effectively identify malware that uses evasive techniques such as obfuscation, fuzzing, and encryption. However, the presentation of dynamic analysis reports is often overly detailed, providing specific details about the behavior of the executable that are irrelevant to the task of identifying malware. The size of the report additionally grows due to the intricacy of the textual representation, which has a detrimental effect on the analysis's time overhead. A dynamic malware classification approach pulls feature vectors from sample behavior reports generated in a controlled environment and then utilizes deep learning or machine learning algorithms to identify and categorize malware. Therefore, quickly classifying malicious samples takes time and is challenging.

2.1.3 Visualization Analysis

The building of malware classification systems based on visualization is a popular study topic in the world of computer security. Researchers have recently conducted various in-depth analyses and examinations of visualization-based malware classification techniques. They have also proposed myriad malware classification systems.

Nataraj et al. [14] proposed an algorithm for converting malware binaries to grayscale images. The experimental results demonstrated the similarity among images from the same malware family. The evaluation was conducted by the researchers on a dataset that contained 25 malware families, and their classification accuracy was 97.18%. Gibert et al. [22] represented the malware as grayscale images to capture minute changes while maintaining the overall structure. The proposed system adopts the CNN-SVM structure. The SVM performs the last classification process utilizing the CNN's output as its input. The authors evaluated the effectiveness of the suggested approach on the Malimg and BIG2015 datasets, achieving classification accuracies of 98.48% and 97.49%, respectively. Choi et al. [23] constructed a classification model using deep learning techniques. The classification accuracy reached 95.66% on a dataset of 30,000 samples (10,000 benign samples and 20,000 malicious samples). Overhead time was not taken into account in this work. Arefkhani et al. [24] applied the perceptual hashing algorithm from image processing to malware classification. They analyzed perceptual hash (PHash), difference hash (DHash), and average hash (AHash), and found that PHash was the most accurate. Su et al. [25] worked on developing a malware classification system that can be applied to real-world scenarios. They proposed a lightweight system that can detect distributed denial of service (DDoS) malware with 94% classification accuracy on real-world datasets. However, this system has a shallow network architecture, and the evaluation dataset contains just two kinds of malware. Narayanan et al. [11] presented a malware classification method, with the aid of ensemble learning, that consists of recurrent and convolutional neural networks. They used assembly (ASM) files as inputs to long short-term memory (LSTM) and BYTE compiled files as inputs to a CNN. The system's classification accuracy was 99.8% when tested against the BIG2015 dataset. The dataset consists of ASM files and BYTE files containing malware from nine families. Yajamanam et al. [26] noticed that deep learning algorithms and global descriptors (GIST) performed comparably in image-based malware classification. The benefit of the deep learning strategy is that it does not require GIST features to be extracted during training. The classification accuracy of this work needs to be further improved, and the overhead time of the system is not considered.

Visualization analysis can identify malware that uses obfuscation, encryption, fuzzing, or other evasion techniques without relying on domain expertise or reverse engineering techniques, and can also avoid excessive time overhead. As a result, this research suggests a malware classification scheme that utilizes image visualization (Table 1).

Table 1: Static analysis, dynamic analysis, and visualization analysis on malware

Year	Authors	Analysis methods	Models	Results
2001	Schultz et al. [16]	Static analysis	Naive bayes	97.11% (Accuracy)
2008	Moskovitch et al. [15]	Static analysis	Boosting decision tree	99% (Accuracy)
2011	Nataraj et al. [14]	Visualization analysis	K-nearest neighbors	97.18% (Accuracy)
2015	Afonso et al. [4]	Dynamic analysis	Random forest	96.66% (Accuracy)
2015	Arefkhani et al. [24]	Visualization analysis	PHash	–
2016	Narayanan et al. [17]	Static analysis	K-nearest neighbors	96.6% (Accuracy)
2016	Dash et al. [20]	Dynamic analysis	Support vector machine	94% (Accuracy)

(Continued)

Table 1: Continued

Year	Authors	Analysis methods	Models	Results
2017	David et al. [18]	Static analysis	MZ-PE	–
2017	Choi et al. [23]	Visualization analysis	CNN	95.66% (Accuracy)
2018	Pektaş et al. [19]	Dynamic analysis	Random forest	98% (Accuracy)
2018	Su et al. [25]	Visualization analysis	CNN	94% (Accuracy)
2018	Yajamanam et al. [26]	Visualization analysis	Machine learning	–
2019	Cai et al. [21]	Dynamic analysis	Random forest	97.8% (F1-score)
2019	Gibert et al. [22]	Visualization analysis	CNN-SVM	98.48% (Accuracy)
2020	Narayanan et al. [11]	Visualization analysis	LSTM + CNN	99.8% (Accuracy)

2.2 Transfer Learning for Malware Classification

Correlations between data are used to transfer knowledge from one domain to another, which is known as transfer learning. It can considerably cut the model's training time and assist in resolving the issue of insufficient data without having to train the model from scratch. Researchers have successfully used transfer learning technology to apply models such as ResNet50, MobilenetV1, MobilenetV2, DenseNet121, Vgg19, and Xception to the field of malware classification [27–29]. Its common practice to transfer the general weights trained on the source domain dataset (usually a natural image dataset) to the malware domain for building an image classification model.

The residual neural network [30] (ResNet) was proposed by Microsoft Research in 2015 and is widely used in malware image classification. It can alleviate the problem of gradients exploding or vanishing in deep networks through the design of shortcut connections. Sudhakar et al. [31] proposed a malware image classification system based on a fine-tuned deep CNN. The system improved the optimizer of the ResNet50 model to NAdam and attained 98.63% accuracy on the Malimg image dataset. Vasan et al. [32] transferred the initial weights trained on ImageNet to a fused malware classification system utilizing ResNet50 and Vgg16. The system can effectively classify packed and unpacked malware, with a classification accuracy of over 98%. Therefore, ResNet50 has the advantages of fewer parameters and higher classification accuracy compared with traditional networks.

MobilenetV1 [33] is a lightweight neural network designed for mobile and embedded devices, proposed by Google in 2017. It decomposes the conventional convolution into depthwise and pointwise convolution using the depthwise separable convolutional structure to decrease the model parameters. MobilenetV2 [34] adopts the inverted residual structure and the new Relu6 activation function, which can improve the classification performance in small-sample scenarios. Bendiab et al. [35] proposed a malware classification system employing MobilenetV1 and visualization techniques. It attained a classification accuracy of 91.32% on a dataset that had a significant amount of zero-day malware. Atitallah et al. [36] proposed a malware multi-classification approach based on transfer learning combining DenseNet161, MobilenetV2, and ResNet18, which achieved 98.74% accuracy and 672 ms classification overhead on the MaleVis dataset. Due to the excellent classification abilities of

MobilenetV1 and MobilenetV2, they have been used widely by researchers in the field of malware classification.

2.3 Ensemble Learning for Malware Classification

To enhance classification performance, ensemble learning is frequently employed in malware classification assignments. Narayanan et al. [11] presented a malware classification structure that combines CNN, AlexNet, ResNet, Vgg16, and LSTM. The system has excellent classification performance, achieving 99.8% classification accuracy on the BIG2015. Ahmed et al. [37] constructed an integrated dynamic and static ransomware classification system with a weighted majority voting strategy. Experiments' findings demonstrate that the approach can successfully classify malware that has not yet been recognized. Vasan et al. [32] proposed a system that fuses Vgg16, ResNet50, and SVM, which achieved a 99.5% accuracy and a 99.46% recall on the Malimg dataset. Taha et al. [12] adopted the Choquet fuzzy integral as an aggregation function to integrate the results of classifiers such as extreme gradient boosting, random forest, decision trees, adaptive boosting, and light gradient boosting machines. Its accuracy on a collection of 15,036 malware samples is 95.08%. Overall, the performance of malware classification systems can be greatly enhanced by the fusion of multiple classifiers.

3 The Proposed Malware Classification System

This section first briefly introduces the VMCTE malware classification system and then gives a detailed description of the feature extraction, dimensionality reduction, and classification parts of VMCTE, respectively.

3.1 Overview of the Proposed System

Fig. 1 depicts the system architecture of the proposed VMCTE. There are three primary parts to it: feature extraction, dimensionality reduction, and classification. Feature extraction is performed using the deep learning frameworks ResNet50, MobilenetV1, and MobilenetV2 to extract features from malware images. Each framework adopts initial weights that were previously trained on the ImageNet dataset, and ResNet50 needs to be fine-tuned. PCA is used to decrease the dimensionality of the extracted feature vectors, removing irrelevant attributes, redundant features, and other unnecessary information. For the classification component, the SVM-based ensemble learning models classify the malware image samples into the appropriate families through the maximum probability algorithm.

3.2 Feature Extraction

Malware that uses obfuscation, fuzzing, encryption, and other evasion detection techniques is difficult to classify using manually extracted features. Therefore, the system employs ResNet50, MobilenetV1, and MobilenetV2 to extract high-dimensional features for classification from raw malware image samples. Both MobilenetV1 and MobilenetV2 transfer the initial weights trained on ImageNet and extract the features contained in the 1,024 and 1,280 neurons of their respective fully connected layers on the Malimg dataset, which are used as the input for the next step. At the same time, ResNet50 is initialized by transferring the weights trained on the ImageNet dataset. For the fine-tuning stage, 25 neurons are employed to replace the original output layer composed of 1,000 neurons and pass the multi-category cross-entropy loss, defined in Eq. (1), and the mini-batch stochastic gradient descent, defined in Eq. (2), to tune the model weights. The features contained in the 2,048 neurons of the fully connected layer of the fine-tuned ResNet50 model are used as the output. The batch size was

32 and the learning rate was 1e-4. The learning rate can affect how quickly the model converges, and a good learning rate can hasten the convergence of the model to the local minimum. We trained 100 epochs to customize the ResNet50 model. The Relu activation function was used in our experiments.

$$L_{mcce} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(\omega_i \cdot x_i\right) \quad (1)$$

$$\omega_{t+1} = \omega_t - \alpha\frac{\partial L_{mcce}}{\partial \omega_t} \quad (2)$$

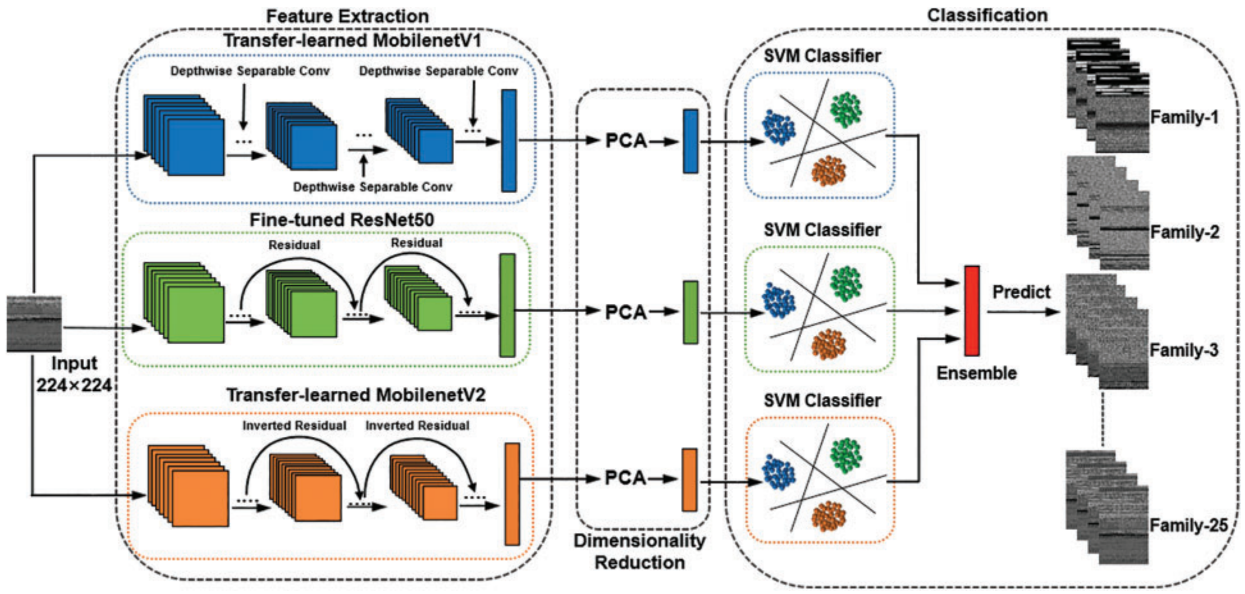


Figure 1: Architecture of VMCTE

Here, L_{mcce} represents the multi-category cross-entropy loss, N is the total number of malicious samples, $\sum$ means to sum the losses of N samples in each batch, y_i is the real family category of the ith sample, $\omega_i \cdot x_i$ is the predicted category of the ith family, ω_t is the current weight value of the ResNet50 model, and ω_{t+1} is the updated model weight value. The learning rate α is utilized to adjust the weight update speed of the ResNet50 model.

3.3 Dimensionality Reduction

Due to the advantages of low sensitivity to noise, low memory consumption, and easy deployment, malware security researchers generally use PCA [38] to lessen the dimensionality of data. In VMCTE, the PCA is utilized to lessen the dimensionality of the extracted high-dimensional features, as described in Algorithm 1.

Algorithm 1: PCA

```
Input:   Sample set $D = \left\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\right\}$, explained variance ratio EVR $\geq 97\%$
Output:  The sample set $X'$ after dimensionality reduction
1:       Decentralize all samples: $x_j^{(i)} \leftarrow x_j^{(i)} - \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$
```

(Continued)

Algorithm 1: Continued

```
2:    Calculate the covariance matrix $XX^T$ of the sample
3:    Eigenvalue decomposition of the covariance matrix $XX^T$
4:    Take the eigenvectors $w_1, w_2, \ldots, w_{d'}$ corresponding to the $d'$ largest eigenvalues
5:    Return the result $X'$ of multiplying the original sample matrix X by the projection matrix
      $W = \{w_1, w_2, \ldots, w_{d'}\}$
```

The high-dimensional feature dimensions extracted by the automatic feature extractors MobilenetV1, MobilenetV2, and ResNet50 are reduced from 1,024, 1,280, and 2,048 to 205, 256, and 205, respectively, by adjusting the explained variance ratio to retain more than 97% of the original information. To keep the information loss as small as possible, the features most conducive to malware classification are extracted as the input of the SVM classifiers.

3.4 Classification

SVM [39] is a well-known supervised learning algorithm that is extensively applied in malware classification tasks. Dash et al. [20] introduced a malware classification system that involved fusing SVM and conformal prediction. The system is capable of effectively solving the data sparsity issue, according to experimental results. Xiao et al. [40] proposed a system based on malware visualization. The system visualized malware binary files as structural entropy graphs and achieved 100% classification accuracy on the BIG2015 dataset using the SVM as a classifier. Liu et al. [41] built a malware classification system by improving the SVM algorithm with kernel functions and parameter optimization. On the CAIDA dataset, it has a 92.5% accuracy and a 5.527% false positive rate. Thus, the identification and classification of malware can be effectively handled by the SVM algorithm. The SVM algorithm flow is as follows:

Suppose the given dataset is $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where $x_i \in R^n$ and $y_i \in \{+1, -1\}$. A positive sample has $y_i = +1$, and a negative sample has $y_i = -1$. SVM splits the dataset by finding a separating hyperplane that maximizes the geometric margin. The separating hyperplane is described by Eq. (3).

$$\omega^* \cdot x + b^* = 0 \tag{3}$$

where $\omega \in R^n$ and b $\in$ R are used to define the hyperplane. To obtain the optimal solution of ω and b, this problem is turned into an optimization problem as shown in Eqs. (4), and (5) is its constraint.

$$\min_{\omega,b,\xi} \frac{1}{2}||\omega||^2 + C\sum_{i=1}^{N} \xi_i \tag{4}$$

$$s.t. y_i (\omega \cdot x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, \ldots, N \tag{5}$$

where $||\omega||^2$ is the L2 norm of ω, ξ_i is the slack variable, and C is the penalty parameter. The primal optimization problem of SVM must be transformed into a dual problem through Eq. (6) to obtain the optimal solution. Eq. (7) is the constraint.

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j \left(x_i \cdot x_j\right) - \sum_{i=1}^{N} \alpha_i \tag{6}$$

$$s.t. \sum_{i=1}^{N} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C, i = 1, 2, \ldots, N \tag{7}$$

where $\alpha(0 \leq \alpha \leq C)$ is the Lagrange multiplier vector. Suppose the solution of α is $\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*)^T$; w^* and b^* can then be obtained by Eqs. (8) and (9).

$$w^* = \sum_{i=1}^{N} \alpha_i^* y_i x_i \tag{8}$$

$$b^* = y_j - \sum_{i=1}^{N} y_i \alpha_i^* (x_i \cdot x_j) \tag{9}$$

The optimal separating hyperplane and classification decision function can then be derived from w^* and b^*. As shown in Eqs. (10) and (11).

$$\sum_{i=1}^{N} \alpha_i^* y_i (x \cdot x_i) + b^* = 0 \tag{10}$$

$$F(x) = sign\left(\sum_{i=1}^{N} \alpha_i^* y_i (x \cdot x_i) + b^*\right) \tag{11}$$

To handle linearly inseparable sample points, SVM introduces a slack variable ξ and a penalty parameter C. This makes SVM resilient to outliers. Additionally, it offers special benefits when resolving high-dimensional and small-sample issues. Therefore, SVM was chosen as the basic malware classifier, which aims to find an optimal separating hyperplane to divide the dimensionally reduced data. To improve performance, this paper combined three high-precision classification models. To achieve ideal fusion performance, the maximum probability fusion strategy is used to compare the decision probability values of different classifiers for the same malware image, to determine the family category to which the malware corresponds. It means to compare the decision results of different classifiers, and the corresponding family category with the highest probability in the decision result is the predicted family category.

$$f_i^* = argmax\left(\max{}^*\left(p_{i,c_1}\right), \max{}^*\left(p_{i,c_2}\right), \max{}^*\left(p_{i,c_3}\right)\right) \tag{12}$$

where i represents different malware image samples, and c_1, c_2, and c_3 represent three SVM classifiers with different structures. The operator max^* indicates the maximum probability value (with family category) of the classifier's determination result for each malware image. $argmax$ is the family category corresponding to the classification probability with the largest value in the classification result. f_i^* is the final predicted family of the ith sample.

4 Experiments

The datasets and assessment criteria utilized in the experiments are briefly introduced in this section. The experimental findings are meticulously analyzed and compared in detail with existing work.

4.1 Dataset

In this research, the performance of VMCTE was assessed using the Vision Research Lab's Malimg dataset [14]. The dataset contains 9,339 grayscale images converted from malware binaries, covering 25 families of malware. Malimg is a benchmark dataset for the classification of malware. Table 2 provides a brief description of the dataset. An example image from each malware family is displayed in Fig. 2.

4.2 Evaluation metrics

This paper used metrics of accuracy, recall, precision, and F1-score to assess the efficacy of the suggested system. An ideal malware classification system generally has high accuracy, F1-score, precision, and recall. To unbiasedly evaluate the effectiveness of malware classification systems, these assessment metrics have been widely employed in the research community [42–44].

Table 2: Description of the Malimg dataset

No.	Family name	Type	No. of samples	No.	Family name	Type	No. of samples
1	Adialer.C	Dialer	122	14	Lolyda.AA2	PWS	184
2	Agent.FYI	Backdoor	116	15	Lolyda.AA3	PWS	123
3	Allaple.A	Worm	2949	16	Lolyda.AT	PWS	159
4	Allaple.L	Worm	1591	17	Malex.gen!J	Trojan	136
5	Alueron.gen!J	Trojan	198	18	Obfuscator.AD	Trojan downloader	142
6	Autorun.K	Worm:AutoIT	106	19	Rbot!gen	Backdoor	158
7	C2LOP.gen!g	Trojan	200	20	Skintrim.N	Trojan	80
8	C2LOP.P	Trojan	146	21	Swizzor.gen!E	Trojan downloader	128
9	Dialplatform.B	Dialer	177	22	Swizzor.gen!I	Trojan downloader	132
10	Dontovo.A	Trojan downloader	162	23	VB.AT	Worm	408
11	Fakerean	Rogue	381	24	Wintrim.BX	Trojan downloader	97
12	Instantaccess	Dialer	431	25	Yuner.A	Worm	800
13	Lolyda.AA1	PWS	213				

Accuracy is the most commonly used evaluation metric and is easy to understand. The accuracy is calculated as the number of samples that were accurately predicted divided by the total number of samples. The model's capacity for classification improves with increasing accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

Another typical evaluation metric is precision. It shows the percentage of samples that were expected to be positive but were really positive.

$$Precision = \frac{TP}{TP + FP} \tag{14}$$

The recall is a measurement of the proportion of all positive samples that were actually identified as such. It assesses the model's capacity to recognize positive samples.

$$Recall = \frac{TP}{TP + FN} \tag{15}$$

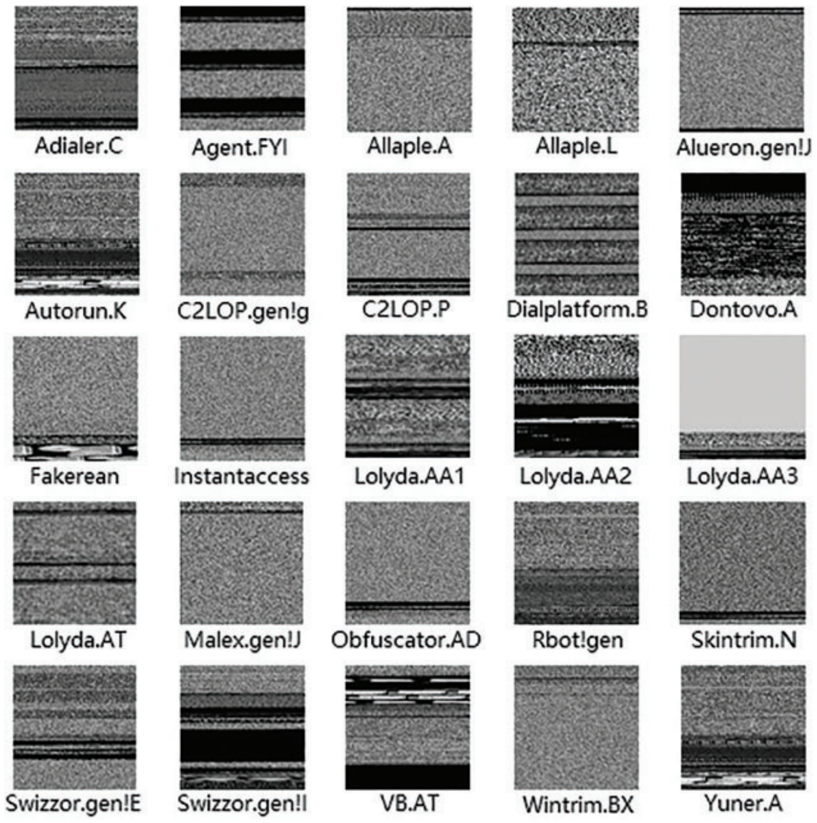


Figure 2: Grayscale images of different malware families

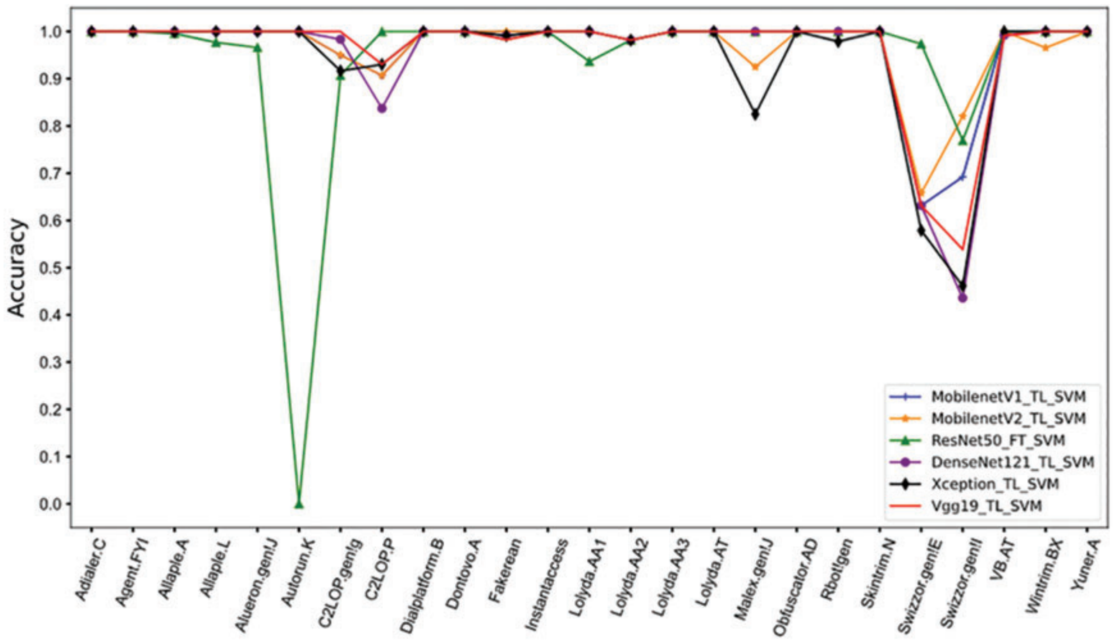
A comprehensive evaluation metric known as the F1-score is defined as the harmonic mean of recall and precision. When recall and precision greatly diverge, the F1-score provides a compromise value.

$$F1 = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{16}$$

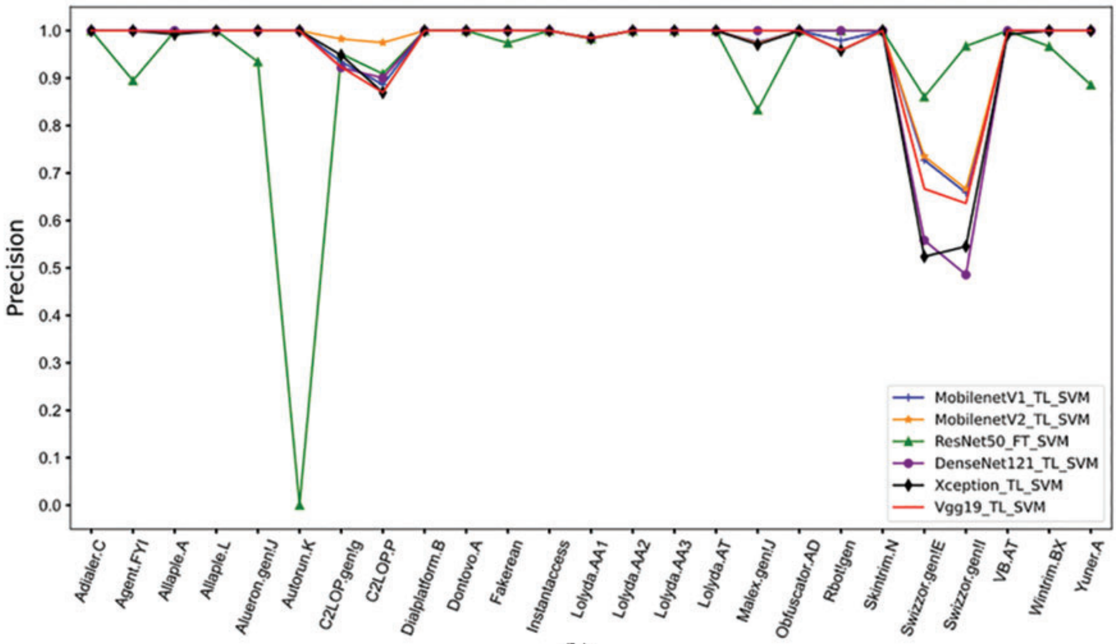
where TP, TN, FP, and FN represent the corresponding totals for true positives, true negatives, false positives, and false negatives.

4.3 Experimental Results and Analysis

Malware researchers have had success classifying malware using techniques from the field of computer vision. This section describes system fusion experiments that were carried out based on six well-known neural network structures in the field of computer vision; they include ResNet50, DenseNet121, MobilenetV1, MobilenetV2, Vgg19, and Xception as feature extractors. This paper used classification systems based on fine-tuned and non-fine-tuned network structures (FT for fine-tuning, TL for transfer learning only) for efficient fusion. In classification precision, accuracy, recall, and F1-score for all 25 families, MobilenetV1_TL_SVM, MobilenetV2_TL_SVM, DenseNet121_TL_SVM, Vgg19_TL_SVM, Xception_TL_SVM, and ResNet50_FT_SVM all perform well, as shown in Fig. 3. Although ResNet50_TL_SVM has poor classification performance for the Autorun.K family, it can distinguish the Swizzor.gen!I and Swizzor.gen!E families, which have high structural similarity, more effectively than the other five systems, with all indicators higher than 75%. Therefore, in this experiment, ResNet50_TL_SVM was effectively fused with five systems to improve the overall classification ability of the system for all families.

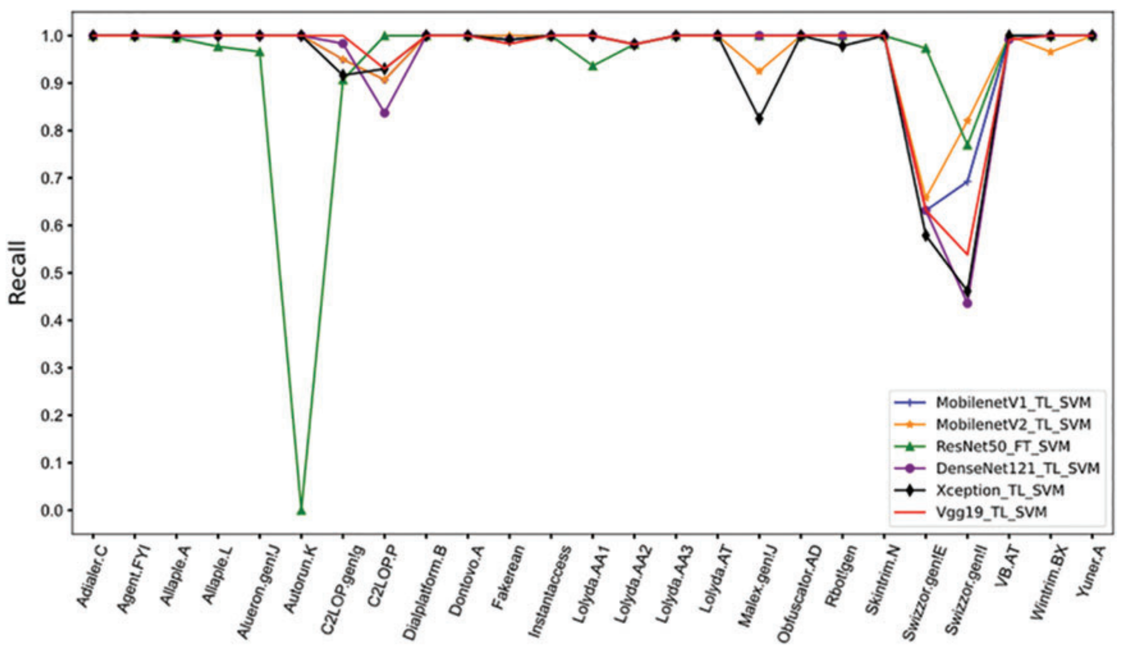


(a)

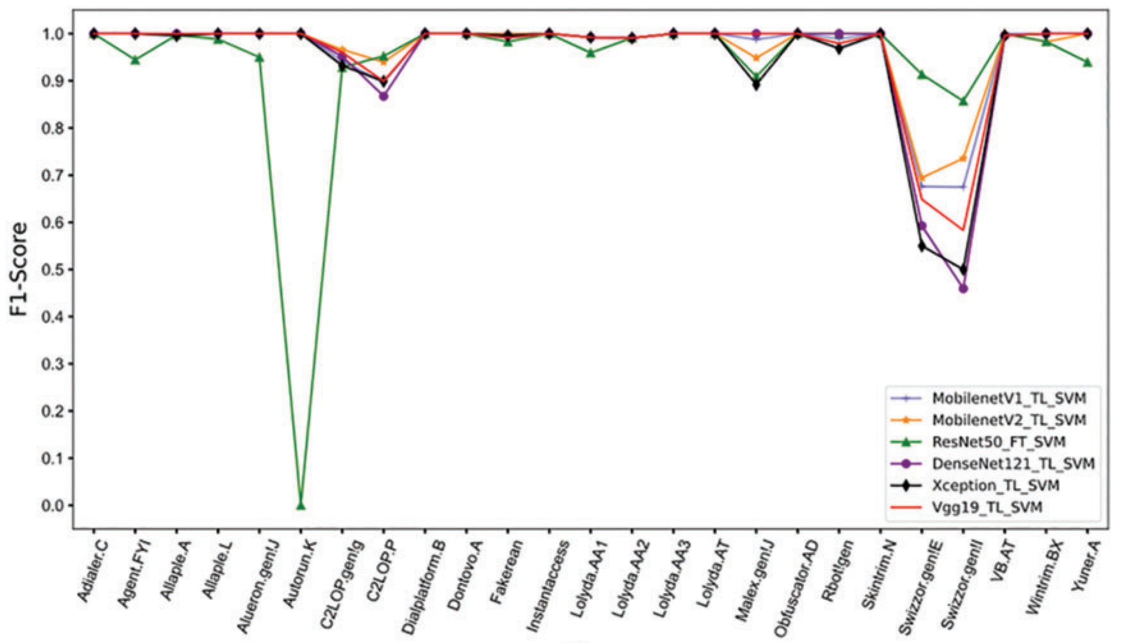


(b)

Figure 3: (Continued)



(c)



(d)

Figure 3: Classification results of MobilenetV1_TL_SVM, MobilenetV2_TL_SVM, ResNet50_FT_SVM, DenseNet121_TL_SVM, Xception_TL_SVM, and Vgg19_TL_SVM. (a) Accuracy (b) precision (c) recall (d) F1-score

Based on ResNet50_TL_SVM, a system fusion experiment was carried out using the maximum probability strategy. Table 3 lists the experimental results of the six fusion systems with the highest accuracy. Among them, R, D, M1, M2, V, and X represent ResNet50_FT_SVM, DenseNet121_TL_SVM, MobilenetV1_TL_SVM, MobilenetV2_TL_SVM, Vgg19_TL_SVM, and Xception_TL_SVM, respectively. R+M1+M2 combines the classification advantages of the respective systems for several malware families and can obtain the best performance of 99.64%, 99.66%, 99.64%, and 99.64%, respectively, in terms of accuracy, precision, recall, and F1-score.

Table 3: Comprehensive performance comparison of different fusion models

Ensemble	Accuracy	Precision	Recall	F1-score
R + D + M1	99.53%	99.56%	99.53%	99.53%
R + D + M2	99.46%	99.50%	99.46%	99.45%
R + M1 + M2	**99.64%**	**99.66%**	**99.64%**	**99.64%**
R + M1 + V	99.50%	99.53%	99.50%	99.49%
R + M2 + V	99.57%	99.59%	99.57%	99.56%
R + V + X	99.43%	99.44%	99.43%	99.42%
R + D + V	99.43%	99.45%	99.43%	99.42%
R + D + X	99.35%	99.38%	99.36%	99.34%
R + M1 + X	99.46%	99.49%	99.46%	99.45%
R + M2 + X	99.46%	99.49%	99.46%	99.45%

Note: R = ResNet50_FT_SVM; D = DenseNet121_TL_SVM; M1 = MobilenetV1_TL_SVM; M2 = MobilenetV2_TL_SVM; V = Vgg19_TL_SVM; X = Xception_TL_SVM.

As shown in Fig. 4, the classification accuracy of VMCTE for both Swizzor.gen!I and Swizzor.gen!E exceeded 84%, especially for families such as Dialplatform.B, C2LOP.gen!g, Allaple.L, Adialer.C, Agent.FYI, and Wintrim.BX, where the classification accuracy reaches 100%. Notably, the fusion of the three systems can improve the classification accuracy of ResNet50_TL_SVM on the Autorun.K family while inheriting the ability to recognize the Swizzor.gen!I and Swizzor.gen!E families.

The performance of the proposed VMCTE system and its three subsystems are contrasted in Table 4. It can be seen that the system built through transfer learning has superior precision, accuracy, recall, and F1-score. M2 is capable of achieving 98.85% classification accuracy. The proposed VMCTE combines the classification advantages of each subsystem; it is also 0.74%, 0.79%, 0.79%, and 0.77% higher than the best subsystem M2 in terms of precision, accuracy, recall, and F1-score, respectively. Compared with the three subsystems, VMCTE significantly improves the classification ability of C2LOP.P, Swizzor.gen!E, Lolyda.AA1, Swizzor.gen!I, Malex.gen!J, and Autorun.K families. Most of the existing systems have low classification accuracy for Swizzor.gen!I and Swizzor.gen!E families. The classification accuracy of VMCTE for these two families can reach 84.62% and 94.74%, respectively. Therefore, ensemble learning can enhance the performance of malware classification systems.

4.4 Comparison with Existing Work

As indicated in Table 5, the proposed VMCTE system is contrasted with a few previously conducted research that used the Malimg dataset. To demonstrate the advantages of the suggested system, the previous work is analyzed and summarized in detail.

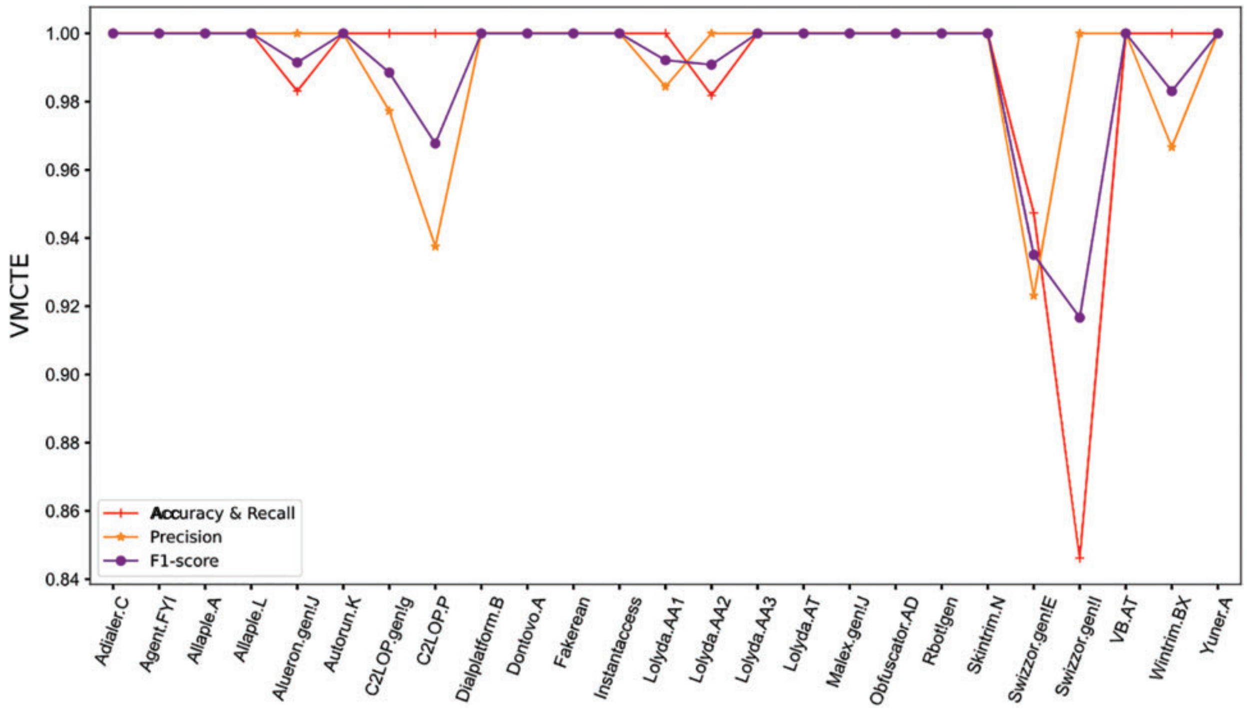


Figure 4: Classification results of VMCTE

Table 4: Comparison of classification performance for VMCTE and its subsystems

Model	Accuracy	Precision	Recall	F1-score
R	97.56%	96.68%	97.56%	97.05%
M1	98.75%	98.73%	98.75%	98.73%
M2	98.85%	98.92%	98.85%	98.87%
VMCTE	99.64%	99.66%	99.64%	99.64%

Table 5: Performance comparison of different classification systems based on the Malimg dataset

Authors	Year	Dataset	Models	Accuracy	Precision	Recall
Nataraj et al. [14]	2011	Malimg	KNN	97.18%	–	–
Kalash et al. [45]	2018	Malimg	M-CNN	98.52%	–	–
Cui et al. [46]	2018	Malimg	IDA+DRBA	94.50%	94.60%	94.50%
Bhodia et al. [47]	2019	Malimg	ResNet34	94.80%	–	–
Cui et al. [48]	2019	Malimg	NSGA-II	97.60%	97.60%	88.40%
Vasan et al. [49]	2020	Malimg	IMCFN	98.82%	98.85%	98.81%
Naeem et al. [50]	2020	Malimg	DCNN	98.79%	98.79%	98.79%
Vasan et al. [32]	2020	Malimg	CNN-SVM	99.50%	99.50%	99.46%

(Continued)

Table 5: Continued

Authors	Year	Dataset	Models	Accuracy	Precision	Recall
Kumar et al. [51]	2021	Malimg	MTIS	98.71%	99.00%	99.00%
Sudhakar et al. [31]	2021	Malimg	MCFT-CNN	99.18%	97.72%	97.76%
Proposed system	–	**Malimg**	**VMCTE**	**99.64%**	**99.66%**	**99.64%**

Nataraj et al. [14] converted malware binary files into grayscale images for the first time and established the Malimg dataset, on which the KNN algorithm was used and had a 97.18% accuracy. Kumar et al. [51] proposed a proactive analysis system, called a malware threat intelligence system (MTIS), capable of collecting and classifying malware. The system used local descriptors, which included gray level co-occurrence matrix, dense scale-invariant feature transform, local binary pattern, and global descriptors to extract global and local texture features, and achieved a classification accuracy of 98.71%.

Vasan et al. [49] introduced a CNN-based malware multi-classification system. The system used data augmentation techniques such as rotation, reflection, flipping, zooming, shifting, scaling, contrast, noise, and color transformation to lessen the effect of unbalanced data on classification performance; it obtained a 98.82% accuracy on the Malimg dataset. Cui et al. [46] presented a malware variant classification system based on visualization methods. The system used the CNN algorithm to classify malware images and the Bat algorithm to balance the data across several malware families; the accuracy, precision, and recall on the Malimg dataset were 94.5%, 94.6%, and 94.5%, respectively. Subsequently, Cui et al. [48] proposed a high-accuracy and low-loss malware classification scheme utilizing image visualization. To address the imbalance in malware family data, the system employed the non-dominated sorting genetic algorithm II (NSGA-II). The outcomes of the experiment proved the algorithm's efficacy; the classification accuracy on the Malimg dataset reached 97.6%. In contrast to the aforementioned work, the family imbalance issue on the Malimg dataset was alleviated in this research by transferring the model's starting weights learned on ImageNet. The classification accuracy, precision, and recall of VMCTE using transfer learning reached 99.64%, 99.66%, and 99.64%, respectively, indicating that the system alleviates the data imbalance problem to some extent.

Sudhakar et al. [31] proposed a system based on fine-tuned CNNs. On the Malimg image dataset, this system, which was based on the transfer learning ResNet50 model, had an accuracy of 99.18%. Kalash et al. [45] presented a Vgg16-based malware classification system based on the image representation of malware binaries. The system had a classification accuracy of 98.52%. Bhodia et al. [47] built a malware classification system by transferring the ResNet34 model and achieved a classification accuracy of 94.8%. Naeem et al. [50] developed a hybrid malware classification system based on color image visualization and deep CNNs that had a classification accuracy of 98.79%. VMCTE significantly improves the classification performance by combining the respective advantages of classification systems based on three neural network models, ResNet50, MobilenetV1, and MobilenetV2, as feature extractors. Vasan et al. [32] proposed a system based on transfer learning that fuses ResNet50, Vgg16, and SVM. The system's accuracy, precision, and recall were 99.50%, 99.50%, and 99.46%, respectively. Compared with the integrated classification system, VMCTE still shows certain advantages in classification performance.

To summarize, VMCTE adopts automatic feature extractors to directly extract effective features from malware images without addressing the concerns of executable file decryption, feature extraction, and fusion. The Malimg dataset has some families with few samples, therefore starting from scratch will have an effect on the system's classification performance. The use of transfer learning mitigates the impact of malware family imbalance on classification performance to some extent. The experimental findings demonstrate that the VMCTE based on ensemble learning has improved classification performance compared with a single model; the accuracy, precision, and recall can reach 99.64%, 99.66%, and 99.64%, respectively.

Fig. 5 displays the confusion matrix for VMCTE using the Malimg dataset. Due to the imbalance of the Malimg dataset, the entries in the confusion matrix are expressed as percentages. The structural composition of the Swizzor.gen!E and Swizzor.gen!I families have a high degree of similarity, so most methods cannot effectively distinguish them. The confusion matrix confirms that the presented approach is effective (94.74% and 84.62%) in differentiating the two malware families. Additionally, Allaple.L and Allaple.A are malware families with multiple layers of encryption. The confusion matrix demonstrates 100% classification accuracy for Allaple.L and Allaple.A, which shows that VMCTE is resilient to code obfuscation in this case.

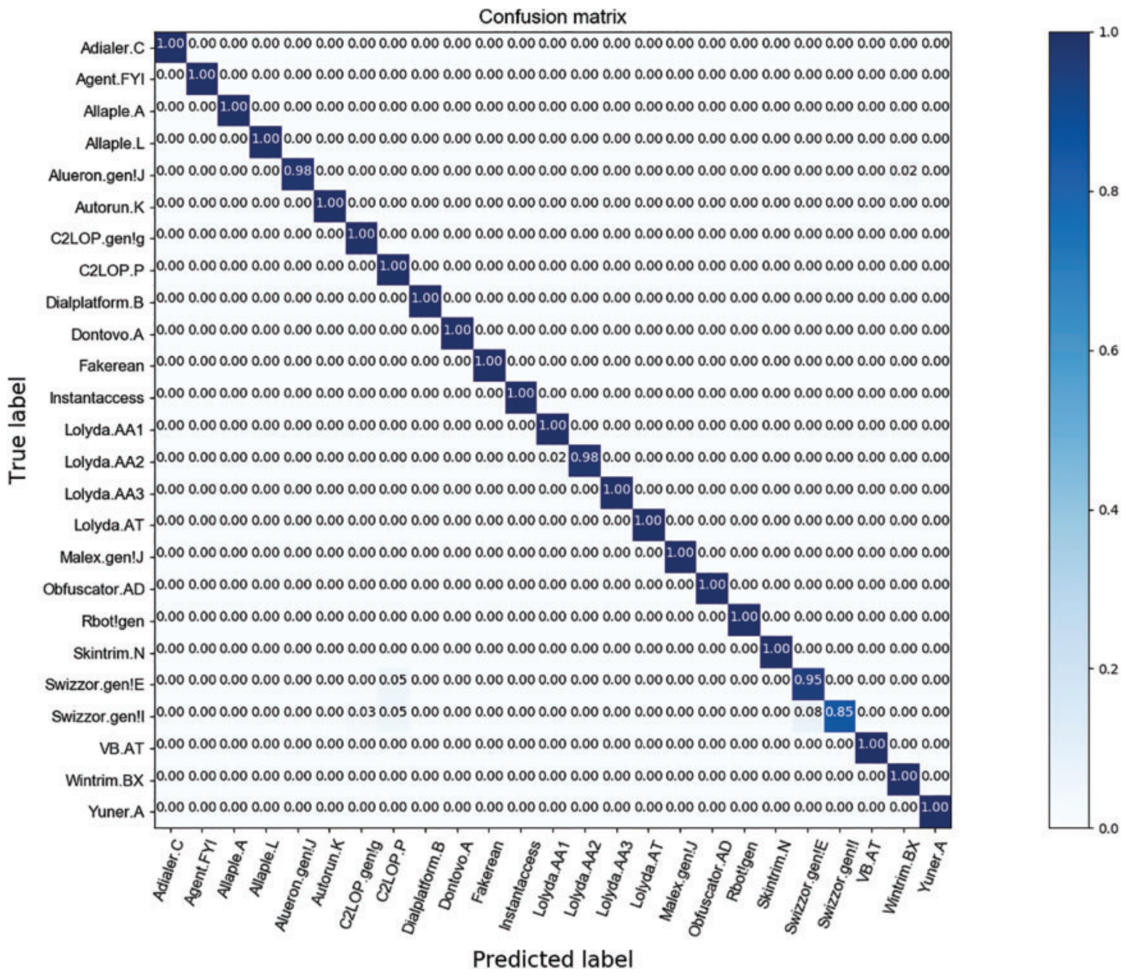


Figure 5: Confusion matrix for VMCTE on the Malimg dataset

5 Conclusion

The rapid growth of malicious software means that the technology for malicious code classification in the existing external security protection system is inadequate. Therefore, this paper presents a visualization-based malware classification system utilizing transfer and ensemble learning. Therefore, this research suggests a transfer and ensemble learning-based malware classification system. The system can accurately classify malicious samples into their appropriate families, which is required to safeguard both business and individual digital properties from infringement. Experiments showed that VMCTE has greater precision, accuracy, recall, and F1-score than existing methods; it can attain 99.66%, 99.64%, 99.64%, and 99.64%, respectively. Therefore, the system can be used to accurately identify and classify malware on a large scale. In the future, we will further develop and optimize the classifier on real malware samples collected in the real world to enhance the generalization ability of the classification system. We will also consider the time complexity of the system to further improve the classification speed for malicious samples.

Funding Statement: This work is supported, in part, by the National Natural Science Foundation of China Grant No. 62102190 and 62272236, in part, by the Natural Science Foundation of Jiangsu Province under Grant No. BK20201136 and BK20191401.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] McAfee, "McAfee Labs Threat Report," (accessed July 2022), 2021. [Online]. Available: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-threats-jun-2021.pdf

[2] Check Point, "CYBER SECURITY REPORT," (accessed July 2022), 2021. [Online]. Available: https://mexicoindustry.com/documentos-tecnicos/sostic/archivos/cyber-security-report-2021.pdf

[3] K. Shaukat, S. Luo and V. Varadharajan, "A novel method for improving the robustness of deep learning-based malware detectors against adversarial attacks," *Engineering Applications of Artificial Intelligence*, vol. 116, no. 4, pp. 105461, 2022.

[4] V. M. Afonso, M. F. de Amorim, A. R. A. Grégio, G. B. Junquera and P. L. de Geus, "Identifying android malware using dynamically obtained features," *Journal of Computer Virology and Hacking Techniques*, vol. 11, no. 1, pp. 9–17, 2015.

[5] S. Yoo, S. Kim, S. Kim and B. B. Kang, "AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification," *Information Sciences*, vol. 546, pp. 420–435, 2021.

[6] T. T. Son, C. Lee, H. Le-Minh, N. Aslam and V. C. Dat, "An enhancement for image-based malware classification using machine learning with low dimension normalized input images," *Journal of Information Security and Applications*, vol. 69, no. 3, pp. 103308, 2022.

[7] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, S. Chen *et al.,* "Performance comparison and current challenges of using machine learning techniques in cybersecurity," *Energies*, vol. 13, no. 10, pp. 2509, 2020.

[8] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed and M. Xu, "A survey on machine learning techniques for cyber security in the last decade," *IEEE Access*, vol. 8, pp. 222310–222354, 2020.

[9] S. Venkatraman, M. Alazab and R. Vinayakumar, "A hybrid deep learning image-based analysis for effective malware detection," *Journal of Information Security and Applications*, vol. 47, no. 11, pp. 377–389, 2019.

[10] S. Ni, Q. Qian and R. Zhang, "Malware identification using visualization images and deep learning," *Computers & Security*, vol. 77, no. 4, pp. 871–885, 2018.

[11] B. N. Narayanan and V. S. P. Davuluru, "Ensemble malware classification system using deep neural networks," *Electronics*, vol. 9, no. 5, pp. 721, 2020.

[12] A. Taha, O. Barukab and S. Malebary, "Fuzzy integral-based multi-classifiers ensemble for android malware classification," *Mathematics*, vol. 9, no. 22, pp. 1–18, 2021.

[13] J. Deng, W. Dong, R. Socher, L. -J. Li, K. Li *et al.,* "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Miami, Florida, USA, pp. 248–255, 2009.

[14] L. Nataraj, S. Karthikeyan, G. Jacob and B. S. Manjunath, "Malware images: Visualization and automatic classification," in *Proc. of the 8th Int. Symp. on Visualization for Cyber Security*, New York, USA, pp. 1–7, 2011.

[15] R. Moskovitch, C. Feher, N. Tzachar, E. Berger, M. Gitelman *et al.,* "Unknown malcode detection using opcode representation," *Intelligence and Security Informatics*, vol. 5376, pp. 204–215, 2008.

[16] M. G. Schultz, E. Eskin, F. Zadok and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Proc. 2001 IEEE Symp. on Security and Privacy*, Oakland, CA, USA, pp. 38–49, 2001.

[17] B. N. Narayanan, O. Djaneye-Boundjou and T. M. Kebede, "Performance analysis of machine learning and pattern recognition algorithms for Malware classification," in *2016 IEEE National Aerospace and Electronics Conf. (NAECON) and Ohio Innovation Summit (OIS)*, Dayton, Ohio, USA, pp. 338–342, 2016.

[18] B. David, E. Filiol and K. Gallienne, "Structural analysis of binary executable headers for malware detection optimization," *Journal of Computer Virology and Hacking Techniques*, vol. 13, no. 2, pp. 87–93, 2017.

[19] A. Pektaş and T. Acarman, "Malware classification based on API calls and behaviour analysis," *IET Information Security*, vol. 12, no. 2, pp. 107–117, 2018.

[20] S. K. Dash, G. Suarez-Tangil and S. Khan, "DroidScribe: Classifying android malware based on runtime behavior," in *2016 IEEE Security and Privacy Workshops (SPW)*, San Jose, California, USA, pp. 252–261, 2016.

[21] H. Cai, N. Meng, B. Ryder and D. Yao, "DroidCat: Effective android malware detection and categorization via app-level profiling," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1455–1470, 2019.

[22] D. Gibert, C. Mateu, J. Planes and R. Vicens, "Using convolutional neural networks for classification of malware represented as images," *Journal of Computer Virology and Hacking Techniques*, vol. 15, no. 1, pp. 15–28, 2019.

[23] S. Choi, S. Jang, Y. Kim and J. Kim, "Malware detection using malware image and deep learning," in *2017 Int. Conf. on Information and Communication Technology Convergence (ICTC)*, Jeju Island, Korea, pp. 1193–1195, 2017.

[24] M. Arefkhani and M. Soryani, "Malware clustering using image processing hashes," in *2015 9th Iranian Conf. on Machine Vision and Image Processing (MVIP)*, Tehran, Iran, pp. 214–218, 2015.

[25] J. Su, D. V. Vasconcellos, S. Prasad, D. Sgandurra, Y. Feng *et al.,* "Lightweight classification of IoT malware based on Image recognition," in *2018 IEEE 42nd Annual Computer Software and Applications Conf. (COMPSAC)*, Tokyo, Japan, pp. 664–669, 2018.

[26] S. Yajamanam, V. R. S. Selvin, F. D. Troia and M. Stamp, "Deep learning versus gist descriptors for image-based malware classification," in *Proc. of the 4th Int. Conf. on Information Systems Security and Privacy(ICISSP)*, Funchal, Madeira, Portugal, pp. 553–561, January 2018.

[27] M. J. Awan, O. A. Masood, M. A. Mohammed, A. Yasin, A. M. Zain *et al.,* "Image-based malware classification using VGG19 network and spatial convolutional attention," *Electronics*, vol. 10, no. 19, pp. 2444, 2021.

[28] Z. Ren, G. Chen and W. Lu, "Malware visualization methods based on deep convolution neural networks," *Multimedia Tools and Applications*, vol. 79, no. 15, pp. 10975–10993, 2020.

[29] J. Hemalatha, S. A. Roseline, S. Geetha, S. Kadry and R. Damaševičius, "An efficient mobilenet-based deep learning model for malware detection," *Entropy*, vol. 23, no. 3, pp. 344–366, 2021.

[30] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, Nevada, USA, pp. 770–778, 2016.

[31] Sudhakar and S. Kumar, "MCFT-CNN: Malware classification with fine-tune convolution neural networks using traditional and transfer learning in Internet of Things," *Future Generation Computer Systems*, vol. 125, no. 5, pp. 334–351, 2021.

[32] D. Vasan, M. Alazab, S. Wassan, B. Safaei and Q. Zheng, "Image-based malware classification using ensemble of CNN architectures (IMCEC)," *Computers & Security*, vol. 92, no. 1, pp. 101748, 2020.

[33] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang *et al.,* "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv: 1704.04861, 2017.

[34] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. -C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *2018 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, Utah, USA, pp. 4510–4520, 2018.

[35] G. Bendiab, S. Shiaeles, A. Alruban and N. Kolokotronis, "IoT malware network traffic classification using visual representation and deep learning," in *2020 6th IEEE Conf. on Network Softwarization (NetSoft)*, Ghent, Belgium, pp. 444–449, 2020.

[36] S. B. Atitallah, M. Driss and I. Almomani, "A novel detection and multi-classification approach for IoT-malware using random forest voting of fine-tuning convolutional neural networks," *Sensors*, vol. 22, no. 11, pp. 4302, 2022.

[37] U. Ahmed, J. C. -W. Lin and G. Srivastava, "Mitigating adversarial evasion attacks of ransomware using ensemble learning," *Computers and Electrical Engineering*, vol. 100, no. 3, pp. 107903, 2022.

[38] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (PCA)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[39] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[40] G. Xiao, J. Li, Y. Chen and K. Li, "MalFCS: An effective malware classification framework with automated feature extraction based on deep convolutional neural networks," *Journal of Parallel and Distributed Computing*, vol. 141, no. 4, pp. 49–58, 2020.

[41] B. Liu, J. Chen, S. Qin, Z. Zhang, Y. Liu *et al.,* "An approach based on the improved SVM algorithm for identifying Malware in network traffic," *Security and Communication Networks*, vol. 14, pp. 5518909, 2021.

[42] H. H. Al-Khshali and M. Ilyas, "Impact of portable executable header features on malware detection accuracy," *Computers Materials & Continua*, vol. 74, no. 1, pp. 153–178, 2022.

[43] A. A. Darem, "A novel framework for windows malware detection using a deep learning approach," *Computers, Materials & Continua*, vol. 72, no. 1, pp. 461–479, 2022.

[44] K. Shaukat, S. Luo, S. Chen and D. Liu, "Cyber threat detection using machine learning techniques: A performance evaluation perspective," in *2020 Int. Conf. on Cyber Warfare and Security (ICCWS)*, Islamabad, Pakistan, pp. 1–6, 2020.

[45] M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang *et al.,* "Malware classification with deep convolutional neural networks," in *2018 9th IFIP Int. Conf. on New Technologies, Mobility and Security (NTMS)*, Paris, France, pp. 1–5, 2018.

[46] Z. Cui, F. Xue, X. Cai, Y. Cao, G. Wang *et al.,* "Detection of malicious code variants based on deep learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3187–3196, 2018.

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

[48] Z. Cui, L. Du, P. Wang, X. Cai and W. Zhang, "Malicious code detection based on CNNs and multi-objective algorithm," *Journal of Parallel and Distributed Computing*, vol. 129, no. 5, pp. 50–58, 2019.

[49] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei *et al.,* "IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture," *Computer Networks*, vol. 171, no. 1, pp. 107138, 2020.

[50] H. Naeem, F. Ullah, M. R. Naeem, S. Khalid, D. Vasan *et al.,* "Malware detection in industrial internet of things based on hybrid image visualization and deep learning model," *Ad Hoc Networks*, vol. 105, no. 1, pp. 102154, 2020.

[51] S. Kumar and B. Janet, "Distinguishing malicious programs based on visualization and hybrid learning algorithms," *Computer Networks*, vol. 201, no. 4, pp. 108595, 2021.