Hindawi
Security and Communication Networks
Volume 2022, Article ID 2206917, 14 pages
https://doi.org/10.1155/2022/2206917

Research Article

MSAAM: A Multiscale Adaptive Attention Module for IoT Malware Detection and Family Classification

Changguang Wang[1,2] **Ziqiu Zhao**[2] **Fangwei Wang**[1,2] **and Qingru Li**[1,2]

[1]*College of Computer & Cyber Security, Hebei Normal University, Shijiazhuang 050024, China*
[2]*Key Laboratory of Network & Information Security of Hebei Province, Hebei Normal University, Shijiazhuang 050024, China*

Correspondence should be addressed to Fangwei Wang; fw_wang@hebtu.edu.cn and Qingru Li; qingruli@hebtu.edu.cn

Received 15 February 2022; Revised 16 May 2022; Accepted 6 June 2022; Published 21 June 2022

Academic Editor: Jinbo Xiong

Nowadays, the attack and defense of malware have presented asymmetric characteristic threats, which has disrupted the pace of IoT research. Traditional detection and family classification methods based on feature extraction, as well as the classical machine learning algorithms, have been afflicted with the problems of high time consuming and unbalanced numbers of malware samples. This paper designs a universal and effective Multiscale Attention Adaptive Module called MSAAM that can combine local and global feature information. It can automatically adjust the arrangement and proportion of channel and spatial submodules by auxiliary classifiers according to actual tasks. The traditional CliqueNet uses a circular feedback structure to improve the DenseNet, optimizes the information flow in a deep network, enhances the utilization of its parameters, and uses a multiscale strategy to prevent a sharp increase of its parameters. As a result, it shows a good effect in the study of image classification. By replacing the attention module in the traditional CliqueNet with the designed MSAAM, we present a new method to process the produced gray-scale images converted from the malware and thus get better results in malware processing. The improved CliqueNet runs on the benchmark datasets of MalImg and Microsoft's BIG 2015 to verify our presented method. After validation on the experimental benchmark datasets, the detection accuracy reaches 99.8%, while the family classification accuracy reaches 99.2% and 98.2% on the above two datasets, respectively. The presented method can solve the problem of unbalanced samples in malware family classification and is also effective against obfuscation attacks.

1. Introduction

Malware refers to a computer code that is written or set up deliberately to pose a threat or potential threat to a network or system. Malware families consist of crypto miners, viruses, ransomware, worms, and spyware, whose purposes are mainly an illegal gathering of information, obstruction of services, or espionage. Nowadays, the popularization of IoT, 5G, and cloud computing increases not only the number and types of malware, but also their threat scope and targets. The McAfee Labs 2021 Threats Report [1] states that Q3 and Q4 of 2020 averaged 588 and 648 malware-related security issues per minute. The two-quarters increased by 169 (40%) and 60 (10%) per minute, respectively, compared to the last quarter. From the third quarter to the fourth quarter, Office malware surged by 199%. Attackers only need to focus on one vulnerability to achieve their goals, but defenders need to patch all vulnerabilities in time to ensure IoT security. This asymmetry is the difficulty faced by malware processing research. And with the increasing application fields affected by malware [2–4], the pressure on IoT security research increases dramatically. Although cybersecurity research continues to grapple with malware threats, malware developers continue to develop new ways to bypass the existing defenses.

Traditional malware processing techniques include static analysis and dynamic analysis. Static analysis means intercepting and analyzing the features such as opcodes, system calls, and Application Programming Interfaces (APIs) from the entire software without executing it. A new malware identification system is proposed according to the frequency of opcodes in portable executable files [5]. The API call

sequence can represent the target's actions and then be used to classify the malware families [6]. Based on API intimacy analysis, their system selects the graph method to analyze the network process and can detect Android malware with high efficiency and precision [7]. The dynamic analysis method enables and controls the running behavior of the target in a closed environment to intercept the behavior characteristics of the target, such as registry modification, system call, file operation, and so on. From five-minute API activities, features are input to a CNN for malware family differentiation [8]. Obfuscated malware is flagged with hooks placed in the system to calculate the time length consumed by the kernel and users [9]. Several feature sets are configured for dynamic malware analysis by extracting features from API calls, and statistical inference is applied to find a set of feature sets that can correctly characterize samples [10]. Nevertheless, both styles of analysis implicit their disadvantages. It is difficult for static analysis with disassembly as the main method to solve the malware that uses various obfuscation techniques for complex reverse engineering. Dynamic analysis has the risk of affecting the system due to the need to actually enable the target. The disassembly technology used in static analysis and the controllable environment required for dynamic analysis of actual activation of target require a large amount of relevant prior professional knowledge. In addition, they require a lot of time.

The spread of machine learning actively drives research in malware detection and family classification [11–14]. But malware defense relying on classic machine learning still has its shortcomings. Their essence lies in feature engineering involving a wide range of fields. Once attackers understand the characteristics of the technology used, they can easily avoid detection.

The deep learning algorithm imitates the learning process of the brain by establishing an artificial neural network with a hierarchical structure and realizing artificial intelligence in a computing system. The outstanding advantages of deep learning are the ability to adapt to the rules of a large amount of data, to extract and filter the input information layer by layer, and to have the ability to learn from mistakes. Due to its powerful feature learning capabilities, many studies have applied deep learning to malware identification [15]. Malware family with imbalanced data is optimized by scheme using convolutional neural network (CNN) and Bat algorithm [16]. Screening the advantages of deep learning architecture and visualization is done to achieve robust and intelligent zero-day malware detection in a big data environment using a hybrid approach of two basic analytics and image processing [17]. In addition, deep learning is widely used for countering malware [18, 19].

In this paper, a new malware processing approach is presented based on Multiscale Attention Adaptive Module (MSAAM) and CliqueNet. The CliqueNet optimizes the information flow in a deep network with a cyclic feedback structure, improves the utilization of parameters in the network, and uses a multiscale strategy to prevent the explosive growth of parameters. MSAAM can combine local and global multiscale feature information in the spatial domain and can automatically adjust the arrangement and proportion of channel and spatial attention submodules by auxiliary classifiers according to actual tasks. This method improves the problem of unbalanced samples of malware family classification while reducing feature engineering and is also effective for obfuscation attacks.

1.1. Contributions of This Study. Multiscale Attention Adaptive Module (MSAAM) which is a new and general module is designed to optimize the expression of CNNs. It contains two parts which are Improved Efficient Channel Attention (IECA) and Multiscale Spatial Attention (MSA), respectively. The proposed MSAAM can automatically adjust the arrangement and proportion of channel and spatial attention submodules by auxiliary classifiers according to actual tasks. Adopting a multiscale strategy combined with Depthwise Convolution and the original spatial attention block of the Convolutional Block Attention Module (CBAM), this study constructs a new attention mechanism called Multiscale Spatial Attention (MSA) that can combine the key information of the local and global attention.

For the first time, we study malware detection and family classification using CliqueNet to expand the information flow and implement feature filtering that in turn enhances the expression of malware processing research.

Without complex feature engineering, our model has good performance on MalImg and BIG 2015 datasets. It deals effectively with the unbalanced samples of malware family classification, and it can also resist malware obfuscation attacks.

The remaining work: Section 2 explores research connected with the model and the method of gray-scale imaging of malware. Section 3 presents our proposed method and introduces MSAAM. Section 4 tests application of this study. Section 5 generalizes our research. Section 6 discusses the limitations of this paper and proposes plans.

2. Related Work

2.1. Malware Visualization. Malware visualization has been an effective technique in malware research in recent years. Nataraj et al. [20] did not focus on the invisible features of malware detection, but a method to detect malware based on visible components is proposed. Transform every byte in the PE file into a pixel with a value in the range of [0, 255] to convert the malware into a visible gray-scale image. They extracted wavelet decomposition-based GIST features from gray-scale images for malware detection. After the conversion, the gray-scale images of malware are visually different for diverse malware families. And as shown in Figure 1, this diversity also lies in benign and malware. And after converting the malware PE file into a gray-scale image, the slight modification made by the malware author to the binary file in the new variant cannot affect the overall structure of the malware gray-scale image. This visualization technique is very effective in detecting malware and its variants. Nowadays, the visualization of malware images has become a routine method. Venkatraman et al. [21] proposed and studied the application of image-based hybrid methods

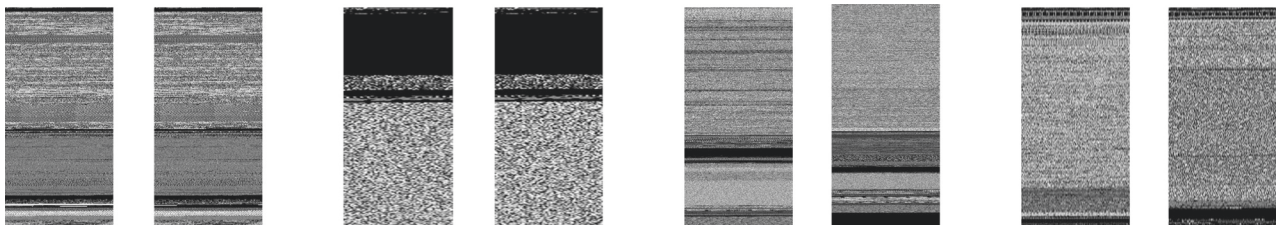

Figure 1: Malware image samples belonging to different families of various malware datasets.

and deep learning architecture to achieve effective malware family classification. Based on image texture similarity, a file-independent deep learning malware family classification method was proposed [22]. Verma et al. [23] demonstrate the idea of using texture analysis to process malware executables. This obfuscated and unbalanced malware discrimination technique combines first-order features extracted on the visualized malware and second-order statistical texture features computed on a gray-scale cooccurrence matrix (GLCM).

However, although these methods solve the problem of code confusion to a certain extent, most of them are based on texture similarity. They require high computational cost feature engineering to extract the complex texture features of malware images (GLCM, Speed-Up Robust Features (SURF), Generalized Search Trees (GIST), Scale-Invariant Feature Transform (SIFT), and Local Binary Pattern (LBP)). When dealing with rapidly iterative malware, they consume a lot of time and are not conducive to processing large datasets. Therefore, reducing the cost of feature engineering and directly using original gray-scale images of malware are the key directions we need to consider. In addition, [24] points out the difficulties faced by texture similarity analysis at this stage. Texture similarity-based analysis is challenging to apply to real-world scenarios with complex malware families, and there are still many practical problems in solving code obfuscation techniques. But [24] considers only the most basic convolutional neural networks in the analysis. We refer to the ideas given in [20] to change target software into gray-scale images. The malware detection and family classification system combined with MSAAM and CliqueNet is used to directly process the original malware gray-scale images to reduce feature engineering and improve efficiency. Through the more robust feature representation capabilities of the new CNNs and the new attention module, the problem that texture similarity analysis is difficult to apply to real-world scenarios with complex malware families is improved.

2.2. Attention Mechanism. Attention mechanisms originating from vision research not only are a focus of recent natural language processing [25–27] but have also been carried forward to research in the image domain [28–30]. It shows strong staying power in the direction of deep learning by imitating the human visual system's attention to important features and eliminating irrelevant information. After SENet [31] demonstrated that attention can strengthen neural network expressive ability by screening significant information areas, the attention mechanism became a conventional means to improve the feature representation of CNNs. The gray-scale images converted from the malware were put into the malware analysis network combining CNN and SENet [32]. By using max and average pooling methods to aggregate features, then creating a network structure combining channel and spatial attention mechanisms, CBAM [33] laid the foundation for the current development of a convolutional neural network attention mechanism. Besides, some other works use adjusted attention blocks to enhance the efficiency of CNNs [34, 35].

After that, the development of the attention module is divided into two aspects: lightweight structure and enhanced feature aggregation. In terms of structural lightweight, ECANet [36] modified SENet based on the idea of lightweight and not reducing the dimensionality, obtaining high efficiency but fewer parameters. In ADCM [37], dropout is integrated into CBAM. We used ECANet and Depthwise Convolution to improve CBAM by the idea of a lightweight and then proposed a new general lightweight convolutional neural network attention module DEAM [38]. Afterward, the DEAM and DenseNet are used to build an effective malware detection and family classification scheme. Some attention modules improved according to the idea of lightweight pay too much attention to the computational efficiency, resulting in the severe lack of feature information used to calculate the attention map, and can not achieve satisfactory results in the vast and complex task model. As a result, their scope of application can only be limited to small task models.

In terms of feature aggregation enhancement, Chen et al., who are dedicated to dynamic scene deblurring research, designed an attention module AAM [39] that can autonomously adjust the position of submodules. Coordattention [40] uses two 1D convolutions with different orientations to aggregate feature information of different dimensions, which enables spatial location information to be embedded into channel attention. The lightweight network using Coordattention can pay attention in a larger region. Multiscale strategies that have performed well in the field of target detection and semantic segmentation are also regarded as an excellent method to improve the feature aggregation effect of the attention mechanism [41–43]. Although the pursuit of feature aggregation will improve the effect of attention mechanisms, sometimes it will

significantly increase the computational cost, which is not suitable for small networks. In particular, the self-attention module developed based on the idea of self-attention mechanism, such as transformer [44], realizes the global reference of each pixel-level prediction, which has great requirements for hardware. Users of small task models often do not have the hardware requirements to use self-attention. At present, some researchers have begun to develop local self-attention in order to reduce the hardware requirements of self-attention mechanism.

Due to the good results shown by DEAM [38], here we use a multiscale strategy to create an effective spatial attention mechanism based on the overall framework of DEAM. The auxiliary classifier is used to automatically adjust the arrangement and proportion of the channel and spatial attention submodules. Multiscale Attention Adaptive Module (MSAAM) which is a new and general module is designed.

2.3. Development of the CNNs. The emergence of convolutional neural networks (CNNs) has brought rapid development to extensive computer vision fields. For strengthening the performance of CNNs, the exploration of network architecture has always been part of the research of CNNs. The original convolutional neural network LeNet [45] consists of five layers. VGGNet [46] has 19 layers and proves that its final performance can be influenced by increasing its depth. GoogLeNet [47] has 22 layers and proves the width is another crucial factor in determining model representation. The three aspects of depth, width, and cardinality have gradually become the most critical factors in determining network architecture. The scale of CNNS has also been increasing with the deepening of exploration. However, the blindly enlarged deep network will make it difficult for the latter layer to obtain gradient information from the previous layer, resulting in the problem of gradient disappearance and parameter redundancy [48]. The emergence of this problem restricts the development of the network architecture in terms of depth, width, and cardinality.

Some experts have explored new paths from the constantly enlarged differences in network architecture, and they have moved towards the exploration of connection modes. ResNet [49] introduces a bypass path so that the top-level network can obtain information from the bottom-level network, strengthens the correlation of the gradients between the network layers, and alleviates the problem of gradient disappearance. It also simplifies network training. After ResNet has shown excellent performance in computer vision orientation, the bypass path is considered to be a key factor to facilitate the work of training these deep networks and solving the problem of gradient disappearance. DenseNet [50] further deepens the idea of ResNet, applying the bypass path to entirety, realizing complex connection, and exploiting the potential of the network through feature reuse. A class-balanced loss is added to the last layer of the DenseNet model for classification for sample imbalanced malware, and this loss is reweighted [51]. But as the dense connection path in DenseNet increases linearly, its parameters will increase sharply. Compared with DenseNet, CliqueNet [52] uses a cyclic feedback structure to further optimize feature flow in deep networks and improve the utilization of parameters in the network. And it also introduces a multiscale feature strategy on the model output to avoid the problem of a sharp increase in parameters in DenseNet.

But as the model continues to expand in the network architecture and connection mode, the hardware burden it brings to researchers also increases dramatically. If researchers want to make a better choice between model performance and requirements, building a general enhancement module in a deep learning model has more room for development than accumulating more nonlinear layers. Therefore, in this paper, new malware processing model is constructed by combining MSAAM and CliqueNet.

3. Proposed Methodology

Our proposed model is composed of CliqueNet and Multiscale Attention Adaptive Module (MSAAM). We obtain CliqueNet suitable for the proposed model based on CliqueNet-S0. Multiscale Spatial Attention (MSA) and Improved Effective Channel Attention (IECA) are two important submodules that make up MSAAM. First, the framework of our proposed malware processing method is described. After that, the architecture of CliqueNet is introduced. Finally, we show our proposed MSAAM and describe MSA.

3.1. Method Overview. Figure 2 depicts the whole framework of our malware detection and family classification approach. The proposed malware processing model uses MSAAM and CliqueNet to automatically extract features at various levels of abstraction from gray-scale images of malware. These features can express the image comprehensively and clearly, and the extracted features are used to train the model. This model can directly process the original malware gray-scale images to reduce feature engineering and improve efficiency.

The following describes the process that the incoming malware samples go through to process them. First, the input malware executable file samples are turned into gray-scale images using the same method as Nataraj et al. [20]. The converted gray-scale images are sent to the trained malware detection model combining MSAAM and CliqueNet to effectively identify benign software and malware. After distinguishing benign software and malware, the malicious samples are sent to the trained malware family classification model combining MSAAM and CliqueNet to effectively identify the family to which each malware belongs.

3.2. Structure of the CliqueNet. To maximize the information flow between layers and solve the problem of a sharp increase in parameters in DenseNet, CliqueNet [52] was created. The most prominent feature of CliqueNet is the use of a cyclic feedback structure with spatial attention effects, so that the model not only has a forward propagation part, but

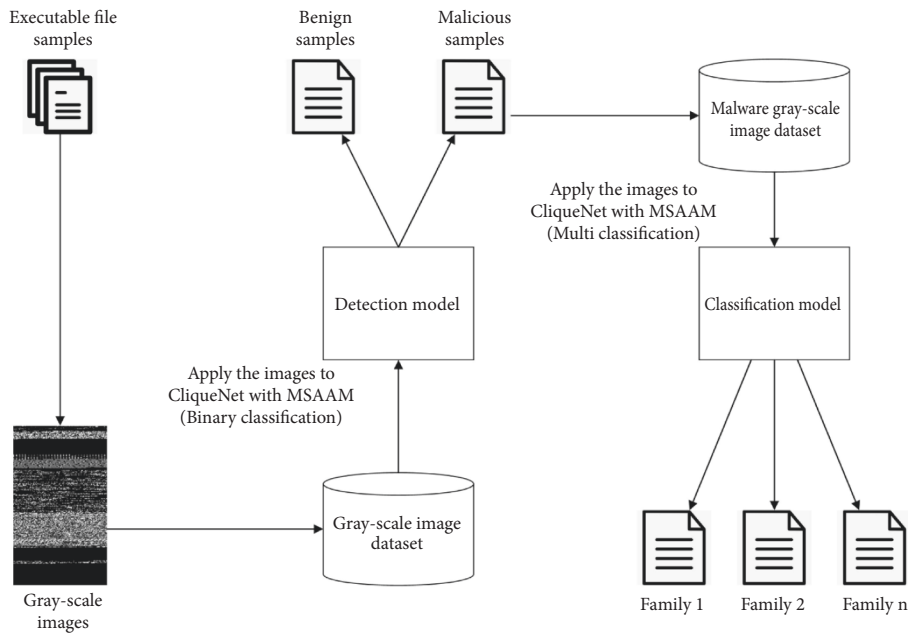

Figure 2: Malware processing method flowchart.

also optimizes the feature map of the previous level based on the output of the next level. Therefore, we can utilize the feature map output in the convolution repeatedly. The modified feature map will consider more important information. Moreover, for avoiding the increased hardware requirements and redundant parameters caused by the explosive growth of parameters in DenseNet, CliqueNet introduces a multiscale feature strategy on the model output. Figure 3 depicts the basic functional module of CliqueNet.

CliqueBlock is the part of CliqueNet that implements the loop feedback structure, as shown in Figure 3. It can be divided into stage-I updated forward and stage-II updated backward. Stage-I is like DenseNet's forward densely connected propagation; the input of each layer will contain the refined feature information of all previous layers. Stage-II realizes the reverse refinement of the model. The input of each convolution operation not only includes the final results of all previously updated layers, but also includes the output feature maps of the subsequent levels. In each step of the stage-II update, the last few feature maps are used to refine relatively earliest feature information, because final feature maps contain relatively higher-level visual information. In this way, the cyclic feedback structure realizes the refinement of the feature maps of each level to achieve the effect of spatial attention. For i-th layer and k-th stage in stage-II, the alternately updated expression is

$$X_i^{(k)} = g\left(\sum_{l<i} W_{li} * X_l^{(k)} + \sum_{m>i} W_{mi} * X_m^{(k-1)}\right), \quad (1)$$

where $k \geq 2$, k denotes the number of stages, and two stages complete a loop. $W * X$ means that the convolution kernel performs convolution operation on the input feature map, and g is nonlinear activation function.

To solve the problem of the sharp increase of parameters in DenseNet, CliqueNet adopts a multiscale feature strategy in the overall structure, as shown in Figure 3. The output of each CliqueBlock consists of two parts. One part is the combination of the output of each layer after reverse refining is called transit_feature, and the other is the combination of input layer and output of each layer after reverse refining is called block_feature. The transit_feature is transmitted to the next CliqueBlock through the transition with the attention mechanism. The block_feature is compressed into a feature vector after global average pooling. Since only the output of stage-II of each CliqueBlock will be used as the input of the next CliqueBlock, the dimension of the feature map of CliqueBlock will not increase super linearly, which has the advantages of parameter amount and calculation amount. And there is no need to use a bottleneck structure like DenseNet to prevent parameter explosion. So the basic structure of CliqueBlock in CliqueNet is BN + ReLU +3 ∗ 3 Conv + Dropout.

We obtain CliqueNet in our method by adapting CliqueNet-S0. Table 1 describes the details of our CliqueNet and CliqueNet-S0 parameters.

3.3. Multiscale Attention Adaptive Module. Here, our purpose is to apply attention block to enhance the effect of malware gray-scale images detection and family classification. For this reason, MSAAM aims to further strengthen the feature extraction capabilities of CliqueNet. The novelty of MSAAM is that it can combine local and global multiscale key information in the spatial domain and can automatically adjust the arrangement and proportion of channel and spatial attention submodules by auxiliary classifiers according to actual tasks to obtain better feature expression ability.

The proposed MSAAM inherits the overall design of CMBA [33], and its construction can be divided into two parts: Improved Efficient Channel Attention (IECA) which

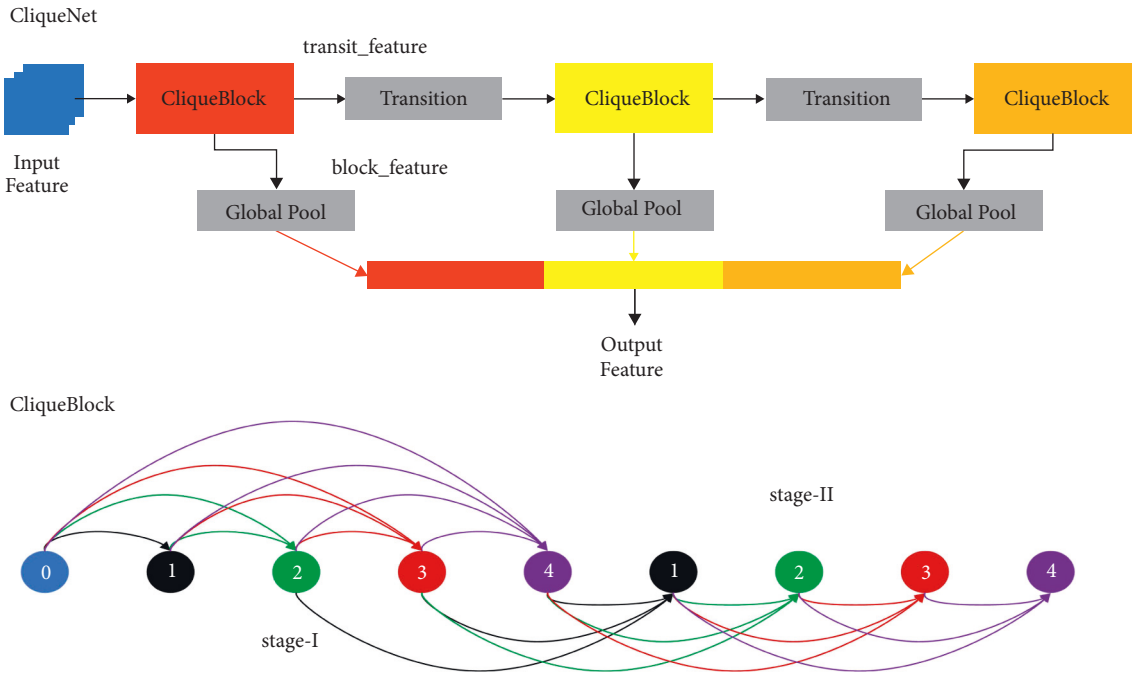

Figure 3: Basic functional modules of CliqueNet [52] and CliqueBlock.

Table 1: The details of our CliqueNet and CliqueNet-S0 parameters. The two numbers in CliqueBlock represent the number of convolutional filters in the block and the number of convolutional layers in the block.

Layers	Our CliqueNet	CliqueNet-S0
Convolution	$7 * 7$ conv, 32, stride 1	$7 * 7$ conv, 64, stride 2
Pooling	$2 * 2$ max pool, stride 2	$3 * 3$ max pool, stride 2
CliqueBlock (1)	$36 * 4$	
Transition	$1 * 1$ conv $2 * 2$ ave pool, stride 2	$36 * 5$
CliqueBlock (2)	$36 * 4$	
Transition	$1 * 1$ conv $2 * 2$ ave pool, stride 2	$64 * 6$
CliqueBlock (3)	$36 * 4$	
Transition	$1 * 1$ conv $2 * 2$ ave pool, stride 2	$100 * 6$
CliqueBlock (4)	$36 * 4$	$80 * 6$

is from [38] and Multiscale Spatial Attention (MSA). Figure 4 shows the overall framework of MSAAM. We use an adaptive method in the comprehensive framework to automatically adjust the arrangement and proportion of the channel and spatial attention submodules. We define the input feature map for MSAAM as $M \in R^{C\times H\times W}$. A one-dimensional channel attention map $M_C \in R^{C\times 1\times 1}$ is obtained by IECA in MSAAM, which can highlight essential feature information in the channel dimension of the feature map. The three-dimensional local space attention map $M_{S1} \in R^{C\times H\times W}$ and the two-dimensional global space attention map $M_{S2} \in R^{1\times H\times W}$ of the feature map are calculated by MSA. The three-dimensional space attention map $M_S \in R^{C\times H\times W}$ is obtained by combining the local and global multiscale key information to highlight more meaningful spatial feature information. Different channel and spatial attention submodule placements in different actual scenes will make the attention mechanism exert different levels of effects.

In this study, we apply learnable matrices to allow the module to adaptively select the placement of attention submodules suitable for the current scene, as well as the proportion of channel and spatial attention submodules that affect the results. The learnable matrices $W_1, W_2 \in R^{C\times H\times W}$ are auxiliary classifiers used to implement the adaptive permutation selection method. We set $W_1 = 0$ means all elements in W_1 are 0, $W_1 = 1$ means all elements in W_1 are 1, and W_2 also has the same setting. The serial placement of channel and space attention submodules is the special case of $W_2 = 0$, and the parallel placement of channel and space attention submodules is the special case of $W_1 = 0$. The calculation process of MSAAM:

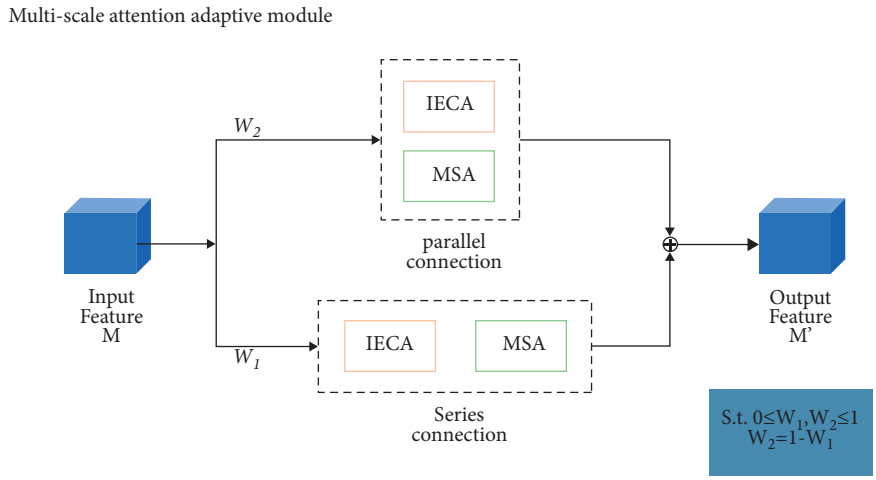

Figure 4: The overall architecture of MSAAM.

$$M' = M_S(M_C(M) \otimes M) \otimes M_C(M) \otimes M \otimes W_1 + M_C(M) \otimes M \otimes M_S(M) \otimes W_2, \quad (2)$$

where $\otimes$ denotes elementwise multiplication, + denotes element summation, and M' denotes the output feature map.

3.3.1. Multiscale Spatial Attention. Differing from the channel attention, the spatial one emphasizes the spatial features. The spatial one computes the likelihood of spatial feature information on the feature map to highlight more meaningful spatial feature information. This study uses a multiscale strategy combined with the spatial attention mechanism of CBAM [33] and Depthwise Convolution to construct a new spatial attention mechanism MSA that can combine the critical information on local attention and global attention, as shown in Figure 5. By collecting feature information on multiple scales, multiscale feature extraction and feature fusion can significantly enhance the information aggregation ability and expand the receptive field of the model. Since spatial information is more fragmented than channel information, more feature information is lost when aggregating. And these two strategies need to consider the application cost when using, so using them on spatial attention can lead to better optimization.

Depthwise Convolution uses convolution and Sigmoid functions on each channel to obtain a spatial attention map without dimensionality reduction. Compared with the spatial attention mechanism in CMBA that uses the max and average pools to compress information, it can obtain better spatial information. However, since Depthwise Convolution is a separate operation for each channel, it focuses on the local spatial information relationships within different regions of the feature map. After visualizing the gray-scale image, the operation of Depthwise Convolution on a single channel of the feature map will divide the image into regions, and the obtained relationship is only the local information relationship in each region. Just using Depthwise Convolution lacks a vision of the global information relationship of the feature map. Figure 6 depicts the visualization of the local and global information relationship of the feature map. Therefore, considering the multiscale strategy, fusion of the key information of local attention extracted by Depthwise Convolution and the key information of global attention extracted by the spatial attention mechanism in CMBA can better pay attention to the spatial information of the features.

In MSA, Depthwise Convolution is applied to the input feature map $M \in R^{C\times H\times W}$, and the Sigmoid function is used for the output feature descriptor $F_2 \in R^{C\times H\times W}$ to obtain a three-dimensional local spatial attention map $M_{S1} \in R^{C\times H\times W}$. The max pool and average pool compression are used for the input feature map $M \in R^{C\times H\times W}$. These two spatial feature descriptors (F^S_{avg}andF^S_{max}) indicate the average pool space information and the max pool space information, respectively. A $7 * 7$ Conv and the Sigmoid function are used for the feature descriptor $F_3 \in R^{2\times H\times W}$ obtained after the two spatial context descriptors are spliced to bring a 2D global spatial attention map $M_{S2} \in R^{1\times H\times W}$. Finally, this attention adds the three-dimensional local space attention map $M_{S1} \in R^{C\times H\times W}$ and the two-dimensional global space attention map $M_{S2} \in R^{1\times H\times W}$ element by element to get the three-dimensional space attention map $M_S \in R^{C\times H\times W}$. $W_3, W_4 \in R^{C\times H\times W}$ are learnable matrices used to determine the proportion of local key information and global key information. The calculation formula of MSA is as follows:

$$M_S(M) = \sigma(\text{DepthwiseConv2D}(M)) \otimes W_3 + \sigma\left(\text{conv}^{7\times 7}\left(\left[F^S_{\text{avg}}; F^S_{\text{max}}\right]\right)\right) \otimes W_4 \quad (3)$$
$$M_S(M) = M_{S1} \otimes W_3 + M_{S2} \otimes W_4,$$

where DepthwiseConv2D denotes Depthwise Convolution, and [;] denotes tensor splicing.

4. Experiments and Analysis

4.1. Experimental Setting and Datasets. We use MalImg [20] and Microsoft's BIG 2015 to assess the proposed approach. Tables 2 and 3 give a detailed introduction of the two benchmark datasets. The MalImg dataset, consisting of 25

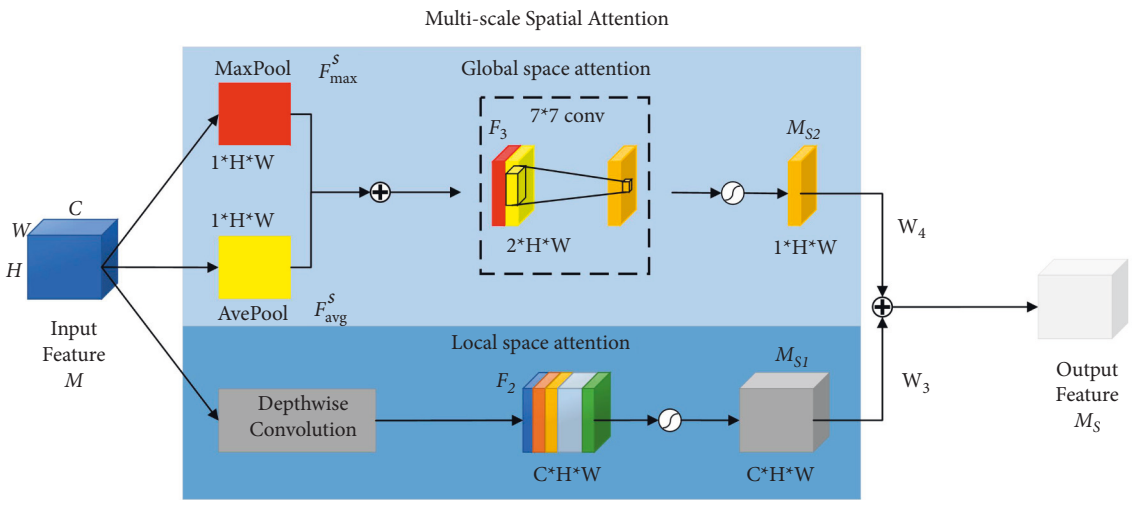

Figure 5: The detailed process of Multiscale Spatial Attention.

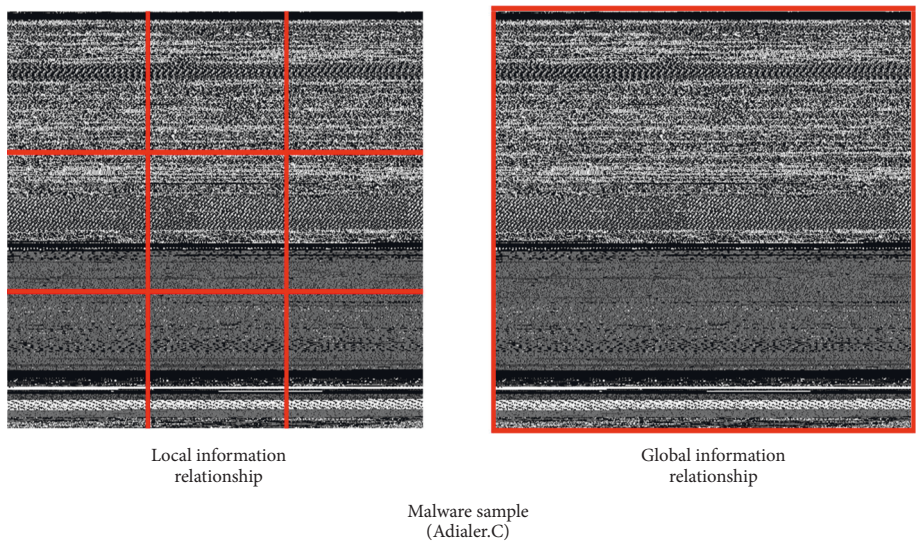

Figure 6: Convolutional receptive field gray-scale images visualization.

Table 2: Sample distribution of MalImg dataset.

No.	Family	Number of samples	No.	Family	Number of samples
1	Adialer.C	122	14	Lolyda.AA2	184
2	Agent.FYI	116	15	Lolyda.AA3	123
3	Allaple.A	2949	16	Lolyda.AT	159
4	Allaple.L	1591	17	Malex.gen!J	136
5	Alueron.gen!J	198	18	Obfuscator.AD	142
6	Autorun.K	106	19	Rbot!gen	158
7	C2LOP.gen!g	200	20	Skintrim.N	80
8	C2LOP.P	146	21	Swizzor.gen!E	128
9	Dialplatform.B	177	22	Swizzor.gen!I	132
10	Dontovo.A	162	23	VB.AT	408
11	Fakerean	381	24	Wintrim.BX	97
12	Instantaccess	431	25	Yuner.A	800
13	Lolyda.AA1	213			9339

Table 3: Sample distribution of BIG 2015 dataset.

No.	Family	Number of samples	No.	Family	Number of samples
1	Ramnit	1541	6	Tracur	751
2	Lollipop	2478	7	Kelihos_ver1	398
3	Kelihos_ver3	2942	8	Obfuscator.ACY	1228
4	Vundo	475	9	Gatak	1013
5	Simda	42			10868

malware families and 9339 malware samples in total, is a massive and unbalanced malware dataset. The BIG 2015 dataset contains 21741 malware samples, consisting of 9 malware families. The training set contains 10868 samples and the test set has 10873 ones. The test set of the BIG 2015 dataset, however, did not give the corresponding label. Thus this paper only uses the training set part. The bytes files containing the original hexadecimal code of the files in the dataset are used to generate gray-scale images of the malware.

To classify malware families, we directly use the MalImg and BIG 2015 as the evaluation benchmark. For the malware detection part, we randomly selected a total of 1087 malware samples out of the 34 malware families contained in the MalImg and BIG 2015 datasets. Using the filtered malicious samples and an equal number of benign samples to create a new malware detection dataset, it contains rich malware families to ensure the experimental scalability.

To effectively evaluate the performance of our model, the dataset splits into three parts where the data is used for training, validation, and test at a ratio of 6 : 2 : 2. Moreover, to abate errors, we repeat every experiment 5 times. Our experiment uses the Adam optimizer and categorical_crossentropy, the batch size is 16, and the learning rate is 0.0005. The environment equipment is as follows: Windows 10, Intel(R) Core (TM) i7-1165G7 CPU @ 2.80 GHz 2.80 GHz and NVIDIA GeForce GTX 1060. The proposed malware processing method is built on the Python framework and the tensorflow2.3 framework.

Malware loses different degrees of information when it converts to gray-scale images of different sizes. However, an immense gray-scale image size will have high requirements on the physical equipment and will bring a huge burden to the training of the model. In [24], the effects of image sizes on malware family classification based on texture feature analysis are experimentally verified. To balance performance and cost, and to guarantee the validity of the experiment, we adjust the gray-scale image size of the model input to $256 * 256$, which is recommended in [24].

The $N * N$ confusion matrix is used to calculate four types of indicators in order to examine our proposed model for detecting malware. These indicators are as follows: *accuracy*, *precision*, *recall*, and *F1 score*. The accuracy rate used alone can only show the expressive ability at the macro level and cannot sensitively reflect the prediction level of the class with a small number of samples. *Precision* reflects how many of all the samples marked as positive are expressed correctly. Recall reflects the ability of the model to identify the target. *F1 score* determines the accuracy and robustness of the model. For the two-class detection task, we directly obtain these values. For the multiclass family classification task, we will enumerate three metrics for each family in detail. Finally, the indicators of each family are combined to calculate macro-precision, macro-recall, and macro-F1.

Indicators other than accuracy are calculated as follows:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ \text{F1} &= \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}, \end{aligned} \tag{4}$$

where TP is the target sample expressed as positive, FP is the target sample expressed as negative, and FN is the nontarget sample expressed as negative.

4.2. Performance of Detecting Malware. Table 4 shows the results in a $2 * 2$ confusion matrix. Our accuracy, precision, recall, and F1 score on malware dataset are 99.8%, 99.8%, 99.8%, and 99.8%, respectively. For the test set with a total number of samples of 421, only one benign sample was classified incorrectly. After validation on the experimental malware detection dataset, our model can effectively distinguish between benign software and malicious software. Table 5 demonstrates that our model outperforms existing methods in detection ability.

4.3. Performance of Classifying Family

4.3.1. MalImg Dataset. The $25 * 25$ confusion matrix on Figures 7 and 8 illustrates our classifying results of the proposed method and CliqueNet on MalImg dataset. Our accuracy, precision, recall, and F1 score on the MalImg are 99.2%, 98.0%, 97.9%, and 97.9%, respectively. Without MSAAM on MalImg dataset, the sequential metrics are 98.6%, 96.7%, 96.3%, and 96.5%, respectively. Table 6 illustrates the comparison results on the MalImg dataset. After validation on the experimental MalImg dataset, our model is equivalent to [23, 51] in precision, recall, and F1 score, but it surpasses these two tasks in terms of accuracy. Compared with other work, our model has achieved a comprehensive surpass in all four evaluation indicators. Table 7 shows the influence of our model on families with smaller samples in two datasets. It achieved F1 scores of 100%, 100%, and 87.5% on Skintrim.N, Wintrim.BX, and Simda families, respectively. These show that our model can effectively complete the classification of malware families

Table 4: Our classification performance on malware dataset.

	Malware	Benign	Precision	Recall	F1 score
Malware	204	0	0.995	1	0.998
Benign	1	216	1	0.995	0.998
Macro			0.998	0.998	0.998

Table 5: The comparison of the binary classification effect between our model and others.

Models	Accuracy (%)
RF [5]	97.0
García and DeCastro-García [10]	99.4
TELM [12]	99.7
SVM [13]	98.5
Zhang et al. [18]	95.1
CRNN [19]	96.2
DenseNet + DEAM [38]	99.3
Hemalatha et al. [51]	97.6
Proposed method	99.8

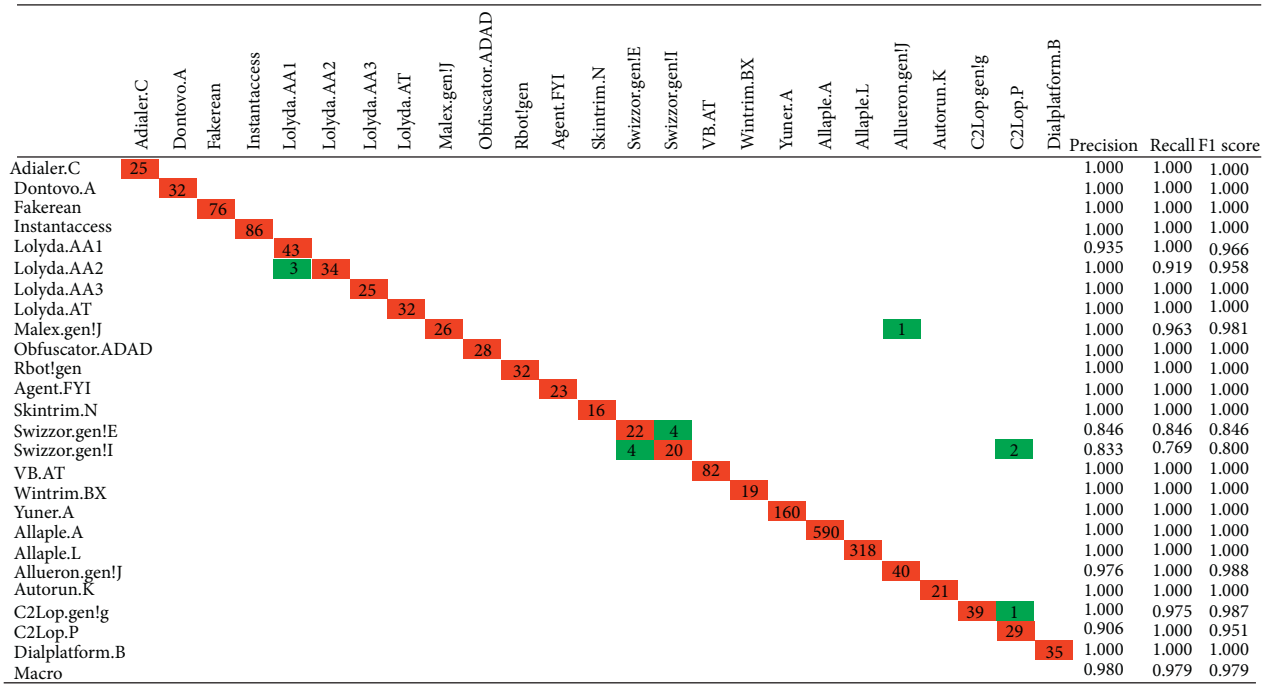

Figure 7: Multiclassification performance of proposed method on MalImg.

and is robust to the problem of imbalance in the classification of malware families. Compared with CliqueNet without MSAAM, it proves that the proposed MSAAM can strengthen the attention to the characteristics of malware. Compared with our previous work [38] and CliqueNet + DEAM, it proves that MSAAM improves the performance of the attention module based on the DEAM.

The comparison between Figures 7 and 8 illustrates that the classification difficulties of the MalImg dataset are concentrated on two families. They are Swizzor.gen!E and Swizzor.gen!I, respectively. The addition of MSAAM has greatly improved the classification effect on these two key families, while the classification effect on other families has also been slightly improved. MSAAM raises the F1 score of Swizzor.gen!E from 71.7% to 84.6% and raises the F1 score of Swizzor.gen!I from 69.2% to 80.0%. This reflects that our proposed MSAAM effectively improves the feature expression ability of CNN. And for the entire network architecture of deep learning, the addition of MSAAM will hardly bring about an increase in computational consumption. There are many samples processed by obfuscation techniques in the MalImg data set. Our model achieves 100% classification on 17 out of 25 families. The families using the packaging technology UPX which makes them

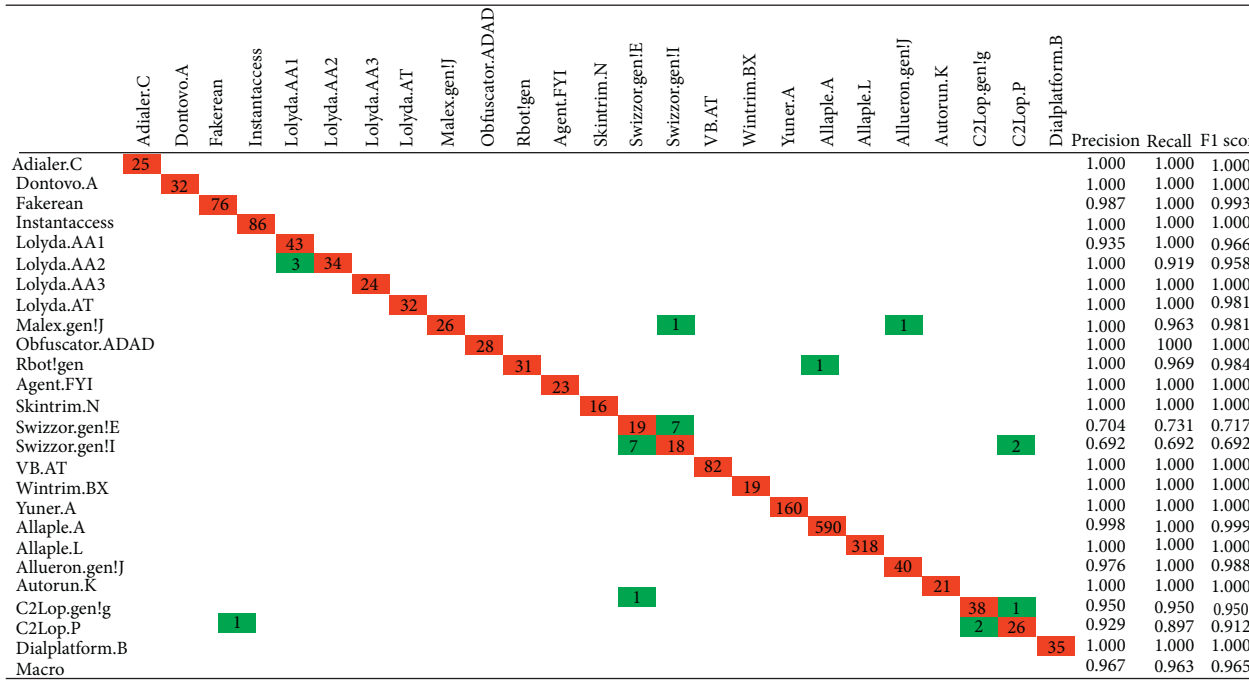

Figure 8: Multiclassification performance of CliqueNet on the MalImg dataset.

Table 6: Comparative analysis of proposed method with others on MalImg.

Models	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
Cui et al. [15]	94.5	94.6	94.5	94.5
Vinayakumar et al. [17]	96.3	96.3	96.2	96.2
Venkatraman et al. [21]	96.3	91.8	91.5	91.6
Gibert et al. [22]	98.5	95.8	96.6	95.8
Verma et al. [23]	98.5	98.0	98.0	98.0
Densenet + DEAM [38]	98.5	96.9	96.6	96.7
Hemalatha et al. [51]	98.2	97.8	97.9	97.9
CliqueNet	98.6	96.7	96.3	96.5
CliqueNet + DEAM	98.8	97.1	96.8	97.0
Proposed method	99.2	98.0	97.9	97.9

Table 7: Influence of our model on families with smaller samples in two datasets.

	Family	Number of samples	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
MalImg	Skintrim.N	80	100	100	100	100
	Wintrim.BX	97	100	100	100	100
Big 2015	Simda	42	87.5	87.5	87.5	87.5

indistinguishable from each other in similar structures in this dataset are Yuner.A, Rbot!gen, Malex.gen!J, VB.AT, and Autorun.K. But our method reaches a 100% classification accuracy in Yuner.A, VB.AT, Autorun.K, and Rbot!gens, and the F1 score on Malex.gen!J is 98.1%. Allaple uses random keys in the code part to encrypt in several layers, but our model makes a perfect distinction between Allaple.A and Allaple.L. Lolyda.AA1 and Lolyda.AA3 belong to the same family variants and are also made a perfect distinction. All these prove that our model is effective for the classification of obfuscated malware.

4.3.2. BIG 2015 Dataset. The $9 * 9$ confusion matrix on Figures 9 and 10 illustrates the classification results of the proposed method and CliqueNet on BIG 2015 dataset. The accuracy, precision, recall, and F1 score of our model on BIG 2015 dataset are 98.2%, 96.6%, 96.3%, and 96.4%, respectively. The accuracy, precision, recall and F1 score of CliqueNet without MSAAM on BIG 2015 dataset are 97.6%, 96.8%, 94.6%, and 95.5%, respectively. After the addition of MSAAM, the evaluation indicators other than precision have been improved. The comparison between Figures 9 and 10 illustrates that MSAAM improves the classification

	Ramnit	Lollipop	Kelihos_ver3	Vundo	Simda	Tracur	kelihos_ver1	Obfuscator:ACY	Gatak	Precision	Recall	F1 score
Ramnit	303	1	1	0	0	0	1	3	0	0.968	0.981	0.974
Lollipop	0	495	0	0	0	0	0	1	0	0.990	0.998	0.994
Kelihos_ver3	0	0	588	0	0	0	0	0	0	0.995	1000	0.997
Vundo	0	1	0	91	1	1	0	0	1	0.968	0.968	0.963
Simda	1	0	0	0	7	0	0	0	0	0.875	0.875	0.875
Tracur	2	1	0	1	0	145	1	0	0	0.967	0.967	0.967
kelihos_ver1	0	0	1	0	0	0	77	1	0	0.975	0.975	0.975
Obfuscator:ACY	2	2	1	2	0	4	0	226	3	0.974	0.922	0.948
Gatak	0	0	0	0	0	0	0	1	201	0.980	0.995	0.988
Macro										0.966	0.963	0.964

Figure 9: Multiclassification performance of proposed method on BIG 2015.

	Ramnit	Lollipop	Kelihos_ver3	Vundo	Simda	Tracur	kelihos_ver1	Obfuscator:ACY	Gatak	Precision	Recall	F1 score
Ramnit	299	1	0	0	0	6	1	2	0	0.955	0.968	0.961
Lollipop	2	492	0	0	0	0	0	0	2	0.984	0.992	0.988
Kelihos_ver3	0	0	588	0	0	0	0	0	0	0.998	1000	0.999
Vundo	0	1	0	91	1	1	0	0	1	0.919	0.958	0.938
Simda	0	0	0	1	6	0	0	1	0	1000	0.750	0.857
Tracur	2	1	0	1	0	144	1	0	1	0.941	0.960	0.950
kelihos_ver1	0	0	0	0	0	0	79	0	0	0.952	1000	0.975
Obfuscator:ACY	10	2	1	5	0	2	1	223	1	0.987	0.910	0.947
Gatak	0	3	0	1	0	0	0	0	198	0.975	0.980	0.978
Macro										0.968	0.946	0.955

Figure 10: Multiclassification performance of CliqueNet on BIG 2015.

Table 8: Comparative analysis of proposed method with others on BIG 2015.

Models	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
ACNN [16]	96.0	95.4	88.3	89.7
Gibert et al. [22]	97.5			94.0
DenseNet + DEAM [38]	97.3	95.3	95.4	95.4
CliqueNet	97.6	96.8	94.6	95.5
Proposed method	98.2	96.6	96.3	96.4

performance on multiple families. Table 8 shows a comparative analysis of the proposed method and others on the BIG 2015. Our approach outperforms other works in malware family classification.

5. Conclusion

An efficient malware processing method is presented by using the newly designed universal and effective Multiscale Attention Adaptive Module (MSAAM) and CliqueNet. MSAAM can combine local and global multiscale feature information in the spatial domain and can automatically adjust the arrangement and proportion of channel and spatial attention submodules by auxiliary classifiers according to actual tasks. This method can directly process the gray-scale images of malware, reducing the feature engineering and improving the problem of unbalanced samples of malware family classification. It is also reliable and effective for obfuscation attacks. After validation on the experimental benchmark datasets, the proposed MSAAM attention module combining adaptive and multiscale strategies achieves the optimization of DEAM attention module in performance. The proposed method can effectively handle malware security issues.

6. Discussion

The proposed method does not perfectly solve the problem that texture similarity-based analysis is difficult to apply to real-world scenarios with complex malware families. In order to deepen the research on the effectiveness and universality of detecting malware, we will work on the deepening of feature engineering and the development of better attention modules. We also plan to optimize the information flow in this network by adding adjustments between different layers of the CliqueNet. The confrontation with code obfuscation technology is also a direction that needs to be studied in the future.

Data Availability

The BIG 2015 dataset can be obtained from https://www.kaggle.com/competitions/malware-classification/overview. The MalImg dataset can be obtained from https://www.researchgate.net/figure/Malimg-dataset_tbl2_323130489.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This research was supported by NSFC under Grant no. 61572170, Natural Science Foundation of Hebei Province of China under Grant no. F2021205004, Science Foundation of Department of Human Resources and Social Security of Hebei Province under Grant no. 201901028 and no. ZD2021062, Science Foundation of Returned Overseas of Hebei Province of China under Grant no. C2020342, and Science and Technology Foundation Project of Hebei Normal University under Grant no. L2021K06.

References

[1] Trellix, "Trellix Threat Labs Research Report," 2021, https://www.mcafee.com/enterprise/en-us/lp/threats-reports/apr-2021.html.

[2] S. Khan and A. Akhunzada, "A hybrid DL-driven intelligent SDN-enabled malware detection framework for Internet of Medical Things (IoMT)," *Computer Communications*, vol. 170, pp. 209–216, 2021.

[3] M. Dib, S. Torabi, E. Bou-Harb, and C. Assi, "A multi-dimensional deep learning framework for iot malware classification and family attribution," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1165–1177, 2021.

[4] Q. Li, J. Mi, W. Li, J. Wang, and M. Cheng, "CNN-based malware variants detection method for internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 16946–16962, 2021.

[5] Y. Abhijit and M. Singh, "Malware detection based on opcode frequency," in *Proceedings of the 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pp. 646–649, IEEE, Ramanathapuram, India, May 2017.

[6] S. Gupta, H. Sharma, and K. Sarvjeet, "Malware characterization using Windows API call sequences," in *Proceedings of the International Conference on Security, Privacy, and Applied Cryptography Engineering*, pp. 363–378, Delhi, India, December 2016.

[7] D. Zou, Y. Wu, S. Yang et al., "IntDroid," *ACM Transactions on Software Engineering and Methodology*, vol. 30, no. 3, pp. 1–32, 2021.

[8] T. Shun, Y. Yukiko, S. Hajime, and I. Tomonori, "Malware detection with deep neural network using process behavior," in *Proceedings of the 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, pp. 577–582, IEEE, Atlanta, GA, USA, June 2016.

[9] D. Javaheri and M. Hosseinzadeh, "A framework for recognition and confronting of obfuscated malwares based on memory dumping and filter drivers," *Wireless Personal Communications*, vol. 98, no. 1, pp. 119–137, 2018.

[10] D. E. García and N. DeCastro-García, "Optimal feature configuration for dynamic malware detection," *Computers & Security*, vol. 105, Article ID 102250, 2021.

[11] A. K. M. A. and J. C. D., "Automated multi-level malware detection system based on reconstructed semantic view of executables using machine learning techniques at VMM," *Future Generation Computer Systems*, vol. 79, pp. 431–446, 2018.

[12] A. Namavar Jahromi, S. Hashemi, A. Dehghantanha et al., "An improved two-hidden-layer extreme learning machine for malware hunting," *Computers & Security*, vol. 89, Article ID 101655, 2020.

[13] R. Sihwail, K. Omar, K. Zainol Ariffin, and S. Al Afghani, "Malware detection approach based on artifacts in memory image and dynamic analysis," *Applied Sciences*, vol. 9, no. 18, p. 3680, 2019.

[14] M. Ali, S. Shiaeles, G. Bendiab, and B. Ghita, "MALGRA: machine learning and N-gram malware feature extraction and detection system," *Electronics*, vol. 9, no. 11, p. 1777, 2020.

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

[16] X. Ma, S. Guo, H. Li et al., "How to make attention mechanisms more practical in malware classification," *IEEE Access*, vol. 7, pp. 155270–155280, 2019.

[17] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, and S. Venkatraman, "Robust intelligent malware detection using deep learning," *IEEE Access*, vol. 7, pp. 46717–46738, 2019.

[18] J. Zhang, Z. Qin, H. Yin, L. Ou, and K. Zhang, "A feature-hybrid malware variants detection using CNN based opcode embedding and BPNN based API embedding," *Computers & Security*, vol. 84, pp. 376–392, 2019.

[19] S. Jeon and J. Moon, "Malware-detection method with a convolutional recurrent neural network using opcode sequences," *Information Sciences*, vol. 535, pp. 1–15, 2020.

[20] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: visualization and automatic classification," in *Proceedings of the 8th International Symposium on Visualization for Cyber Security—VizSec'11*, pp. 1–7, Pittsburgh, PA, USA, July 2011.

[21] 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, pp. 377–389, 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] V. Verma, S. K. Muttoo, and V. B. Singh, "Multiclass malware classification via first- and second-order texture statistics," *Computers & Security*, vol. 97, Article ID 101895, 2020.

[24] B. Tamy and F. B. Marcus, "A.L(a)yingin(Test)Bed," in *Proceedings of the International Conference on Information Security*, pp. 381–401, New York, NY, USA, September 2019.

[25] L. Yang, P. Wang, H. Li, Z. Li, and Y. Zhang, "A holistic representation guided attention network for scene text recognition," *Neurocomputing*, vol. 414, pp. 67–75, 2020.

[26] M. Liu, L. Li, H. Hu, W. Guan, and J. Tian, "Image caption generation with dual attention mechanism," *Information Processing & Management*, vol. 57, no. 2, Article ID 102178, 2020.

[27] Y. Cheng and Y. Morimoto, "Triple-stage attention-based multiple parallel connection hybrid neural network model for conditional time series forecasting," *IEEE Access*, vol. 9, pp. 29165–29179, 2021.

[28] Y. Yang, C. Xu, F. Dong, and X. Wang, "A new multi-scale convolutional model based on multiple attention for image classification," *Applied Sciences*, vol. 10, no. 1, p. 101, 2019.

[29] Z. Huang, Y. Zhao, X. Li et al., "Application of innovative image processing methods and AdaBound-SE-DenseNet to optimize the diagnosis performance of meningiomas and gliomas," *Biomedical Signal Processing and Control*, vol. 59, Article ID 101926, 2020.

[30] J. Shi, K. Wu, C. Yang, and N. Deng, "A method of steel bar image segmentation based on multi-attention U-net," *IEEE Access*, vol. 9, pp. 13304–13313, 2021.

[31] J. Hu, Li Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7132–7141, IEEE, Salt Lake, UT, USA, June 2018.

[32] H. Yakura, S. Shinozaki, R. Nishimura, Y. Oyama, and J. Sakuma, "Neural malware analysis with attention mechanism," *Computers & Security*, vol. 87, Article ID 101592, 2019.

[33] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: convolutional block attention module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19, Munich, Germany, September 2018.

[34] G. Huang, Y. Gong, Q. Xu, K. Wattanachote, K. Zeng, and X. Luo, "A convolutional attention residual network for stereo matching," *IEEE Access*, vol. 8, pp. 50828–50842, 2020.

[35] M. Zhu, L. Jiao, F. Liu, S. Yang, and J. Wang, "Residual spectral-spatial attention network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 1, pp. 449–462, 2021.

[36] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Efficient Channel attention for deep convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11531–11539, Francisco, CA, USA, June 2020.

[37] Z. Liu, J. Du, M. Wang, and S. S. Ge, "ADCM: attention dropout convolutional module," *Neurocomputing*, vol. 394, pp. 95–104, 2020.

[38] C. Wang, Z. Zhao, F. Wang, and Q. Li, "A novel malware detection and family classification scheme for IoT based on DEAM and DenseNet," *Security and Communication Networks*, vol. 2021, Article ID 6658842, 16 pages, 2021.

[39] L. Chen, Q. Sun, and F. Wang, "Attention-adaptive and deformable convolutional modules for dynamic scene deblurring," *Information Sciences*, vol. 546, pp. 368–377, 2021.

[40] Q. Hou, D. Zhou, and J. Fengi, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13713–13722, IEEE, Nashville, TN, USA, June 2021.

[41] Y. Jiang, H. Yao, C. Wu, and W. Liu, "A multi-scale residual attention network for retinal vessel segmentation," *Symmetry*, vol. 13, no. 1, p. 24, 2020.

[42] Y. Zhu, R. Yang, Y. He et al., "A lightweight multiscale Attention semantic segmentation algorithm for detecting laser welding defects on safety vent of power battery," *IEEE Access*, vol. 9, pp. 39245–39254, 2021.

[43] A. Sinha and J. Dolz, "Multi-scale self-guided attention for medical image segmentation," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 1, pp. 121–130, 2021.

[44] V. Ashish, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[45] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[46] S. Karen and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations (ICLR)*, IEEE, Kuala Lumpur, Malaysia, November 2014.

[47] S. Christian, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, IEEE, Boston, MA, USA, June 2015.

[48] H. Gao, "Deep networks with stochastic depth," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 646–661, Amsterdam, The Netherlands, October 2016.

[49] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, IEEE, Las Vegas, NV, USA, June 2016.

[50] H. Gao, Z. Liu, and L. Van Der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700–4708, IEEE, Honolulu, HI, USA, July 2017.

[51] J. Hemalatha, S. Roseline, S. Geetha, S. Kadry, and R. Damaševičius, "An efficient DenseNet-based deep learning model for malware detection," *Entropy*, vol. 23, no. 3, p. 344, 2021.

[52] Y. Yang, Z. Zhong, T. Shen, and Z. Lin, "Convolutional neural networks with alternately updated clique," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2413–2422, IEEE, Salt Lake, UT, USA, June 2018.