Method for constructing multi-dimensional feature map of malicious code

Haocong Ma1,3,4,*, Ji Zhang2, Junhua Zhou1,3,4, Xiang Zhai1,3,4, Junjie Xue1,3,4 and Hang Ji1,3,4

1State Key Laboratory of Intelligent Manufacturing System Technology, Beijing Institute of Electronic System Engineering, Beijing 100854, China
2School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
3Beijing Complex Product Advanced Manufacturing Engineering Research Center, Beijing simulation Center, Beijing 100854, China
4Science and Technology on Space System Simulation Laboratory, Beijing Simulation Center, Beijing 100854, China

*3220180839@bit.edu.cn

Abstract. Malicious code is characterized by a large number of types, rapid increase in number, continuous update of transmission routes, and continuous enhancement of back analysis and back detection methods. Therefore, how to effectively detect and analyze malicious code has been a problem of great concern. This paper studies the features of binary file and disassembly file of malicious code, introduces the concept of information gain, and proposes a method to construct the multi-dimensional characteristic graph of malicious code. Finally, the convolutional neural network is used to classify the multi-dimensional feature graph of malicious code, which provides a new idea for the feature extraction of malicious code.

1. Introduction
While the number of malicious codes is growing rapidly, the technology of malicious code production is also improving rapidly. The emergence of technologies such as polymorphism, deformation, anti-tracking, port bounce and Rootkit has brought great difficulties to the detection of malicious code. Therefore, the effective detection and classification of malicious code has become an urgent problem.

The characteristic processing of malicious code is the premise of malicious code classification. The malicious code visualization method is a visual representation method, which reduces the overhead of malicious code feature extraction in the traditional malicious code detection, and can be directly learned through the neural network model. In recent years, it has been widely studied and applied in the field of malicious code detection, showing a good detection rate in various experiments. In order to solve the one-sidedness problem of single feature classification, it has become a new trend to extract multi-dimensional features for comprehensive treatment.

2. Related work
The main malicious code detection technologies include static detection technology and dynamic detection technology. With the development of machine learning and deep learning techniques,
especially deep neural networks, such techniques have been widely used in malicious code detection research. In 2001, Shultz et al. extracted DLL information from malicious code as features, and for the first time proposed a method to detect malicious code using data mining technology. In 2008, Conti et al. first mentioned the idea of visualizing malicious code binary data in their paper. Nataraj et al. proposed to visualize binary files in the form of grayscale images in 2011, and use texture features in images to cluster malicious codes. In 2012, Shabtai et al. extracted the n-gram sequence of the opcode from the disassembly file as a feature to identify unknown malicious code. HUANG et al. proposed a multi-task deep learning architecture in 2016, which extracts features from the dynamic analysis of malicious files and benign files to train each model. In 2018, Cui et al. designed a classification network to automatically extract the characteristics of malware images and convert the malicious code into grayscale images, and use convolutional neural networks to identify and classify the images. In 2019, Zhu et al. expressed malware as opcode sequence and used Deep Belief Network (DBN) for detection, which provided a high accuracy rate.

To sum up, this paper analyzes the malicious code files from multiple perspectives, extracts the static features of the malicious code binary files and disassembly files, and uses the convolutional neural network to learn the features after fusion, and proposes a new method to construct the multi-dimensional characteristic map of malicious code.

3. Construction of multi-dimensional feature graph of malicious code

From the perspective of imagology, gray map divides white and black into 0 to 255 different gray scales according to logarithmic relation. There are 256 levels in total, representing 256 kinds of information. RGB images get different colors through changes and superpositions of R, G, and B components. For an RGB image of size m×n, it can be represented as a 3×m×n array. The value range of each pixel is 256=16777216. Based on the binary, the disassembly characteristics and operation code, we build multidimensional characteristics. The selected features are described below.

3.1. Binary features

Binary features are generated by malicious code binaries that convert every two hexadecimal digits into decimal digits 0-255 as the pixels of the image. Specific steps are as follows:

Read the binary file by line, and the value range of each two hexadecimal number is 0-255, which is regarded as a pixel;
1) Calculate all the pixels and get a vector composed of pixels;
2) If the length of the one-dimensional vector is greater than 65536, cut its length to 65536. If the length of the one-dimensional vector is less than 65536, expand it to 65536 with 0;
3) Expand the one-dimensional vector into a two-dimensional matrix with a size of 256×256.

3.2. Disassembly features

The disassembly file has the suffix ".asm " and is an Assembly Language source program. As a kind of machine-oriented programming language, assembly language use mnemonics instead of the operation of the machine instruction code, address symbols and label instead of instruction or the address of the operand. Disassembly files contain static API, symbolic features, opcodes and other information, which is of great significance for detecting malicious code. Disassembly feature processing mainly consists of the following steps.

1) Remove the header information of the disassembly file and the address information in each line;
2) Convert the file into a hexadecimal file;
3) Convert all hexadecimal numbers to decimal numbers, the range is 0-255, to get a one-dimensional vector;
4) If the length of the one-dimensional vector is greater than 65536, cut its length to 65536. If the length of the one-dimensional vector is less than 65536, expand it to 65536 with 0;
5) Expand the one-dimensional vector into a two-dimensional matrix with a size of 256×256.
3.3. Opcode features

Operation code (opcode) is a very important piece of information in a malicious code disassembly file. It represents the Operation to be performed by the file, while the sequence of opcodes represents the entire flow of malicious code. Traditionally, data mining is used to analyze opcode sequences, such as opcode N-gram information. N-gram is a concept of linguistic and probabilistic scope, specifically a sequence of N consecutive items in a text or speech. In this paper, 3-gram model is adopted for the opcode feature processing to calculate the information gain of 3-gram sequence. Information gain is a concept of probability and information theory. The greater the information gain is, the better the selectivity of this feature will be. It is defined as the difference between the information entropy of the set to be classified and the conditional entropy of the selected features:

\[
IG(Y \mid X) = H(Y) - H(Y \mid X)
\]

Where, information entropy is the expected value of information, which is used to measure uncertainty. The higher the entropy value, the greater the uncertainty of \(X= Xi\), which is denoted as:

\[
H(X) = \sum_{i=1}^{n} p(x_i) \log_2 p(x_i)
\]

The conditional entropy of random variable \(X\) under given conditions \(Y\) is the mathematical expectation of conditional probability distribution of \(y\) under given conditions \(x\), and its formula is as follows:

\[
H(Y \mid X) = \sum_{i} P(x)H(Y \mid X = x)
\]

Extract the opcode 3-gram sequence of the malicious code sample and calculate its information gain. The specific algorithm is shown in Table 1:

Table 1: Algorithm of Calculation of Malicious Code Operation Code Features

| Algorithm 1 Calculation of Malicious Code Operation Code Features |
|---------------------------------------------------------------|
| Input: the malicious code file                                   |
| Output: information gain value of opcode 3-gram                 |
| A certain 3-gram feature \(m\) divides the set \(D\) into \(D_1, D_2, D_3, \ldots, D_k\), the number of samples in the subset \(D_i\) is recorded as \(|D_i|\); the set of samples belonging to \(C_j\) in \(D_i\) is recorded as \(D_{ij}\). The number of samples is \(|D_{ij}|\). |
| Function getInformationGain( ):                                    |
| return \(\rightarrow\) Map()                                         |
| Extract the opcode sequence of each disassembled file and record the relative position of this sequence in the malicious code file; |
| Extract the 3-gram sequence of opcodes corresponding to each file and get the set \(M\); |
| Extract all the malicious code file set \(D\), and record the number as \(|D|\). \(D\) is divided into \(n\) categories in total, each category is denoted as \(C_i\), and the number of samples in this category is \(|C_i|\). |
| Calculate the information entropy \(H(D)\) of the entire set \(D\) |
| \(H(D) = -\sum_{j=1}^{n} \frac{|C_j|}{|D|} \log_2 \frac{|C_j|}{|D|}\) for \(m\) in \(M\)                      |
| The feature \(m\) divides the set \(D\) into \(D_1, D_2, D_3, \ldots, D_k\) for \(i\) in \(k\)                             |
| Calculate the number \(|D_i|\) of samples in the subset \(D_i\)   |
| Calculate the number of samples \(|D_{ij}|\) contained in the sample set \(D_{ij}\) belonging to \(C_j\) in \(D_i\) end for |
| Calculate the conditional entropy \(H(D\mid m)\) of feature \(m\) |
| \(H(D\mid m) = \sum_{j=1}^{n} \frac{|D_j|}{|D|} H(D_j) = -\sum_{j=1}^{n} \frac{|D_j|}{|D|} \sum_{i=1}^{k} \frac{|D_{ij}|}{|D_j|} \log_2 \frac{|D_{ij}|}{|D_j|}\) Calculate the information gain \(g(D, m)\) of feature \(m\) |
The information gain of all 3-gram sequences in the data set is calculated by this method. Select the feature whose information gain value is the first 65536 and map it to the range of 0-255 after normalization processing. When extracting the 3-gram sequence of opcode, the relative position of this sequence in the malicious code file is recorded at the same time, so that the position of the 3-gram sequence of opcode can correspond to the corresponding position of binary features. According to the position of opcode in the malicious code file, it is marked in a two-dimensional matrix of 256×256=65536. By combining binary and disassembly features in this way, the one-sideality matrix for a single feature is improved so that no opcode 3-gram position is filled with 0. Three kinds of feature information of malicious code files are obtained by extracting the features of three different dimensions. According to the generation principle of RGB image, the matrix of three dimensions is superposed together and saved into a file in the format of "BMP", which is stored according to the actual value in the original matrix.

4. Experiment

4.1. Experimental conditions
In this paper, the experiment using a personal computer, its hardware environment mainly as follows: CPU for i5 8400, GPU is 980 ASUS raptor, coupled with 16g of memory and hard disk 1T. The operating system used in the experiment was Windows 10, and Pytorch1.6 was used for code writing. The malicious code files tested in this article contain a total of 10,124 malicious code samples. All the samples fall into nine categories, with one binary and one disassembly file for each malicious code. The 10,124 samples were divided into three parts according to the proportion of "Training set: Verification set: Test set =8:1:1".

4.2. Classification model
In order to fully compare the differences between feature maps, this section adopted convolutional neural network as the classification algorithm, and designed three conv1, conv3 and conv5 classification models to conduct classification learning of three single-dimensional feature maps and one multi-dimensional feature map, and compared their classification indexes.

The structure of the three classification models is shown in Table 2.

| conv1          | conv3          | conv5          |
|----------------|----------------|----------------|
| Input 256×256 image | Input 256×256 image | Input 256×256 image |
| 5×5 conv.16 ReLU.   | 5×5 conv.16 ReLU.   | 5×5 conv.16 ReLU.   |
| FC. Output layer.   | 5×5 conv.32 ReLU.   | 5×5 conv.32 ReLU.   |
|                  | 5×5 conv.16 ReLU.   | 5×5 conv.16 ReLU.   |
|                  | FC. Output layer.   | 5×5 conv.16 ReLU.   |
|                  |                 | 5×5 conv.8 ReLU.   |
|                  |                 | FC. Output layer.   |

4.3. Result
In the experiment, the accuracy and F1 values of each feature graph verification set during the training of the three models were recorded. In order to better compare the effect of each feature graph, it is plotted as a curve for comparison, as shown in Figure. 1.
The experimental results on the test set and the training time comparison are shown in Table 3.

### Table 3. The experimental results on the test set

| Feature | Model | Accuracy | Precision | Recall | F1      | Time (s) |
|---------|-------|----------|-----------|--------|---------|----------|
| ASM     | conv1 | 92.12%   | 92.89%    | 92.12% | 91.98% | 521      |
| GREY    |       | 86.88%   | 87.61%    | 86.88% | 86.88% | 711      |
| I_G     |       | 86.38%   | 87.14%    | 86.38% | 86.02% | 699      |
| RGB     |       | 92.33%   | 92.05%    | 92.33% | 92.08% | 806      |
| ASM     | conv3 | 94.15%   | 94.03%    | 94.15% | 94.05% | 781      |
| GREY    |       | 91.42%   | 91.59%    | 91.42% | 91.43% | 805      |
| I_G     |       | 91.93%   | 92.37%    | 91.93% | 91.91% | 789      |
| RGB     |       | 95.76%   | 95.91%    | 95.76% | 96.62% | 823      |
| ASM     | conv5 | 95.96%   | 96.04%    | 95.96% | 95.84% | 815      |
| GREY    |       | 92.84%   | 93.07%    | 92.84% | 92.89% | 812      |
| I_G     |       | 93.84%   | 94.77%    | 93.84% | 93.83% | 838      |
| RGB     |       | 96.67%   | 96.65%    | 96.67% | 96.61% | 923      |

The experimental results show that the convolutional neural network has a prominent performance in the processing of malicious code image classification, and the multi-dimensional feature map of malicious code achieves an accuracy rate of 96.67% in the test set, both of which are better than the single feature map. By analyzing the training time of the model, it can be seen that the multi-dimensional feature graph of malicious code carries more information, so the training time is larger than the other three feature graphs. But considering the particularity of the research problem, it is more meaningful to improve the classification efficiency. From the classification index of the experimental test set, it can be seen that after training the multidimensional characteristic diagram of the malicious code, the model has a good generalization ability, and its ROC curve is shown in Figure 2.
5. Conclusion

Aiming at the shortcomings of the traditional malicious code visualization methods, this paper proposes a malicious code based on binary files and the disassembly building multidimensional feature diagram method, the malicious code from the disassembly characteristics of malicious code file, binary features and operation code in constructing multi-dimensional characteristic figure, and designed a three convolution model of neural network classification characteristics of four types of figure for training. After several times of optimization and improvement, the classification accuracy rate of the multi-dimensional feature map finally reached 96.67%, and the AUC of the model reached 0.99298. The main contribution of this paper is to propose a complete multi-dimensional feature map of malicious code and design the classification experiment, and realize the feature extraction and classification detection of sample malicious code files.

This paper mainly introduces the feature extraction and classification model of malicious code files, but does not conduct in-depth research on the distribution of experimental data sets and the setting of model parameters, and will continue to do so in the future. The subsequent work needs to optimize the classification model and classification model parameters, and solve the problem of uneven distribution of original samples of malicious code, so as to further improve the accuracy and stability of classification training.

References

[1] Schultz M G, Eskin E, Zadok F, et al. Data Mining Methods for Detection of New Malicious Executables[C]// IEEE Symposium on Security & Privacy. IEEE, 2001.

[2] Conti G, Dean E, Sinda M, et al. Visual Reverse Engineering of Binary and Data Files[C]// International Workshop on Visualization for Computer Security. Springer, Berlin, Heidelberg, 2008.

[3] Nataraj L, Karthikeyan S, Jacob G, et al. Malware images: visualization and automatic classification[C]. Proceedings of the 8th international symposium on visualization for cyber security, ACM, 2011:4.

[4] Shabtai, A., Moskovitch, R., Feher, C. et al. Detecting unknown malicious code by applying classification techniques on OpCode patterns[J]. Security Informatics, vol. 1, no. 1, p. 1, 2012.

[5] HUANG W, STOKES J W. Mt Net: a multi-task neural network for dynamic malware classification[C]. The International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA). 2016: 399-418.

[6] Zhihua Cui, Fei Xue, Xingjuan Cai, et al. Detection of Malicious Code Variants Based on Deep Learning[J]. IEEE transactions on industrial informatics, 2018, 14(7):3187-3196.

[7] Zhu Siyi, Ding Yuxin. Malware detection based on deep learning algorithm[J]. Neural computing & applications, 2019, 31(2).