A malicious code family classification method based on self-attention mechanism

Ru Zhang1*, Xinjian Zhao2, Jiaqi Li1, Song Zhang3 and Zhijie Shang3

1School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China
2Information & Telecommunication Branch, State Grid Jiangsu Electric Power Co. Ltd, Nanjing, China
3State Grid Information & Telecommunication Branch, Beijing, China
*Corresponding author’s e-mail: zhangru@bupt.edu.cn

Abstract. Malicious code families have become a major threat to network security. Many current methods convert malicious code into images and use deep learning to classify malicious code families. However, the family classification method based on deep learning incorporates the overall characteristics of the malicious code into the classification model, which may cause interference with redundant information in the malicious code. This paper proposes a malicious code family classification method based on a self-attention mechanism. When analysing a noisy data structure such as malicious code images, the attention mechanism is introduced to filter the interference information in the malicious code. The experimental results show that the classification accuracy rate of this method for malicious code families is 99.56%, and the recall rate is 98.06%. Rigorous theoretical analyses and numerous experiments prove our method is efficient and reliable.

1. Introduction
Currently, malicious code attacks on the network are more organized. Family-based malicious codes often appear in major hacker attacks. At the same time, the number of malicious codes is showing a rapid development trend. By modifying the code reuse and other means, An attacker with a certain classic malicious code blueprint is able to write more powerful and danger variants quickly which adapt to the latest testing methods. Using the derived correlation between malicious code can quickly locate the source of attack or attacker, and have a certain deterrent effect on the attacker. It plays an important role in deterring hacker attacks and improving the network security system [1].

With the continuous improvement of malicious code countermeasures and detection methods, traditional malicious code classification methods are gradually unable to meet the needs of malicious code classification. By transforming the problem of malicious code classification into classification problems of other types of data has become a trend. It takes the advantages of deep learning in natural language processing, image recognition, and feature extraction. However, the current network has some interference with redundant information in the process of malicious code images, which will cause the characteristics of non-malicious code to be included in the classification model during the model training phase. Therefore, this paper proposes a malicious code family classification method based on the self-attention mechanism. When analysing the noisy data structure such as malicious code images, the self-attention mechanism is introduced into the network to filter out redundant information in the code.
2. Related Work

Nowadays, the malicious code family classification is mainly represented by NLP-based classification and CV-based classification. Abou-Assaleh et al. [2] used N-Gram to extract character-level malicious code features, and used KNN algorithm to classify malicious codes. However, it is limited by the small amount of data and needs to be verified on a larger data set. Kolosnjaji et al. [3] also used N-Gram as the feature extraction of malicious code, but it researches the API call sequence of malicious code.

CV-based malicious code image detection method was first proposed by Nataraj and Karthikeyan of the University of California in 2011 [4]. They converted the binary file of malicious code into the form of image, thus turning the classification problem of malicious code into the classification problem of unknown image. Then through deep learning to complete the classification of the image, use the texture features in the image to cluster the malicious code, and complete the establishment of the malicious code family. Subsequently, the researchers combined malicious code image technology with deep learning, and conducted research on malicious code by exploring the characteristics of malicious code images and the classification of malicious code images. Xiaolin et al. [5] research on the detection method of malicious code variants based on texture fingerprinting.

3. Neural network design

When classifying samples, there will inevitably be some redundant information in the samples. Generally speaking, the sample is likely to contain information irrelevant to the current classification task, and these noises may adversely affect the classification effect.

This paper proposes a malicious code family classification method based on self-attention mechanism shown as Figure 1. Compared with the traditional deep learning network for malicious code classification, this method introduces the self-attention mechanism and soft threshold mechanism into the network structure, and automatically sets the threshold for each feature channel.

3.1. Sampling Layer
The main goal of the sampling layer is to down-sampling the input malicious code image. This paper uses a 7x7 large convolution and a 3x3 max pool as the sampling structure of the network, and the step size is set to 2. As the Figure 2 shows, the image (a) is the original image of the malicious code, and the image (b) is the feature map of the malicious code processed by the sampling layer.

3.2. Residual Shrinkage Convolutional Layer
In the process of feature learning, the residual shrinkage network (RSN) introduces "soft thresholding" as a "shrinkage layer" into the residual module, and proposes an adaptive threshold setting method that can eliminate redundant information.
(1) Convolution calculation uses 1*1 convolution to reduce the dimensionality of the data, to prevent the parameters from being too large, and to reduce the amount of calculation of the neural network.

(2) Replace the traditional convolution with depthwise separable convolution (DSC) to further reduce the amount of parameters and calculations of the neural network.

(3) The attention mechanism and soft threshold mechanism are introduced. The calculation process of each convolutional layer is shown in Figure 4. It consists of an identity shortcut module, a threshold calculation module and a soft threshold module. A sub-network is used to automatically set the threshold. And apply it to all channels of the feature map.

(4) The attention mechanism used by this module is not shared between different convolutional layer channels. The Residual shrinkage module and Convolutional layer structure are displayed by Figure 3 and Figure 4.

3.3. Spatial Pyramid Pooling
This paper introduces spatial pyramid pooling (SPP) in the deep residual shrinkage network to replace the original GAP, so that the network can accept images of different sizes. This article takes the maximum scale of 4 as an example. Assuming that the input image size is (w, h), SPP performs corresponding scale pooling for each part of the feature map. The specific selection process is as follows

(1) The first level of the pyramid divides the entire image into 16 blocks. In the case of image size (w, h), the size of each extracted block is (w/4, h/4).
(2) The second level of the pyramid divides the entire picture into 4 blocks. In the case of image size 
(w, h), the size of each extracted block is (w/2, h/2).

(3) The third layer of the pyramid divides the entire picture into a whole block. In the case of the 
image size (w, h), the extracted block size is (w, h).

4. Experimental Evaluation and Comparative Analysis

4.1. Experimental Environment and Data Set

This article uses pytorch to construct neural networks, including traditional CNN networks, spatial 
pyramid pooling + CNN networks, resnet50 networks, and the deep residual network that combines 
spatial pyramid pooling and separable convolutions (spp-resnet) proposed in this article.

The data set used in this article is provided by the kaggle malicious code classification competition 
Microsoft Malware Classification Challenge (BIG 2015). There are a total of 10896 malicious code 
training sets provided by this data set. The data set contains 9 malicious code families. 80% is used as 
the training set and 20% is used as the test set. Use accuracy, precision, recall, FPR, and F-measure to 
evaluate experimental results.

4.2. Experimental Results and Analysis

For intuitive display, the feature map display under the same network depth selects the output of the 
shallow network. The output of the feature map of block1 with a total of 256 channels is displayed as 
the experimental result to observe the effect of the attention mechanism on the feature map that the 
convolutional neural network can learn.

![Figure 5 block1 output feature map](image)

As shown in Figure 5, compared with the same malicious code image, the RSN’s performance is 
better in the reflection of the malicious code image texture. In the output feature map without introducing 
the attention mechanism network, some images can no longer inform the texture features of the original 
image. However, in the introduction of the attention mechanism network, each feature map can observe 
some features of the original image.

Table 1 shows the classification results after introducing the attention mechanism, and Tables 2 and 
3 show the comparison results of malicious code family classification on the same data set.
Table 1  Residual shrinkage network classification results

|                | TPR   | FPR   | Accuracy | Recall rate | F-measure | Precision |
|----------------|-------|-------|----------|-------------|-----------|-----------|
| Ramnit         | 98.0% | 0.004 | 97.11%   | 98.0%       | 0.9761    | 99.31%    |
| Lollipop       | 98.7% | 0.003 | 98.98%   | 98.7%       | 0.9868    | 99.52%    |
| Kelihos_ver3  | 98.3% | 0.006 | 98.13%   | 98.3%       | 0.9821    | 99.06%    |
| Vundo          | 97.35%| 0.004 | 94.00%   | 97.89%      | 0.9538    | 99.58%    |
| Tracur         | 97.35%| 0.004 | 99.22%   | 97.35%      | 0.9832    | 99.76%    |
| Kelihos_ver1  | 95%   | 0.0004| 98.7%    | 95%         | 0.9681    | 99.76%    |
| Obfuscator.ACY | 99.2% | 0.002 | 98.38%   | 99.2%       | 0.9879    | 99.70%    |
| Gatak          | 100%  | 0.003 | 97.59%   | 100%        | 0.9878    | 99.76%    |
| Entire         | --    | --    | --       | --          | --        | --        |

Table 2  Recall rates of each family based different neural network

|                | Without attention mechanism | With attention mechanism |
|----------------|-----------------------------|--------------------------|
| Ramnit         | 97.08%                      | 98.0%                    |
| Lollipop       | 97.37%                      | 98.7%                    |
| Kelihos_ver3  | 97.45%                      | 98.3%                    |
| Vundo          | 96.84%                      | 97.89%                   |
| Tracur         | 95.36%                      | 97.35%                   |
| Kelihos_ver1  | 96.25%                      | 95%                      |
| Obfuscator.ACY | 97.15%                      | 99.2%                    |
| Gatak          | 96.06%                      | 100%                     |

Table 3  Accuracy of each family based different neural network

|                | Without attention mechanism | With attention mechanism |
|----------------|-----------------------------|--------------------------|
| Ramnit         | 95.23%                      | 97.11%                   |
| Lollipop       | 97.97%                      | 98.98%                   |
| Kelihos_ver3  | 96.47%                      | 98.13%                   |
| Vundo          | 89.32%                      | 93.00%                   |
| Tracur         | 96.66%                      | 99.32%                   |
| Kelihos_ver1  | 97.46%                      | 98.7%                    |
| Obfuscator.ACY | 98.38%                      | 98.38%                   |
| Gatak          | 97.98%                      | 97.59%                   |

As the data shows, the recall rate of the neural network with the attention mechanism is generally higher than that of the neural network without the attention mechanism. After introducing the attention mechanism, some malicious code images that were originally classified incorrectly were classified into the correct category. The accuracy rate represents the proportion of correct results in the final result of the classification. As shown in Table 3, after the introduction of the attention mechanism, the accuracy of the classification results of the same family has been increased, and the network can better classify malicious codes in family units.

On the other hand, the attention mechanism based on soft thresholds amplifies a certain feature of each category, and filters out the feature map output of the remaining channels. The attention mechanism has a more obvious improvement for malicious code families with a small number of samples. But at the same time, it will also have an additional impact on relatively similar malicious code families.
Table 4  FPR of each family based different neural network

|                  | Without attention mechanism | With attention mechanism |
|------------------|-----------------------------|--------------------------|
| Ramnit           | 0.008                       | 0.004                    |
| Lollipop         | 0.006                       | 0.003                    |
| Kelihos_ver3     | 0.013                       | 0.006                    |
| Vundo            | 0.004                       | 0.003                    |
| Tracur           | 0.002                       | 0.004                    |
| Kelihos_ver1     | 0.0001                      | 0.0004                   |
| Obfuscator.ACY   | 0.002                       | 0.002                    |
| Gatak            | 0.002                       | 0.003                    |

Our method also performs well on the FPR values. After the introduction of the attention mechanism, except for Kelihos_ver1, the FPR of the classification results of the rest of the families decreased.

5. Conclusion
This paper proposes a malicious code family classification method based on the self-attention mechanism. Compared with the traditional deep learning network for malicious code classification, this method introduces the attention mechanism and soft threshold mechanism into the network structure, which can filter redundant information to a certain extent. Based on this conclusion, the structure of the convolutional neural network is revised. Experimental results show that the introduction of an attention mechanism into the network can bring positive feedback to the malicious code family classification network, which can amplify the characteristics of the malicious code itself while suppressing the redundant information in the malicious code.

Acknowledgments
The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is supported by Science and Technology Project of the Headquarters of State Grid Corporation of China, “The research and technology for collaborative defense and linkage disposal in network security devices” (5700-202152186A-0-0-00).

References
[1] Wenna S., Guojun P., Jianming F., Huanguo Z., (2019). Research on Malicious Code Evolution and Traceability Technology. Journal of Software, 30(8).
[2] Abou-Assaleh, T., Cercone, N., Keselj, V., & Sweidan, R. (2004, September). N-gram-based detection of new malicious code. In Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004. (Vol. 2, pp. 41-42). IEEE.
[3] Kolosnjaji, B., Zarras, A., Webster, G., & Eckert, C. (2016, December). Deep learning for classification of malware system call sequences. In Australasian joint conference on artificial intelligence (pp. 137-149). Springer, Cham.
[4] Nataraj, L., Karthikeyan, S., Jacob, G., & Manjunath, B. S. (2011, July). Malware images: visualization and automatic classification. In Proceedings of the 8th international symposium on visualization for cyber security (pp. 1-7).
[5] Xiaolin, Z., Yiman, Z., Xuhui, L., & Quanbao, C. (2018, August). Research on malicious code homology analysis method based on texture fingerprint clustering. In 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE) (pp. 1914-1921). IEEE.