ISSN 2320 2629
Volume 10, No.3, May - June 2021
International Journal of Information Technology Infrastructure
Available Online at http://www.warse.org/IJITI/static/pdf/file/ijiti011032021.pdf
https://doi.org/10.30534/ijiti/2021/011032021

Malicious Code Invariance Based On Deep Learning

Sameena S, Akshara Sivan, Sreelakshmi K V, Shilpa Suresh, Seethu George
Department of Computer Science and Engineering, AISAT, Kalamassery, Samheenparwin4444@gmail.com
Department of Computer Science and Engineering, AISAT, Kalamassery, aksharasivan2000@gmail.com
Department of Computer Science and Engineering, AISAT, Kalamassery, sreelakshmikv916@gmail.com
Department of Computer Science and Engineering, AISAT, Kalamassery, shilpasuresh2109@gmail.com
Department of Computer Science and Engineering, AISAT, Kalamassery,seethugeorge@aisat.ac.in

ABSTRACT

The malicious code detection is critical task for in the field of security. The malicious code detection can be possibly by using convolutional neural network (CNN). The malicious code can be categorized in to different families. The malicious code identification helps to identify the affected malware on the system. Malicious code theft data from our system and it yields high security issues in real time. The neural network architecture classifies the malicious code based on the collected dataset. The dataset contains different families of malicious code. The malicious code detection can be done with the help of model created from CNN architecture.

Keywords: Malicious Codes, Security Breaches, Convolutional Neural Network(CNN), Data Augmentation, Feature Extraction, Malware Families.

1. INTRODUCTION

Machine learning is an application of artificial intelligence that provides systems the ability to learn with data and improve from their experience without clear programming. Machine learning focuses on developing computer programs, which can access data and learn by themselves. In this project we are using deep learning which is branch of machine learning. Now a days, Malware detection is crucial with malware's prevalence on the internet because it functions as an early warning system for the computers secure regarding malware and cyber-attacks. It keeps the hackers out of computer and prevents the information from getting compromised. The ability to detect variants of malicious code is critical for protection against security breaches, data theft and some other dangers. Current method for recognizing the malicious code have demonstrated poor detection accuracy and low detection speeds. Our system uses the deep learning approach to detect the malicious family.

Our system automates the malware detection. In this approach we are using 17 families in which the image which is malicious will be classified in to the corresponding 15 malware families, the image which is malware but not known will be classified in to the unknown family and the image which is not a malicious image will be classified as non-malicious class. CNN is used to classify the 17 classes in to corresponding families. The automated Feature extraction in CNN is responsible for this.

2. LITERATURE SURVEY

Malware Detection based on Feature Analysis[1]*:* As described previously in this paper, there are two main parts of feature analysis techniques for malware detection: static analysis and dynamic analysis. With respect to static analysis, several methods have been put forward by analysing the codes. For example, Isohara et al. developed a detection system using kernel behaviour analysis that performed well detecting the malicious behaviours of unknown applications. However, static method is easily deceived by obfuscation techniques. To solve this problem, Christodorescu et al. proposed a novel malware-detection algorithm that used trace semantics to characterize the behaviour of malware. This model proved effective in combating the obfuscation of instructions (e.g., rearrangement of instructions, insertion of garbage code, register redistribution). However, the approach was limited to feature extraction and analysis at the instruction level of the model. Moreover, the pattern-matching was complex.

Dynamic analysis monitors and analyses the runtime characteristics of applications (apps) based on assessment of behaviours, such as accessing private data and using restricted API calls. Given this information, a behaviour model is established to detect malicious code. Such techniques improved detection performance, but they were still challenged by the variety of countermeasures developed to generate unreliable results. In addition, dynamic analysis is time-consuming because of the large amount of computational overhead, leading to low efficiency when exposed to a large dataset.

Malicious Code Visualization[2]:Currently, many tools can visualize and analyse binary data, e.g., common text editors and binary editors. Several studies have proposed utilization of malware visualization. Yoo et al. employed self-organizing maps to visualize computer viruses. A similar but another approach was proposed by Trinius et al. who used two visualization techniques tree maps and thread graphs to detect and classify the malware in the system. Apart from a single detection result, Goodall et al. aggregated the results of different types of malware analysis tools into a visual environment, providing an increase in the vulnerability of detection coverage of single malware tools. The above studies focused mainly on the visualization of malware behaviour, but the software source code may imply more meaningful patterns. As previously mentioned, Nataraj et al. put forward a new visualization approach for malware detection based on binary texture analysis. First, they

converted the malicious executable file into grayscale images. Then they identified malware according to the texture features of these input images. Compared with the dynamic analysis method, their approach produced equivalent results. In similar work, Han et al. transformed malware binary information into color image matrices, and arranged malware families by using an image processing method.

Image Processing Techniques for Malware Detection[3]: Whenever the malware has been visualized as grayscale images, malware detection can be converted into an image recognition problem. Nataraj et al. used a GIST algorithm for extracting the features of malware images. However, the GIST algorithm was time-consuming. Now a days, more powerful image processing techniques have been proposed. Daniel et al. developed a bio-inspired parallel implementation for identifying the representative geometrical objects of the homology groups in a binary 2D image. For image fusion, Miao et al. presented an image fusion algorithm based on shearlet and genetic algorithm. In their model, a genetic algorithm is employed to optimize the weighted factors in the fusion rule. Experimental results demonstrated their method could acquire better fusion quality than other methods. These traditional models, however, were challenged by the high time cost required for complex image texture feature extraction. To label this challenge, we employed deep learning to identify and classify images efficiently. In the next section, we present our research on malware detection based on deep learning.

Malware Detection based on Deep Learning[4]: Deep learning is an area of machine learning research that has emerged in recent years from the work on artificial neural networks. Neural networks can approximate complex functions by learning the deep nonlinear network structure to solve complex problems. Deep learning, which is more powerful than back propagation, uses a deep neural network to simulate the human brain's learning processes. Deep learning has the ability to learn the essential characteristics of data sets from a sample set. As a powerful tool of artificial intelligence, deep learning has been applied in many fields, such as recognition of handwritten numerals, speech recognition, and image recognition. Because of its powerful ability to learn features, many scholars have applied deep learning to malware detection. Using deep learning techniques, Yuan et al. created and implemented an online malware detection prototype system, named Droid-Sec.

Their model attained high accuracy by learning the features extracted from both static analysis and dynamic analysis of Android apps. David et al. presented a similar but more compelling method that did not need the type of malware behaviour. Their work was based on the deep belief network (DBN) for automatic malware signature generation and classification. Compared with conventional signature methods for malware detection, their model demonstrated increased accuracy for detecting new malware variants. And, these methods remained based on the analysis of features extracted by static analysis and dynamic analysis. Therefore, to a greater or lesser extent, they continuedto the limitations of feature extraction. To address this problem, we employed a CNN network to learn the malware image features and classify them automatically. Features of the images like frequency, way of attack and other are used to compare the training set and the input image

3. PROPOSED SYSTEM

In our Proposed Model, Deep learning is used to detect malware family. In our Model, we have 17 classes of which 15 are the malware families namely 1. Agent, 2. Adopshel, 3.Allaple, 4.BrowseFox, 5.Dinwod, 6.Elex, 7.Expiro, 8.Fasong, 9.Hlux, 10.Injector, 11.Neshta, 12.Vilsel, 13.Regrun, 14.Stantiko, and 15.VBKrypt. The 16'th family consist of the Non-Malicious class and 17'th is the Unknown class.

We worked a CNN network to learn the malware image features and classify them automatically. The CNN is a convolutional neural network which consists of different layers that perform different functions. The CNN is used to train the different malware families in the dataset. In our proposed paper, we are using coloured images instead of grayscale images. The features of the images which we are inputting will be automatically extracted by the CNN and compare it with the trained dataset and then it will predict the class of the inputted images. This is how our model works.

The main advantage of our proposed system is the ability to generate data whenever there exist a data imbalance. The data augmentation technique is used to solve the data imbalance. Image Data Augmentation Technology. In deep learning, to avoid overfitting problem, we usually need to enter sufficient data to train the model. If the data sample is small, we can use data augmentation to increase the sample, thereby restraining the influence of imbalanced data. The appropriate data augmentation method can avoid overfitting problems and improve the robustness of the model.

Generally, transformation of the original image data (changing the location of image pixels and ensuring that the features are still unchanged) is used to generate new data. There are many kinds of data augmentation techniques for images, for example, rotation=reflection, flip, zoom, shift, scale, contrast, noise, and color transformation.

A mechanism to convert .exe files to colored image is used. The input will be images from the malware images. There will be a graphical user interface (GUI) for inputting the image data. The image will be uploaded through this interface. The output of the system will be the predicted malicious family of given image. The system predicts the malicious family based on the model file created during training.

The first one is the input layer, which brings the training images into the neural network. Next are the convolution and sub-sampling layers. The former layer can enhance signal characteristics and noise can be reduced. The latter can reduce the amount of data processing while retaining various useful information. Then there are several fully connected layers that convert a 2-dimensional feature into a 1-dimensional feature that conforms to the classifier criteria. Finally, the classifier finds and sorts the malware images into different families according to their characteristics.

3.1 Convolution Layers

This layer scan reduce the number of image parameters while preserving the main features, termed invariance, including translation invariance, rotation invariance, and scale invariance. This process can avoid overfitting problem effectively, and improve the generalization ability of the model. The input is several maps, and the output will be the maps after dimension reduction. Each map will be a combination of convolution values of input maps that belong to the upper layer [1], and can be given by the following equation:

$$x_j^l = f(\sum_{i \epsilon Mj} x_j^{l-1} * K_{ij}^l + b_j^l) \quad (1)$$

where Mj is the collection of input maps, klij is the convolution kernel used for the connection between the ith input feature map and the jth output feature map, blj is the bias similar to the jth feature map and f is the activation function. The sensitivity is:

$$\delta_j^l = \delta^{l+1}{}_j \, W_j^{l+1} \delta^{\circ} f^l(u^l) = \beta_j^{l+1} up(\delta_j^{l+1})^{\circ} f^l(ul) \quad (2)$$

where l+1 layer is the sampling layer, the weight W represents
a convolution kernel, and its value $\beta^{l+1^{\circ}}$ up(.) is an up sampling operation. The partial derivative of the error cost function with respect to bias b and convolution kernel k can be given as:

$$dE/d_{bj} = \sum_{u,v} (\delta_j^l)_{u,v} \quad (3)$$

$$dE/dk^l i_j = \sum_{u,v} (\partial^{lj})_{u,v} (p_l^{1-i})_{u,v} \quad (4)$$

3.2 Sub-sampling Layers

This layer is also called the pooling layer. Generally, its and it We are using a custom Algorithm and the Convolutional Neural Network to train our proposed model that is the Malware Classification Model.reduce the dimension of the feature map, improves the model's accuracy, and avoids overfitting. In CNN, for each output of the sampling layer, the feature map is given as follows:

$$x_j^i = f(down(x_j^{l-1}) + b_j^i) \quad (5)$$

where down(:) represents a sub-sampling function, and b is bias. The sensitivity is calculated as shown:

$$\delta_j^l = \delta_j^{l+1} \, W_j^{l+1^{\circ}} f^l(u^l). \quad (6)$$

4. ALGORITHM

There are two major algorithms.

- One is the custom algorithm, and
- Another one is the Convolutional Neural Network

They are following:

The steps that we used in this algorithm is following. The first algorithm which is the custom algorithm it is a set of binding dynamics that are generated on an individual campaign basis and designed to deliver outcomes that are aligned to a specific goal. The second one is the CNN which is a convolutional neural network which consists of different layers to carry out feature extraction, classification, image recognition. By using this CNN we achieved more accuracy. It consists of Deep neural networks which is a powerful tool for classification.

4.1The Custom Algorithm Used for Converting exe File

Select the .exe file then perform the following,

- In this algorithm, we first convert .exe file to binary format.
- The binary array will be converted to blocks.
- Each block will be converted to pixel value.
- With pixel values create the image with defined height and width.

4.2 CNN Classification Method

CNNs are most efficient when it comes to image classification. In this, the mathematical function of convolution which is a special kind of linear operation wherein two functions are multiplied to produce a third function which expresses how the shape of one function is modified by the other one.

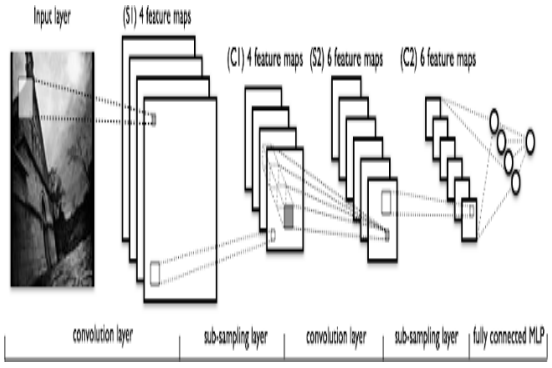

Figure4.1: The stages in CNN

- First, we pass an input image to the convolutional layer. The output is obtained as an activation map. The filters applied in the convolution layer will extract the required features from the input image to pass it.
- Each filter shall give a different feature to grant the correct class prediction. In any case that we need to retain the size of the input image, we use same padding (zero padding), otherwise valid padding is used since it helps to reduce the number of features to it.
- Pooling layers can reduce the number of parameters

- Several convolution and pooling layers are provided before the prediction is made. Convolutional layer help in extracting features. When we go deeper into the network more specific features are extracted as compared to a shallow network where the features are more generic.
- The output layer in a CNN is a fully connected, where the input from the other layers is flattened and sent so as the transform the output into the number of classes as intended by the network.
- The generated output through the output layer and is compared to the output layer for error generation. A loss function is defined in the fully connected output layer to get the mean square loss. The gradient of error is then calculated.
- Then the error is back propagated to update the filter and bias values.

5. DATASET

The dataset refers to a file that contains one or more record. Record is the basic unit of information that is used by a program running on z/OS. Any named group of record is called a data set. Our dataset consists of 17 classes. Each class contains 350 images. Therefore, we have 5950 images in total. The first 15 classes are the malicious families, the 16'th class is the non-malicious class and 17'th class contains the unknown images.

5.117 Classes

The following shows the 17 classes in our dataset that we used to train our model.

5.1.1 Agent

Agent is also called as Trojan. Trojan:W32/Agent is a very large family of programs, most of which download and install adware or malware to victim's machine. Agent variants may also change the configuration settings for Windows Explorer and/or for the Windows interface.

5.1.2 Adopshel

Adopshel is classified as a type of Riskware. Riskware is any potentially unwanted application that is not classified as malware, but may utilize system resources in an frightful or annoying manner, and/or may pose a security risk.

5.1.3 Allaple

Allaple is a multi-threaded, polymorphic network worm capable of spreading to other computers connected to a local area network (LAN).

5.1.4 BrowseFox

BrowseFox is Malwarebytes' detection name for a large family of adware that uses different methods of browser hijacking and monetizing to get their message across.

5.1.5 Dinwod

This Trojan arrives on a system as a file dropped by other malwares or a file downloaded unknowingly by users when visiting malicious sites.

5.1.6 Elex.

Elexis the detection name for a family of adware variants that go to any lengths to infiltrate and stay on user systems. Because of these methods, this trojan users are so aggressive, some of their components are classified as Trojans.

5.1.7 Expiro

Expiro is a class of malware that is a class of virus. This type viruses are Windows executable file infecting virus. It is also capable of stealing credit card information gathered from the affected machines.

5.1.8 Fasong

This Trojan arrives on a system as a file dropped by other malwares or a file downloaded unknowingly by users when visiting malicious sites. Same as Dinwod.

5.1.9 Hlux

Hlux describes software with malicious behavior that aims to gather information about a person or organization and send such information to another entity in a way that harms the users.

5.1.10 Injector

Injector trojans insert malicious code into processes running on a computer in order to perform various actions such as downloading additional malwares, interfering with web browsing activities or monitoring the user's actions.

5.1.11 Neshta

Neshta is an older file infector that is still general in the wild. It was initially percieve in 2003 and has been previously associated with BlackPOS malwares. It prepends malicious code to infected files. This threat is commonly introduced into an environment through unintented downloading or by other malwares.

5.1.12 Vilsel

Vilsel is detection name for a family of Trojans that change the system's proxy settings, bypass the Windows firewall, and downloads and executes other malwares.

5.1.13 Regrun

Regrun is a devious Trojan horse that may be installed onto a PC through a malicious link or even a web browser attack. After installed, Trojan. Regrun may open up the infected system to a remote hacker. The remote hacker could then obtain personal data and files from the infected PC. To completely eliminate the threat that has comes with the installation of Trojan. Regrun, a computer user should use a reputable anti-spyware program.

5.1.14 Stantiko

Stantiko can be used to execute certain operations such as searches, filling out forms, signing up for email lists that you're unaware of, and even allowing other backdoor activities. The backdoor has a loader to execute any practicable, and allowing the threat operators to execute any

code on the thousands of machines that belong to this botnet. It contains two malicious Windows services that can reinstall

5.1.15 VBKrypt

This malware family is written in the Visual Basic programming language, which is its main distinguishing traits from other malware families. This is a different class of malware

5.1.16 NonMalicious

This class consists of the images which does not belong to the malicious classes. Which means the non-malicious images will be classified in to this class.

5.1.17 Unknown

The images which are malicious but doesn't belong to any of the above 15 malware families will be classified in to this class.

6. EVALUATION

To validate the effectiveness and efficiency of the proposed approach, we designed our experiments. To verify the validity
of our proposed data equalization method, including the image data augmentation technique (IDA). It showed a high accuracy The confusion matrix for our model is shown below along with the graph.

Table1: Accuracy of malicious code invariance

Methods	Accuracy	Precision	Recall	RunningTime
GIST+KNN	91.9	92.1	91.7	60ms
GIST+SVM	92.2	92.5	91.4	64ms
GLCM+KNN	92.5	92.7	92.3	45ms
GLCM+SVM	93.2	93.4	93.0	48ms
Grayscale based	94.5	94.6	94.5	20ms
Our approach	96.8	96.8	96.6	30ms

The performance of our approach achieved a good result with respect to accuracy, even if the data set was not processed. The accuracy of our model with respect to the Gray scale based method is high. Our approach showed 96.8% accuracy. But the Gray scale-based approach only showed 94.5% accuracy.
The graph given below shows accuracy in the Y-axis and epochs in the X-axis. The table1 given below shows the results of our approach compared with the outcomes of the 5 models: Gray scale-based model, GIST+KNN, GIST+SVM, GLCM + KNN, and GLCM+SVM.
The Graph and the confusion matrix for our approach is shown below in figure 6.1 and 6.2. It shows the accuracy obtained with different malware families that is the seventeen classes. The graph shows the different losses and accuracies for training and validation sets.

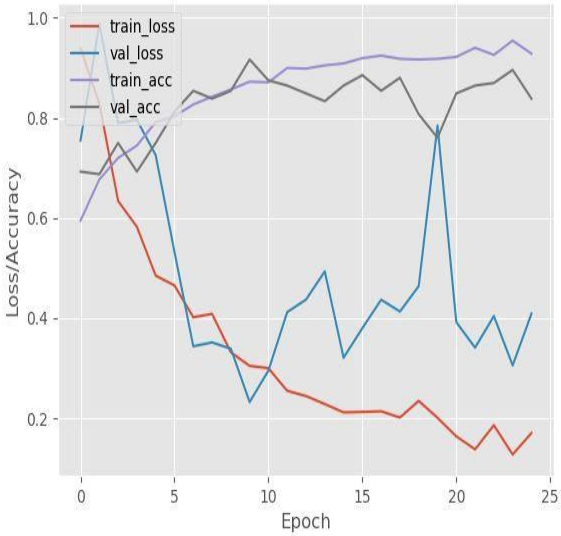

Figure 6.1: Graph for the proposed approach showing loss/accuracy

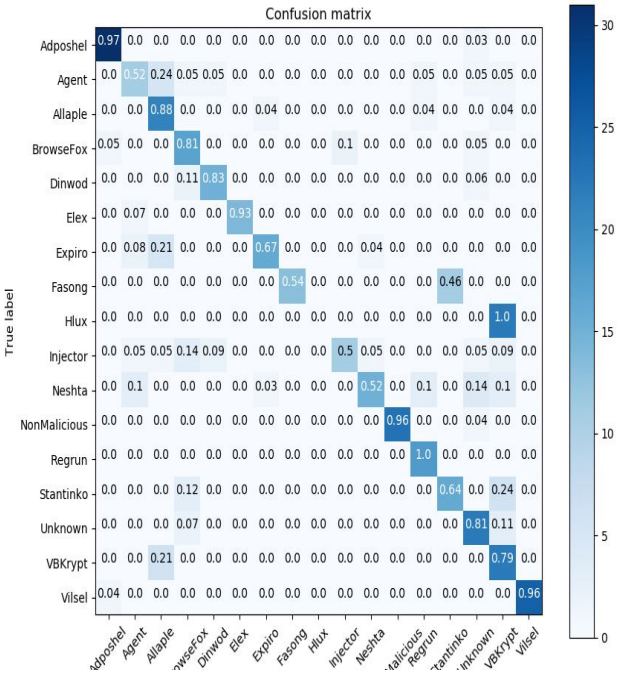

Figure6.2: Confusion Matrix for our approach

7. EXPERIMENTATION AND RESULT

Malicious code invariance based on deep learning uses 17 classes.15 classes are malicious families, 1 is non -malicious class and the last one is the unknown class. Each class consists of more than 300 images. Our model is able to classify the given image.

And there is a provision to convert the .Exe files in to the corresponding coloured image and after converting it, we can predict the class of the image. The output of different classes is showing below.

One shows the output of malicious class, one shows the output of non-malicious class and the other is the output of unknown class. In addition our model provided an effective solution for data imbalance problem among different malware families. Our experimental results on 5,950 images of 17 classes showed that the proposed approach achieved 96.8% accuracy with good detection speed. The new system has overcome most of the difficulties and limitations of the existing system and works according to the design specification given. The developed systems dispense the problem and meet the needs by providing reliable and comprehensive information. All the requirements of the user have been met by the system.

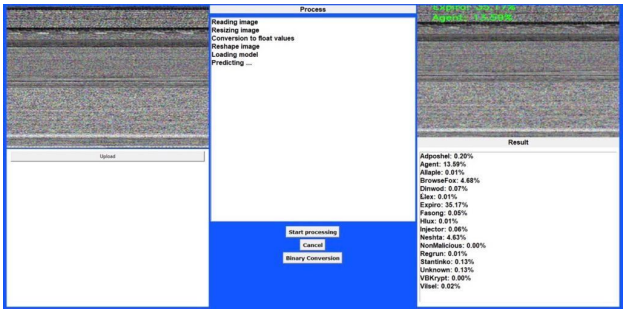

Figure 7.1: Output of malicious class

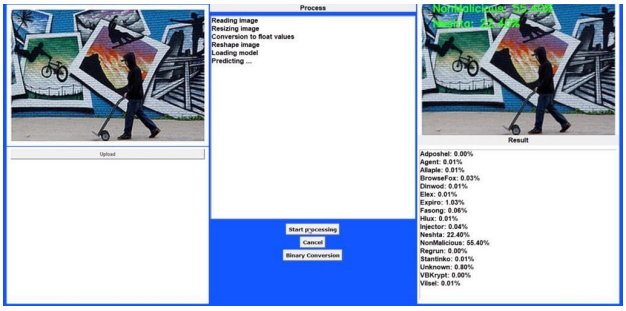

Figure7.3: Output of Non-Malicious class.

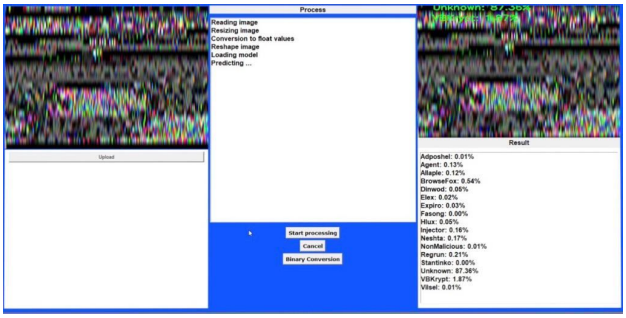

Figure7.2: Output of unknown class

8. CONCLUSION

The malicious code invariance based on deep learning proposed a novel method to improve the detection of malware variants through the application of deep learning along with convolutional neural networks. First, the input images were identified and classified by a convolutional neural network that could extract the features of the malware images automatically. Because of the effectiveness and efficiency of the CNN for identifying malware images, the detection speed of our model was significantly faster than speeds achieved by other approaches because of the effectiveness and efficiency of the convolutional neural network and the deep learning algorithm.

There are other algorithms like back propagation algorithm it works on perceptron, but our approach that is deep learning which consist of a deep neural network so it has higher efficiency than any other methods. Deep learning-based method gained better accuracy than other algorithms.

REFERENCES

[1] H. Gao, Y. Du, and M. Diao. **Quantum-inspired glowworm swarm optimization and its application**. *International Journal of computing Science and Mathematics. International Journal of Computing Science and Mathematics,8(1):91-100, 2017.*

[2] D. D' iaz-pernil, A. Berciano, F. pena-Cantillana, and M. A.Gutierrez Naranjo. **Bio-inspired parallel computing of representative geometrical objects of holes of binary 2d-images.***International Journal of Bio Inspired Computation, 9(2):77-92,2017.*

[3] G. -G. Wang, X. Cai, Z. Cui, G. Min, and J. Chen. **High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm.***IEEE Transactions on Emerging Topics in Computing, 2017.*

[4] Y. Ye, T. Li, D. Adjeroh, and S. S. lyengar. **A survey on malware detection using data mining techniques**. *ACM Computing Surveys (CSUR), 50(3):41, 2017.*

[5] J. Bouvrite**. Notes on convolutional neural networks.***2006.*

[6] X. Cai, X.-z. Gao, and Y. Xue. **Improved bat algorithm with optimal forage strategy and random disturbance strategy.** *International Journal of Bio-Inspired Computation, 8(4):205–214, 2016.*

[7] X. Cai, H. Wang, Z. Cui, J. Cai, Y. Xue, and L. Wang**. Bat algorithm with triangle-flipping strategy for numerical optimization.** *International Journal of Machine Learning and Cybernetics, 9(2):199–215, 2018.*

[8] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant. **Semantics-aware malware detection.** *In 2005 IEEE Symposium on Security and Privacy, pages 32–46. IEEE, 2005.*

[9] Z. Cui, Y. Cao, X. Cai, J. Cai, and J. Chen. **Optimal leach protocol with modified bat algorithm for big data sensing systems in internet of things.***Journal of Parallel and Distributed Computing, 2018. (doi:10.1016/j.jpdc.2017.12.014).* Z. Cui, B. Sun, G. Wang, Y. Xue, and J. Chen. **A novel oriented cuckoo search algorithm to improve dv-hop performance for**

cyber–physical systems. *Journal of Parallel and Distributed Computing, 103:42–52, 2017.*

[10] O. E. David and N. S. Netanyahu. Deepsign: **Deep learning for automatic malware signature generation and classification.** *In Neural Networks (IJCNN), 2015 International Joint Conference on, pages 1–8. IEEE, 2015.*

[11] J. R. Goodall, H. Radwan, and L. Halseth**. Visual analysis of code security**. *In Proceedings of the seventh international symposium on visualization for cyber security, pages 46–51. ACM, 2010.*

[12] K. Han, J. H. Lim, and E. G. Im. **Malware analysis method using visualization of binary files.** *In Proceedings of the 2013 Research in Adaptive and Convergent Systems, pages 317–321. ACM, 2013.*

[13] H. Gao, Y. Du, and M. Diao. **Quantum-inspired glowworm swarm optimization and its application**. *International Journal of computing Science and Mathematics. International Journal of Computing Science and Mathematics,8(1):91-100, 2017.*

[14] D. D' iaz-pernil, A. Berciano, F. pena-Cantillana, and M. A.Gutierrez Naranjo. **Bio-inspired parallel computing of representative geometrical objects of holes of binary 2d-images.***International Journal of Bio Inspired Computation, 9(2):77-92,2017.*

[15] G. -G. Wang, X. Cai, Z. Cui, G. Min, and J. Chen. **High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm.***IEEE Transactions on Emerging Topics in Computing, 2017.*

[16] Q. Miao, R. Liu, Y. Quan, and J. Song. **Remote sensing image fusion based on shearlet and genetic algorithm.***International Journal of Bio-Inspired Computation, 9(4):240–250, 2017.*

[17] A. Moser, C. Kruegel, and E. Kirda. **Limits of static analysis for malware detection.** *In Computer security applications conference, 2007. ACSAC 2007. Twenty-third annual, pages 421–430. IEEE, 2007.*

[18] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath. **Malware images: visualization and automatic classification**. *In Proceedings of the 8th international symposium on visualization for cyber security, page 4. ACM, 2011.*

[19] L. Nataraj, V. Yegneswaran, P. Porras, and J. Zhang. **A comparative assessment of malware classification using binary texture analysis and dynamic analysis.***In Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, pages 21–30. ACM, 2011.*

[20] P. Trinius, T. Holz, J. G¨obel, and F. C. Freiling. **Visual analysis of malware behavior using treemaps and thread graphs.** *In Visualization for Cyber Security, 2009. VizSec 2009. 6th International Workshop on, pages 33–38. IEEE, 2009.*

[21] J. Schmidhuber. **Deep learning in neural networks: An overview.** *Neural networks, 61:85–117, 2015.*

[22] A. Khoshkbarforoushha, R. Ranjan, R. Gaire, E. Abbasnejad, L. Wang, and A. Y. Zomaya**. Distribution based workload modelling of continuous queries in clouds**. *IEEE Transactions on Emerging Topics in Computing, 5(1):120–133, 2017.*