REVIEW ARTICLE

SAMRIDDHI Volume 14, Issue 1, 2022 Print ISSN: 2229-7111 Online ISSN: 2454-5767

A Review on Malware Detection Methods

Jaishri M. Waghmare[1]*, Mayuri M. Chitmogrekar[2]

[1]Associate Professor, Department of Computer Science Engineering, SGGIE&T, Nanded, Maharashtra, India

[2]P.G. Student, Department of Computer Science Engineering, SGGIE&T, Nanded, Maharashtra, India

Abstract

With the expeditious improvement of the internet and information technology, malicious codes and attacks pose a critical threat to internet and computer security. These attacks and malicious codes are increased exponentially and caused huge amounts of financial damage. The potential to recognize several malicious code variants and threats is crucial for preservation across unauthorized access to computer data, information, data theft, security breaches, etc. Recent methods for identifying malicious codes and threats have indicated less precision and deficient speeds. This paper aims to conduct a brief and systematic survey on the malware detection methods based on the soft computing model. This work is expected to help researchers to understand the malware detection technology and the direction of its research development.

Keywords: CNN, Deep Learning, Malware Detection.

SAMRIDDHI : A Journal of Physical Sciences, Engineering and Technology (2022); DOI: 10.18090/samriddhi.v14i01.6

Introduction

In the last several years, the exponential evolution in the area of the internet, computerized data, and communication technology, the expeditious rise of malware play a primary threat to the internet and computer security. The interconnections of the internet grew exponentially, which led to the significant growth of malicious attacks. Malware is any program or software which is deliberately designed to cause harm to computer networks and information security. It includes Viruses, Trojans, Worms, Ransomware, Adware, Spyware, etc. Cybercriminals take advantage of such things.

According to the current Internet security threat report from Symantec (Security solution provider) published in April 2017.[21] Based on this report, India appeared as a vulnerable country in cyber threat, ranked as the third position in 2017. The United States (With 26.61% global threats) was the first most vulnerable country, Second was China (With 10.95% global threats) in 2017, and India (With 5.09% global threats), which was in a third position. These threats are based on eight metrics: malware, ransomware, web attacks, phishing, network attacks, spam, bots, etc.

According to these statistics, India was the second most affected by bots and spam, the third most affected by network assaults, and the fourth most harmed by ransomware. Symantec says that after the United States, India was most affected by targeted attacks between 2015 and 2017. This report also showed that in 2016, 401 million malicious codes were found, which includes malicious code variants as 357 million. The emergence of malware is not only on the computer system and on the internet but also in Smartphone appliances and things related to the internet, which further grows expeditiously. Up to date, malicious code reported as more than 10 thousand and malicious code families up to 68.

Panda Labs reported that whichever malware was detected by their antivirus engine out of that 27% of all malware were newly discovered in 2015.[7] Similarly, every year huge amounts of new malware are created. Therefore, it is important to simplify the detecting procedure of new malware and its variants.

This rapid growth of malware and malicious code variants placed a challenge in cloud computing for malware detection.[14,15] In terms of security protection from threats, exposing malware variants is challenging. In addition, to detect the malware or harmful code, this method primarily consists of the following analysis methods Static and dynamic detection methods. In static analysis, it observed only code without actually running the program. It is debugging a software program without executing the software program. It analyses the malicious code or malware. For example, the disassembling code needs knowledge of assembly language and requires a good disassembler. IDA pro. Static detection works by disassemblers and network analyzers, which may

Corresponding Author: Jaishri M. Waghmare, Associate Professor, Department of Computer Science Engineering, SGGIE&T, Nanded, Maharashtra, India, e-mail: jmwaghmare@sggs.ac.in

How to cite this article: Waghmare, J.M., & Chitmogrekar, M.M. (2022). A Review on Malware Detection Methods. *SAMRIDDHI : A Journal of Physical Sciences, Engineering and Technology,* 14(1), 38-43.

Source of support: Nil

Conflict of interest: None

be used to examine malware without actually running it to learn how it works.

Another approach is dynamic analysis. It executes the suspected malicious code to analyze its behavior functionality and observes the technical indicators. Once it executes the malicious content in a secure domain like a sandbox now, it's easy to analyze the behavior. The sandbox allows security professionals to observe malware in action without risking infecting their own computer. Both techniques rely on a feature-based revealing approach. The malicious content is discovered by assessing the given qualities, which were discovered by collecting word-based characteristics or behavioral elements. After extraction of features, we found some quantifiable property which is helpful to detect the malware. Several research attempts to detect malware using data mining methodologies have been published recently. These procedures efficiently analyze earlier unknown malware patterns recognizing families of malware that belong to different malicious samples.

Figure 1 shows malicious code identification by using the data mining technique. These techniques have efficient ways to classify previously unknown malware samples and identify malicious samples from malware families. In the first step, numerous features such as API calls, string, control graphs, and program behaviors are collected statically or dynamically to get the features of file samples. In the second step, most useful algorithms such as clustering or classification are used to classify the above file samples into different graphs dynamically. This approach for malware detection is well organized, and it has less false positive rate as compared to other detection methods. Unfortunately, the methods which depend on the feature

Analysis such as static and dynamic methods are discomposed. The obfuscation technique converts malware binary data into a uniquely structured binary. Due to this, obstruction can occur on the efficacy of static code evaluation.

Dynamic simulation avoids some kinds of malicious code variants since the implementation platform does not obey

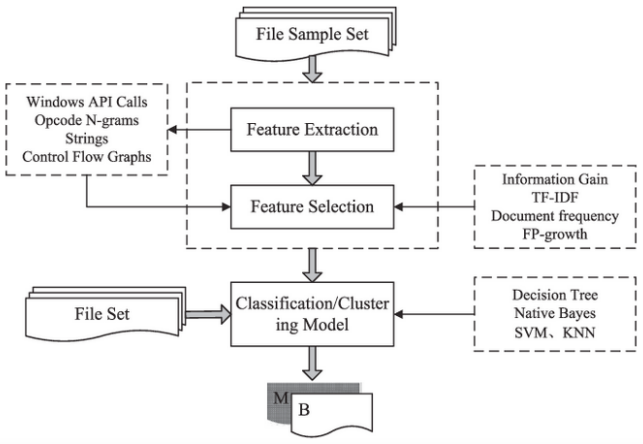

Figure 1: Malware detection using data mining

the guidelines. Dynamic characteristics evaluation is also conflicted by many types of attempts that are designed to construct irrelevant results. Instead of concentrating on hidden aspects for malware categorization, Natraj. *et al*.[19] offers malware visualization that provides a novel technique for analyzing malware and visualization based on image processing. The malware can appear in the form of zeros and ones. This vector representation can be reshaped into a matrix form, and this process is visualized as a gray-scale image. Based on the images, we found many similarities in the images for the malware that belongs to the same family. We extracted the image features and proposed a classification method that gives good accuracy for malware classification.

Malware Detection Methods

Malware detection methods are fundamentally categorized in different categories from other points of view. In this survey paper, we assume three categories for malware detection methods. These are recurrent neural networks, convolutional neural networks, and deep learning-based methods.

Recurrent Neural Networks

Recurrent neural networks (RNN) have been the best option of the most profound learning-based malware identification frameworks. RNN-based MDS creators think that the consecutive information contained inside machine directions and API calls obtained from the malware is helpful to distinguish malware, just like content order.[12,18] An average RNN based malware discovery framework encodes every API call/guidance as a one-hot vector of measurement M, where M is the quantity of aggregate Programming interface calls/directions. Given that API calls/directions are named from 0 to M – 1, for an API call/guidance named i, all numbers in the vector are 0 except record i, where the number is 1. A succession of implanted API calls is assembled into a grid and is taken care of into the RNN for grouping. A few varieties of RNN have been assessed,[1] appearing LSTM network with max-pooling and strategic relapse as characterization beats character-level CNN.

One of RNN's biggest flaws is that it can only learn one language environment or pattern. Attackers can make use of this flaw by using a replacement RNN to anticipate the language that a specific RNN-based malware detection system has learned and then using another RNN to write adversarial code to get through malware detection systems.

Convolutional Neural Networks

The convolutional neural network (CNN) collects hierarchical local characteristics from data sets independent of their location, making it a popular tool for picture categorization. Despite this, we believe CNN is a better choice for malware detection than RNN. Applying repetitive API calls or computer commands to a picture correlates to characteristic interpretation or displacement, which a CNN may be trained to identify. A hand in a picture, for example, remains a

hand regardless of its location or alignment, but a hand in a text segment might be a noun or a verb based on the circumstances. By putting "the" in front of the word "hand" in RNN-based MDS, attackers may mislead the MDS into thinking it is a noun when it should be a verb. Adding text across the picture, on the other hand, would alter or interpret the characteristic, which CNN can readily identify.

Sadly, arranging malware with CNN has not been widely examined in the writing. Yuan *et al.* proposed an MDS that first concentrates a bunch of highlights with crossbreed examination also uses CNN to arrange the separated element vectors.[20] Basically, utilizing only a CNN with API calls or machine instructions was infrequently done.

Malware Images

First, we convert benign and malware files into images to detect malware.[1] In common, there are numerous methods for converting binary coding data into graphics. In this study, the visualization of executable malware raw information files[19] was employed. A malware binary file or string can be broken down into various categories 8-bit substring, and each substring is treated as a single image particle. Because an undefined value is used to express an 8 bits string which can be in the range of 0 to 255, for example, If an offered bit string is 0011001010110011, the process (Figure 2) is further 0011001010110011→00110010,10110011→50,179. An 8 bit binary number is represented as B = (b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0) Which can be converted into decimal number D as follows:

$$D = b_0 \times 2^0 + b_1 \times 2^1 + b_2 \times 2^2 + b_3 \times 2^3 + b_4 \times 2^4 + b_5 \times 2^5 + b_6 \times 2^6 + b_7 \times 2^7 \quad ...(1)$$

As compared to API sequences, we can get fast images from the files. We have to run each file for several minutes to get API sequences. To prevent malware from running in real-time, we must check whether the file is malicious or not for several seconds. So we observed that, for malware detection, malware image is good as compared to extracting API sequences (Figure 3).

Here each image size is 64KB. The remainder is discarded when the file size is greater than 64KB. When the file size is less than 64KB, the remainder of the image is padded with zero. If we use 256 * 256 images for the detection of malware

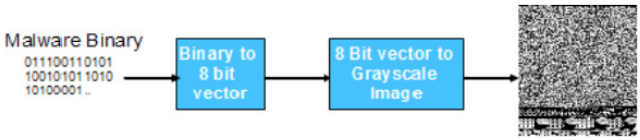

Figure 2: Visualizing malware as a gray-scale image.

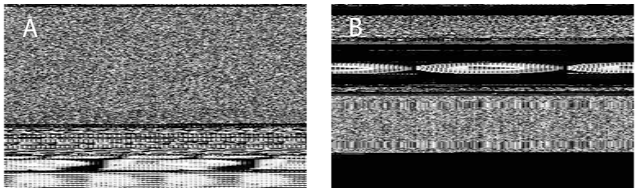

Figure 3: A) Normal file image; B) Malware image

in deep learning, it runs out of memory. So that we down-sample of the images to 32*32 images. Once we convert data into binary, the malware string data can be converted into decimal form. The 1-D array is represented as a gray-scale image for a particular width. Finally, the malicious data form is represented as a gray-scale image. Based on experimental observation, Table 1 shows some image widths for several file sizes.

As we can see, the similar malware group has visibly similar images and different images belonging to different families. For example, Figure 4(a) illustrates four alternative forms of the malware assistant, which is known as the agent. FY1 family. The length of every agent is distinct, but still, they have the same functionality since new malware variations are frequently developed from the old software. In addition to this, if families have similarities, pictures can show their dissimilarities easily. When compared to the swizzor.gen! E family has black lines. Motivated by the similarities in malware pictures, we can identify and detect harmful software, which can be done by using image recognition methods. To recognize malware images, We employed CNN in this current research.

Malware Image Classification Based on CNN

By using CNN models, researchers have classified malware. First, it is used images. Convolutional neural networks have developed quickly in the area of speech evaluation and picture identification. The number of network models is reduced due to their local perception and weight-sharing network structure properties. If the input is multidimensional, it is a good advantage for CNN. For our work, the input is an image of the network. When opposed to standard

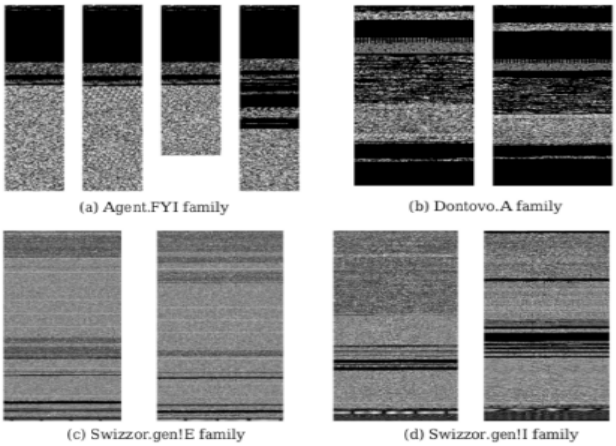

Figure 4: Different families of malware.

Table 1: Image width for different file sizes

File size	Image width	File size	Image width
<10 kB	32	100 kB~200 kB	384
10 kB~30 kB	64	200 kB~500 kB	512
30 kB~60 kB	128	500 kB~1000 kB	768
60 kB~100 kB	256	>1000 kB	1024

recognition algorithms, this technique is never obstructed by sophisticated feature collection and data reconstruction. Convolutional neural networks are powerful in terms of image deformation, such as scale, rotation, transformation. Also, convolutional networks are multi-layer perceptrons designed for recognizing 2-D shapes.

To classify malware in this research, we developed a convolutional neural network. Figure 5 gives a description of malware categorization based on a convolutional neural network. The convolutional neural network contains various fragments for gray-scale image recognition, as shown in Figure 5. First part is an input layer, which can put training data or images into the neural network. Next are sub-sampling and convolution layers. The former layer is used to reduce noise and enhance single characteristics. There are various fully connected layers that convert 2-D features into 1-D features. Finally, according to the characteristics of malware images, the classifier recognizes and ranks malware pictures into distinct families. The detailed information of each layer and its iterative formula can be given as follows:

1. Convolution Layer

A biologically inspired method, the convolutional neural network generally consists of convolution and sub-sampling operations. The last architecture of CNN consists of the fully connected layer. The convolution layer can minimize the number of picture parameters while keeping the essential invariance aspects such as rotation, translation, and scale invariance. This method can avoid the overfitting problem efficiently while simultaneously enhancing the model's ability to generalize. Traditional artificial neural networks' extensive training and feature collection are also avoided. A convolution layer is made up of a collection of kernel layers. The previous layer's feature map is convolved using learnable kernels in this layer. Then a non-linear transformation is applied using the activation function. So that it can form an output feature map, here, each output feature map is related to numerous input maps. CNN can automatically learn a unique set of features by performing convolution operations.

The iterative formula of convolution operation is given as follows:

$$x_j^l = f\Big(\sum_{i \in M_j} x_j^{l-1} \times k_{ij}^l + b_j^l\Big) \tag{2}$$

Where represents a collection of input maps, represents convolution kernel, which is used for the connection between *i* th input feature map and *j* th output feature map. shows bias corresponding to *j* th feature map, and finally, f is an activation function. Convolutional neural networks learn and optimize the best sets of convolutional kernels as compared to traditional machine learning approaches.

By including such kernels with proper pooling operation, which extracts applicable on appropriate features to perform tasks such as image classification, segmentation, or recognition. Convolutional neural networks are used in the segmentation process. In that, each image pixel is individually classified for image classification.[13] To use such an approach, we need first to convert malware binary data into an image as described earlier. By using a simple CNN model Cui *et al*.[6]

Detected versions of code that are harmful after converting to gray-scale and presenting it with patches. Which are extracted from a particular pixel.

We may now repeat the above calculation for each map j in the convolution layer that can be paired with the appropriate map from the sub-sampling layer.

$$\delta_j^l = \beta_j^{l+1} up\big(\delta_j^{l+1}\big) \circ f'(u^l) \tag{3}$$

Where up (·) is a sampling operation, W denotes convolution kernel, and *l +1* is the sampling layer.

The partial derivative of convolution kernel k and error cost function with regard to b is shown as follows:

$$\frac{\partial E}{\partial b_j} = \sum_{u,v} \big(\delta_j^l\big)_{u,v} \tag{4}$$

The bias gradient may be determined immediately by simply adding all of the elements in δ_j^l. The kernel weights are calculated via backpropagation gradients. The gradient of the specified weight is then averaged over all connections that reference it.

$$\frac{\partial E}{\partial k_{ij}^l} = \sum_{u,v} \big(\delta_j^l\big) u, v\big(p_i^{l-1}\big)_{u,v} \tag{5}$$

Where $(p_i^{l-1})_{u,v}$ denotes patch for every convolution of and k_{ij}^l,*u,v* is the center of the patch.

2. Sub-sampling Layer

The sub-sampling layer is also called the pooling layer. This is never altered by backward propagation. This sub-sampling layer can reduce the impact of image warping—for example, translation, rotation, scale, etc. Dimension of the feature map is reduced by using this layer, and also it improves the accuracy of the model and overfitting problem avoids. The sub-sampling layer provides a downsampled version of the

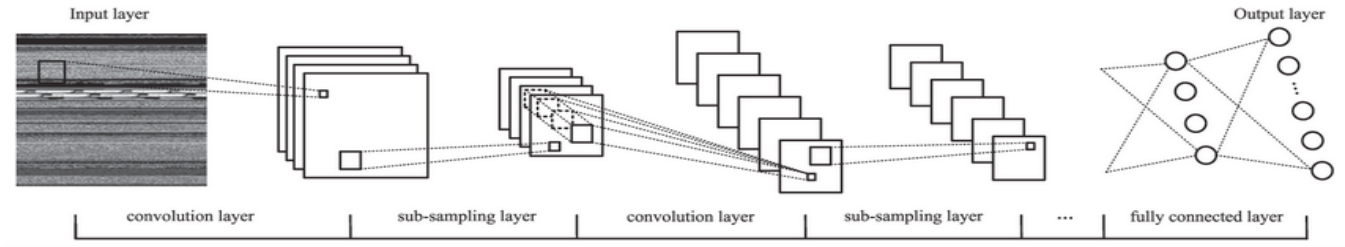

Figure 5: CNN architecture

input maps; there will be the same number of output feature maps if there are N input feature maps.

However, the output feature map graph is smaller than the graph of the input feature map.

For every result of the sampling layer, the feature map in the CNN is illustrated as follows:

$$x_j^l = f(down(x_j^{l-1}) + b_j^l \quad (6)$$

Where is indicated as sub-sampling function and b is represented as bias.

The sensitivity is determined as follows:

$$x_j^l = f(down(x_j^{l-1}) + b_j^l \quad (7)$$

With respect to bias b, the partial derivative of the erroneous objective functions can be stated as follows:

$$\frac{\partial E}{\partial b_j} = \sum_{u,v} (\delta_j^l)_{u,v} \quad (8)$$

Until now, we've been able to obtain CNN's weight updating formula and use it to categorize malicious data.

A structured network requires a fixed-length picture, even if the connection layer's connection matrix W remains constant in size during training. If the input and output neurons have sizes of 30 and 60, correspondingly, the weight matrix size must be 30 from the convolution layer to the complete connection layer (30, 60). The malware file's size is not fixed, regrettably, but depends on the length of the software.

3. Residual Network or ResNet 34

A final important innovation in convolutional neural nets that we will review was proposed by Kaiming He, *et al.*[22] The authors proposed a very deep model called a Residual Network in the paper (Figure 6).

Resnet34 is a state-of-the-art image categorization model with 34 layers of convolutional neural networks. This is a model that has been pre-trained on the ImageNet dataset, which has 100,000+ pictures divided into 200 categories. It is, however, distinct from typical neural networks in that it employs the residuals from each layer in the succeeding linked layers (similar to residual neural networks used for text prediction).

Resnet34's architecture where the 34 layers and the residuals from one layer to another are visualized.

The Resnet34 model first needs to load the model into a GPU device after visualizing model architecture. To train the model, initially, ResNet layers need to freeze and trained for some epochs. It will train the final layer to start classifying the images. Next, by applying unfreeze full network is trained. The accuracy starts to increase steeply after 3rd epoch. This trend can be visualized by the training and validation losses.

Deep Learning Model

Deep learning is widely used in various fields, especially in the area of image recognition. It shows good results. Deep learning contains three layers: input, output, and several hidden layers. When the deep learning model is trained with some training data, it will automatically extract the features from the data and classify the data into various classes. For example, if image recognition is the purpose of deep learning, it can classify the images into various image classes, such as a dog, cat, flowers, etc. To classify or detect malware, a deep learning approach is widely used. Training data of malware is required to detect or classify the malware.

Deep learning emerged widely in recent years from the concept of artificial neural network. To solve the complex problems, neural networks are used to estimate complex functions by studying deep non-linear networks. Deep learning is more powerful as compared to backpropagation. Deep learning uses neural networks to replicate the individual's intelligence learning process. From a sample set, deep understanding has the capability to study the necessary features of data sets. Deep learning plays a very important role. It is a powerful tool of AI so it is applicable in many areas such as image recognition, speech identification, and handwritten identification. Since it has a strong capability to learn characteristics and detect malware, many scholars have applied deep learning technology. By using deep learning technology, Yuan *et al.*[30] proposed and implemented a malware detection prototype as an online basis, named Droid-sec.

Their idea accomplished good correctness by learning the characteristics from android apps which are statically and dynamically analyzed. Another approach which is presented by David *et al.*[7] proposed the same but large fascinating technique. There is no need for different types of malware behavior for this method.

Choi *et al.*[11] proposed a malware detection technique based on malware images and deep learning. To begin, the writers created pictures from both benign and malicious files. Second, the authors utilized a deep learning approach based on convolutional neural networks to identify malware since this model learns characteristics from pictures. The deep learning model consists of three convolution layers, a pooling layer, and two fully connected layers. Deep learning is resilient to tiny changes in the pictures because of the pooling layer.

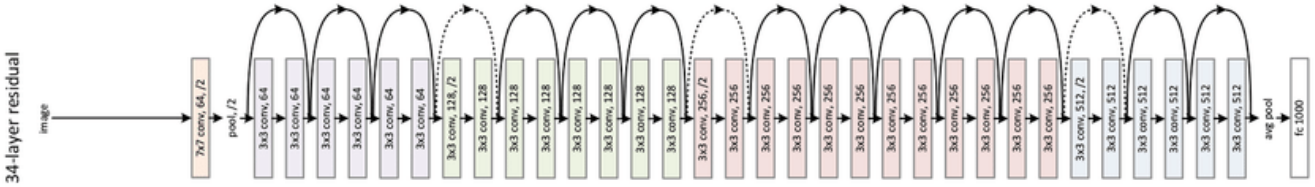

Figure 6: Deep learning model

Deep belief networks were used to classify malware automatically and develop malware signatures. Compared to traditional malware detection approaches, the aforesaid strategy gives a high correctness rate for detecting new malware variants. Regrettably, approaches continue to depend on the evaluation of properties collected by static and dynamic study. As a result, they are still subject to the restrictions of characteristic collection to a greater or lower level. To resolve the above issue, we used a CNN, which automatically classifies and extracts malware features. For that, we first need to convert malware data into images and then apply the deep learning module.

Conclusion

Malware causes a serious threat to computer systems, networks, the internet, and information. This survey paper briefly reviewed malware types and presented three malware detection techniques based on the soft computing model used. These are malware detection using RNNs, CNNs, deep learning models, and Resnet34. The objective of the survey is to provide a quick idea about present methods and their procedures, which could be suitable for further studies and to develop other malware detection techniques.

References

[1] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, Malware Images: Virtualization and Automatic Classification, ACM VizSec, 2011

[2] LIU Wu, REN Ping,LIU Ke, DUAN Hai-xin. Behaviour-based malicious code evaluation and identification. In *Proceeding First International conference on Complexity and Data Mining,* 2011.

[3] Z. Yuan, Y. Lu, Z. Wang, and Y. Xue. Droid-sec: deep learning in android malware detection. In *ACM SIGCOMM Computer Communication Review,* ACM, 2014.

[4] 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,* IEEE, 2005.

[5] Hauri, https://www.hauri.co.kr.

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

[7] Panda, http://www.pandasecurity.com/mediacenter/press-releases/all-recorded-malware-appeared-in-2015.

[8] D. Diaz-Pernil, A.Berciano, F. Pena-Cantillana, and M. A. Gutierrez-Naranjo. Bio inspired parallel computing of geometrical things that are representational geometrical objects of binary holes 2-d images, *International Journal of Bio-Inspired Computation,* 2017.

[9] I. Yoo. Visualizing windows executable viruses using self-organizing maps. In *Proceeding of the 2004 ACM workshop on visualization and data mining for computer security,* ACM, 2004.

[10] J. R. Goodall, H. Radwan, and L. Halseth. Visual analysis of code security. *In Proceedings of the seventh international Symposium on visualization of cyber security.* ACM, 2010.

[11] 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,* ACM, 2013.

[12] T. Isohara, K. Takemori, and A. Kubota. Kernel based behavioural analysis for android malware detection. In *2011 Seventh International Conference on Computational Intelligence and Security.* IEEE, 2011.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrel. Caffe: Convolutional architecture for fast feature embedding. In *Proceeding of the 2nd ACM International Conference on Multimedia,* ACM, 2014.

[14] A. Khoshkbarforoushha, A. Khosravian, and R. Ranjan. Elasticity management of streaming data analytics flows on clouds. *Journal of Computer and System Sciences,* 2017.

[15] 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,* 2017.

[16] P. Li, J. Zhao, Z. Xie, W. Li, and L. Lv. General central firefly algorithm based on different learning time. *International Journal of Computing Science and Mathematics ,* 2017.

[17] P. Trinius, T. Holz, J. Gobel, and F. C. Freiling. Visual analysis of malware behaviour using treemaps and thread graphs. In *Visualization of Cyber Security, 2009. VizSec 2009. 6th International Workshop on , IEEE, 2009.*

[18] 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 ,* IEEE, 2007.

[19] L. Nataraj, S Karthikeyan, G. Jacob, and B. Malware Images: visualization and automatic classification. In processing of the 8th international symposium on visualization for cyber security, 2011.

[20] L. Nataraj, V.Yegneswaran, P.Porras, and J. Zhang. A comparative assessment of malware classification using sequential text evaluation and dynamic evaluation. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence,* ACM, 2011.

[21] Symantec. Internet security threat report, 2017.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Deep Residual Learning for Image Recognition. proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.