Recognition of Vindictive Code Variations Dependent on Deep Learning

Srinivas Achar G[1], Rashmi K R[2], Rakshitha P[3] , S Vino David[4]
Department of Computer Science Engineering
[1,2,3,4] Atria Institute of Technology

Abstract- *With the improvement of the Web, malignant code assaults have expanded exponentially, with vindictive code variations positioning as a key danger to Web security. The capacity to identify variations of vindictive code is basic for insurance against security ruptures, information robbery, and different perils. Current techniques for perceiving malignant code have exhibited poor location exactness and low discovery speeds. This paper proposed a novel technique that utilized deep learning figuring out how to improve the discovery of malware variations. In earlier research, Deep learning exhibited astounding execution in picture acknowledgment. To actualize our proposed identification strategy, we changed over the malignant code into grayscale pictures. At that point the pictures were recognized and characterized utilizing a convolutional neural system (CNN) that could extricate the highlights of the malware pictures consequently. Also, we used a bat calculation to address the information lopsidedness among various malware families. To test our methodology, we directed a progression of trials on malware picture information from Vision Exploration Lab. The test results showed that our model accomplished great precision and speed as contrasted and other malware identification models.*

Keywords- Malware variants, Grayscale image, Deep learning, Convolution neural network, Bat algorithm

I. INTRODUCTION

With the fast improvement of data innovation, the exponential development of malevolent code has turned out to be one of the fundamental dangers to Web security. An ongoing report from Symantec demonstrated that 401 million malignant codes were found in 2016, including 357 million new noxious code variations [22]. The presence of malware in cell phones and the web of things (IoT) additionally developed quickly. To date, 68 new noxious code families and in excess of 10 thousand malignant codes have been accounted for. This development has represented a test for pernicious code identification in distributed computing [14], [15].

As a key piece of security insurance, revealing pernicious code variations is especially testing. Malware discovery techniques comprise principally of two kinds of methodologies: static recognition and dynamic location. Static location works by dismantling the malware code and investigating its execution rationale. Dynamic identification dissects the conduct of noxious code by executing the code in a safe virtual condition or sandbox. Both static and dynamic location are include based discovery strategies. To begin with, the printed or conduct highlights of the malignant code are removed, and afterward the noxious code is identified or classified by investigating these separated highlights. As of late, a few researchers have utilized information mining strategies to dissect the highlights of malignant code [28]. This methodology has turned into the backbone of malware location since it is exceedingly efficient and has a low rate of false positives contrasted and conventional heuristic-based identification strategies. Fig. 1 illustrates the process of detecting malicious code using data mining.

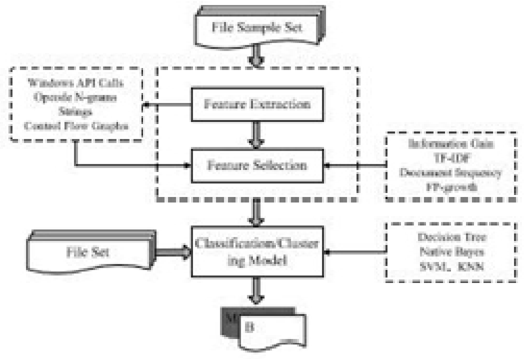


Fig. 1. Malicious code detection based on data mining

Challenges: Malware identification strategies depend mostly on examination of the highlights of pernicious codes (e.g., static highlights, dynamic highlights). All the more dominant identification strategies dependent on different AI methods additionally utilize these highlights to reveal pernicious codes or their varieties. In any case, these methodologies become less successful when identifying vindictive code variations or obscure malware. The malware representation strategy can deal with code jumbling issues, yet it experiences the high time cost required for complex picture surface component extraction (e.g., Substance, GLCM). Besides, these component extraction techniques likewise exhibit low-efficiency when presented to vast datasets. The test for structure malware recognition models is to find a methods for separating highlights viably and naturally. Additionally, the information irregularity issue forces another test. Of the extensive amount of malware produced every year, a generous part incorporates variations that have a place with existing noxious code families or gatherings. For the most part, the quantity of vindictive code varieties contrasts enormously among different code families. The test is to manufacture an all inclusive identification demonstrate that can manage the gigantic volume of varieties, so it can function admirably crosswise over malware families.

Contributions. To address the above challenges, this paper offered the following contributions:

- We introduced a technique for converting a malware binary to an image, thereby transforming malware detection into an image classification problem.
- We proposed a novel method for detecting malware variants based on a convolutional neural network (CNN).
- To resolve the data imbalance problem among different malware families, we designed an effective data equilibrium approach based on the bat algorithm.
- Extensive experimental results demonstrated that our proposed method was an effective and efficient approach for malware detection.

II. RELATED WORK

In this section, we present the related research regarding malware, including malware detection based on feature analysis, malicious code visualization, image processing techniques for malware detection, and malware detection based on deep learning.

A. Malware Detection based on Feature Analysis

As portrayed already in this paper, there are two fundamental classes of highlight examination strategies for malware discovery: static investigation and dynamic examination. Regarding static investigation, a few techniques have been advanced by breaking down the codes. For instance, Isohara et al. [12] built up an identification framework utilizing bit conduct examination that performed well distinguishing the vindictive practices of obscure applications. Be that as it may, the static strategy is effectively deluded by jumbling methods [18].

To take care of this issue, Christodorescu et al. [4] proposed a novel malware-discovery calculation that utilized follow semantics to portray the conduct of malware. This strategy demonstrated viable in fighting the obscurity of directions (e.g., improvement of guidelines, addition of trash code, register redistribution). In any case, the methodology was restricted to include extraction and investigation at the guidance level. Also, the example coordinating was unpredictable. Dynamic examination screens and breaks down the runtime qualities of uses (applications) in view of evaluation of practices, for example, getting to private Page | 2

information and utilizing limited Programming interface calls. Given this data, a conduct display is set up to identify vindictive code. Such procedures accomplished improved discovery execution, yet they were still tested by the assortment of countermeasures created to produce questionable outcomes [20]. Moreover, dynamic investigation is tedious as a result of the vast measure of computational overhead, prompting low efficiency when presented to a substantial dataset.

B. Malicious Code Visualization

Currently, many tools can visualize and manipulate binary data, e.g., common text editors and binary editors. Several studies have proposed utilization of malware visualization [24]. Yooetal. [29] employed self-organizing maps to visualize computer viruses. A similar but richer approach was proposed by Triniusetal.[23],who used two visualization techniques—tree maps and thread graphs—to detect and classify malware. Rather than a single detection result, Goodalletal. [10] aggregated the results of different malware analysis tools into a visual environment, providing an increase in the vulnerability of detection coverage of single malware tools.

C. Image Processing Techniques for Malware Detection

Once the malware has been imagined as grayscale pictures, malware location can be changed over into a picture acknowledgment issue. In [19], Nataraj et al. utilized a Substance calculation to extricate the highlights of malware pictures. In any case, the Substance calculation was tedious. As of late, increasingly amazing picture handling strategies have been proposed. Daniel et al. [8] built up a bio-motivated parallel execution for finding the agent geometrical objects of the homology bunches in a twofold 2D picture. For picture combination, Miao et al. [17] proposed a picture combination calculation dependent on shearlet and hereditary calculation. In their methodology, a hereditary calculation is utilized to upgrade the weighted factors in the combination rule.

Exploratory outcomes showed their technique could obtain preferable combination quality over different strategies. These customary methodologies, be that as it may, were tested by the high time cost required for complex picture surface component extraction. To address this test, we utilized profound figuring out how to recognize and order pictures efficiently. In the following subsection, we present our exploration on malware location dependent on Deep learning.

D. Malware Detection based on Deep Learning

Deep learning [21] 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 (BP), 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 widely in many fields, such as recognition of handwritten numerals, speech recognition, and image recognition. Because of it powerful ability to learn features, many scholars have applied deep learning to malware detection. Using deep learning techniques, Yuan et al. [30] designed and implemented an online malware detection prototype system, named Droid-Sec. Their model achieved high accuracy by learning the features extracted from both static analysis and dynamic analysis of Android apps. David et al. [7] presented a similar but more compelling method that did not need the type of malware behavior. Their work was based on a deep belief network (DBN) for automatic malware signature generation and classification. Compared with conventional signature methods for malware detection, their approach demonstrated increased accuracy for detecting new malware variants.

III. MALWARE DETECTION BASED ON A CONVOLUTIONAL NEURAL NETWORK

This area introduces our improved noxious code variation recognition strategy dependent on a CNN, which notwithstanding: (1) the vindictive code mapping as the grayscale picture; and (2) the CNN structure for grayscale picture location. Fig. 2 shows a diagram of these two procedures. In the first place, the double files of malevolent code are changed into the grayscale pictures. Next, the convolution neural system is utilized to recognize and order the pictures. As indicated by the aftereffects of picture classification, we understood the programmed acknowledgment and classification of noxious programming.

A. Binary Malware to Gray Image

In general, there are several ways to transform binary code into images. In this paper, we used the visualization of executable malware binary files [19]. A malware binary bit string can be split into a number of substrings that are 8 bits in length. Each of these substrings can be seen as a pixel, because the 8 bits can be interpreted as an unsigned integer in the range 0 to 255.

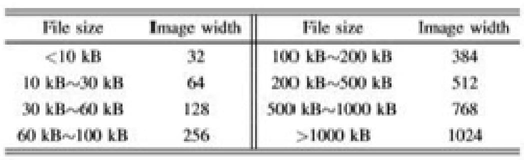
For example, if a bit string is 0110000010101100, 0110000010101100→01100000,10101100→96,172.

An eight-bit binary number B=(b7,b6,b5,b4,b3,b2,b1,b0) can be converted into a decimal number I as follows:
I =b0 ∗20 + b1 ∗21 + b2 ∗22 + b3 ∗23 + b4 ∗24 +
b5 ∗25 + b6 ∗26 + b7 ∗27

(1)

After binary conversion, the binary malware bit string has been converted into a 1-D vector of decimal numbers. According to a specified width, this one-dimensional array can be treated as a 2-D matrix of a certain width. Finally, the malicious code matrix is interpreted as a grayscale image. For simplicity, the width of the image is fixed, and the height of the image varies depending on the size of the file. Table I, taken from [19], presents some recommended image widths for various file sizes, based on empirical observations.

TABLE I
IMAGE WIDTH FOR VARIOUS 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

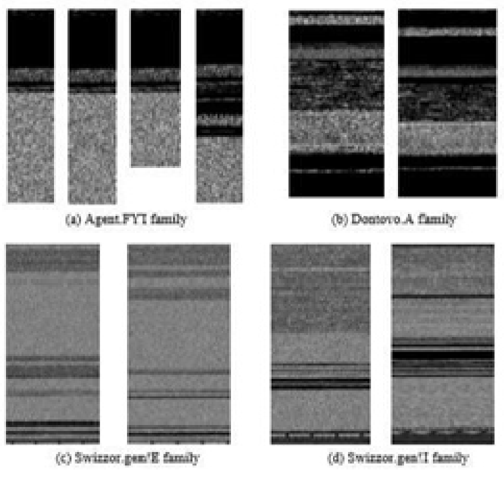


Fig. 3. Illustration of malware images

Fig. 3 shows examples of malware images from different families [19].

B. Malware Image Classification based on Convolutional Neural Network

In this paper, we developed a CNN to classify malware. First 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 reduce noise. The latter can reduce the amount of data processing while retaining useful information. Then there are several fully connected layers that convert a two-dimensional feature into a one-dimensional feature that conforms to the classifier criteria. Finally, the classifier identifies and sorts the malware images into different families according to their characteristics. The detailed iterative formula of each layer can be given as follows:

(1) Convolution layers

The convolution layer can reduce the number of image parameters while preserving the main features, termed invariance, including translation invariance, rotation invariance, and scale invariance This area introduces our improved noxious code variation recognition strategy dependent on a CNN, which notwithstanding: (1) the vindictive code mapping as the grayscale picture; and (2) the CNN structure for grayscale picture location. Fig. 2 shows a diagram of these two procedures. In the first place, the double files of malevolent code are changed into the grayscale pictures. Next, the convolution neural system is utilized to recognize and order the pictures. As indicated by the aftereffects of picture classification, we understood the programmed acknowledgment and classification of noxious programming.
xl j = f(X i∈Mj
xl−1 j ∗klij + bl j) (2)

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, bl j is the bias corresponding to jth feature map, and f is the activation function.

(2) Sub-sampling layers

The sub-sampling layer is also called the pooling layer. Generally, its convolution kernels are the maximum or mean of patch (maximum pooling, average pooling), and are not modified by backward propagation. This layer can weaken the influence of image deformation (e.g., translation, scale, rotation). It reduces 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:

xl j = f(down(xl−1 j) + bl j) ………………..(6)

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

δl j = δl+1 j Wl+1 j ◦f0(ul) …………(7)

So far, we can get the weight updating formula of the CNN, and use it to classify malware images. However, in the CNN, a fixed size image is needed for a structured network because that the connection matrix W of the full connection layer is a fixed size after training. For example, if the input Page | 4

and output sizes are 30 and 20 neurons, respectively, from the convolution layer to the full connection layer, then the size of the weight matrix must be (30,20). Unfortunately, the image size of malware is not fixed, but varies with the size of the software. Therefore, we cannot input these malware images directly into a CNN.

IV. MALWARE IMAGE DATA EQUILIBRIUM

Great information is the way to AI and profound learning. Unpleasant example information will influence the nature of the model. Conversely, a model prepared by excellent information is progressively vigorous (abstains from overfitting), and can accelerate show preparing. Malevolent programming more often than not comprises of a few families, and the quantity of tests of every family differs broadly. This variety in information will prompt low precision and poor vigor of the preparation show. To address this issue, we utilized an information enlargement procedure to improve information quality. We proposed an information adjustment strategy dependent on a keen streamlining calculation (i.e., dynamic resampling dependent on a bat calculation) to determine the issue of imbalanced information between various families.

A. Image Data Augmentation Technology

In deep learning, to avoid overfitting, we usually need to enter sufficient data to train the model. If the data sample is relatively 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 effectively, 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 remain 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.

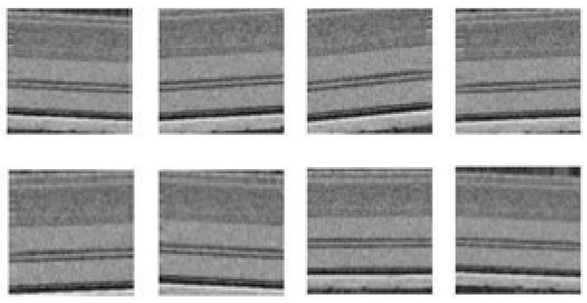


Fig. 6. Images generated by data augmentation for Swizzor.gen!E family

Let A = [a1,a2,...,a8] be a collection of these techniques. Mi=am,...,an is the sequence of operations, and i is the length of M. M2 = a1,a2 represents data augmentation by using rotation transformation and flip transformation. Typically, each augmentation technique has a weight λi, so the sequence of operations of weighted data augmentation techniques can be given b

y Mi = λmam,...,λnan ………………..(9)

where λm is the weight of the augmentation technology, and am

⊆ A. Usually it is necessary to perform multiple data augmentations. For a grayscale image matrix m, Mk(m) represents k augmentations. Fig. 6 shows some images generated by data augmentation for the malware Swizzor.gen!E family. Clearly, most of the newly generated images retained the original texture features. However, a few images lost some information through inappropriate transition (i.e., overrotation). The images viewed in the lower-left corner lack a texture feature (black dots at the bottom) of the Swizzor.gen!E family In classification, resampling is a simple method used to This approach works the best in guidance of fellow researchers. handle an unbalanced training set. The resampling method converts the imbalanced data set into a balanced dataset by In this the authors continuously receives or asks inputs from processing the training set. It consists of two implementations: their fellows. It enriches the information pool of your paper versampling and undersampling. Oversampling is used to make with expert comments or up gradations. And the researcher multiple copies of subsets. Undersampling is used to remove feels confident about their work and takes a jump to start the some samples from the sample set (i.e., to select only some paper writing.samples). For instance, take the Allaple.A family (2,949 images)

B. Data Equalization based on a Bat Algorithm and the Skintrim.N family (80 images). If the training set for the former consists of 1,000 samples, we can choose 500 of them. Imbalanced information is a typical issue in AI, and it can For the latter, because of the insufficient number of data, we influence the exactness of the model in classification issues. In must repeat sampling until the training sample reaches 500. this paper, the malware picture information was extremely Because the training sample is less than the total sample, the uneven, as appeared in Fig. 7. The most extreme proportion was trained classification model for the Skintrim.N family will suffer about 36:1. For instance, the Allaple.A family had 2,949 from over fitting. pictures, while the Skintrim.N family had 80. Such contrasts Therefore, the proper proportion of samples is have a major effect on a CNN's act in picture classification. The important for the training set. Unfortunately, it is difficult to find characterizing model prepared by the information of the two an optimal proportion due to the complex weight combinations families may accomplish 97% exactness when every one of the with dozens of families. A swarm intelligent algorithm is an examples in the Skintrim.N family are not identified. Be that as effective solution for complex optimization problems. Many it may, it's anything but a sensible classifier in light of the fact that it can just distinguish a few information. researchers have made several in-depth studies about swarm intelligence and its applications [9], [16], [26]. Wang et al. [25] used the evolutionary multi-objective optimization (EMO) algorithms to deal with the floorplan problem in cyber-physical social system. Cui et al. [6] applied a modified cuckoo search algorithm to improve the performance of DV-Hop.

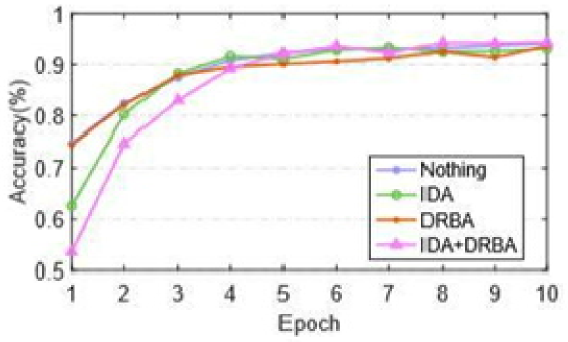
V. EXPERIMENTAL EVALUATION

A. Experimental Results

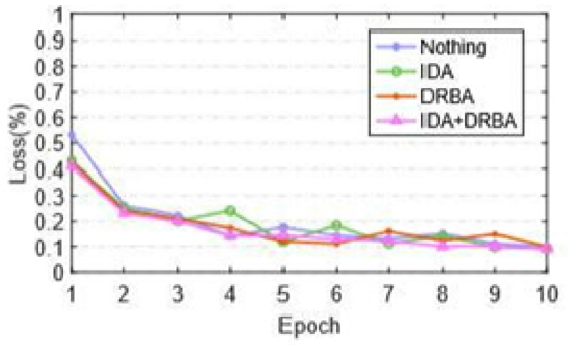
To validate the effectiveness and efficiency of the proposed approach, we designed experiments to (1) verify the efficiency of different data equalization methods, (2) ascertain the effects of different malware image sizes, and (3) compare the results of our proposed approach with the outcomes of other methods for detecting malicious code.

Efficiency of malware image data equalization: To verify the validity of our proposed data equalization method, including the image data augmentation technique (IDA) and

DRBA, we compared the results when no equalization method was used (Nothing) with models using IDA, DRBA, and IDA+DRBA. The performance of the different classification models on the training set is shown in Fig. 8. As seen from Fig. 8(a), all methods achieved a good result with respect to accuracy, even if the data set was not processed. The Nothing method may have demonstrated overfitting. Fig. 8(b) shows the Loss curve, for which the results of the four methods were similar.



(a) *Accuracy*

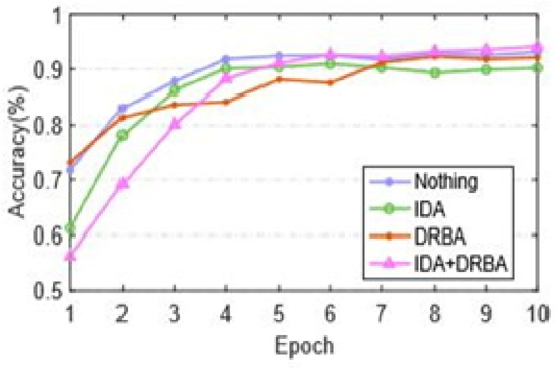


(b) *Loss*

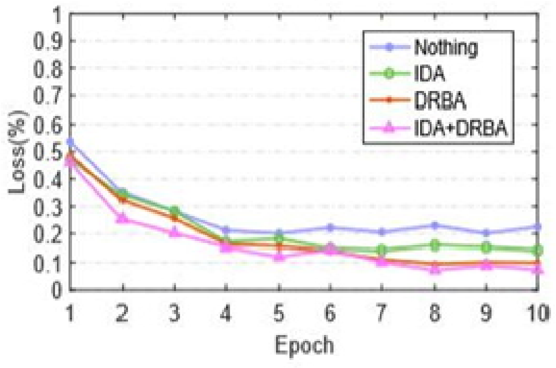
Fig. 8. Performance of different classification models on the training set

Fig. 9 shows the performance of the different classification models on the validation set. As we can see from Fig. 9(a), the Nothing method reflected the classification of all the data with the basic CNN model. Because of the imbalance of the samples, it was easy to cause over-fitting, so the Nothing method appeared to show a high level of accuracy. The accuracy of the DRBA method was the best. The accuracy rate increased continuously with the increase of the epoch. The accuracy of the IDA method was not as good as its performance on the training set, perhaps because some features of the image were lost in the image transformation process. There were some differences between training data and test data. As seen from Fig. 9(b), the loss curve of DRBA and IDA+DRBA was decreasing, while the curve of Nothing was oscillating within a certain range, which meant it was overfitting.

Table III shows the results of our approach compared with the outcomes of the four combination models: GIST+KNN, GIST+SVM, GLCM+KNN, and GLCM+SVM. We can see that our method had higher accuracy and faster detection speed. The performance of the other methods was somewhat poorer because their use of GIST or GLCM was complex and time-consuming.



(a) *Accuracy*



(b) *Loss*

Fig. 9. Performance of different classification models on the validation set

VI. CONCLUSION AND FUTURE WORK

This paper proposed a novel method to improve the detection of malware variants through the application of deep learning. First, this method transformed the malicious code into grayscale images. Next, the 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. In addition, our model provided an effective solution for the data imbalance problem among different malware families. Our experimental results on 9,342 grayscale images of 25 malware families showed that the proposed approach achieved 94.5% accuracy with good detection speed. In this study, the CNN framework required all input images to have a fixed size, which limited our model. In future work, we would like to use the SPP-net model to allow images of any size to be used as input. The SPP-net can extract features at variable scales thanks to a spatial pyramid pooling layer. We can introduce that layer into our model between the last subsampling layer and the fully connected layer to improve our models flexibility. In addition, the transformation of malicious code into color images would be a good topic for future research.

REFERENCES

[1] J. Bouvrie. Notes on convolutional neural networks. 2006.

[2] 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.

[3] 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.

[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, pages 32–46. IEEE, 2005.

[5] 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).

[6] 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.

[7] O.E.DavidandN.S.Netanyahu. Deepsign:Deeplearningforautomatic malware signature generation and classification. In Neural Networks (IJCNN), 2015 International Joint Conference on, pages 1–8. IEEE, 2015.

[8] D. D´ıaz-Pernil, A. Berciano, F. Pe˜na-Cantillana, and M. A. Guti´errezNaranjo. Bio-inspired parallel computing of representative geometrical objects of holes of binary 2d-images. International Journal of BioInspired Computation, 9(2):77–92, 2017.

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

[10] 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.

[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, pages 317–321. ACM, 2013.

[12] T. Isohara, K. Takemori, and A. Kubota. Kernel-based behavior analysis for android malware detection. In 2011 Seventh International Conference on Computational Intelligence and Security (CIS), pages 1011–1015. IEEE, 2011.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 22nd ACM international conference on Multimedia, pages 675–678. ACM, 2014. Khoshkbarforoushha, A. Khosravian, and R. Ranjan.

[14] Elasticity management of streaming data analytics flows on clouds. Journal of Computer and System Sciences, 89:24–40, 2017.

[15] 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.

[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, 8(5):447–456, 2017.

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

[18] A.Moser,C.Kruegel,andE.Kirda. Limits of statican alysis for malware detection. In Computer security applications conference, 2007. ACSAC 2007. Twenty-third annual, pages 421–430. IEEE, 2007.

[19] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath.

[20] Malware images: visualization and automatic classification. In Proceedings of the 8th international symposium on visualization for cyber security, page 4. ACM, 2011.

[21] 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

[22] ACM Workshop on Security and Artificial Intelligence, pages 21– ACM, 2011.

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

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

[25] 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.

[26] M. Wagner, F. Fischer, R. Luh, A. Haberson, A. Rind, D. A. Keim, and W. Aigner. A survey of visualization systems for malware analysis. In Eurographics Conference on Visualization (EuroVis), pages 105–125, 2015. [25] 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.

[27] L. Wang, H. Geng, P. Liu, K. Lu, J. Kolodziej, R. Ranjan, and A. Y. Zomaya. Particle swarm optimization based dictionary learning for remote sensing big data. Knowledge-Based Systems, 79:43–50, 2015.