International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 22 (2018) pp. 15622-15628

A Combination of Artificial Immune System and Deep Learning for Virus Detection

Vu Thanh Nguyen[1], Le Hoang Dung[1], Tuan Dinh Le[2]

[1]*University of Information Technology, Vietnam National University, HCM City, Vietnam.*

[2] *Long An University of Economics and Industry, Tan An, Long An Province, Vietnam.*

Abstract

This paper proposes a virus detection model to classify a file as benign or malware using a combination of artificial immune system (AIS) and deep learning with high accuracy detection rate. The approach contains the following stages: the first stage is data extraction of portable executable headers and creating PE feature set. In the second stage, the AIS is used to build a clonal generation of malware detectors and improve the accuracy of unknown viruses detection rate. Finally, a Deep Belief Network (DBN) is implemented to compute and train dangerous level of files, then evaluate performance of the system. As a result, our method can achieve a high detection rate of 98.8% on average with a very low false positive rate.

Keywords: Portable executable (PE), Artificial Immune System (AIS), Negative Selection Algorithm (NSA), Clonal Selection Algorithm (CLONALG), Deep Learning, Deep Belief Network (DBN)

1 INTRODUCTION

In recent years, virus recognition and elimination become critical and important problems. Many antivirus programs are developed to detect and remove viruses with different algorithms. However, the development of malware systems is complicated because of the continuous change in virus signatures and attacking methods. Researchers have been looking for other approaches can solve computer viruses as human problems. As a result, bio-inspired algorithms are developed as prospective models due to their ability to adapt naturally to the environment where they are applied.

Artificial Immune System (AIS), a model based on the principles of the biological immune system [1] is implemented for the computer viruses detection. It is a significant growing area employing immunological bio-mechanisms to solve virus computer problems.

Negative Selection Algorithm (NSA) and Clonal Selection Algorithm (CLONALG) [2] are two algorithms of AIS [7]. They have been showed to be efficient detection malwares in [3] and [4]. In their approaches, a set of detectors is generated by some randomized processes that uses a collection of self (benign) and non-self (virus) as the inputs. NSA eliminates candidate detectors that match any of the clean samples and keeps the unmatched ones, whereas CLONALG builds a mutation of detectors to produce a pool of suitable detector for solving a particular problem.

In this paper, we present a hybrid approach to build a virus detection model with a higher performance and overcome the disadvantages of previous others. Unlike other related approaches, we combine the AIS algorithms with a deep belief network (DBN) to train the samples. We decide to choose the Portable Execute headers of files as our main feature set due to the effeciency of them for recognizing how a file can be a malware or clean file [11]. Then we uses the AIS algorithms to generate malicious detectors to enhance the ability of unknown virus detection. We also use a DBN [5] [6] as a classify model because in recent years the Deep Learning has shown a more efficiency in classification problems than machine learning techniques.

This paper is organized as follows. Section 2 describes the related work. Section 3 introduces our model for virus detection in details. Section 4 presents our experimental results before the paper is concluded in Section 5.

2. RELATED WORK

In [7], Rui Chao and Ying Tan introduced a virus detection system based on AIS. They combined NSA and Clonal Selection Algorithm which was proposed to generate detectors. They also presented a mechanism to determine the danger level of files. Their model achieved a strong detection ability with good performance. However the discrimination error often happens when the size of a file is too small, because of little information utilized by the virus detection system (VDS).

In [8], the most relevant features are extracted from Portable Executable (PE) structure of executable files by Fisher Score and then is learned by an artificial neural network. Although their approach can identify unknown virus patterns, they use

only one deployed artificial neural network as learning model which is not efficient in both training cost and performance for large data.

In [9], the authors combined Negative Selection Algorithm with Artificial Immune Network. Negative Selection Algorithm is used to create the first generation of detectors and AiNet is used to improve detectors' coverage and enhance the ability of unknown virus detection. The limitation of their approach is that they use strings for detecting strings.

In [10], ANN is integrated with Clonal Selection Algorithm (CSA) to create a new virus detection approach. The CSA is used to train a pool of immature detector to adapt with the problem-space. However, the coverage of detector has not been examined and many irrelevant detectors are obtained, which cause low detection rate.

In [11], Liao selected only five fields of PE headers as the feature and implemented a hand-crafted rule-based algorithm for classification. Liao's algorithm achieved 99% accuracy and 0:2% FP for unknown malware. The limitation of this approach is that it cannot detect all the malware from the dataset.

In [12], Baldangombo used the three feature sets DLL calls, API calls and PE header fields value as both separately and as combined. Information Gain and PCA were used for features selection and these features were used to train SVM, J48, and NB classifiersand. Their approach achieved a detection rate of 99.6% with J48. The cons of this method is the complexity of the feature set with top 88 PE header, 130 DLL, and 2453 API function features to train the system.

In [13] and [14], they approached a malware detection with deep neural network (DNN) with detection rates are 95% and 96% in that order. Although these methods experienced lower detection rates than others, both approaches has been showing the efficiency of using Deep Learning as a new approach in solving computer viruses.

3. THE PROPOSED APPROACH

In this paper, we implement a model of detection based on the combination of Artificial Immune System and Deep Learning. We decide to choose a deep belief network as our training model because of its efficiency in classification problems [19]. The stages are proposed as follows: we extract PE features by processing Portable Executable (PE) file's header fields of the samples, then use AIS algorithms to generate the final virus detectors and calculate dangerous levels of detectors to choose the best candidates, finally all dangerous level set will be used to train a DBN classify model.

3.1 Extracting feature set

3.1.1 PE header

In our proposed work, we consider the PE files for the experiment since they are one of the most used file formats of Windows operating system and secondly the most common files submitted to VirusTotal are PE files. Figure 1 shows the number of different file types submitted to VirusTotal and Figure 2 describes the structure of a portable execute file.

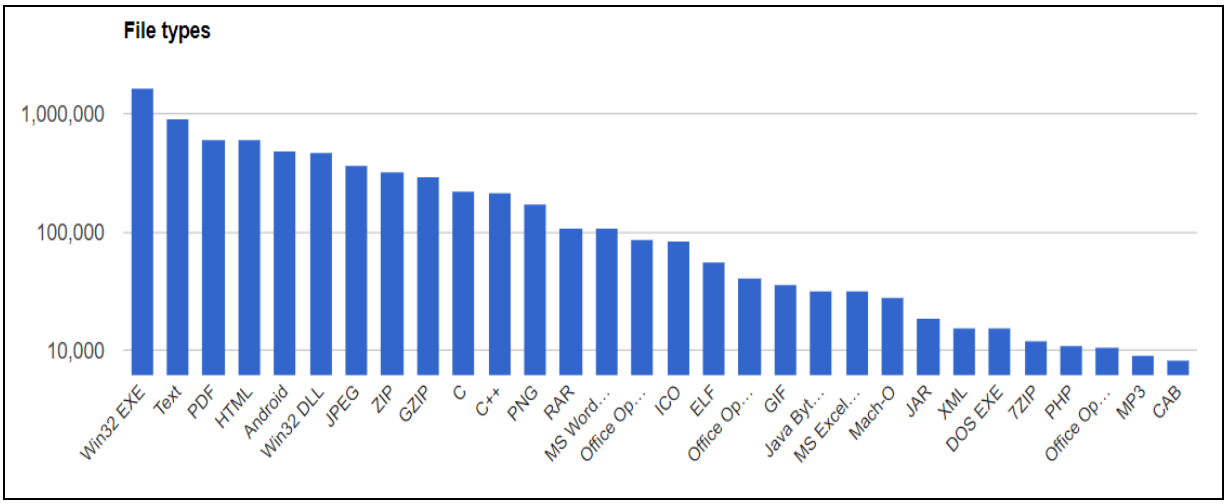

Figure 1: The figure for file types submitted to VirusTotal (8 August 2018) [1]

[1] https://virustotal.com/en/statistics/, (Last Accessed: August 8, 2018)

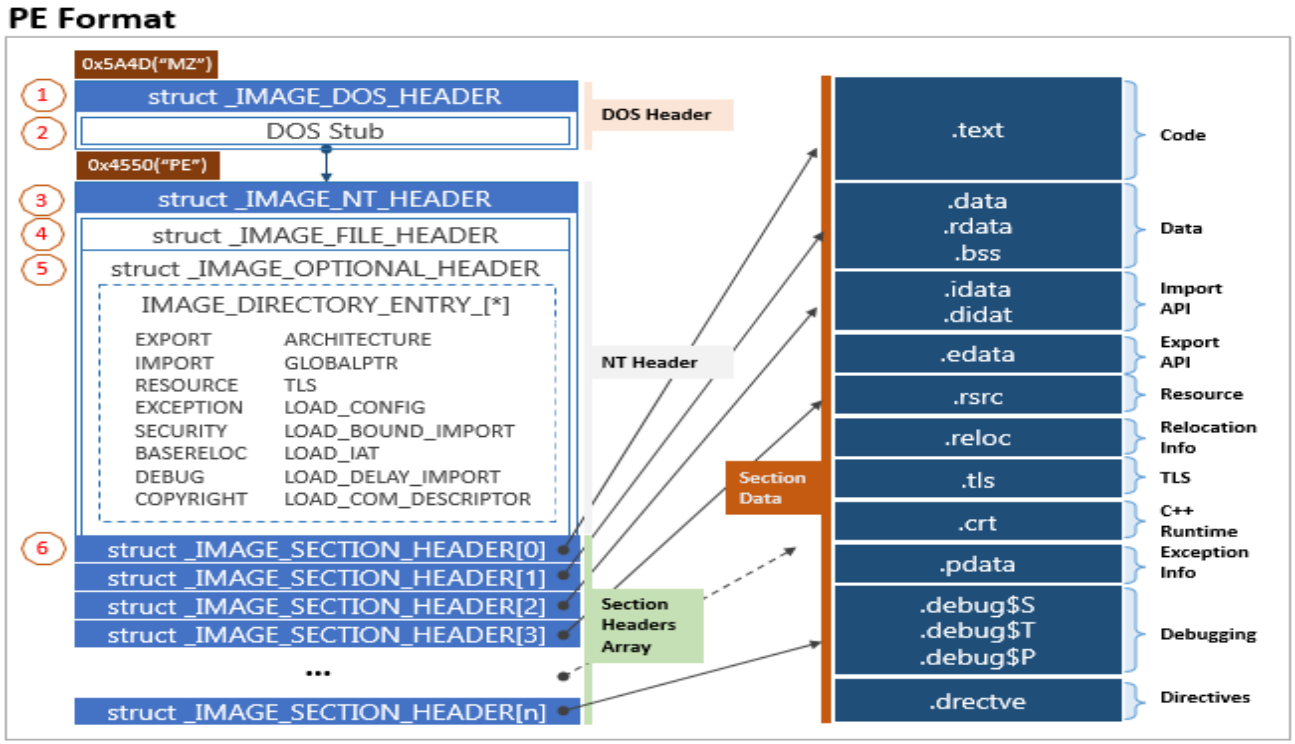

Figure 2: The PE format[2]

3.1.2 PE feature set

In our method, we choose some of fields of three main headers (DOS header, File Header and Optional header), then integrate with derived features [15] and finalise the PE feature set.

Firstly, we extract all raw values of three headers, totally 53 features from these headers. Then we need to select which fields that presents a significant difference between clean and virus file. By analyzing the Fisher-score and a statistical mean we select 28 fields for raw feature set.

DOS header	e_minalloc - e_ovno - e_sp - e_lfanew
File Header	Machine – NumberOfSections – CreationYear - PointerToSymbolTable - NumberOfSymbols – SizeOfOptionalHeader - Characteristics
Optional header	Magic – MajorLinkerVersion – MinorLinkerVersion - SizeOfCode - BaseOfCode - BaseOfData - FileAlignment - MajorOperatingSystemVersion - MajorImageVersion - MinorImageVersion - MajorSubsystemVersion - MinorSubsystemVersion - SizeOfHeaders – SizeOfStackReserve - SizeOfStackCommit – Subsystem - DllCharacteristics

Figure 3: Raw feature set

We also extract derived features that are introduced in [15]. These features are not values directly extracted from the header's field, but they are derived from the raw value of PE header by validating with a set of rules. In the proposed work, we reuse *Suspicious Section Name*, *Package Info, File Info* and *Entropy* that defined in [15]. Besides that we also select some other derived features such as *Machine*, *Import_Address_Table* and *Image_Data_Directory*.

Import_Address_Table: The import table DLLs may help our model to capture the semantics of the external function calls that a given input binary relies upon [13]. Therefore it can detect either heuristically suspicious files or files with a combination of imports that match a known malware family. We extract the import address table as a pair (count, value) from the binary program by hashing each tuple of DLL name and multiply with in the index counter.

Image_Data_Directory: Resources, symbol tables, debugging information, import, export tables, etc, are accessable from that nifty DataDirectory member of the optional header. This member is an array of IMAGE_DATA_DIRECTORY's that can be used to access other structures containing this information. We will extract 16 data directory as a derived feature.

[2] PE File Format – Kevin Attic for Security Research http://dandylife.net/blog/archives/388 August 8, 2018

Machine: The Machine field is a field of *File Header* which has one of the following values that specifies its CPU type. There are 25 types[3] that we can index them and convert it values into the index value.

Finally, PE feature set is created by combining a selected raw features and a set of derived features. Figure 3 summarizes the count for raw features and derived features present in PE feature set.

Raw	Derived	Total
28	24	52

Figure 3: PE feature set

3.2 Artificial Immune System

3.2.1 Negative selection process

After extracting features from training data, NSA is used to generate detectors that have ability to discriminate between benign and virus genes. Because some detectors can detect not only benign genes but also virus ones, if any of these detectors can recognize a benign genes, they will be eliminated from the set.

The process of negative selection is shown as follow:

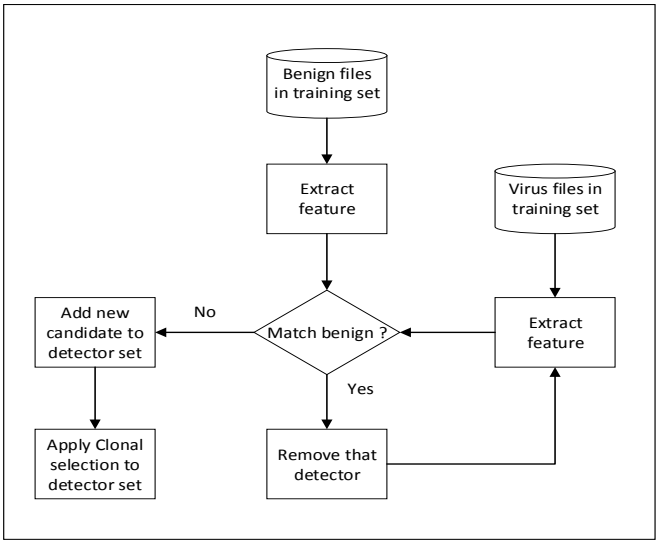

Figure 4: Negative selection process

3.2.2 Clonal selection process

The CLONALG [16] is employed in training the detectors. The main operators are described as follows:

Generate Candidate Genes Randomly: We randomly select L elements from detector set of the virus files, this set will be used for gene mutation process.

Gene Mutation: To perform candidate gene mutations, we provide 5 methods to get new detectors: 1. Copy r-length of a benign or virus to the current candidate; 2. Set value 0 to r-length cells of the current candidate; 3. Set value 1 to r-length cells that are zero; 4. Set value of a cell as median with neighboring candidates; 5. Set r-length cells value is average of 10 candidates (benign or virus).

Select Best Genes: The best genes will be selected based on affinity of it with the antigen trained in artificial immune network. This method present virus detectors the highest accuracy of capability of detecting.

The detectors generation algorithm is described as follows:

```
Notation

S  : a set of antigens
nt : suppression threshold
ct : matching threshold
N : set of memory detectors (antibodies)

While not reach target generation
Begin
  Generate set of random set from the original detector set
  For all antigens ag S do
    Calculate affinity of all detectors b of N with ag
    Select highest affinity set and place them in C
    Mutate all detectors clones of C and determine affinity each element c with ag
    Select highest affinity clones of C and place in D
    Remove all elements of D whose affinity with ag is less than ct
    Suppress all elements of D whose affinity with each other is greater than nt
    Remove detectors that can recognize some benign genes (NSA)
    Insert remaining elements of D into N.
  End For
  Remove duplicated detectors in N
  Determine affinity between each pair of elements in N
  Remove all elements whose affinity to another is less than nt
  Introduce new candidates into N
End
```

Figure 5: The pseudo code of the Clonal selection process

3.2.3 Affinity

In the theory of artificial immune systems, affinity is a concept that refers to the degree of genetic similarity. There are several methods to calculate the affinity such as the Euclidean distance, Hamming distance… depending on the type of data represented. This paper proposes a way to measure the similarity of set A and B by using a combine of Euclidean distance and Cosine similarity as the formula below:

$$\mathbf{aff = 1 - (\alpha.cosine + \beta.distance) / (\alpha+\beta)}$$

[3] https://docs.microsoft.com/en-us/windows/desktop/debug/pe-format#machine-types August 8, 2018

While cosine looks at the **angle** between vectors (thus not taking into regard their weight or magnitude), euclidean distance is similar to using a ruler to actually measure the distance. Since the range of the formula is [0;1] so that our method will normalize Euclidean distance to reduce the distance value by:

distance = min (euclidean / 10, 1.0)

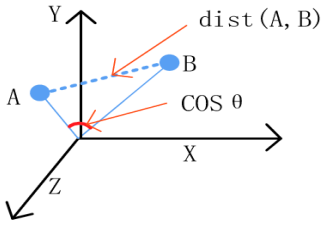

Figure 6: Euclidean distance (d) and Cosine similarity (θ)[4]

In previous section, we extract PE feature when the magnitude of the vectors is really important because there are many outliers in the set. Therefore, an Euclidean distance is a major choice to measure the similarity; however, we also include a cosine similarity to the formula to get higher accuracy in not only position but also direction, with $\alpha < \beta$.

3.3 Combination with learning algorithm

In our approach, dangerous level is used to determine how dangerous a file is based on Danger Theory [17][18]. The dangerous level of a file is a vector that is calculated based on a set of k-highest affinity values of detectors with that file. A file with a large value of dangerous level is considered as dangerous.

The dangerous levels of files in training data are calculated and are used to construct new training data, whose elements have k-dimensions. A deep belief network (DBN) object is then deployed as a learning model to study these data and is used to detect files in the testing set.

4. EXPERIMENTAL RESULTS

A virus detection program was implemented in Python to evaluate our approach. The program allows the users to define parameters manually as well as observe the learning process and the experimental results.

4.1 Evaluation Metrics

We have used the standard 5-fold cross-validation process in our experiments and the next fold has a larger size than the previous one. The dataset is randomly divided into 10 smaller subsets, where 7 subsets are used for training and 3 subset is used for testing. To evaluate our system we were interested in several quantities listed below:

True Positives (TP): the number of malicious executable examples classified as malicious executable

True Negatives (TN): the number of benign programs classified as benign

False Positives (FP): the number of benign programs classified as malicious executable

False Negatives (FN): the number of malicious executable classified as benign executable

Because our samples are unbalanced between the number of benigns and malwares, we consider to use assesment indicators such as precision, recall, f-score to evaluate the system.

Precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances:

$$precision = \frac{tp}{tp + fp}$$

Recall (also known as sensitivity) is the fraction of relevant instances that have been retrieved over the total amount of relevant instances

$$recall = \frac{tp}{tp + fn}$$

F-score harmonic mean of precision and recall:

$$F = \frac{2.precision.recall}{precision + recall}$$

Accuracy are measured by:

$$accuracy = \frac{tp + tn}{tp + fp + tn + fn}$$

4.2 Training and Testing Data

In this experience, we evaluate the approach with a dataset with more 9300 sample files, including clean and virus files, then we split randomly all of them to five different datasets. For each dataset, the ratio of training set and benign set is 7:3, as shown in Figure 7.

Dataset	Training files		Testing file	
	Benign	Virus	Benign	Virus
Dataset 1	317	985	137	423
Dataset 2	636	1971	273	845
Dataset 3	954	2956	410	1268
Dataset 4	1273	3942	546	1690
Dataset 5	1591	4928	683	2112

Figure 7: The number of files in file datasets

4 https://www.researchgate.net/publication/320914786_An_Opportunistic_Routing_for_Data_Forwarding_Based_on_Vehicle_Mobility_Association_in_Vehicular_Ad_Hoc_Networks

4.3 Experiments

In our experiment, the threshold of negative selection (z_threshold) and the threshold for clonal selection (ct, nt, loop) are *z_threshold=0.7*, *ct=0.7*, *nt=0.85* respectively. We use $\alpha=1$ and $\beta=2$ for calculating the affinity value. Then we select the first 32 highest affinities of a file to use as dangerous vector. For the learning algorithm trains DBN we execute parameters that are *hidden_layers [128, 256, 128], epochs=20, learning_rate=0.01, backprop=500* and *batch_size=128* as the best learning performance. The result is shown in below.

Dataset	Deep Belief Network			
	Accuracy	Precision	Recall	F-Score
Dataset 1	0.9875	0.9879	0.9875	0.9876
Dataset 2	0.9803	0.9816	0.9803	0.9806
Dataset 3	0.9869	0.9875	0.9869	0.987
Dataset 4	0.9919	0.9922	0.9919	0.992
Dataset 5	0.9943	0.9944	0.9943	0.9943
Average	0.9882	0.9887	0.9882	0.9883

Figure 8: The detection rate of DBN, loop = 1

We also try to increase the generation iterator of training process with loop = 2 and loop = 5, the result still shows a slightly higher accuracy. The result is showns in Figure 9 and Figure 10

Dataset	Deep Belief Network			
	Accuracy	Precision	Recall	F-Score
Dataset 1	0.9911	0.9914	0.9911	0.9911
Dataset 2	0.9830	0.9839	0.9830	0.9832
Dataset 3	0.9881	0.9886	0.9881	0.9882
Dataset 4	0.9875	0.9879	0.9875	0.9876
Dataset 5	0.9936	0.9937	0.9936	0.9936
Average	0.9887	0.9891	0.9887	0.9887

Figure 9: The detection rate of DBN, loop = 2

Dataset	Deep Belief Network			
	Accuracy	Precision	Recall	F-Score
Dataset 1	0.9893	0.9897	0.9893	0.9894
Dataset 2	0.9857	0.9863	0.9857	0.9858
Dataset 3	0.9905	0.9906	0.9905	0.9905
Dataset 4	0.9897	0.99	0.9897	0.9898
Dataset 5	0.9943	0.9944	0.9943	0.9943
Average	0.9899	0.9892	0.9888	0.9889

Figure 10: The detection rate of DBN, loop = 5

As can be seen from the tables, our approach archive an average detection rate of 98.8%. Our method not only has a better average score than [13], [14], [19] which also use deep learning model but also has a very high accuracy as slightly similar as other related work. The accuracy also increases when the size of samples grows up. In our system, the combination of AIS and DBN enhances the performance of training process and by using the PE headers as feature set we can save our computer storage than storing signatures, opcodes or strings of file.

As the result we can conclude that the PE header feature with the new approach using the combination of deep learning and artificial immune system can be effectively used for detecting many malwares. Our hybrid model has opened a new prospect for dealing with the virus detection problem.

5. CONCLUSION

In this paper we introduced a deep learning based bio-inspired algorithms that achieves a detection rate of 98.8% over an experimental dataset of total around 9300 software binaries. This research indicates that the new approach using the combination of deep learning and artificial immune system opens a new prospect for dealing with the virus detection problem.

At the present the time detection of learning model is depended on the number of detectors and training set because it need to convert into dangerous levels. Therefore, we will investigate the technique to reduce the time execution with large data in future study.

ACKNOWLEDGMENT

This research is funded by Vietnam National University, Ho Chi Minh City (VNUHCM) under grant number C2018-26-06.

REFERENCES

[1] Read, M., Andrews, P., Timmis, J.: Artificial immune systems, pp. 4–5 (2012)

[2] J. Al-Enezi, M. Abbod, and S. Alsharhan.: Artificial immune systems-models, algorithms and applications. (2010).

[3] Sahu, Agnika and Prabhat Ranjan Maharana.: Negative Selection Method for Virus Detection in a Cloud. International Journal of Computer Science and Information Technologies 4, 771-774 (2013).

[4] Al-Sheshtawi, Khaled A. and Hatem M. Abdul-Kader.: Artificial Immune Clonal Selection Classification Algorithms for Classifying Malware and Benign Processes Using API Call Sequences. (2010).

[5] David, O. E., & Netanyahu, N. S.: DeepSign: Deep learning for automatic malware signature generation and classification. International Joint Conference on Neural Networks (IJCNN). doi:10.1109/ijcnn. 2015.7280815. (2015).

[6] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09). ACM, New York, NY, USA, 609-616. (2009).

[7] Chao, Rui and Ying Tan.: A Virus Detection System Based on Artificial Immune System. International Conference on Computational Intelligence and Security 1: 6-10. (2009).

[8] S. Shah, H. Jani, S. Shetty, and K. Bhowmick.: Virus detection using artificial neural networks. International Journal of Computer Applications, vol. 84, no. 5. (2013).

[9] V. T. Nguyen, T. T. Nguyen, M. T. Khang, and T. D. Le.: A combination of negative selection algorithm and artificial immune network for virus detection," in Future Data and Security Engineering. Springer, 2014, pp. 97–106. (2014).

[10] V. T. Nguyen, N. P. Anh, M. T. Khang, N. H. Ngan, N. Q. Thai, and N. T. Quoc.: A combination of clonal selection algorithm and artificial neural networks for virus detection, in Advances in Computer Science and its Applications. Springer, 2014, pp. 95–100. (2014).

[11] Liao, Y., 2012. Pe-header-based malware study and detection. Retrieved from the University of Georgia: http://www. cs. uga. edu/˜liao/PE Final Report. pdf.

[12] Baldangombo, U., Jambaljav, N., Horng, S.-J., 2013. A static malware detection system using data mining methods. arXiv preprint arXiv:1308.2831.

[13] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features", 10th International Conference on Malicious and Unwanted Software (MALWARE), Fajardo, 2015, pp. 11-20. (2015).

[14] Tobiyama, Shun & Yamaguchi, Yukiko & Shimada, Hajime & Ikuse, Tomonori & Yagi, Takeshi.: Malware Detection with Deep Neural Network Using Process Behavior. 577-582. 10.1109/COMPSAC.2016.151. (2016).

[15] Kumar, Ajit & Kuppusamy, K S & Gnanasekaran, Aghila. (2017). A learning model to detect maliciousness of portable executable using integrated feature set. Journal of King Saud University - Computer and Information Sciences. 10.1016/j.jksuci.2017.01.003.

[16] M. T. Khang A Combination of Artificial Neural Network and Artificial Immune System for Virus Detection, REV Journal on Electronics and Communications, Vol.5, No.3-4, July-December 2015. ISSN: 1859-378X; p. 52-p.57.

[17] V. T. Nguyen, C. N. Tuan, L. T. Dung, V. M. Hai, N. T. Toan. (2016). Computer Virus Detection Method Using Feature Extraction of Specific Malicious Opcode Sets Combine with aiNet and Danger Theory. 199-208. 10.1007/978-3-319-48057-2_14.

[18] Aickelin, U., Cayzer, S.: The danger theory and its application to artificial immune systems, pp. 4–6 (2008).

[19] P. T. Anh, T. H. P. An (2018), Research methods of virus detection by machine learning and deep learning algorithms. Master thesis of Ho Chi Minh City University of Information Technology, Vietnam (2018).