How to Cite:
Manoj, M., & Rani, V. G. (2022). Ransomeware classification using fuzzy neural network algorithm. *International Journal of Health Sciences*, *6*(S2), 11268–11278.
https://doi.org/10.53730/ijhs.v6nS2.8026

Ransomeware classification using fuzzy neural network algorithm

M. Manoj
Research Scholar, Sri Ramakrishna College of Arts &Science for Women, Coimbatore, India
Corresponding author email: manomca24@gmail.com

Rani V G
Associate Professor, Department of Computer Science
Sri Ramakrishna College of Arts & Science for Women, Coimbatore, Coimbatore India
Email: ranisrcw@gmail.com

***Abstract*---**Traditional malware classification relies on known malware types and significantly large datasets labelled manually which limits its ability to recognize new malware classes. For unknown malware types or new variants of existing malware containing only a few samples each class, common classification methods often fail to work well due to severe over fitting. In this paper, we propose a new neural network structure called fuzzy neural network based algorithm for ransom ware classification. Malware classification with fuzzy neural network classifier yields 98% accuracy which means greater performance than classification with neural network.

***Keywords*---**malware, classification, neural network, ransomware.

Introduction

Malware (short for “malicious software”) is a file or code, typically delivered over a network, that infects, explores, steals or conducts virtually any behavior an attacker wants. And because malware comes in so many variants, there are numerous methods to infect computer systems. Though varied in type and capabilities, malware usually has one of the following objectives:

- Provide remote control for an attacker to use an infected machine.
- Send spam from the infected machine to unsuspecting targets.
- Investigate the infected user’s local network.
- Steal sensitive data.

International Journal of Health Sciences ISSN 2550-6978 E-ISSN 2550-696X © 2022.
Manuscript submitted: 27 March 2022, Manuscript revised: 09 April 2022, Accepted for publication: 18 May 2022

Ransomware affects an infected computer system in some way, and demands payment to bring it back to its normal state. There are two variations of ransomware, being crypto ransomware and locker ransomware. Locker ransomware just locks down a computer system without encrypting its contents, whereas the traditional ransomware is one that locks down a system and encrypts its contents. For example, programs such as Crypto Locker encrypt files securely, and only decrypt them on payment of a substantial sum of money

Malware Classification is the process of assigning a malware sample to a specific malware family. Malware within a family shares similar properties that can be used to create signatures for detection and classification. Signatures can be categorized as static or dynamic based on how they are extracted. A static signature can be based on a byte-code sequence, binary assembly instruction, or an imported Dynamic Link Library (DLL). Dynamic signatures can be based on file system activities, terminal commands, network communications, or function and system call sequences.

Given the severity of the problem, malware classification is an active research area [1], but the escalating threat indicates the problem is clearly not solved. Achieving very low false positive rates is extremely challenging and having access to a very large number of labeled malware and a benign example is required to even begin to obtain reasonable accuracies. Earlier research has been done on relatively small malware sample collections [2, 3] limiting the accuracy of these systems.

In section II, we shortly explain the related works that have been done. In section III, we present the precise explanation of work might be the most important part of this thesis, and how the models are designed. Surprisingly, this has not been previously undertaken in this field. In section IV, we present a comparison of obtained results in the proposed method to others that are using different algorithm and analyzing the best among them. Its efficiency is proved by illustrating tables and plots to show the accuracy and other metrics of different algorithms. In section V, includes the results of our work, conclusion and future works are an interesting section portraying the map as a guideline to a future supposition.

Literature Survey

Yan, Qi and Rao (2018) present an *ensemble* method for detecting malware based on a deep neural network. The approach uses a *convolutional* neural network and a memory technique to learn raw data and make inferences regarding the existence or nonexistence of malware. The inferences are based on patterns extrapolated from both the structure and code of the malicious file. (A convolutional recurrent neural network is a blend of the recurrent neural network and the convolutional neural network. Convolutional neural networks can be characterised as those that apply convolutions (a kind of mathematical operation) and that classify data regardless of the positioning.) This approach is similar to the *recurrent neural network ensemble* proposed by Rhode, Burnap and Jones (2018).

The ensemble studies behavioural data and makes inferences regarding the maliciousness of an executable file. This is done during the execution by collecting a small sample of behavioural data with a view to detecting and blocking malicious processes before them cause damage. In order to classify this behavioural data, a classifier is presented based on a convolutional recurrent neural network (Alsulami & Mancoridis, 2018), in order to classify families of malware and to extrapolate better patterns for improving detection accuracy. The method extracts features adaptively from MS Windows files to classify them.

In the same vein, Kabanga and Kim (2018) apply the convolutional neural network to the classification of malware *image*. Instead of using text and other forms of data as inputs, image vectors are used to train neural networks. The convolutional neural network is set up with three layers, in order to achieve the classification function. In order to identify and classify complex patterns in data for malware detection, Le, Boydell, Namee and Scanlon (2018) present a classification method based on deep learning. The approach uses data-driven techniques to identify features for classification. Multiple deep-learning architectures are utilised, and each input is classified into a malware class in terms of various neural network layers, whereby vectors are generated for feature extraction and classification. This classification method contrasts with mechanisms that rely on expert domain knowledge.

In order to detect and classify malware in unseen files, Rad, Nejad and Shahpasand (2018) apply a binary classifier to MS Windows files. The training of the neural network classifier is done with a view to giving it the ability to distinguish malicious files from benign ones. Many of the aforementioned approaches provide novel solutions, but the malware landscape has dramatically changed in recent years. The changes are mostly epitomised in the shifting optima (Souri & Hosseini, 2018), i.e., shifts in the most favourable solution among a set of constantly changing feasible solutions. The shifting optimum makes tracking difficult, and makes it overwhelmingly difficult for static and slow-evolving solution approaches to find optima. Thus, the more efficient malware countermeasures will be those that are highly adaptive and dynamic in their search and optimisation processes for malware detection.

Mitsuhashi R et al [1] proposed a strategy to select a Deep learning model that fits the malware visualization images. First, they solved the problem of sample data imbalance between malware families using the under sampling technique. Second, they selected a CNN model that fits the malware visualization images using fine-tuning method. Finally, selected the VGG16 fine-tuning model and achieved high accuracy of classifying malware family. In the future, there is an approach to improve the proposed strategy by preparing a dataset with a larger malware sample.

Asrafi N et al [2] elaborated that Stacking machine learning provides an instrument to use different algorithms for training and testing data for different aspects. Boosting enhances weak learners by aggregating each of their strengths. In this paper, author proposes an automated stacking model that combines stacking and boosting to malware classifications. Their results show that the best performance coming from the MLP-Adaboost Classifier. In the future, different feature selection methods could be explored to better describe the malware characteristics other than APIs.

Kim DW et al [3] proposed a method to effectively classify malware based on a much smaller number of features. Using the recursive feature elimination method, the number of features examined was reduced while maintaining an accuracy of up to 78% with Random Forest and a margin error of just 0.4%. All three classifiers were capable of performing malware classification using all the features at an accuracy of around 70%. An identical performance level was achieved using around 30% of the number of features, after selection based on RFE. In addition, a comparison of the performance assessments with respect to the major types of malware showed that, the marginal error, was around 1 to 3%. The biggest factor for malware came from the category, System call, and classifying the malware type features of the System call category play the most important role in the malware operation.

Tang Z et al [4] novelly use few-shot learning approaches to address malware classification when few samples are available and propose a new model, ConvProtoNet, to improve the performance over existing few-shot models. The new model enriches the representation of embedding space, makes deep prototype induction, and prevents from gradient vanishing problem. Experiment results show that contemporary few-shot approaches are of enough ability to solve few-shot malware classification problem where considerable accuracy is achieved on several dataset and converting malware to images can retain useful information to do classification task.

Yuan B et al [6] proposed a byte-level malware classification method called MDMC which converted malware binaries into markov images and classified malware markov images with deep learning. Only the binaries of malware were used without the reverse analysis and dynamic analysis. MDMC could be applicable to various systems such as windows and android. Compared with similar methods such as GDMC, MDMC could significantly improve the accuracy of malware classification.

Narayanan BN et al [7] proposed ensemble approach provided a further boost in performance. Extracting features from proposed CNN and RNN and later classifying those using logistic regression and SVM provided a further improvement in performance. Extracting a total of 45 features from all these networks helped in distinguishing the malware programs effectively. An overall accuracy of 99.5% and 99.8% is achieved using logistic regression and SVM, respectively. Representing malware programs both in terms of compiled and assembly level files helped in overcoming a lack of information present in either of those file types. Representing such huge malware programs in terms of a simple suite of 45 features helps in reducing data complexity and computational resources.

Hosseini S et al [9] showed how Deep Neural Network can be effective in malware classification, mostly in a great compound convolutional neural network and RNNs. The proposed method is a great malware classification algorithm which is very comprehensive in checking lib.so files in a deep separate Neural Network. As the experimental results show, maximum accuracy is achieved by 5-Fold crossvalidation is 98.8 which is about one percent more than CNN and Ensemble-learning methods and also two percent more than *SVM* algorithm. Although the results demonstrate an effective and efficient method for android malware classification, it is possible to make it better and more robust.

Proposed Methodology

Data Set Collection

https://www.kaggle.com/sapere0/bitcoinheist-ransomware-dataset

Data's are collected for malware classification from bit coin heist.

Ransomware Families Used

montrealAPT, montrealComradeCircle, montrealCryptConsole, montrealCryptoLocker, montrealCryptoTorLocker2015, montrealCryptXXX, montrealDMALocker, montrealDMALockerv3, montrealEDA2, montrealFlyper, montrealGlobe, montreal GlobeImposter, montrealGlobev3,montrealJigSaw, montrealNoobCrypt, montrealRazy, montrealSam, montrealSamSam, montrealVenusLocker, montrealWannaCry, montrealXLocker, montrealXLockerv5.0, montrealXTPLocker, paduaCryptoWall, paduaJigsaw, paduaKeRanger, princetonCerber, princetonLocky, Normal.

Dataset Pre-Processing

Data are pre processed using mean imputation method and set of new data's are obtained.

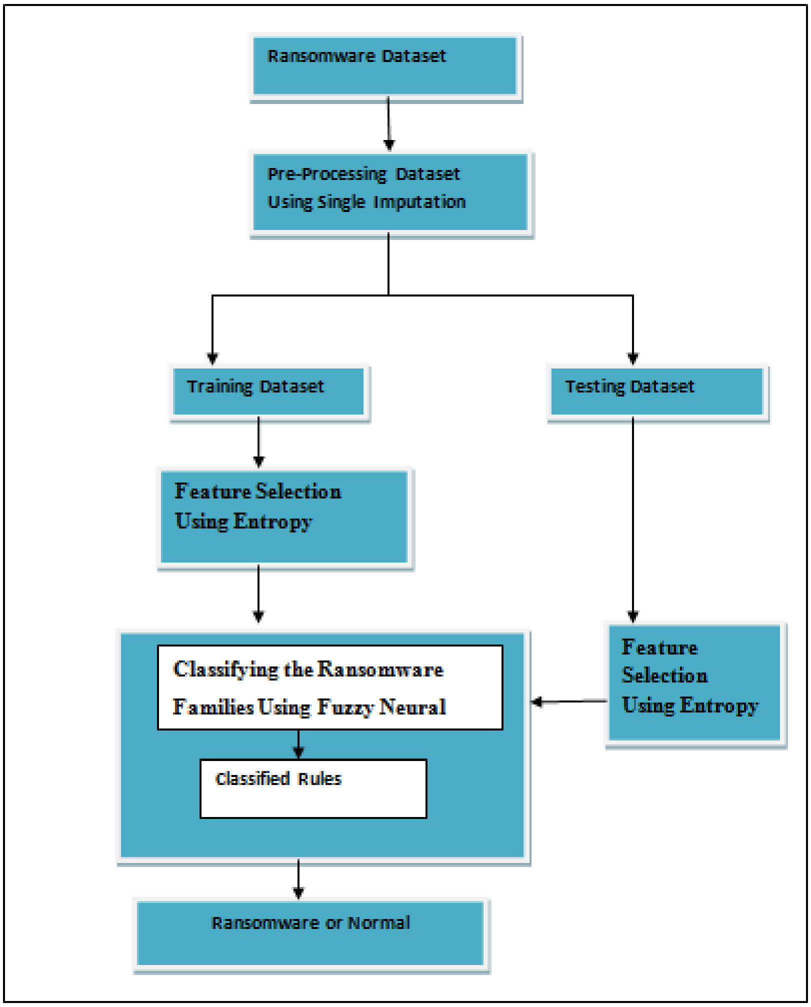
Ransomware Dataset
Pre-Processing Dataset Using Single Imputation
Training Dataset
Testing Dataset
Feature Selection Using Entropy
Classifying the Ransomware Families Using Fuzzy Neural
Classified Rules
Feature Selection Using Entropy
Ransomware or Normal

Fig 3.1 Proposed Frame work

Feature Selection Using Entropy

Entropy is a measure of the amount of uncertainty in the outcome of a random experiment, or equivalently, a measure of the information obtained when the outcome is observed. First, Entropy-based feature selection is based on the criteria of IG. It selects those features that provide most gain in information. For calculating information gain of features first we have to calculate probability and entropy of classes present in data set. With the creation of huge databases and the consequent requirements for good machine learning techniques, new problem arise and novel approaches to feature selection are demand. Feature selection plays an important role in classification. Feature selection is an important pre-processing step to machine learning. It selects an effective subset from the original features according to a certain criterion so that it can improve the performance of later data processing, such as classification and clustering. In real-world applications, there are many irrelevant and redundant attributes in relations of relational database, in which are little contribution to classification accuracy. Hence, feature selection is an essential data processing step in multi-relational data mining.

By applying entropy based feature selection techniques, we can improve classification accuracy, achieve good time performance, and enhance comprehensibility of the models. Feature selection reduces the number of features, removes irrelevant, redundant, or noisy data, and brings the immediate effects for applications: speeding up a data mining algorithm, improving mining performance such as predictive accuracy and result comprehensibility. In fact, feature selection techniques have been widely employed in a variety of applications, such as genomic analysis, information retrieval, and text categorization.

Feature selection is a process that selects a subset of original features. The optimality of a feature subset is measured by an evaluation criterion. As the dimensionality of a domain expands, the number of features N increases. Finding an optimal feature subset is usually intractable and many problems related to feature selection have been shown to be NP-hard.

Classifying the Ransom ware Families Using Fuzzy Neural Network algorithm

Phase1: Pre processing

Step 1 : Upload the dataset
Step 2 : Pre processing is done using
mean imputation method
Step 3 : New dataset is obtained

Phase 2: Fuzzification

Step 1 : Each instance is considered for fuzzification.
Step 2 : Fuzzy membership value for each attribute is calculated
Step 3 : Replace the value obtained with the linguistic variable (High, Medium and Low)
Step 4 : Fuzzy rules are generated from new table. Duplicate rules are eliminated

Phase 3: Ensemble Fuzzy with Neural Network

```
Input: New dataset is Di….Dn ;
Output: Fuzzy Neural Network Classifier NN
1. Initialize the weights in the network random dataset Di,….Dn, Hidden Neurons weight Wi……Wn (random) , bias b
2. While (di != null) for each samples D in the training set do
2.1 Ot = neural network output (network,D);
2.2 To = trainer output for S
2.3 Calculate error (To - Ot) at output unit
3. Compute all Neuron weight from hidden layer to output layer
```

$y= \sum(\text{weight} * S) +b$

$z=(D1*W1 + D2*W2 + D3*W3 +\ldots\ldots\ldots+ Dn*Wn)$

activate function
$\text{sigmoid}(z) =1/(1+e^{-z})$
backward pass

4. Compute delta-wi for all weights from input layer to hidden layer ; backward pass continued
5. Update the weights in the network End while
6. Until all samples are classified correctly or stopping criterion satisfied return(step 1)

Performance Evaluation

Precision

Precision is a method used to describe random errors and is calculated accordingly.

Precision P = tp/(tp + fp)

Where tp is true positive and fp is false positive

Recall

Recall is fraction of relevant data that are retrieved and it is computed using the equation.

Recall R = tp/(tp + fn)

Where fn is false negative

Accuracy

Helps to know how accurate the decision tree is formed. It is calculated with the help of

Accuracy A =tp+tn / (tp+tn+fp+fn)

Where tp is true positive and fp is false positive

Result and Discussion

Experimental Results

To evaluate the accuracy of classification, we measured the percentage of correctly classified samples for each classifier.
https://www.kaggle.com/sapere0/bitcoinheist-ransomware-dataset
Data's are collected for malware classification from bit coin heist.

Ransomware Families Used

Montreal APT, montreal Comrade Circle, mont real CryptConsole, montrealCryptoLocker, montreal Crypto T or Locker 2015, mont real Crypt XXX , mont real DMA Locker, mont real DMA Lockerv3, mont real EDA2, mont real Flyper, mont real Globe, mont real Globe Imposter, mont real Globev3, mont real JigSaw, mont real Noob Crypt, montreal Razy, montreal Sam, mont real SamSam, mont real Venus Locker, mont real WannaCry, montreal XLocker, montreal XLockerv5.0, montreal XTP Locker, padua Crypto Wall, padua Jigsaw, padua KeRanger, prince ton Cerber, Princeton Locky, Normal.

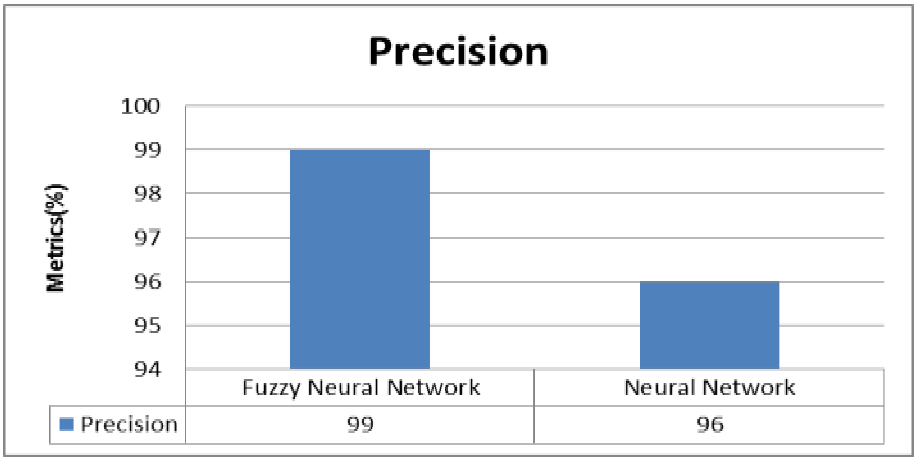

Fig 4.1 Precision of Neural Network and Fuzzy Neural Network

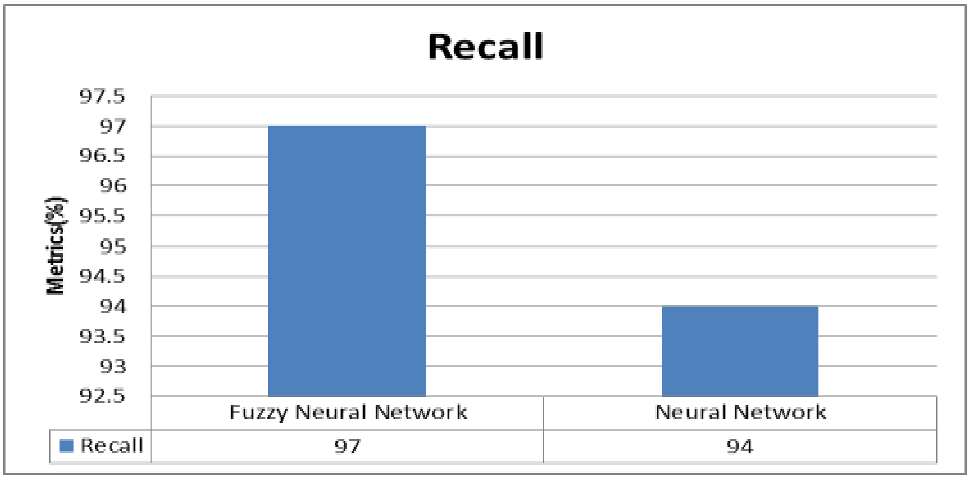

Fig 4.2 Recall of Neural Network and Fuzzy Neural Network

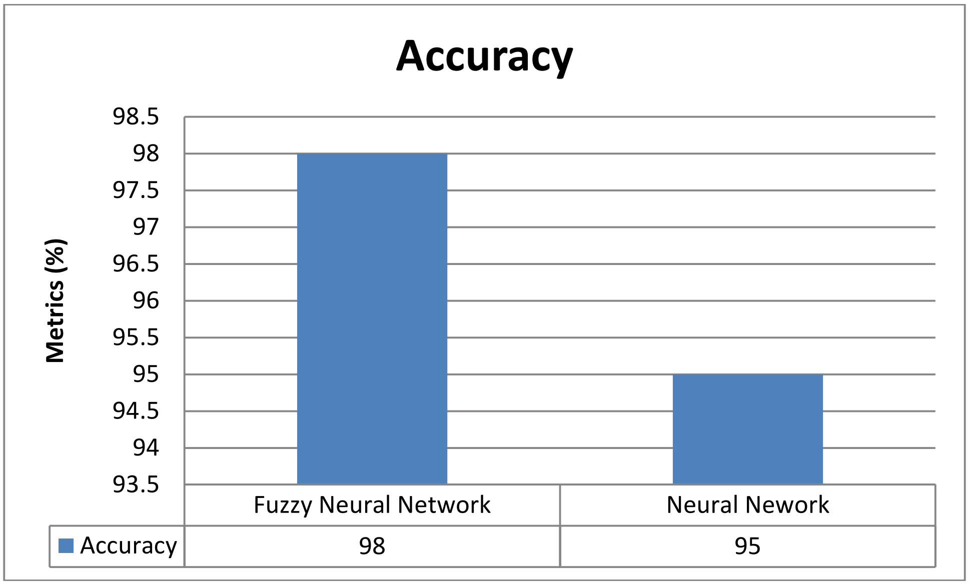

Fig 4.3 Accuracy of Neural Network and Fuzzy Neural Network

The pictorial representation represents the performance evaluation factors such as accuracy, precision and recall. The figure shows malware classification yields better results with fuzzy neural network compared to classification with neural network.

Conclusion

In this paper we presented malware classification based on fuzzy neural network. Feature Selection is based on entropy. By applying feature selection techniques, we can improve classification accuracy, achieve good time performance, and enhance comprehensibility of the models. Feature selection reduces the number of features, removes irrelevant, redundant, or noisy data, and brings the immediate effects for applications. The experimental result shows that the proposed classifier is effective in respect to the classification accuracy. For the future work, we can apply our proposed classifier to the more relational dataset to measure the performance of our classifier.

References

[1]. Mitsuhashi R, Shinagawa T. High-accuracy malware classification with a malware-optimized deep learning model. arXiv preprint arXiv:2004.05258. 2020 Apr 10.

[2]. Asrafi N, Lo DC, Parizi RM, Shi Y, Chen YW. Comparing performance of malware classification on automated stacking. InProceedings of the 2020 ACM Southeast Conference 2020 Apr 2 (pp. 307-308).

[3]. Kim DW, Shin GY, Han MM. Analysis of feature importance and interpretation for malware classification. Computers, Materials & Continua. 2020 Jan 1;65(3):1891-904.

[4]. Tang Z, Wang P, Wang J. ConvProtoNet: Deep prototype induction towards better class representation for few-shot malware classification. Applied Sciences. 2020 Jan;10(8):2847.

[5]. Nisa M, Shah JH, Kanwal S, Raza M, Khan MA, Damaševičius R, Blažauskas T. Hybrid malware classification method using segmentation-based fractal texture analysis and deep convolution neural network features. Applied Sciences. 2020 Jan;10(14):4966.

[6]. Yuan B, Wang J, Liu D, Guo W, Wu P, Bao X. Byte-level malware classification based on markov images and deep learning. Computers & Security. 2020 May 1;92:101740.

[7]. Narayanan BN, Davuluru VS. Ensemble malware classification system using deep neural networks. Electronics. 2020 May;9(5):721.

[8]. Vu DL, Nguyen TK, Nguyen TV, Nguyen TN, Massacci F, Phung PH. HIT4Mal: Hybrid image transformation for malware classification. Transactions on Emerging Telecommunications Technologies. 2020 Nov;31(11):e3789.

[9]. Hosseini S, Nezhad AE, Seilani H. Android malware classification using convolutional neural network and LSTM. Journal of Computer Virology and Hacking Techniques. 2021 Apr 29:1-2.

[10]. Dahl GE, Stokes JW, Deng L, Yu D. Large-scale malware classification using random projections and neural networks. In2013 IEEE International Conference on Acoustics, Speech and Signal Processing 2013 May 26 (pp. 3422-3426). IEEE.

[11]. Ronen R, Radu M, Feuerstein C, Yom-Tov E, Ahmadi M. Microsoft malware classification challenge. arXiv preprint arXiv:1802.10135. 2018 Feb 22.

[12]. Nari S, Ghorbani AA. Automated malware classification based on network behavior. In2013 International Conference on Computing, Networking and Communications (ICNC) 2013 Jan 28 (pp. 642-647). IEEE.

[13]. Kalash M, Rochan M, Mohammed N, Bruce ND, Wang Y, Iqbal F. Malware classification with deep convolutional neural networks. In2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS) 2018 Feb 26 (pp. 1-5). IEEE.

[14]. Anderson B, Storlie C, Lane T. Improving malware classification: bridging the static/dynamic gap. InProceedings of the 5th ACM workshop on Security and artificial intelligence 2012 Oct 19 (pp. 3-14).

[15]. Milosevic N, Dehghantanha A, Choo KK. Machine learning aided Android malware classification. Computers & Electrical Engineering. 2017 Jul 1;61:266-74.