Malware Detection using Artificial Neural Network

Anirban Chakraborty, Dr. Shishir K Shandilya, Dr. Praveen Lalwani

Abstract

In this paper, we introduce Artificial Neural Network in the detection of Malwares. Malwares and Viruses are a big problem in today's world. Capable of devastating arrays of computer systems and critical infrastructures across the globe, it is the need of the hour to devise an intelligent system which can be used for detecting malicious entities. We use one of the Deep Learning Techniques knows as Artificial Neural Network with suitable parameters and optimizations to achieve an accuracy rate of more than 98% on the Malware Classification Dataset. The dataset contains Malware features like the PE Header information's and the packer types used to obfuscate the malicious program. Subsequently selected useful features were taken to train the ANN model so that it can achieve maximum accuracy possible. This is a binary classification problem and the output of the model will be either Malicious (1) or Non-Malicious (0) if applied on a test dataset.

Keywords: Malware, Deep Learning, Artificial Neural Network, Machine Learning, Cyber Security

1. Introduction

Detection of malwares is difficult given how many types and the ranges of attack types that they inflict upon a host system. It is therefore an open challenge to detect such malicious programs and stop them before any damage can be done.

Since the beginning of this century and the substantial progress in Computer hardware technology, various methods have been proposed, some are rule based that pertain to some cases of malwares namely the Heuristic Approach and Machine Learning Approaches which given the model and the type of data will predict and reason if a particular piece of software is benign or malicious. The state of the industry however right now prefers manual and automatic created rules and signatures over machine learning because of low false positives achievable by rule and signature based methods [1].

However since the substantial and huge progress of mathematics concerning Machine Learning and the development of various techniques and methods handling big data, there is now a confidence upon the use of Machine Learning Models in detecting Malwares. The main cause of this is the availability of huge datasets which were mined for years and the availability of labels which accurately describe the data. Computing power and processing has also become a lot more affordable and Machine Learning and Deep Learning libraries by Google and Meta now

have extensive documentations and many interesting and easy to use features. Also the models are not GPU heavy so they can be deployed even on a basic piece of hardware.
This sparks the usage of Machine Learning algorithms or Deep Learning techniques to simply solve a binary classification problem of whether a given file is benign or malicious. Due to the vast amount of datasets which are available publicly it is possible for Deep Learning Techniques to get an almost perfect predictability score.
In this paper we describe how Artificial Neural Network can be used to detect Malwares with 98% accuracy. We surely believe that this is the best accuracy which can be achieved and is better than most other Machine Learning algorithms used so far.
By taking relevant data and dropping unnecessary data based on the Portable Executable headers of Windows Executables using Data Correlation methods, it was possible to further streamline and clean the data such that the model does not unnecessarily works on data that isn't needed. The Non-Linear ReLU function is used that will output the input directly if it's positive. Adam optimizer with binary cross entropy to take care of the cost functions.

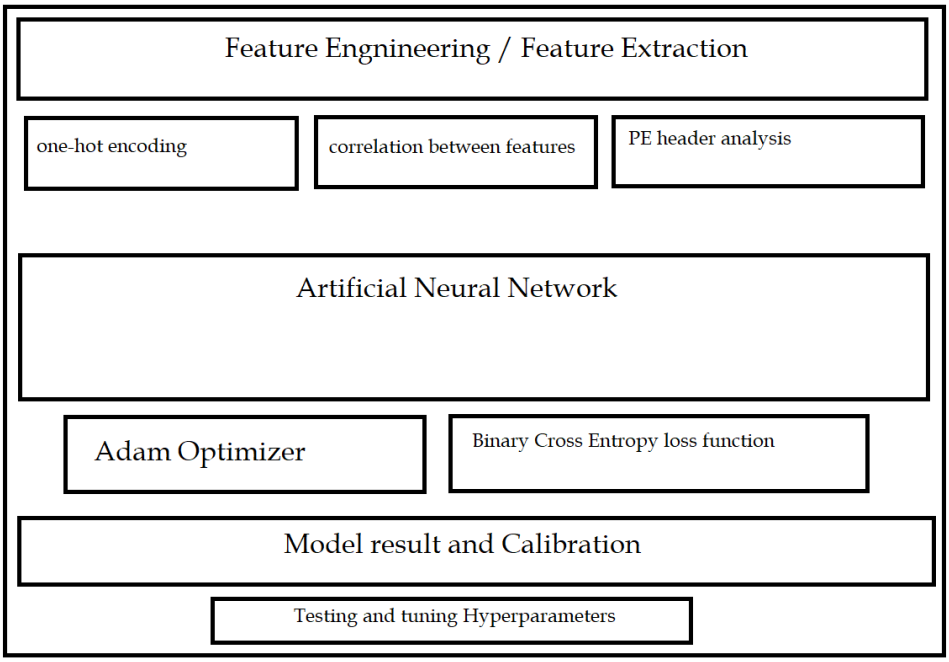
2. Previous Work in the Domain (Current state of the art)

For deep learning based CNN approaches to be used for the task of malware detection, the binary files are usually represented in a two dimensional array. This would make it easier to allow and use the benefits of Convolutional Neural Networks, such as the most important one, which compensates for translational variance. A concept borrowed from Natural Language Processing is used here which is Embedding Space. At a high level, the essence of the embedding space approach is to transform the n length binaries into two dimensional n x m matrix where each row is a vector associated with an instruction. The use of 1-dimensional Convolutional Neural Networks has significant performance advantages. The usage of Term Frequency-inverse document frequency (TF-IDF), is a measure which as been used extensively in the field of information retrieval. It uses a two-part formula to assess a phrase's relative commonness, with the elements capturing the frequency of the term while accounting for its information richness. [2]
Santos et al.[3] Presented semi-supervised learning to detect unknown malware in 2011 after deducing that supervised learning needed tagged data. Santos et al revisited the frequency of the emergence of operating codes in 2011. They employed the information gain approach to pick features and a variety of classifiers, including DT, KNN, Bayesian Network, and Support Vector Machine (SVM), with SVM outperforming the others with 92.92 percent for one opcode sequence length and 95.90 percent for two opcode sequence length. Shabtai et al. Eemployed the opcode n-gram pattern feature in 2012, using the Document Frequency (DF), G-mean, and Fisher Score methods to find the best feature.
Ban Mohammed Khammas et al. [4] used a variety of classifiers in their method, with Random Forest outperforming the others with 95.146 percent accuracy.
Unstructured characteristics, such as n-grams, were brought to malware detection by certain researchers. n-grams are all substrings of a bigger string in the context of malware detection. In most cases, a string is simply split into N-length substrings. Many research using n-grams features used the same method. The most relevant n-grams features were picked as the input

of the machine learning algorithms from a file scanned by slide window to generate the original n-grams features. Several experiments were carried out in order to find the best combination of fixed length N, number of n-gram features, and classification method.
The results of these trials showed that detection based on these types of traits is unlikely to provide excellent result which is in contrast to previous research.

3. Design and Algorithm

The Design of the whole model is described by the image below:



3.1. Methodology

The full classification framework as shown in the figure consists of three main components. The first component is related to feature engineering and extracts complementary features from the static benign and malicious binaries. The second layer is the deep neural network classifier which has three hidden layers with 64, 128 and 256 nodes respectively using RelU activation function and the output layer having sigmoid as activation function.

3.1.1 Feature Engineering

Entropy / Histogram Features

We first start of by plotting a scatter plot graph, that will represent the relationship between different variables we have on the data-set. The features represent the data points on a two-dimensional plane or on a Cartesian System. The independent variable or attribute is plotted on the X-axis while the dependent variable is plotted on the Y-axis. Since we have a large volume of data we have a trend line or line of best fit.

To better visualize the data presented to us by the dataset, we have used the dabl library also known as the Data Analysis Baseline Library which is especially used to make supervised machine learning easier and reduce boiler plate for common tasks. We can now plot the categorical variables and have a better understanding of the dataset. Since the dataset contains binary file information of Windows executables some features can be ignored (since they are same) or can be omitted if any null values are found while feature engineering. The dabl library allows seeing correlation between benign and malicious files for each of the features.

We have also used linear Discriminant Analysis which helps immensely to perform dimensionality reduction of a dataset. The new variables are chosen in a way that maximizes the linear separability of a certain set of classes in the underlying data [5].

3.1.1.1 Omission of certain features

According to windows documentation on windows executables, every executable must have some certain common properties be it benign or malicious. This kind of features such as “NumberOfSections” which denotes the size of the section table which follows the headers can be omitted since they are mostly the same for all binary types. Similarly “CreationYear” can also be dropped from the dataset because the date of creation is not really necessary for our neural network model for prediction.[6]

3.1.1.2 Data Preprocessing

Since we are working with windows executables, binary packers come into an important role in here. The type of packer can determine whether the binary is malicious or benign. We used the preproceeing function from sklearn library to split them into different features for different packers. Since packer_type in the dataset is an object (i.e a categorical variable). One hot encoding is used to achieve this.

3.1.1.3 Labeling

We need precise labels for our malware and benign-ware files to train and evaluate our model with minimal false positive rates. We do this by putting all of our data through its paces.

Through VirusTotal, this scans the binaries for malware using 55 different engines. We split the test train model by 20:80 using sklearn’s preprocessing. Then Standard Scaler function was used to standardize the features by subtracting the mean and then scaling them to unit variance. [7].

3.1.2 Neural Network Model

To solve the classification problem, we use an Artificial Neural Network using the Keras module from the Tensorflow API. It is a sequential model having the three

hidden layers, one input and one output layer. The three hidden layers have RelU activation function since it converges much more quickly and reliably than training a deep network with sigmoid activation in the hidden layers [8]. The hidden layers have 64, 128, 256 nodes respectively. The input layer also has activation function of RelU. The output layer has one node (since it's a binary classification problem) and has an activation function of sigmoid.

3.1.2.1 Cost Functions and Optimizers

We have selected the industry standard, Adam optimizer and Binary Cross Entropy as the loss function which is defined by

$$\text{Loss} = -\frac{1}{\text{output size}} \sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i)$$

Since our output is using sigmoid activation function, binary cross entropy seems to be a perfect fit as it is the only activation function compatible with it.

3.1.2.2 Model Learning

Setting the batch size to 10 and epochs to 100 we have a accuracy score of 98.74%. Further increase in batch size and epoch levels is not observed to bing any changes to the acquired accuracy score which is marked against the test dataset.

Thereafter suitable metrics are used to get the result.

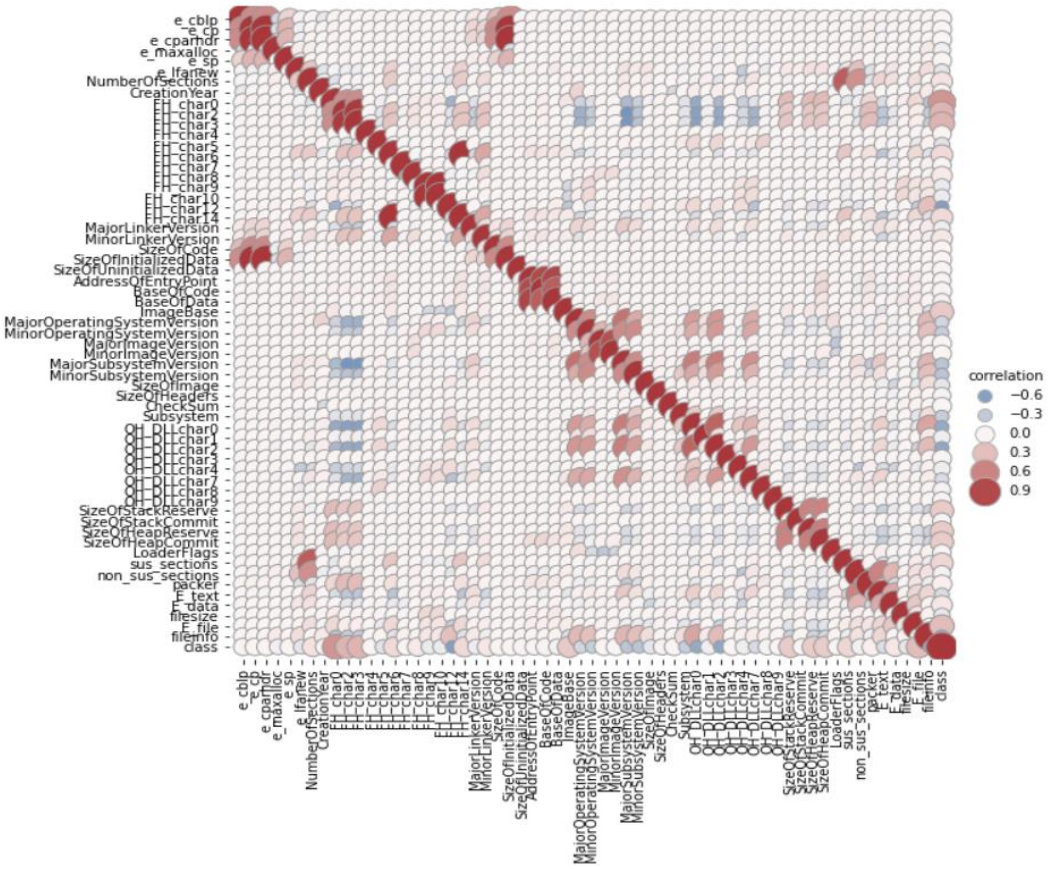
3.2. Results

The Dataset used for the research is Classification of Malwares (CLaMP)[9].

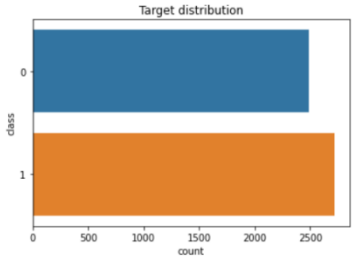
3.2.1 Exploratory Data Analysis

For any model to have high percentage of accuracy it is necessary to explore the features and use a dimensioanlity reduction reduction concept if needed to wrangle the data so that the model does not get confused. Here we have broken down the steps below and using the inferences we get from the data visualization we apply them on the dataset then feed it into the model.

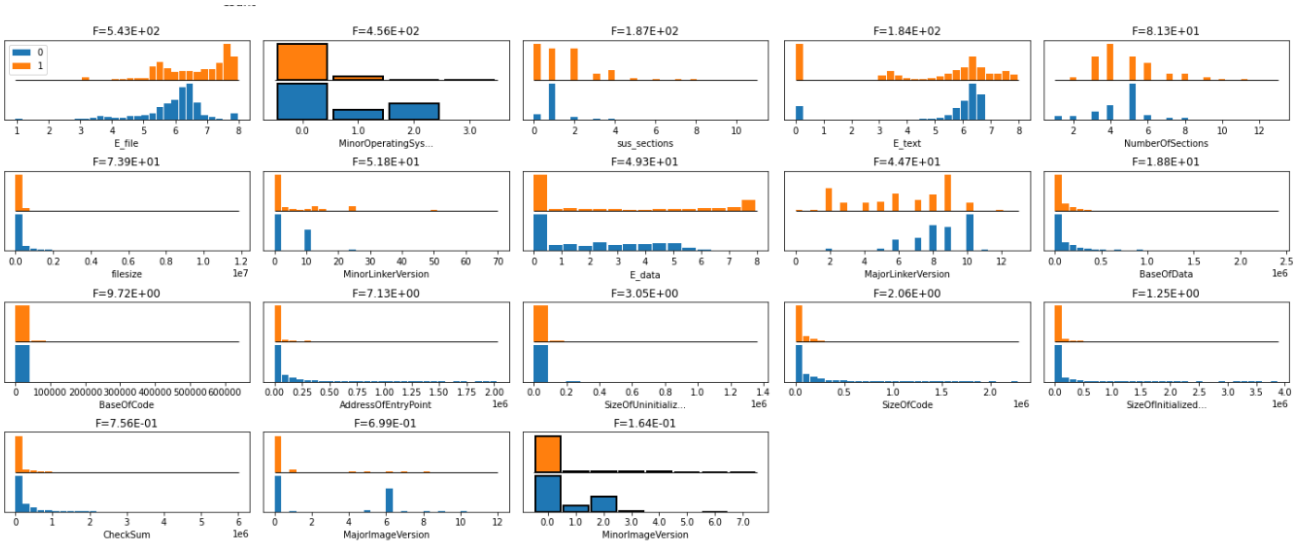
After the training has been completed, graphs are plotted to showcase the effectiveness and the drawbacks of the model.



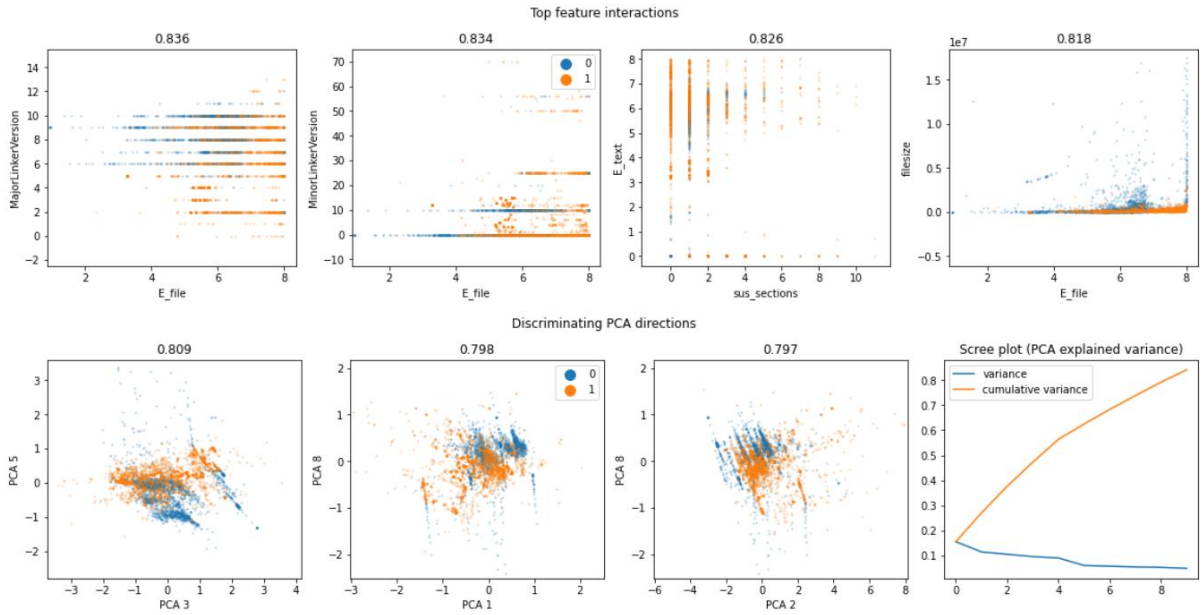
As we can clearly observe from this, a line of best fit is seen. Using the dabl library we can plot the classes available in the dataset (now converted to a dataframe). With 0 being benign and 1 being malicious



We have all the features being compared individually using the same 0-1 scale to explore benign malicious distribution



Conviniently the Principle Component Analysis and Top feature extraction can also be plotted (which is done already by the dabl library)



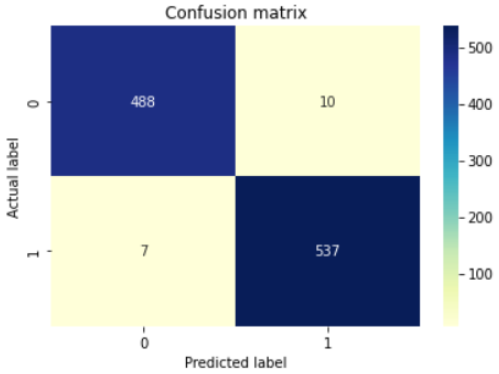
One important thing to notice here is Top feature interaction, which gives certain hints about how the classification can be predicted from the dataset.

The packer_type in the dataset being a categorical variable has to be one hot encoded to be able to be used by the model.

The model as discussed above is implemented using keras.

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_16 (Dense)            (None, 61)                3782

 dense_17 (Dense)            (None, 64)                3968

 dense_18 (Dense)            (None, 128)               8320

 dense_19 (Dense)            (None, 256)               33024

 dense_20 (Dense)            (None, 1)                 257

=================================================================
Total params: 49,351
Trainable params: 49,351
Non-trainable params: 0
_________________________________________________________________
```

Thereafter training the model and testing it against the test dataset and plotting the heatmap of the result.



We have the confusion matrix plotted with the help of the seaborn library. This figure shows that our model successfully predicted 488 binaries which are benign and 537 binaries as malicious. However it is also seen that the model wrongly predicted 17 binaries.

Finally using sklearn metrics we get the accuracy score, precision and f1-score

```
#accuracy score
print(accuracy_score(y_test, y_pred)*100)
```

```
98.36852207293666
```

```
#classification report_final
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       498
           1       0.98      0.99      0.98       544

    accuracy                           0.98      1042
   macro avg       0.98      0.98      0.98      1042
weighted avg       0.98      0.98      0.98      1042
```

The model provides over 98% accuracy using Artificial Neural Network on the given dataset.

4. Your Contribution

– We present an efficient way of malware detection using Deep Learning. There is a lot of scope for the growth of Deep Learning in this century and the developments in this field are rapid. With growing developments much more accuracy and precise detection of Malwares can be done.
– As Malware authors get creative and malwares became more and more dangerous, behavioral analysis will be ineffective. This is where Deep Learning will come into a greater effect and as progress in hardware is made, the computational overheads and complexity will be reduced making this technique a perfect matches for embedded system, IoT and large scale process where computation matters the most
– The proposed model of malware detection is also independent of the complexities that Anti-Virus software has and can be deployed quickly.

5. Conclusion

In this paper we have introduced a deep learning based – artificial neural network to detect Malwares. The model was successfully able to achieve an accuracy rate of more than 98% on the Classification of Malware dataset. We have also shown that this model can be compiled and deployed under the most modest of hardware i.e. the model does not require a high processing power and is hardware friendly hence economical and practical.

The binary classification problem of detecting malwares from benign binaries was attempted using Neural Networks. In this model we have three hidden layers having RelU activation and the output having sigmoid activation. Adam optimizer and binary cross entropy loss function is used to minimize the cost.

However it is to be noted that though the model achieves high accuracy compiling the model with bigger volumes of dataset can lead to higher false positives and an overall less accuracy. No matter what the depicted restrictions, we trust the layered methodology of neural network gives a significantly higher accuracy than all other Machine Learning Algorithms in practice today.

5.1 Future Work / Potential Research

The Research can be further extended by training and testing the model on much bigger datasets where samples had been collected for over five years and then compare the accuracy of detection.

5.2 Software and Data

The software and datasets are publicly available in the github repository link below:
https://github.com/sakai026/ANN-based-Malware-Detection/tree/master

6. References

[1] Joshua Saxe, Konstantin Berlin, Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features, 2015.

[2] TF-IDF Vectorization from sklearn.feature_extraction documentation, scikit-learn.org

[3] Igor Santos, Yoseba K, Penya, Jaime Deveda, Pablo Garcia Bringas, N-grams-based File Signatures for Malware Detection, Proceedings of the 11th International Conference on Enterprise Information Systems, Volume AIDSS, Milan, Italy, May 6-10, 2009

[4] Ban Mohammed Khammas, Ransomware Detection using Random Forest Technique, Al-Nahrain University, Iraq, 2020

[5] Reza Mirzazadeh, Mohammad Hossein Moattar, Majid Vafaei Jahan, Metamorphic Malware Detection using Linear Discriminant Analysis and Graph Similarity.

[6] PE Format, Win32 Apps, docs.microsoft.com

[7] Standard Scaler from sklearn.preprocessing documentation, scikit-learn.org

[8] Ke He, Dong Seong Kim, Malware Detection with Malware Images using Deep Learning Techniques, 18th IEEE International Conference On Trust, Security and Privacy in Computing And Communication, Australia, 2019

[9] Ajit Kumar, K.S.Kuppusamy, G.Aghila, A learning model to detect maliciousness of portable executable using integrated feature set

[10] Sanjay Sharma, C. Rama Krishna, Sanjay K Sahay, Detection of Advanced Malware by Machine Learning Techniques

[11] M Yeo, Y.Koo, Y.Yoon, T.Hwang, J.Ryu, J.Song, C. Park, Flow-based Malware Detection Using Convolutional Neural Network, University of Seoul, South Korea, 2018

[12] Westyarian, Yusep Rosmansyah, Budiman Dabarsyah, malware Detection on Android Smartphones using API Class and Machine Learning, School of Electrical Engineering and Informatics, Institut Teknologi, Bandung, Indonesia, 2015

[13] Zhiyang Fang, Junfeng Wang, Jiaxuan Geng, Xuan Kan, Feature Selection for Malware Detection Based on Reinforcement Learning, College of Computer Science, Sinchuan University, Chengdu, China, 2018