Zero-day Detection: Classification Experiments on Malware with Small Sample Size

1st Everette Li
San Jose State University
San Jose, U.S.
everetteli12@gmail.com

***Abstract*—"Zero-day", in computer security, refer to the situation when the computer system has no knowledge on a possible incoming attack. The "zero" here is indicating that no previous warning or flag has risen for a security system to prepare. This paper conduct experiments on "Zero-day" malware classification. The "security system" here is a Deep Neural Network model that has been trained on detect 3 different types of malware - Zbot, Winwebsec and Zeroaccess. The experiment focus on testing the robustness of the DNN model under small amount of training data. In order to simulate the real-world "Zero-day" scenario where only a few data is available.**

I. Introduction

With the help of artificial neural network, and the right feature extracted, most of the neural network malware classification algorithms do have good performance. However, the robustness of a neural network model replies heavily on data. The quality and the quantity of training data are straightly co-related with the model performance. For the model, the less data it seen the worse accuracy it obtained. In order to test the sustainability of Deep Neural Network on data shortage, experiments has been introduced to push the data boundary and observe the model's behaviour. Two types of model has been selected for the "Zero-day" experiment - the traditional DNN model, and the LSTM DNN model. Python has been used through all the experiments. Section II will go over how the raw data looks like and how raw the data has been processed. Section III explain the setup of different experiments and the results from the them. Section IV discusses the interesting observation in the results. In the end, a conclusion section will summarise the paper.

II. Data and Preprocessing

A. Raw Data

The original malware data comes in Opcode text files. Each text file contains a sequence of Opcodes which the malware will perform during the execution. The Opcode turns out to be a good signature of a malware. Though the Opcode representation of malware is reasonable, many of these files are long and "noisy". Malware creators knows that people are using such detection technique so they start putting meaning less operations in their malware to confuse the detection. Preprocessing needs to be done in order to make full use of these malware Opcodes.

B. Data Preprocessing

There are two goals for this preprocessing. One is to convert string of Opcode into numerical data. The other is to abstract the feature of a malware out of the noisy opcode. The preprocessing follows the steps below.

1) Step 1: For each file, the Opcodes with highest token frequency will be selected to form a list. This list then become the representation of that file. At the end of this step, a list of text files will turns into a list of list with high frequency Opcodes. The list of all Opcode lists then become $MatrixA$ which will be used in the later steps.

2) Step 2: For each Opcode in the $MatrixA$, put down the token frequency of that Opcode within its file. This yield $MatrixB$.

3) Step 3: Encoding $MatrixA$. This will map each Opcode to an integer. $MatrixC$ is the encoded Opcode matrix.

4) Step 4: For each encoded Opcode, multiply it with the token frequency. That is performing an element wise multiplication between $MatrixB$ and $MatrixC$ to obtain $MatrixD$.

5) Step 5: Scale the $MatrixD$ to get $MatrixE$.

After these 5 steps, the data with Opcode text file is now being transferred into a numerical data with its feature obtained. At the end, $MatrixE$ will be shuffled and pair with target in one-hot encoding of the malware type. Three datasets have been generated from the raw data as shown in Table I. Opcode means number of high frequency opcodes selected from each malware file.

TABLE I
DATASET INFO

	small	medium	large
opcode	5	10	13
length	2400	2400	2400

III. Experiment

The experiments have been designed following the control variate approach. The ”Number of Opcodes” case variate on high frequency opcodes length. It horizontally compare the three datasets and use the accuracy of the model to benchmark the datasets. The ”Model Compare” focus on the performance difference between traditional DNN model and LSTM model when they are feed with small dataset. Finally, as a side note, all the experiment runs machine learning algorithm on Tensorflow 2.1.0.

A. Number of Opcodes

This experiment run sample with various Opcode lengths on the same model. Opcode length is the number of high frequency Opcodes that have been selected from the text file. These Opcodes are considered the features or signatures that have been extracted from the malware file. How to pick the right number of Opcodes is then an issue. The original malware file contains noise. Moreover, the malware program, as a computer program, follow the Opcode frequency distribution of a normal program. Just like in English literature the word ”the” is likely to the one with highest frequency no matter it is a novel or an academic writing. In order to determine best the opcode size for ”Zero-day” detection, a reduce sample test will be conducted on all three sample types - small, medium, large. Small sample select 5 Opcodes from each file. This is 10 for medium and 13 for large. The configuration of the DNN model is in Table II.

TABLE II
DNN Model

Loss Function	categorical crossentropy	
Optimizer	ADAM	
Epoch	50	
	Number Nodes	Activation
Layer 1 Dense	varies on Opcode size	tanh
Layer 2 Dense	200	relu
Layer 3 Dropout	set drop out rate 0.01	
Layer 4 Dense	3	softmax

1) small: This experiment use 5 Opcodes each file to conduct a reduce sample test. The overall result is demonstrated as figure 5. The figure 6 and 7 shows the confusion matrix on 2400 and 30 small samples. From the figure, one can easily observe that the whole model loss its accuracy with a high false positive prediction rate on the ”zeroaccess”.

2) medium: This experiment use 10 Opcodes each file to conduct a reduce sample test. The overall result is demonstrated as figure 8. The figure 9 and 10 shows the confusion matrix on 2400 and 30 medium samples. The overall result of medium sample on reduce sample test follows the same tendency as the small sample. The confusion matrix on 30 samples indicating that using medium sample will slightly improve the prediction outcome.

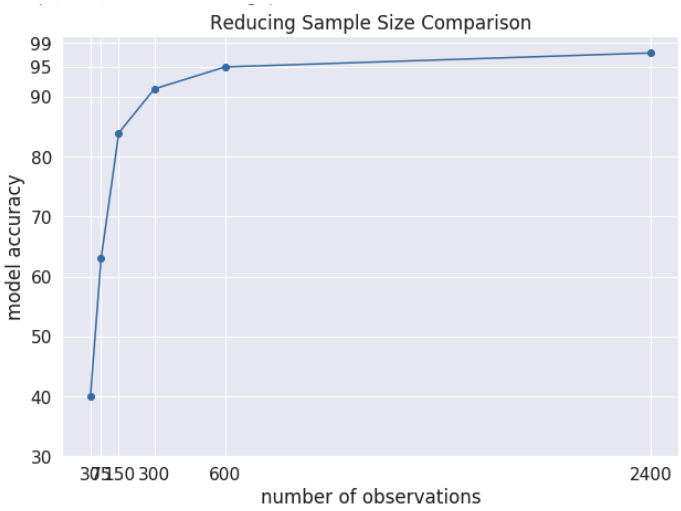

Fig. 1. DNN on Small sample

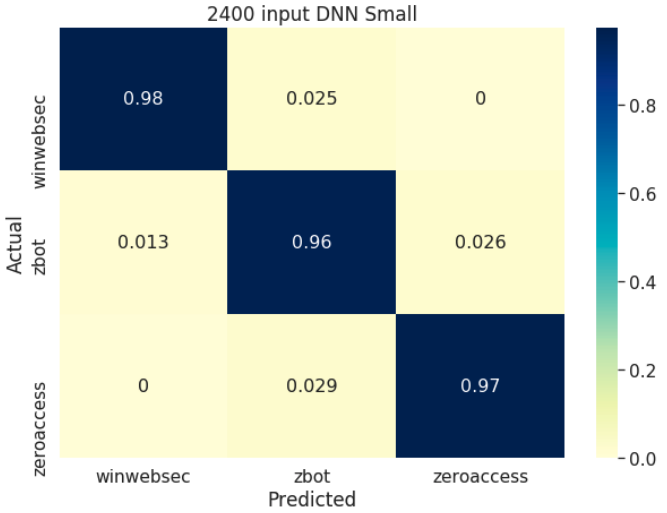

Fig. 2. Confusion Matrix: DNN Small 2400 sample

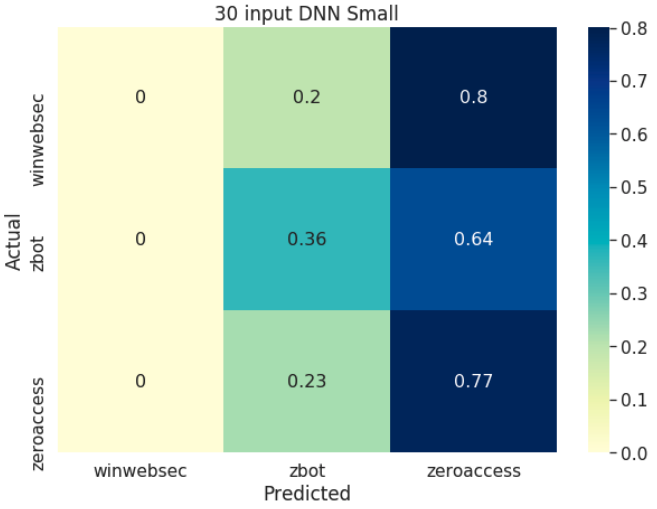

Fig. 3. Confusion Matrix: DNN Small 30 sample

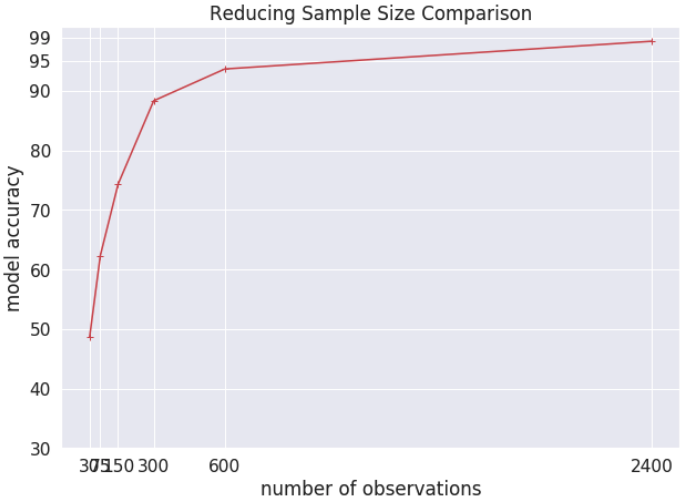

Fig. 4. DNN on Medium sample

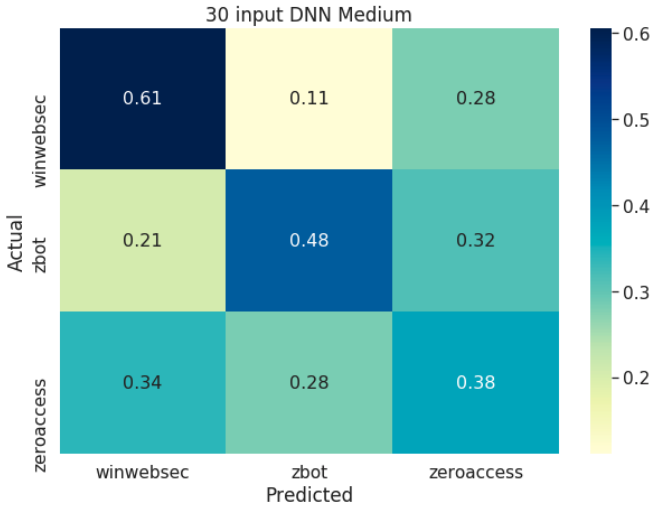

Fig. 6. Confusion Matrix: DNN Medium 30 sample

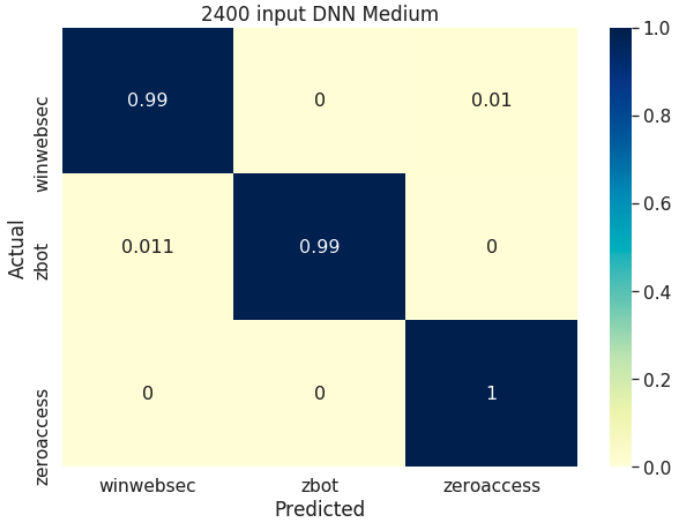

Fig. 5. Confusion Matrix: DNN Medium 2400 sample

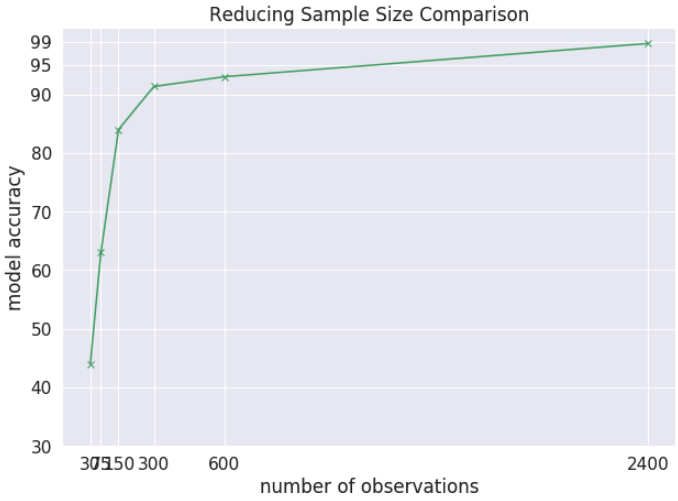

Fig. 7. DNN on Large sample

3) large: This experiment use 13 Opcodes each file to conduct a reduce sample test. The overall result is demonstrated as figure 11. The figure 12 and 13 shows the confusion matrix on 2400 and 30 large samples. The confusion matrix on 30 large sample is then repeating the similar false positive pattern as small samples. Just by observing the experiment outcome, the medium size sample with 10 Opcodes is the est option among the three. However, for the overall performance, all three model shows the same curve of accuracy reduction as shown in the figure 14. The accuracy detail is given in Table V.

B. Model Compare

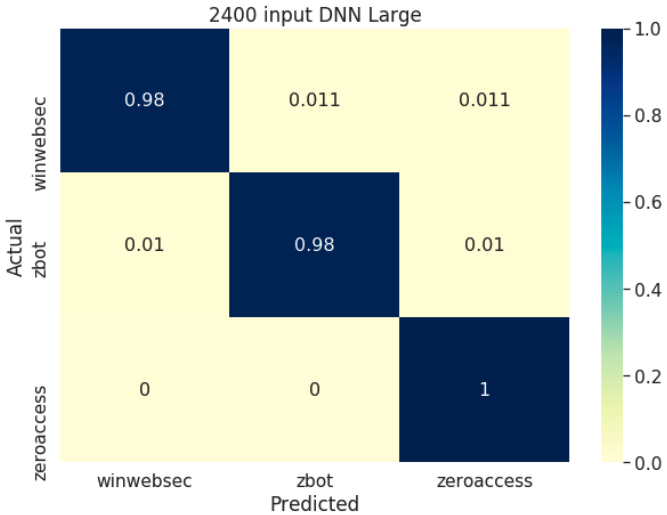

Fig. 8. Confusion Matrix: DNN Large 2400 sample

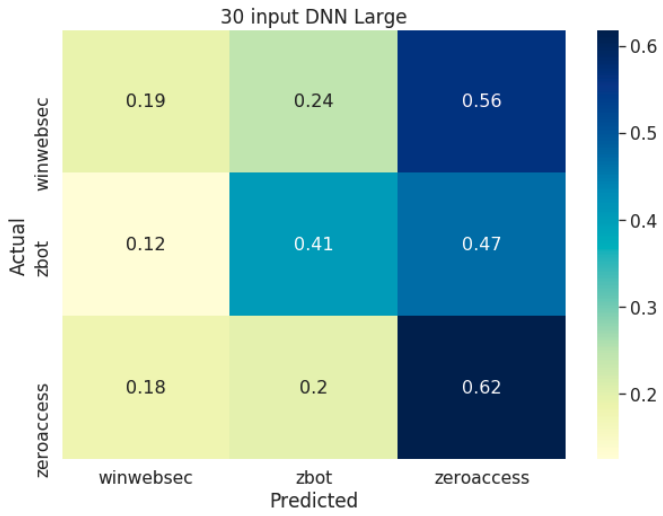

Fig. 9. Confusion Matrix: DNN Large 30 sample

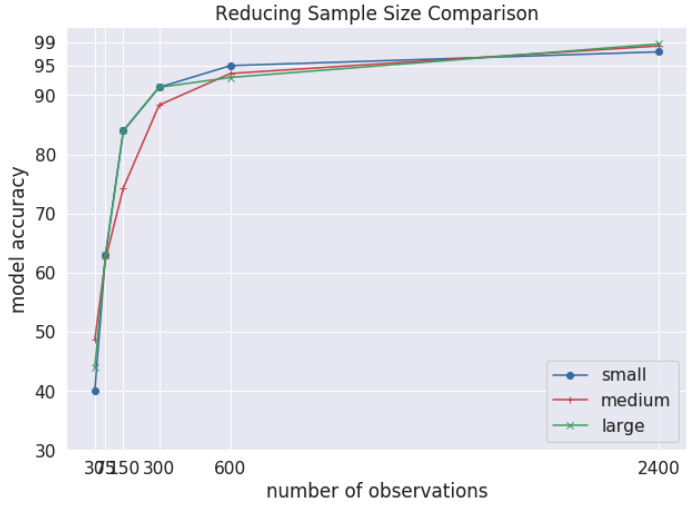

Fig. 10. Small vs Medium vs Large

TABLE III
THREE SAMPLES REDUCE SAMPLE ACCURACY

Sample Size	Small	Medium	Large
2400	%97.33	%98.33	%98.67
600	%95.00	%93.67	%93.00
300	%91.33	%88.33	%91.33
150	%84.00	%74.33	%84.00
75	%63.00	%62.33	%63.00
30	%40.00	%48.67	%44.00

1) LSTM: This experiment uses the medium dataset on tested on LSTM model. LSTM(Long-Short Term Memory) model is a well-known type of RNN(Recurrence Neural Network). Each LSTM cell has a inner weight matrix that integrate the previous learning outcome with the next incoming data. LSTM is popular dealing with sequential data. Since the malware Opcode is a type of sequential data. It is worth given it a try. Table III here is the LSTM model used in the experiment. It has an accuracy of %99.00 when training on full 2400 data from the training set. As the Figure 1 is showing the model obtain a high accuracy with low false positive rate. This proves that LSTM is suitable for malware classification.

TABLE IV
LSTM MODEL

Loss Function	categorical crossentropy	
Optimizer	ADAM	
Epoch	80	
	Number Nodes	Activation
Layer 1 Dense	10	tanh
Layer 2 LSTM	200	relu
Layer 3 Dropout	set drop out rate 0.01	
Layer 4 Dense	3	softmax

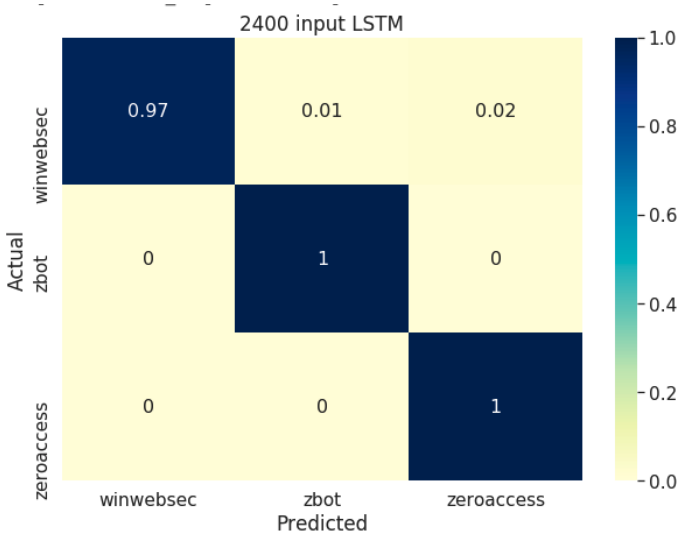

Fig. 11. Confusion Matrix: LSTM with 2400 inputs

2) Compare LSTM and DNN: This experiment compare LSTM with normal DNN. Both models will be run a reduce sample test where training sample will reduce in size for each round. In the end, A comparison will be done. The traditional DNN model used here is the same one from Table II but with 80 epochs. In this reduce sample test, the initial size of training data is 2400, the later sample size follow the list below.

$$[2400, 600, 300, 150, 75, 30]$$

The DNN and LSTM accuracy is shown in Table IV for comparison. As the figure 2 is showing, LSTM model in general obtain higher accuracy in compare with traditional

TABLE V
LSTM REDUCE SAMPLE ACCURACY

Sample Size	DNN Model Accuracy	LSTM Model Accuracy
2400	%98.00	%99.00
600	%94.33	%96.67
300	%90.67	%94.33
150	%83.00	%91.67
75	%75.33	%85.33
30	%50.00	%68.33

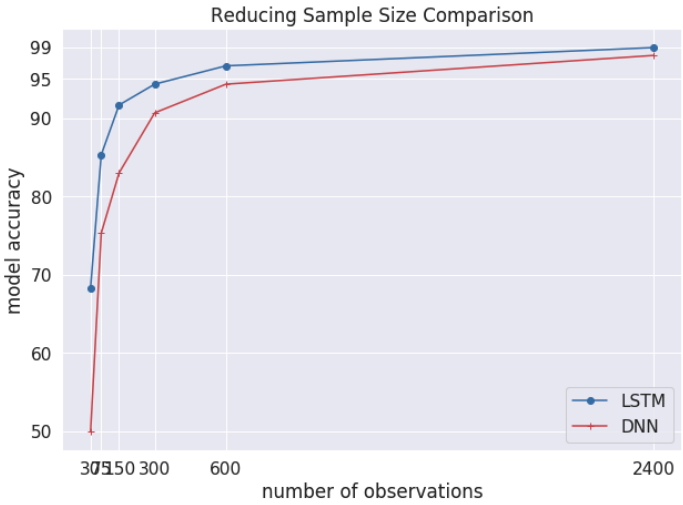

Fig. 12. Reduce Sample Test: LSTM v. DNN

DNN. The confusion matrix of both models on 30 samples are demonstrated in figure 3 and 4. It is straight forward that the LSTM predictions have lower false positive rate than the DNN predictions. Though the false positive rate is generally high for ”zeroaccess” when it comes to low sample size.

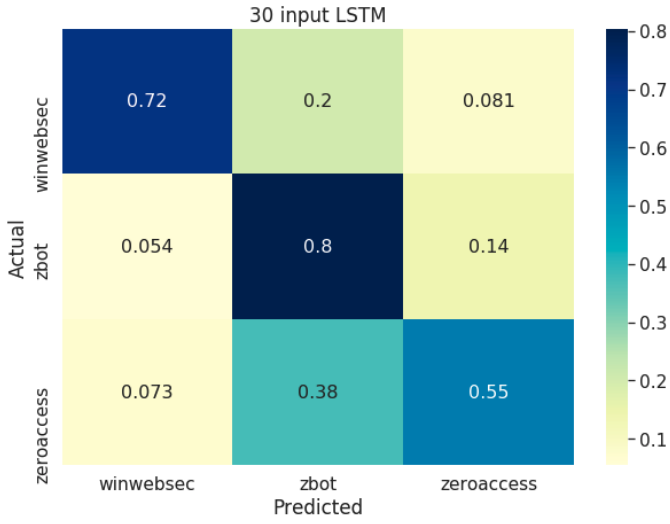

Fig. 13. Confusion Matrix: LSTM with 30 inputs

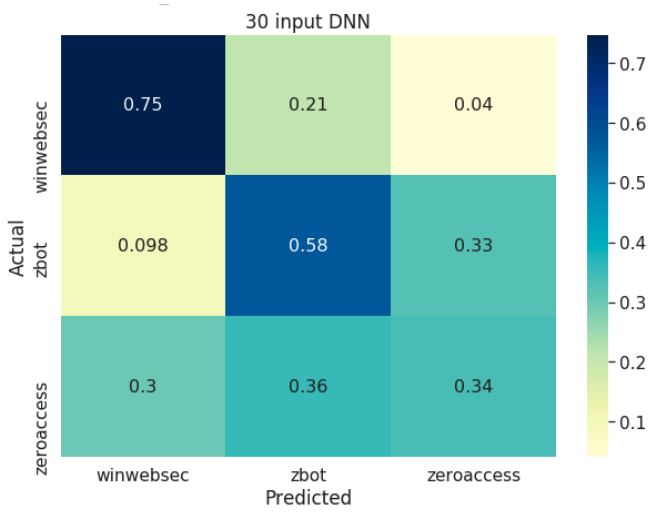

Fig. 14. Confusion Matrix: DNN with 30 inputs

IV. DISCUSSION

Two set of experiments have been conducted addressing the issue of number of Opcodes and model type. In the Opcode size selection. The results are very close to each other. This may indicating that the slight difference between Opcodes selection will not affect overall model classification that much. More extreme cases need to be tested in order to draw fair conclusion on this issue. For example, 2 Opcodes for small, 10 for medium and 30 for large. However, some of the file doesn't contain that many types of Opcode which makes it hard to test on large Opcode sample. In the model selection, LSTM model out perform the traditional DNN in all test. Besides the confusion matrix of LSTM does provide useful evidence supporting the idea that LSTM model works better than DNN on small observation close to ”Zero-day”. Another useful size note is when comparing the accuracy of traditional DNN with 80 epochs to the DNN with 50 epochs. Accuracy does increase when the same DNN is given more epochs. However, increasing numbers of epochs leads to overfitting when the observation is large. Proportionally given more epoch to the model while decreasing the number of observations may yield good results. In the end, due to the uncertainty of the model training process itself, more experiments needs to be done before drawn any reasonable conclusions.

V. CONCLUSION

This paper simulated the ”Zero-day” scenario with reducing sample test. The tests are designed to investigate the robustness of Artificial Neural Network. The experiment focus on two approaches - comparing samples using same model and comparing models using same samples. In the experiments on samples, the results gained from the model are very close to each other. This, combined with the uncertain nature of model training process, makes it hard to come up with reason conclusion. More extreme cases need to be tested in future experiments. In the experiment on model type, LSTM is doing

a better job in general than traditional DNN model with small number of observations. However, the accuracy of the two models are almost the same since they are all close to $\%100$.