Network Anomaly Detection

Tien-Hsuan Wu | Supervisor: Dr. S. M. Yiu

FYP16021

Department of Computer Science, HKU

Introduction

The Internet is expanding and its scale is increasing as more and more devices are connected to the Internet. In November 2016, Google indexed 46 billion webpages, and the annual global Internet traffic was expected to reach 1 zettabyte (10^{21} bytes) by the end of 2016. With the popularization of the Internet, its usage has become necessary in various areas. However, alongside the advantages of Internet use is the increasing potential threat of cyber attack. According to Symantec, there were 54 zero-day (unseen) vulnerabilities discovered each week in 2015, which was twice as many as those in 2014. Therefore, without appropriate security measures, it is likely that the systems will be compromised, causing great losses to individuals and companies. Intruders may gain unauthorized privileges, or simply overload the server and make it unavailable. Both of these may incur huge losses for the system owners.

In our project, we focused on implementing an intrusion detection system with a deep learning model as the backend. We have built a deep learning model for preliminary evaluation for a classification task, and the model was further used to detect anomalies.

Background

Traditional machine learning methods, such as support vector machines and decision trees, are able to find patterns from a set of data. Deep learning is also able to do this. However, what differentiates deep learning from machine learning is the number of learning methods used. In machine learning, typically a single method is used; whereas in deep learning, we can use multiple methods, with each method being based on the result of previous one. The deep structure comes from the multiple steps between the input and the output [1].

Methodology

Our ultimate goal was to identify packet outliers. The process of anomaly detection is shown in Figure 1. We retrieved Internet packets that went through department network as the training and testing data for the normal samples. After the data were collected, we adjusted them so that they were suitable for deep learning. We used the Transmission Control Protocol (TCP) packets and joined the payloads of each stream. Since the model required a fixed size of input, we must truncate payloads that were too large and padded the payloads that were too small. We set the size of payloads to 500 bytes since previous research by Wang showed that most important bytes are located in the first few bytes (see Figure 2) [2].

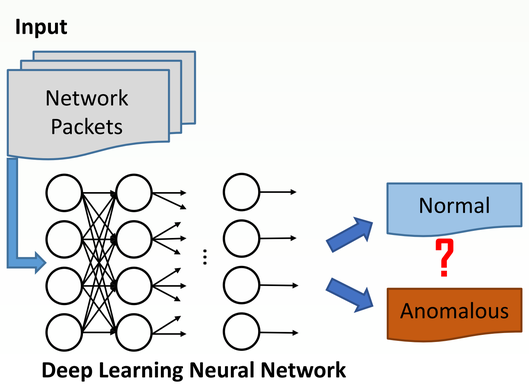

Figure 1: Anomaly detection process

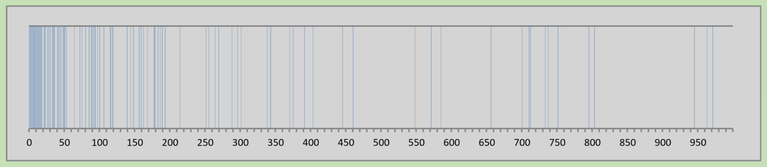

Figure 2: Most important 100 locations of the data

We included some publicly available anomalous data found in the Internet including as anomalies. These data were some scenarios from the CTU-13 botnet dataset prepared by Czech technical university. The preprocessing steps were the same as what we did for the normal samples. Anomalous packets were expected to be distinguished from normal flow in the output.

Experiment Results

We conducted several experiments based on different scenarios and different configurations. The results reported in this poster are based on the dataset that contained the information shown in Table 1. To detect network anomalies from deep learning model, we created a multiple layer perceptron (MLP) and a 1D convolutional neural network (CNN) using the configurations in Table 2. Both models were trained on the final year project server (single core Intel i7 CPU, 2GiB memory). The learning rate was set to 0.01 for the first 100 epochs, and 0.005 for epochs 101-200. The batch size was 32, and different activations were used, including rectified linear units (ReLU), exponential linear units (ELU, α=1) and LeakyReLU (α=0.3, only for MLP). After training for 200 epochs, the results are reported in Table 3. The weight associate with each byte of input is shown in Figure 3.

Type	Train	Test	Description
Normal	3822	764	Samples of 2-hour traffic in HKUCS network
Anomalous	2125	425	CTU-13#1: 1718 samples using Neris bot CTU-13#3: 283 samples using Rbot bot

Table 1: Dataset used for network anomaly detection

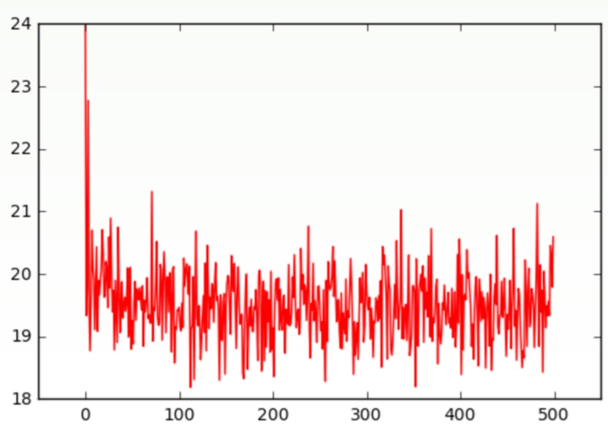

Figure 3: Weight associated with each input byte

MLP
500-byte input
500 units
Activation
Dropout (0.1)
500 units
Activation
Dropout (0.2)
500 units
Activation
Dropout (0.2)
500 units
Activation
Dropout (0.3)
500 units
Activation
Dropout (0.3)
2-way softmax

CNN
500-byte input
200 filters 2x1
Activation (ReLU/ELU)
1D Maxpooling (pool length = 2)
Dropout (0.1)
300 filters 2x1
Activation (ReLU/ELU)
1D Maxpooling (pool length = 2)
Dropout (0.2)
400 filters 2x1
Activation (ReLU/ELU)
1D Maxpooling (pool length = 2)
Dropout (0.2)
2-way softmax

Table 2: Configurations of MLP and CNN

Type	MLP			CNN	
Activations	ReLU	ELU	LeakyReLU	ReLU	ELU
Test Accuracy	**0.9727**	0.9462	0.9672	0.9681	0.9599
Precision	0.9489	0.9770	0.9770	0.9489	0.9790
Recall	0.9604	0.8702	0.9279	0.9604	0.8981
Test Loss	0.1185	0.3053	0.1892	0.0168	0.2025
Train Accuracy	0.9996	0.9960	0.9991	0.9969	0.9918
Train Loss	0.0010	0.0131	0.0017	0.0119	0.0223

Table 3: Results of MLP and CNN

From Table 3, we can see that when we used the same configurations as the ones in protocol classification, the performance of MLP was better than CNN. However, in the last 10 training epochs of CNN and MLP (using ReLU as activations), the test accuracy fell in [0.9681, 0.9781] and [0.9699, 0.9754], respectively, and the accuracy was neither strictly increasing nor strictly decreasing. Note that this situation did not apply to classification task, where the test accuracy of CNN [0.8585, 0.8689] was better than MLP [0.8807, 0.8978] in the last 10 epochs. We concluded that the two models had similar performance and both demonstrated satisfactory level of accuracy in detecting anomalies.

Conclusion

We have shown the performance of various activation functions when applied to network data. Some issues arising from this project can be further pursued. Firstly, it is possible to include more information other than payload as the input of deep learning neural network. Secondly, the model can be adjusted to tradeoff between positives against false negatives. Thirdly, more types of anomalies can be included, and finally, some activation functions can be devised to work on network packets.

References

[1] Goodfellow I, Bengio Y, Courville A. Deep Learning. 2016.

[2] Wang Z. The Applications of Deep Learning on Traffic Identification. Black Hat; 2015.