Deep Learning for Ransomware Detection

Aragorn Tseng[†], YunChun Chen[⋆], YiHsiang Kao[‡], and TsungNan Lin[⋆‡]

[†] Department of Engineering Science and Ocean Engineering, National Taiwan University
[⋆] Department of Electrical Engineering, National Taiwan University
[‡] Graduate Institute of Communication Engineering, National Taiwan University

E-mail : b02505023@ntu.edu.tw, b03901148@ntu.edu.tw, yhkao@ntu.edu.tw

Abstract Ransomware is a kind of malware that installs covertly on a victim's computer or smartphone, executes a cryptovirology attack and demands a ransom payment to restore it. Ransomwares have been the most serious threat in 2016, and this situation continues to worsen. Because of high reward for Ransomwares, more and more Ransomware families appear, and it makes us more difficult to detect them. There are different signatures or behaviors among different families (i.e. Locky, Cerber, Cryptowall to name a few) or versions (i.e. CryptXXX2.0, CryptXXX3.0) of Ransomwares. It will be wonderful if there is a way that can detect potential Ransomware threats.

In this paper, we will use deep learning method to detect Ransomwares. At first, we will introduce how we label the data with different behaviors and what features we choose. Afterwards, we will present our model for detecting various Ransomwares and prevent them from encrypting victim's data. Experimental evaluation demonstrates that our deep learning model can detect the latest Ransomwares in high-speed network timely.

Key words Ransomware, Deep Learning, Machine Learning, Cyber Attack

1 Introduction

According to recent statistics [1], more than 140 million types of various malwares were found in 2015. A large part among them were Ransomwares, which will freeze the system of victims'. They usually initiate attacks through encrypting important files such as photos or documents, and subsequently request users to pay a ransom to recover the damage.

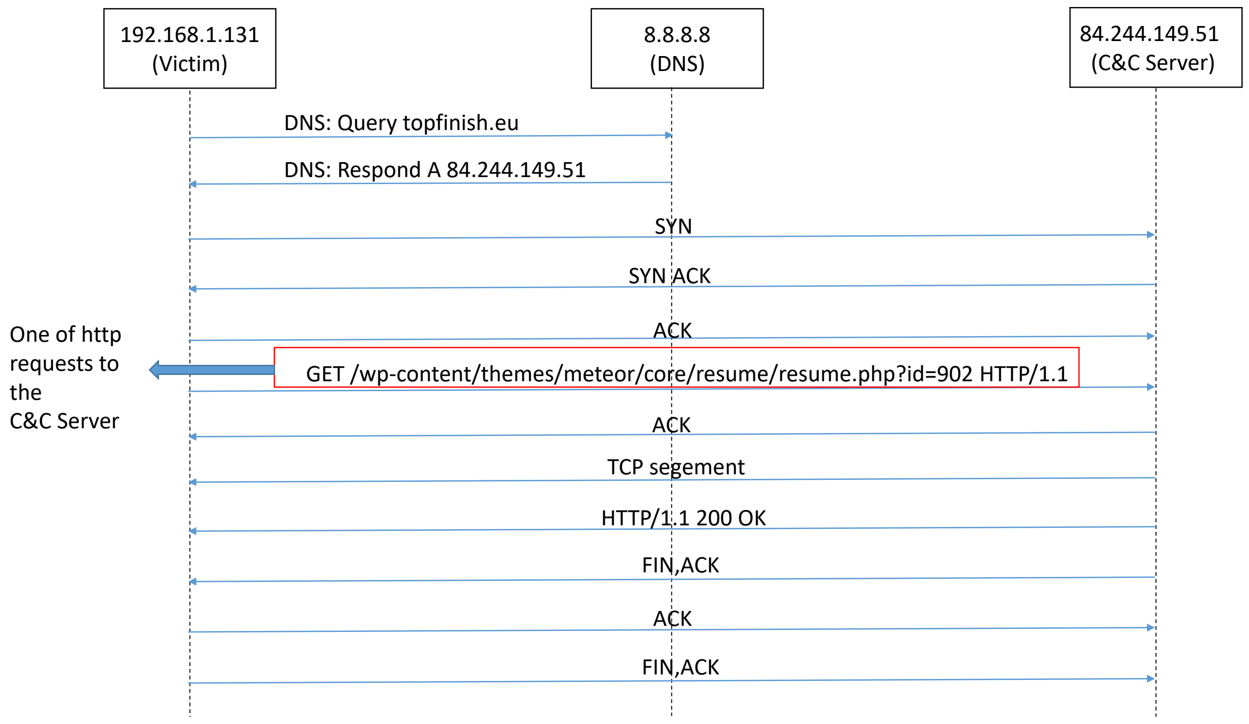
Nowadays, the traditional ways to detect Ransomware in anti-virus software are through matching binary patterns and monitoring APIs. However, these methods are signature-based. If Ransomwares change their behavior or use packers to camouflage, they are no longer being seen until the anti-virus software updates. That is, the anti-virus software cannot defend novel attacks. Besides, the former one has limit that it cannot detect immediately when infection starts.

Based on our observation, the network behavior of different Ransomware families are similar. By taking advantage of this characteristic, we apply machine learning methods to find patterns between different Ransomwares. Doing this, we will be capable of improving flaws in traditional detection system, especially by means of deep learning approach. Deep learning has been widely used in different aspects of fields, and this technique allows models to learn abstract representations directly from low-end data such as images and speeches. With this benefit, we can intuitively append our result from deep packet inspection (DPI) to deep learning model and create a feasible classifier on Ransomware detection.

In this work, we constructed a deep neural network and trained the perceptron with critical payloads selected from packets which were extracted from real network traffic. Experimental result shows that by means of deep learning approach, Ransomware detection is feasible. Furthermore, unlike the traditional signature-based methods which detect almost at last session of the infection process, our model can discover the threat in the beginning of the infection process. At last, this method could easily be implemented on SDN switch, and therefore, has the potential to be well-developed in real world network architecture.

2 Related Works

Ransomware has been even more rampant throughout these years, and lots of methods have been proposed to detect it. However, due to its variability, anti-virus vendors strive to keep the pace with mutative malware variants. As a matter of fact, signature-based mechanisms are prone to be tricked especially when new variants start to spread. For example, only 6% signatures of unique exploit kits (EK), which is one of the main parts in Ransomware, are found in VirusTotal among 3000 different signatures. Moreover, current Ransomwares consist of complex packaging technologies in order to avoid detection, and one of them is to render the static analysis of importing tables and calling obfuscated APIs. Because of these weakness, it is easy to avoid detection of signature-based anti-virus



192.168.1.131 (Victim)
8.8.8.8 (DNS)
84.244.149.51 (C&C Server)
DNS: Query topfinish.eu
DNS: Respond A 84.244.149.51
SYN
SYN ACK
ACK
One of http requests to the C&C Server
GET /wp-content/themes/meteor/core/resume/resume.php?id=902 HTTP/1.1
ACK
TCP segement
HTTP/1.1 200 OK
FIN,ACK
ACK
FIN,ACK

Figure 1. Cryptowall3.0 conversation with C&C

software. If new variants of Ransomwares appear, the anti-virus software may be unable to protect users from their infection. To solve this problem, an innovative method to detect Ransomware was shown as a Dynamic Behavior Analysis [2]. In the analysis, they used API Invocations, Registry Keys, Files/Directory Operations, Dropped Files, and Embedded Strings as their metrics. Another method was mentioned in Nolen′s work [3], the author used a set of behavior related to file operation. In our work, we proposed an architecture using both deep packet inspection and machine learning skills. The idea of seeking from traffic fingerprint was inspired by Arnon′s work [4], which they fed time difference into a LZ78 structure and use K-means algorithm to classify the malware traffic.

3 Ransomware Characteristic

3.1 Ransomware Network Behavior

Ransomware behaved in numerous ways. Some were known to be delivered as attachments from spammed email (i.e. Locky Ransomware), some were forcefully downloaded from malicious pages through ads, and the others launched when exploit kits (i.e. Angler EK) were planted on vulnerable systems (i.e. CryptXXX). When an infection began, the Ransomware will firstly request a DNS query to the DNS server for the C&C information for its configuration file, and then contact those C&C servers. Another traffic (i.e. Locky Ransomware) resulted from Domain Generation Algorithm, which came out about four domains of C&C server depending on users information. After connected to the C&C server, the Ransomware usually transferred the user information, including the version of devices and OS system to the C&C server. In the next step, if the C&C server had confirmed that the OS system could be invaded based on previously sent information, it will send the encryption key and ransom pages in .html or .txt to the victim. In the end, after the encryption process was completed, the total number and the path of encrypted files will be sent to the C&C server.

Additionally, we also observed the process of paying ransom. But most of the process are under Tor browser, and thus we only focus on the encrypting process with C&C communications.

3.2 Packet Selection

The communication between CryptoWall3.0 and its C&C server is shown in Figure 1. In this process, only DNS query and http requests are useful for traffic analysis. They may contain critical features implied, but DNS query merely provide the C&C server with the domain information. Besides, given the fact that C&C domain varies with ease, the http requests are undoubtedly become more and more important. It is for sure that the http requests are the key point to the infection session. For instance, in most Locky Ransomware family infection pcap files, there are many http request headers such as POST /1.php and HTTP /1.1. These payloads usually represent the communication with the C&C server. We choose these essential http requests and raw payloads as our training data. Later in Section 5, how this selection reduces the data size and thus obtain a fine result will be mentioned.

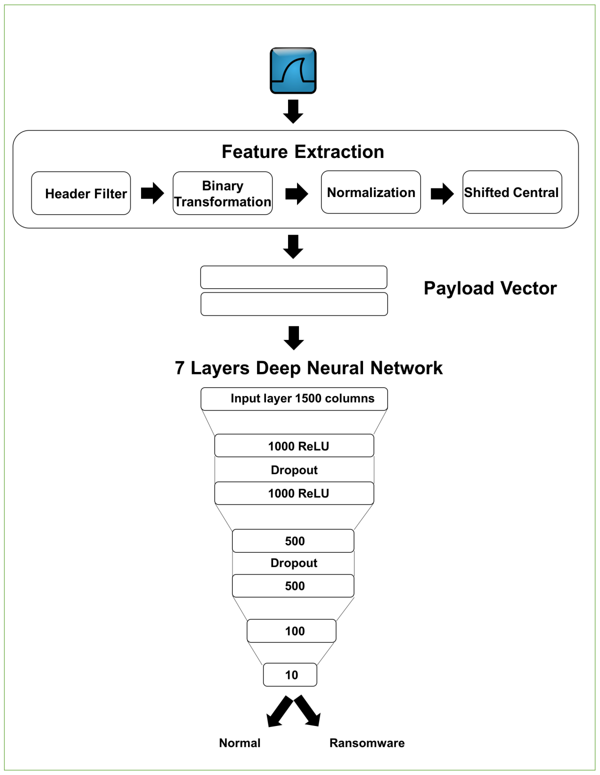


Figure 2. Model Flow Chart.

4 Deep Neural Network

Since our objective is to detect Ransomware and maintain the cyber security from being infected, the model we choose will play a crucial role in the whole process. Because the input data we intend to classify are significantly in a raw level, a Deep Neural Network will be less limited in its ability than conventional machine learning techniques [5]. Furthermore, the reason why we select deep neural network rather than a shallow but wide one is based on the understanding that deep architecture is able to extract and build better features than shallow models.

4.1 Architecture

As shown in Figure 2, we use deep neural network consisting of 7 layers, where the second and fourth layer set to filtrate the noise, and rectified linear units are applied as the activation function in all layers.

4.2 Reducing Overfitting

Dropout method has been proved to make huge improvements on raw dataset such as image recognition [6], and is an efficient way to reduce or even prevent overfitting in neural network [7]. The technique randomly ignores neurons during the training process, which means that their contribution in the forward pass is temporarily removed and any weight updates are not applied to the neuron on the backward pass. The formal description is:

$$O_i^h = f(y_i^h) = f(\sum_{l<h}\sum_{j} w_{ij}^{hl} b_j^l O_j^l)$$

where O_i^h is the output of unit i in layer h, y_i is the output vector of the ith layer, w are the weights and f serves as the activation function. With b_j^l being the Bernoulli random vector, where $\mathrm{P}(b_j^l = 1) = p_j^l$, the dropped neuron will depend on the Bernoulli random vector [8].

Instead of standard weight optimizer based on the l_1 and l_2 norms, dropout seeks weights at each node that are complementary to weights in other nodes. Therefore, it is a feasible way to correct classification throughout the whole neural network.

4.3 Accelerating the Training Process

Rectified linear units (ReLU) have been proved to be more efficient and are capable of speeding up the whole training process significantly [9]. It not only substantially speeds up the training process but also possesses some advantages when comparing to the traditional activation function including logistic function and hyperbolic tangent function [10]. Selecting ReLU is a more efficient way when considering the time cost of training the vast amount of data. Therefore, we refer to neurons with this nonlinearity following Nair and Hinton [11].

4.4 Root Mean Square Propagation

Root mean square propagation (RMSprop) is a technique to automatically adjust the learning rate such that the step size is on the same scale as the gradient [12]. The mathematical form is:

$$V(w,t) = \gamma V(w,t-1) + (1-\gamma) \bigtriangledown Q(w)^2$$

$$w^+ = w - \frac{\eta}{\sqrt{V(w,t)}} \bigtriangledown Q(w)^2$$

where γ is the forgotten coefficient, $\bigtriangledown Q(w)^2$ is the gradient at w. This technique has a similar effect as weight decay [13] since it keeps a moving average of the gradient. In our architecture, we apply γ as 0.8.

5 Data Sets

We captured 23 families of Ransomware pcap files, including CryptoWall, TeslaCrypt, CryptXXX, Locky, CrypMIC, and Cerber. Most pcap files labeled as Ransomware invasion were retrieved from malware-traffic-analysis websites [14] with additional captures provided by the author′s capture in a sandboxed environment. Other normal connections were the capture of real users doing normal tasks. We gathered 155 different Ransomwares, extracted all TCP and UDP connections, and initially got 623823 packet payloads labeled malware and 206732 labeled normal. After applying our observation mentioned in Section 3, we could reduce these high volumes to about 0.3%, that is, 2631 critical payloads of Ransomwares intrusion.

5.1 Divide into Training and Testing

To be sure that we have the ability to predict the existence of novel Ransomware, the training sets include 80 records caught from February 2015 to May 2016, while the testing sets include 77 different Ransomware programs discovered after June 2016.

5.2 Normalization and Central Slide

To extract essential payloads from the capture, other irrelevant traffic will be automatically filtered based on the previous discovered domain knowledge. In addition, for data normalization, Maximum Transmission Unit is used to set each input to equal vector. We apply the widely used standard in Ethernet as size of 1500. In the final step, to reduce the noise caused by different length, all string-like vectors will be applied with Central-Slide, shifting them to the middle of the vector and padding zeroes surrounded the margin.

6 Experiment Setup

All the experiment and pre-processing program were written in Python with the third-party library dpkt to decode the payloads in the original pcap files, and Google′s TensorFlow to build the multi-layer neural networks. In default initialization, we set our learning rate as 0.01, decay in RMSprop as 0.8, and 128 inputs for one mini-batch training.

6.1 Experiment Result

The best performance could achieve a 93.92% overall accuracy on testing sets. We conducted several separate experiments tuning different parameters, and it turned out that the dropout rate is an important factor to determine the overall accuracy on testing since it is potentially more resilient to imperfect data [15], which is common in extracted payloads because it contains unobvious requests.

6.2 Evaluation

The learning curve displayed in Figure 3. visualizes the process of training with different keep rates. We shall clearly see how overfittings were prevented in this scenario as more columns kept in the hidden layer, more rapidly the model fit in the whole noisy data. Table 1. shows the TN and FP ratio in various trained models predicting testing dataset.

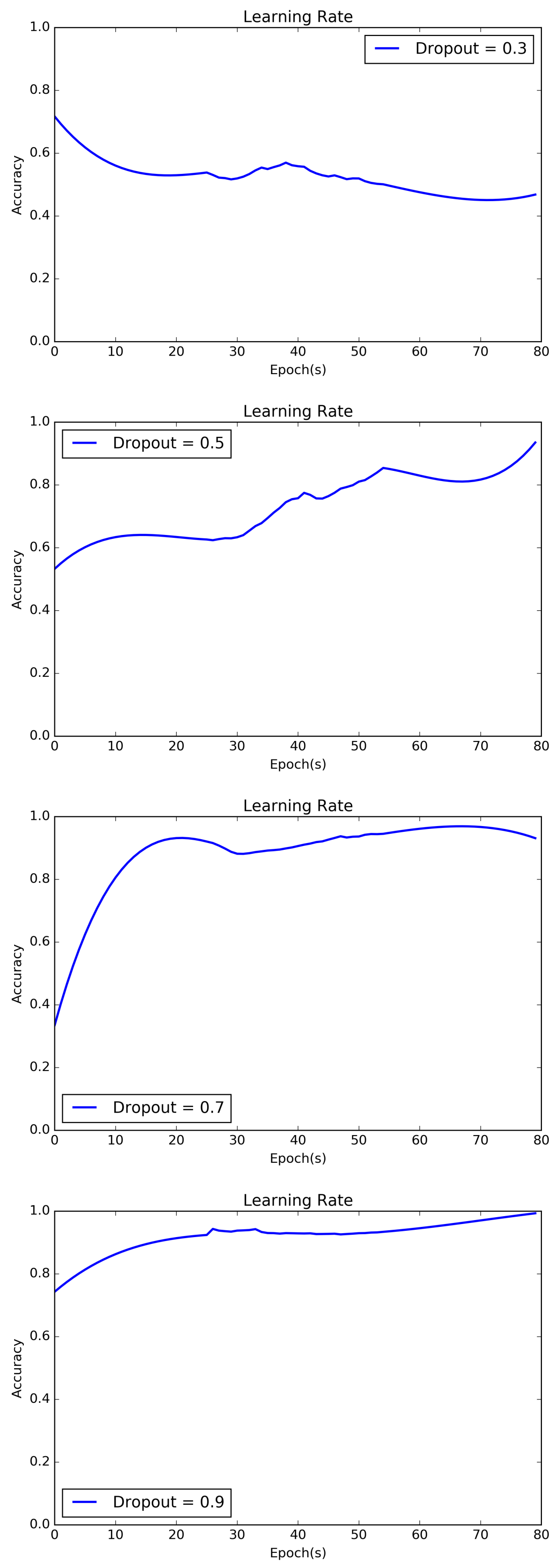


Figure. 3: Learning curves for various dropout rates

6.3 Model Selection

Though in Figure. 4, we can easily choose the one with the highest AC rate. However, for intrusion detection, a false accept is a costly disaster similar to the fingerprint verification [16]. Thus, the TN rate can be tolerated. We will choose the one with the lowest FP rate.

Another observation we conducted after training is to snoop on the testing data in order to find the relation between models. By doing so, we discover that among our final decisions, the recommended model best fits in both training set and testing set, that is to say, by means of dropping out some weights in deep neural network, some advancements are guaranteed. We draw out this answer in Figure 4.

Dropout rate	TN rate	FP rate	AC rate
0.9	0.0418	0.38	0.9041
0.7	0.0684	0.24	0.9233
0.5	0.057	0.1	0.9361
0.3	0.038	0.12	0.9392

Table. 1 Different metrics in testing data sets

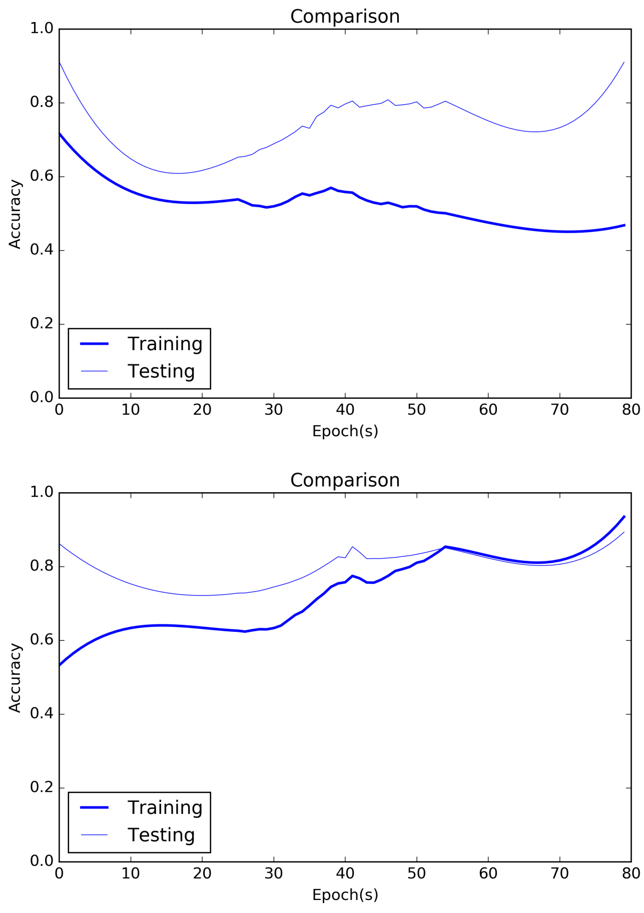


Figure. 4 The learning curve comparison of training to testing. The above one dropped 0.7 and the other dropped 0.5.

7 Conclusion

In this paper, we present a Ransomware detection method. To the best of our knowledge, this is the first architecture to combine Deep Packet Inspection with Machine Learning over network traffic, and it can perform about 93% AC rate from our experiment results. Besides, our model is also a viable and effective approach to detect new variants and families of Ransomware as a complement for anti-virus. It also shows that deep learning is a great technique to use in terms of Ransomware detection.

8 Future Work

Currently, our model still executes in a static state, and thus it is our goal to implement it on the SDN switch, which is set in real network environment, and also use Intel DPDK. For reaching that objectives could we have the confidence to claim that we are able to protect users from being infected by Ransomware.

Another option we may improve is the feature we selected in pcap files. Because infection with Ransomware is a process, we want to design an algorithm that can compare the contents with the previous or following packets to enhance our detection rate.

9 Acknowledgement

The authors gratefully acknowledge the financial support of the Ministry of Science and Technology, Taiwan, R.O.C. through its grants 105-2221-E-002-117-MY3

10 Reference

[1] ′5 highlights from the HPE Cyber Risk Report 2016′, http://techbeacon.com/5-highlights-hpe-cyber-risk-report-2016, Oct. 5. 2016. .

[2] Daniele Sgandurra, Luis Muoz-Gonzlez, Rabih Mohsen, Emil C. Lupu, Automated Dynamic Analysis of Ransomware : Benefits, Limitations and use for Detection, eprint arXiv:1609.03020, Sep.2016

[3] Nolen Scaife, Henry Carter, Patrick Traynor, Kevin R. B. Butler, "CryptoLock (and Drop It) : Stopping Ransomware Attacks on User Data", 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), vol. 00, no. , pp. 303-312, Jun. 2016.

[4] Arnon Shimoni and Shachar Barhom, ′Malicious traffic detection using traffic fingerprint′, https://github.com/arnons1/trafficfingerprint, 2014.′,

Oct. 5. 2016. .

[5] Yann LeCun, Yoshua Bengio & Geoffrey Hinton, Deep learning, Nature, vol. 521, no. 7553, pp. 436-444, May 2015.

[6] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ”Imagenet classification with deep convolutional neural networks”, Advances in neural information processing systems, 2012.

[7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, ′Improving neural networks by preventing co-adaptation of feature detectors,arXiv preprint arXiv:1207.0580, 2012.

[8] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, ”Dropout: A Simple Way to Prevent Neural Networks form Overfitting”, Journal of Machine Learning Research, 15.1, 1929-1958, 2014.

[9] Xavier Glorot, Antoine Bordes, Yoshua Bengio, ”Deep Sparse Rectifier Neural Networks”, Aistats. Vol. 15. No. 106. 2011.

[10] Andrew L. Maas, Awni Y. Hannun, Andrew Y. Ng, ”Rectifier Nonlinearities Improve Neura Network Acoustic Models”,2013 Proceeding ICML, vol. 30, no. 1, September 2013.

[11] Vinod Nair, Geoffrey E. Hinton, ”Rectified Linear Units Improve Restricted Boltzmann Machines”, Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010.

[12] Geoffrey Hinton, Nitish Srivastava, Kevin Swersky, Lecture 6: Overview of mini-batch gradient descent, Coursera Lecture slides ′https://class.coursera.org/neuralnets-2012 -001/lecture′, accessed Oct. 5. 2016.

[13] http://www.malware-traffic-analysis.net/, Oct. 5. 2016. .

[14] Anders Krogh, John A Hertz, ”A Simple Weight Decay Can Improve Generalization”, Advances in neural information processing systems, pp. 950-957, 1995.

[15] Saxe, J., Berlin, K.: ”Deep neural network based malware detection using two dimensional binary program features”. arXiv preprint 2015. arXiv:1508.03096v2

[16] Yaser S. Abu-Mostafa, Malik Magdon-Ismail, Hsuan-Tien Lin, Learning from Data, 2012