Computers, Materials & Continua CMC, vol.62, no.1, pp.399-411, 2020

A Convolution-Based System for Malicious URLs Detection

Chaochao Luo[1], Shen Su[2,*], Yanbin Sun[2], Qingji Tan[3], Meng Han[4] and Zhihong Tian[2,*]

Abstract: Since the web service is essential in daily lives, cyber security becomes more and more important in this digital world. Malicious Uniform Resource Locator (URL) is a common and serious threat to cybersecurity. It hosts unsolicited content and lure unsuspecting users to become victim of scams, such as theft of private information, monetary loss, and malware installation. Thus, it is imperative to detect such threats. However, traditional approaches for malicious URLs detection that based on the blacklists are easy to be bypassed and lack the ability to detect newly generated malicious URLs. In this paper, we propose a novel malicious URL detection method based on deep learning model to protect against web attacks. Specifically, we firstly use auto-encoder to represent URLs. Then, the represented URLs will be input into a proposed composite neural network for detection. In order to evaluate the proposed system, we made extensive experiments on HTTP CSIC2010 dataset and a dataset we collected, and the experimental results show the effectiveness of the proposed approach.

Keywords: CNN, anomaly detection, web security, auto-encoder, deep learning.

1 Introduction

The world is becoming more than ever dependent on digital technology, with vital sectors such as health-care, energy, transportation and banking relying on networks of digital computers to facilitate their operations. The rapid development of big data and cloud technology has affected people's daily lives, and people are more inclined to keep their data in the cloud which is supported by companies. Cloud brings people considerable convenience, but the security services provided by cloud providers are the only data protection that users can rely on. Thus, users are still at a risk of information leakage. Particularly, works in Chen et al. [Chen, Tian, Cui et al. (2018); Han, Tian, Huang et al. (2018); Li, Sun, Jiang et al. (2018); Qiu, Chai, Liu et al. (2018); Tian, Shi, Wang et al.

[1] Institute of Computer Application, China Academy of Engineering Physics, Mianyang, 621054, China.

[2] Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, 510006, China.

[3] School of Mechanical and Electrical Engineering, Heilongjiang State Farm Science Technology Vocational College, Harbin, 150431, China.

[4] Department of Computing and Software Engineering, Kennesaw State University, Kennesaw, GA 30144, USA.

[*] Corresponding Authors: Shen Su. Email: sushen@gzhu.edu.cn;

Zhihong Tian. Email: tianzhihong@gzhu.edu.cn.

(2019); Tian, Su, Shi et al. (2019)] proposed new techniques for Internet of Things and cloud computing while studies in Shi et al. [Shi (2018); Tian, Cui, An et al. (2018); Wang, Tian, Zhang et al. (2018); Yu, Tian, Qiu et al. (2018); Xiao, Rayi, Sun et al. (2007); Du, Xiao, Guizani et al. (2007); Xiao, Du, Zhang et al. (2007); Du and Chen (2008); Du, Guizani, Xiao et al. (2009); Tian, Gao, Shi et al. (2018)] focused on the security problems for Internet of Things and cloud techniques.

According to the statistics of OWASP Curphey et al. [Curphey, Williams and Konda (2017)] in 2017, SQL injection and Cross-Site Scripting are ranked as the first and third in the list of the most critical web application security attack methods. Typical methods for intrusion detection through HTTP request are matching strings which are pre-defined by domain experts. Reports from internet security companies indicted that these traditional ways were not safe. Firstly, it is easy to bypass Web Application Firewall (WAF) by replacing keywords of existing malicious requests or encoding themselves multiple times [Lupták (2011)]. Secondly, extremely large attack pattern set or requests with long lengths consumes lots of computing resources to finish pattern comparing [Liang, Zhao and Ye (2017)]. Protecting the ever-growing attack surface from determined and resourceful attackers requires the development of effective, innovative and disruptive defense techniques [Hendler, Kels and Rubin (2018)].

A nationwide wave of deep learning is sweeping the world after AlphaGo, a computer program by Google with techniques of artificial intelligence, defeated the world champion of board game Go. Deep learning models have received great success in the field of natural language processing (NLP) and computer vision. Gradually, more and more researches concentrate on deep learning models for cyber security. Recent scientific achievements in machine learning in general, and deep learning [Goodfellow, Bengio and Courville (1999)] in particular, provide many opportunities for developing new state-of-the-art methods for effective cyber defense [Hendler, Kels and Rubin (2018)]. Deep learning algorithms are able to extract more comprehensive features from raw data automatically. Compared with time consuming feature extraction manually, it doesn't only save time but also bring significant increasement in performance. In this paper, we propose a new approach for the anomalous URL detection by utilizing auto-encoder and convolutional neural networks. In particular, we will process URLs with removing useless information and tokenizing in the phase of preprocessing. Subsequently, the pretrained auto-encoder will transforms URLs into vectors in order to fit deep learning model's input. Eventually, a well-designed deep model will be used to detect anomalous requests. Experimental results show the proposed approach's capability of detecting anomalous URLs. Specifically, our work makes three contributions:

- We use auto-encoder to do feature representation that converts URLs into vectors. To the best of our knowledge, this is the first time that auto-encoder is used to represent URLs.
- We utilize neural networks to detect anomalous requests is free of feature selection. It is more convenient and can save costs compare with traditional rule-based approaches.
- We propose a system contains a special neural network to detect anomalous URLs

and we empirically evaluate this system. The results demonstrate that this system is capable of detecting potential web attacks with low false alarms and underreports.

The rest of the paper is organized as follows. In Section 2, we review some related works. Next, the details of our system are provided in Section 3. In Section 4, we introduce the dataset we work with and experimental results. Finally, we conclude our work in Section 5, also present a brief discussion for future work.

2 Related works

Numbers of literatures concentrate on web attacks detection have been proposed. Especially, n-grams was widely used to extract sensitive features from the URLs to detect anomalous payloads. Especially, in the work of Zhang et al. [Zhang, George and Shujaee (2016)], an incoming URL will be broken down to a series of 5-grams, G1, G2, …, Gn, using a sliding window and be vectorized. Then, Naïve Bayesian model was trained to learn patterns of normal URLs from these vectors. And the requests will be detected as normal if the probability calculated by system is smaller than the threshold and as anomalous otherwise. Similarly, Rieck et al. [Rieck and Laskov (2006)] used n-grams in their work while they considered a similarity measurement between n-gram sequences additionally.

Besides n-grams, neural networks have been also applied in web attacks detection. Kim [Kim (2014)] has proved that convolutional neural networks have capability for sentence classification. Character-level convolutional neural network (CLCNN) and global max-pooling were used to extract the feature of HTTP request in Ito et al. [Ito and Iyatomi (2018)] for detecting anomalous URLs. In this work, they encoded URLs again with Unicode to filter '%' in raw URL queries. Then, each character is replaced by an 8-bit numeric string. These strings were input into a neural network specially designed to identify whether the request is normal or anomalous. Likewise, Liang et al. [Liang, Zhao, and Ye (2017)] proposed a novel approach to detect anomalous requests. They focused on the phase of feature extraction, and presented an architecture with (Long Short Term Memory neural network) LSTM to select features automatically. And it is different from our work in Luo et al. [Luo, Wang and Lu (2018)] which used LSTM as a classifier. Works in Zhang et al. [Zhang, Li, Zhou et al. (2017)] also concentrated on detecting malicious URL. Different from Zhang et al. [Zhang, George and Shujaee (2016); Ito and Iyatomi (2018); Liang, Zhao and Ye (2017)], Zhang et al. [Zhang, Li, Zhou et al. (2017)] used semi-supervised learning to detecting anomalous requests. Interestingly, the data used in this work are composed of small set of labeled requests and large set of unlabeled requests. Experimental results showed that using unsupervised learning to detect anomalous requests is also effective.

3 The system model

In this section, we describe the detail of the system model for URL detection. Our system frees us form selecting features and takes advantage of convolution neural networks to identify anomalous requests. The model of proposed system is depicted in Fig. 1.

We now proceed to introduce, in Section 3.1, the phase of data processing includes preprocessing and feature representation for URLs, and, in Section 3.2, the details about the structure of deep learning model.

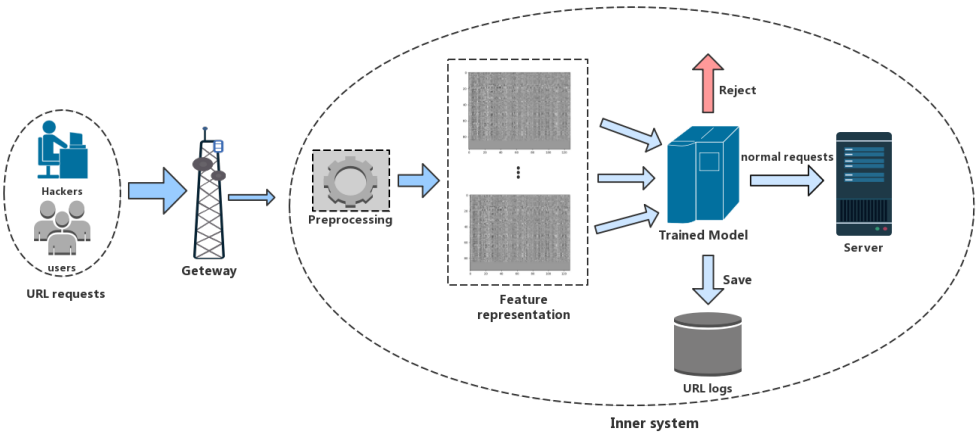


Figure 1: The model of our system

3.1 Preprocessing and feature representation

URLs are textual data and need to be converted into numerical data if we want to take advantage of deep learning. Below, we have a brief explanation of the method for URL preprocessing and feature representation. Since the URL is textual data which has certain syntax, it is natural to utilize techniques of natural language process (NLP) for data analysis. And in the context of NLP, considering text as a stream of characters have gained recent popularity and have been shown to outperform state of art methods [Jozefowicz, Vinyals, Schuster et al. (2016); Zhang and LeCun (2015)]. Thus, we use the same way in NLP to analyze URLs. Initially, all characters will be lowered and the first part of URL (e.g., 'http://xxx.com') will be deleted, because it is useless in web attacks detection. Then, we consider a URL as a sequential text and tokenize it with punctuations and space, and get a dictionary composed of first M words that most frequently appear in all tokenized requests, 'UNK' for words are not in the dictionary and punctuations. Specifically, these punctuations include:

, : . ! ? : ' " / | _ @ # $ % ^ & * ~ + - = < > ( ) { } [ ] \

These punctuations are included in the dictionary because they are primarily used by attackers to structure anomalous requests. In this way, the structural information and semantic information of URLs are reserved. In particular, every word or punctuation is replaced by a one-hot vector and a URL is represented with several one-hot vectors.

In feature presentation, we use these one-hot vectors belong to one URL to train an auto-encoder that is an unsupervised learning model in deep learning to learn the special syntax of URLs and represent them. As is shown in Fig. 2, in feature representation, an auto-encoder takes one-hot vectors $\mathbf{x} \in R^v$ as input. The first part is to map the input vector $\mathbf{x} \in R^v$ to a hidden representation $\mathbf{h} \in R^N$ through a simple mapping $\mathbf{h} = W \cdot \left(\sum_{i=1}^{k} X_i\right) + b$. $W \in R^{V \times N}$ is a matrix that means weights between input layer and hidden layer and $\mathbf{b}$ is a bias vector. So, the representation of the output layer is mapped as $\mathbf{y} = \mathbf{softmax}(\mathbf{h}W' + \mathbf{b}')$. The error needs to be minimized can be formulated as:

$$E = \arg\min_{(W,b,W',b')} \frac{1}{n} \sum_{i}^{\infty} L(x_i, y_i) \quad (1)$$

where L is loss function and every $\mathbf{x}_i$ is mapped to a $\mathbf{y}_i$.

Significantly, every word or punctuation in one URL will be represented as a vector from $h_i = W \cdot X_i + b$. After that, we concatenate these vectors as a URL-matrix and we can get an image view in Fig. 3.

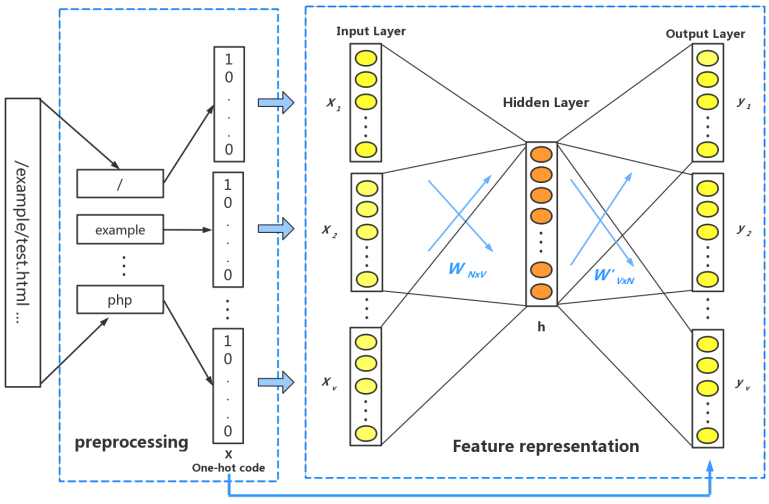
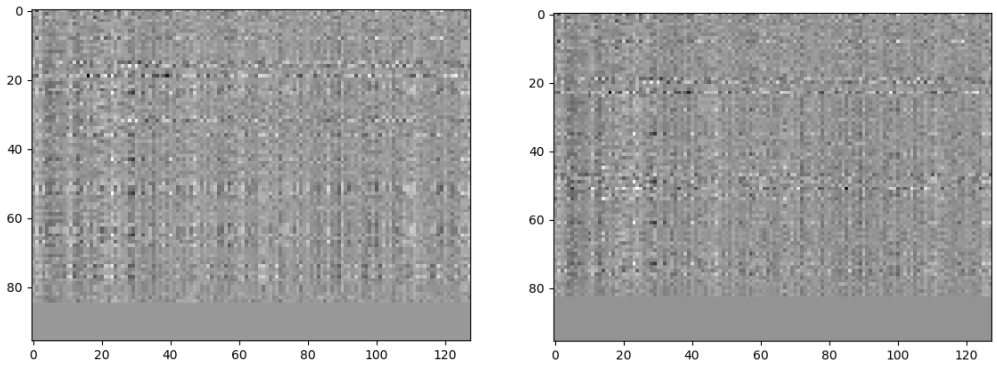
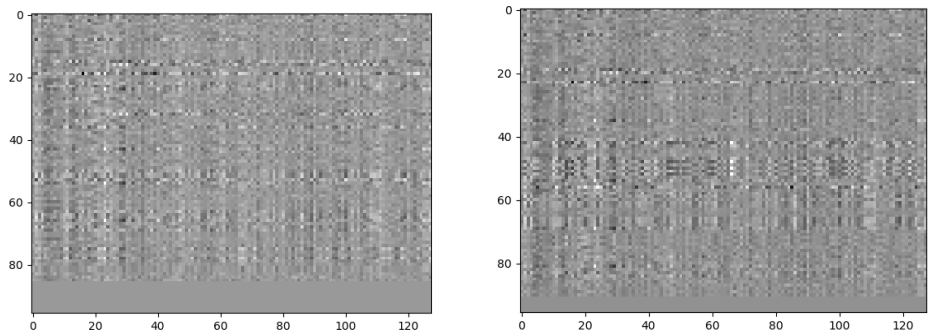


Figure 2: The structure of preprocessing and feature representation. Every word or punctuation in a URL is replaced by a one-hot vector, and all vectors of one URL will be input into the input layer in feature representation



(a) Images of normal URL



(b) Images of anomalous URL

Figure 3: Image view for feature representation of URLs

3.2 Composite neural network

Convolution neural network (CNN) proposed by Lecun et al. [LeCun, Boser, Denker et al. (1989); LeCun, Bottou, Bengio et al. (1998)] is a typical type of deep learning structure that is widely used in computer vision. Convolution operation is the key operation that has high capability of extracting features from an image and, specifically, has been proved to be able to do works for sentence classification in Kim [Kim (2014)]. So, we designed a new structure for detecting malicious URLs with convolution operations. The new structure, we call it Composite neural network block (Comp-block) below, is presented in Fig. 4. Particularly, $\mathbf{X}$ is a matrix that stands for pixels of an image. There are three branches for $\mathbf{X}$. In the first branch (from left to right), there is a convolutional layer which is applied to extract a new feature from a window of k pixels. For example, a new feature S_i is generated from a window of pixels $x_{iii+h-1}$ by

$$S_i = f(w \cdot x_{iii+h-1} + b) \tag{2}$$

where b is a bias term and function f is Relu function [Nair and Hinton (2010)] which is a rectified linear function. In the second branch, there is a max pool layer which takes the maximum value $\mathbf{X}' = \max\{\mathbf{X}\}$ from a window of $\mathbf{k} \times \mathbf{L}$ pixels as a new feature. In the third branch, we keep the input $\mathbf{X}$ invariant because we want to retrain enough useful information that may be filtered in the first two branches. In the end, we add these three branches together and add a max pool layer followed. Most importantly, we set a factor for every branch which will be updated in the training phase. In other words, we don't determine the exact value for every factor but let itself learn a perfect one for every factor in the training phase. So, the final output of Comp-block can be formulated as:

$$\mathrm{H}(\mathbf{x}) = \mathrm{pool}(\alpha \mathrm{F1}(\mathbf{x}) + \beta \mathrm{F2}(\mathbf{x}) + \gamma \mathrm{F3}(\mathbf{x})) \tag{3}$$

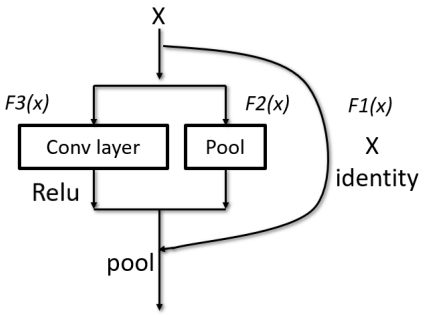


Figure 4: Composite neural network block architecture

Specifically, our network is composed of one convolutional layer, four Comp-block layers and normalization and drop out layer after each, three full connected layers and drop out layers after each. The structure in which normalization layers and drop out layers are omitted is illustrated in Fig. 5. It must be emphasized that feature maps after convolution layer are composed of original feature representation before convolution layer and new feature maps from convolution layer. We did not use Convolution layer but Comp-block in the next layers because convolution layers get a feature map simply from the input and drop some useful information from the input while the Comp-block keeps the original information.

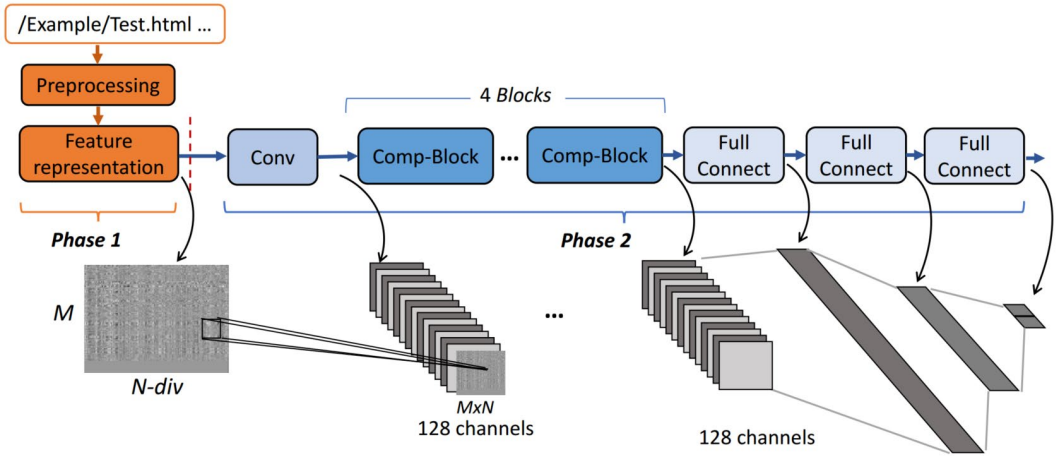


Figure 5: The architecture of neural network in the system

4 Experiments

In order to evaluate the proposed system, we conduct experiments with two datasets while one is a public dataset widely used in web attack detection and another is a dataset created by us. Particularly, we run all experiments under environment with an Intel Core i7-6900k, 4 Kingston DDR4 16G, 2 CUDA-enabled MSI GTX 1080Ti and Ubuntu 16.04. In the following, we introduce the datasets we used and experimental results with a brief discussion.

4.1 Dataset

The HTTP CSIC 2010 dataset includes thousands of web requests automatically generated and was created by web application developed at the "Information Security Institute" of CSIC (Spanish Research National Council). In particular, there are 72000 requests labeled normal and 25065 requests labeled anomalous in the dataset. And the types of attacks in the dataset include buffer overflow, information gathering, SQL injection, files disclosure, XSS (Cross-site scripting), CRLF injection, parameter tampering and so on. Since we focus on detecting malicious URL, we extract the URLs from the HTTP GET requests from the original data and then divide them into a training dataset and a test dataset randomly. The extracted dataset is presented in Tab. 1.

Table 1: HTTP CSIC 2010 dataset

Dataset	Normal	Anomalous	Total
Train	33629	9023	42652
Test	22371	6065	28436

Furthermore, we used another dataset which is collected and sampled from a web application deployed in our lab. Particularly, we used several tools include sqlmap, xssya, vega scanner to generate anomalous samples. Note that there are two parts in the dataset: a large dataset of normal URLs and a set of anomalous URLs that includes SQL injection, Cross Site Script (XSS), command injection. The dataset is showed in Tab. 2.

Table 2: Data collected in our lab

Dataset	Normal	Anomalous	Total
Train	28797	28953	57750
Test	19329	19172	38501

4.2 Results and Discussion

4.2.1 Results

For our experiments, we define detection rate of anomalies (DRA) and detection rate of normal (DRN) as performance metrics in this work. Accuracy is the rate of requests that are classified correctly. The DRA represents the capability of detecting anomalous URLs and the DRN denotes the rate of normal data that detected to be normal from all normal data.

$$\text{Accuracy} = \frac{\text{numbers of all url requests detected correctly}}{\text{all data}} \times 100\% \quad (3)$$

$$\text{DRA} = \frac{\text{numbers of anomalous url requests detected correctly}}{\text{all anomalous url requests}} \times 100\% \quad (4)$$

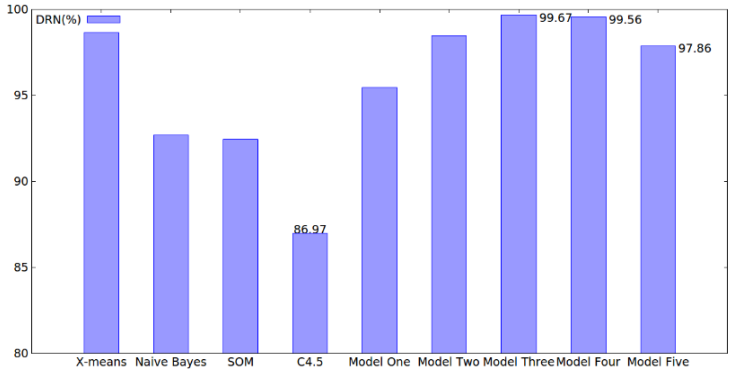
$$\text{DRN} = \frac{\text{numbers of normal url requests detected correctly}}{\text{all normal url requests}} \times 100\% \quad (5)$$

We evaluated our system on two datasets mentioned above. Among these experiments, we used ASCII and auto-encoder to represent URLs respectively. As for discrimination model, we

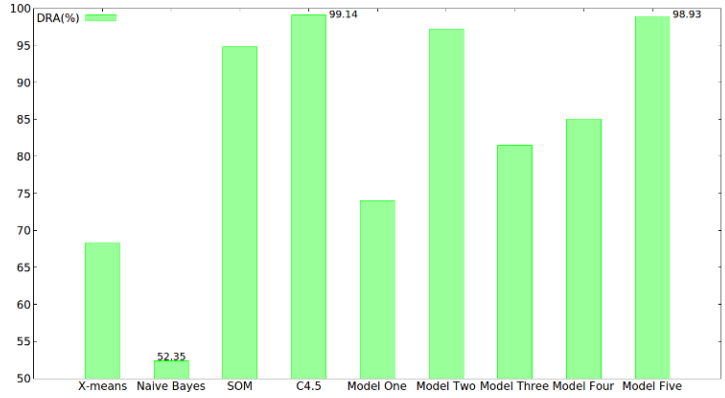
applied proposed structure of neural network (Comp-block) and convolutional neural network.

The results of SOM, C4.5, Naïve Bayes, X-means and EM were evaluated in Le [Le (2017)]. The DRA score of C4.5 evaluated in Le [Le (2017)] that is 99.14% was the best in Fig. 6(b) while the DRN score was unsatisfactory which means too much false alarms would appear if C4.5 algorithm was used. The results of model with GRU in Liang et al. [Liang, Zhao and Ye (2017)] were competitive, because these three scores were all at a high level. All our models have high scores in DRN and ACC, but the DRA scores of our models with ASCII are imperfect. It seems that the ability of detecting anomalies of our models with ASCII code for feature representation is deficient. On the contrary, our model with auto-encoder and Comp-block seems to get the best results that got competitive scores in three metrics (98.20% in ACC, 98.93% in DRA, 97.86% in DRN). Consequently, these results indicate that our system have high ability of detecting anomalous URLs with low false alarms.

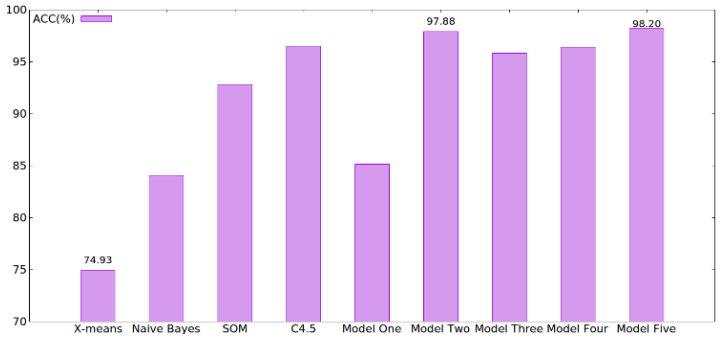
The Fig. 7 shows comparison on the new dataset generated in our lab. Two models have almost the same scores in DRN while the performances of model with auto-encoder and Comp-block defeated the one with auto-encoder and CNN in other metrics. Specifically, the model with auto-encoder and Comp-block win by 1.19% in Accuracy and 5.03% in DRA respectively.



(a) DRN results of comparison



(b) DRA results of comparison



(c) ACC results of comparison

Figure 6: Comparison on HTTP CSIC2010 (Model One refers to model with RNN in Liang et al. [Liang, Zhao and Ye (2017)], Model Two refers to model with GRU in Liang et al. [Liang, Zhao and Ye (2017)], Model Three means our model with ASCII code for feature representation and CNN for discrimination, Model Four means model with ASCII code for representation and Comp-block for discrimination, Model Five stands for our model with auto-encoder for representation and Comp-block for discrimination)

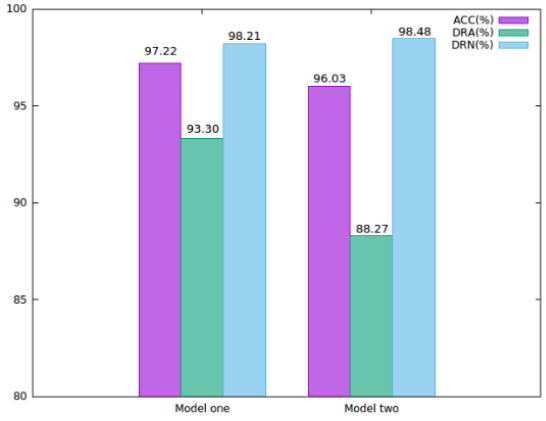


Figure 7: Comparison on new dataset (model one means model with auto-encoder for feature represnetation and comp-block for discrimination, model two stands for model with auto-encoder for feature represnetation and CNN for discrimination)

4.2.2 Discussions

We conduct experiments on model with two methods of feature representation that includes ASCII code and auto-encoder for URLs. Experimental results in Fig. 6 demonstrate that ASCII code is able to represent URLs with retaining some useful information to some extent while auto-encoder seems to retain much more useful information of URLs. Although models with ASCII code for representation got highest scores in DRN, poor performance in DRA shows that they are not able to do this work well. On the contrary, we can see noticeable increase when auto-encoder is used. We are of the opinion that semantic information is lost when URLs are represented by ASCII code while auto-encoder retains it.

To achieve better performance, we designed Comp-block that is a special structure of convolution neural network. Comparisons in experiments with two datasets showed its better capacity of extracting features and discrimination in anomalous URLs detection. Overall, our system with auto-encoder and Comp-block has prominent capability of representing URLs and detecting anomalous URLs.

It should be added that there are limitations in our system. Importantly, our model is very dependent on training data, errors in training data have obvious impact on the system. Another limitation is that the representation part and discrimination part both need to be updated when required.

5 Conclusion and future work

In this work we propose a novel system for malicious URLs detection to protect against web attacks. Particularly, we use auto-encoder to represent URLs firstly, and then transform the detection problem into a classification problem with utilizing deep learning methods. The proposed system uses NLP techniques for analysis and auto-encoder for representing the URLs with vectors automatically. Moreover, we designed a new structure of convolutional neural network that has better capability for discrimination in this work. To evaluate this system, two datasets are used in our experiments. Experimental results show that the proposed system has prominent capacity of extracting features automatically and detecting anomalous URLs.

There are several directions for our future work. One is to optimize the structure of this system so that it will be convenient to update it. How to reduce the dependence on training data is another problem to be resolved. And it will enhance the stability of the system to a large extent. Another is to investigate the attack against deep learning models which is a potential threat to our system. At the same time, we will think about applying these techniques to other security researches.

Acknowledgement: This work is supported in part by the National Natural Science Foundation of China (61871140, 61872100, 61572153, U1636215, 61572492, 61672020), the National Key research and Development Plan (Grant No. 2018YFB0803504), and Open Fund of Beijing Key Laboratory of IOT Information Security Technology (J6V0011104).

References

Carmen, T. G.; Alejandro, P. V.; Gonzalo, Á. M. (2018): HTTP DATASET CSIC 2010. http://www.isi.csic.es/dataset/.

Chen, J.; Tian, Z.; Cui, X.; Yin, L.; Wang, X. (2018): Trust architecture and reputation evaluation for internet of things. *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-9.

Curphey, M.; Williams, J.; Konda, M. (2018): OWASP top 10 most critical web application security risks.

Du, X.; Chen, H. H. (2008): Security in wireless sensor networks. *IEEE Wireless Communications*, vol. 15, no. 4, pp. 60-66.

Du, X.; Guizani, M.; Xiao, Y.; Chen, H. H. (2009): Transactions papers a routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks. *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1223-1229.

Du, X.; Xiao, Y.; Guizani, M.; Chen, H. H. (2007): An effective key management scheme for heterogeneous sensor networks. *Ad Hoc Networks*, vol. 5, no. 1, pp. 24-34.

Duc Jr, L. (2017): An unsupervised learning approach for network and system analysis. http://hdl.handle.net/10222/72785.

Han, W.; Tian, Z.; Huang, Z.; Li, S.; Jia, Y. (2018): Bidirectional self-adaptive resampling in internet of things big data learning. *Multimedia Tools and Applications*, pp. 1-16.

Hendler, D.; Kels, S.; Rubin, A. (2018): Detecting malicious PowerShell commands using deep neural networks. *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pp. 187-197.

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project.

Ito, M.; Iyatomi, H. (2018): Web application firewall using character-level convolutional neural network. *IEEE 14th International Colloquium on Signal Processing & Its Applications*, pp. 103-106.

Jozefowicz, R.; Vinyals, O.; Schuster, M.; Shazeer, N.; Wu, Y. (2016): Exploring the limits of language modeling. ArXiv e-prints, vol.abs/1602.02410.

Kim, Y. (2014): Convolutional neural networks for sentence classification. ArXiv e-prints, vol.abs/1408.5882.

LeCun, Y.; Bengio, Y.; Hinton, G. (2015): Deep learning. *Nature*, vol. 521, no. 7553, pp. 436.

LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E. et al. (1989): Backpropagation applied to handwritten zip code recognition. *Neural Computation*, vol. 1, no. 4, pp. 541-551.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. (1998): Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324.

Li, M.; Sun, Y.; Jiang, Y.; Tian, Z. (2018): Answering the min-cost quality-aware query on multi-sources in sensor-cloud systems. *Sensors*, vol. 18, no. 12, pp. 4486.

Liang, J.; Zhao, W.; Ye, W. (2017): Anomaly-based web attack detection: a deep learning approach. *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, pp. 80-85.

Luo, C.; Wang, L.; Lu, H. (2018): Analysis of LSTM-RNN based on attack type of KDD-99 dataset. *International Conference on Cloud Computing and Security*, pp. 326-333.

Lupták, P. (2011): Bypassing web application firewalls. *Proceedings of 6th International Scientific Conference on Security and Protection of Information*, pp. 79-88.

Nair, V.; Hinton, G. E. (2010): Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th International Conference on Machine Learning*, pp. 807-814.

Qiu, J.; Chai, Y.; Liu, Y.; Gu, Z.; Li, S. et al. (2018): Automatic non-taxonomic relation extraction from big data in smart city. *IEEE Access*, vol. 6, pp. 74854-74864.

Rieck, K.; Laskov, P. (2006): Detecting unknown network attacks using language models. *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 74-90.

Shi, C. (2018): A novel ensemble learning algorithm based on DS evidence theory for IoT security. *Computers, Materials & Continua*, vol. 57, no. 3, pp. 635-652.

Tan, Q.; Gao, Y.; Shi, J.; Wang, X.; Fang, B. et al. (2018): Towards a comprehensive insight into the eclipse attacks of tor hidden services. *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1584-1593.

Tian, Z.; Cui, Y.; An, L.; Su, S.; Yin, X. et al. (2018): A real-time correlation of host-level events in cyber range service for smart campus. *IEEE Access*, vol. 6, pp. 35355-35364.

Tian, Z.; Shi, W.; Wang, Y.; Zhu, C.; Du, X. et al. (2019): Real time lateral movement detection based on evidence reasoning network for edge computing environment. *IEEE Transactions on Industrial Informatics*, pp. 1.

Tian, Z.; Su, S.; Shi, W.; Du, X.; Guizani, M. (2019): A data-driven method for future Internet route decision modeling. *Future Generation Computer Systems*, vol. 95, pp. 212-220.

Wang, Y.; Tian, Z.; Zhang, H.; Su, S.; Shi, W. (2018): A privacy preserving scheme for nearest neighbor query. *Sensors*, vol. 18, no. 8, pp. 2440.

Xiao, Y.; Du, X.; Zhang, J.; Hu, F.; Guizani, S. (2007): Internet protocol television (IPTV): the killer application for the next-generation internet. *IEEE Communications Magazine*, vol. 45, no. 11, pp. 126-134.

Xiao, Y.; Rayi, V. K.; Sun, B.; Du, X.; Hu, F. et al. (2007): A survey of key management schemes in wireless sensor networks. *Computer Communications*, vol. 30, no. 11-12, pp. 2314-2341.

Yu, X.; Tian, Z.; Qiu, J.; Jiang, F. (2018): A data leakage prevention method based on the reduction of confidential and context terms for smart mobile devices. *Wireless Communications and Mobile Computing*, vol. 2018.

Zhang, X.; LeCun, Y. (2015): Text understanding from scratch. arxiv: vol.abs/1502.01710.

Zhang, Y. L.; Li, L.; Zhou, J.; Li, X.; Liu, Y. et al. (2017): Poster: a PU learning based system for potential malicious URL detection. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2599-2601

Zhang, Z.; George, R.; Shujaee, K. (2016): Efficient detection of anomolous HTTP payloads in networks. *SoutheastCon*, pp. 1-3.