IJICTR
International Journal of Information & Communication Technology Research

Volume 11- Number 3 – Summer 2019 (49 -56)

Auto-Encoder LSTM Methods for Anomaly-Based Web Application Firewall

[1]Ali Moradi Vartouni, [2]Soheil Mehralian, [3]Mohammad Teshnehlab[*], [4]Saeed Sedighian Kashi
Faculty of Electrical and Computer Engineering
K.N. Toosi University of Technology
Tehran, Iran
e-mails: [1,2]{alimoradivartouni, mehralian}@ee.kntu.ac.ir, [3,4]{teshnehlab, sedighian}@eetd.kntu.ac.ir

Received: 3 March 2019 - Accepted: 20 June 2019

Abstract—**Web Application Firewall (WAF) is known as one of the Intrusion Detection System (IDS) solutions for protecting web servers from HTTP attacks. WAF is a tool to identify and prevent many types of attacks, such as XSS and SQL-injection. In this paper, deep machine learning algorithms are used for enriching the WAF based on the anomaly detection method. Firstly, we construct attributes from HTTP data, to do so we consider two models namely *n*-gram and one-hot. Then, according to Auto-Encoder LSTM (AE-LSTM) as an unsupervised deep leaning method, we should extract informative features and then reduce them. Finally, we use ensemble isolation forest to train only normal data for the classifier. We apply the proposed model on CSIC 2010 and ECML/ PKDD 2007 datasets. The results show AE-LSTM has higher performance in terms of accuracy and generalization compared with naïve methods on CSIC dataset; the proposed method also have acceptable detection rate on ECML/PKDD dataset using *n*-gram model.**

Keywords-Web Application Firewall; Anomaly Detection; LSTM; AE-LSTM; Ensemble Isolation Forest

I. INTRODUCTION

Nowadays, attacks to web applications and targeting their vulnerabilities are the source of a significant part of data breaches, to avoid this problem, security experts proposed Web Application Firewalls (WAFs). A WAF checks every request to the web application from clients to find and block abnormal activity, compromising the security of the web server.

In general, the WAFs methods are divided into two groups based on signature and anomaly detection. The signature-based techniques use a set of pre-identified rules and patterns created and tuned by experts to block specific HTTP requests and prevent attacks. On the other hand, in anomaly-based methods, machine-learning (ML) algorithms can be easily automated to continuously learn from the most recent data without the mediation of human. Also, the strength of predictive models in detecting similar patterns secures it against attack obfuscation techniques. Finally, their generalization and ability to learn new unknown pattern can help us to discover new types of attacks.

To evaluate attack detection assured with ML, the reducing false positives and false negatives are important. Thus, for the sake of performance evaluation of the WAF, not only accuracy measure is essential, but also metrics that are related to the generalization of learning are crucial.

Two essential phases in ML are feature extraction and model training. The performance and efficiency of machine learning algorithms highly depend on data

[*] Corresponding Author

representation [1], which is the result of the first phase, feature extraction. In other words, a good data representation can lead to high performance for a relatively more straightforward machine learner [2]. This issue is more important in WAFs because the HTTP data have non-stationary behavior. Thus, feature creation and extraction are critical points of WAFs.

The rest of the paper is organized as follows: Section II summarizes the literature review about WAF based on anomaly detection especially deep learning methods. In Section III, firstly, we review LSTM structure and then, we present the various architectures of AE-LSTM. In Section IV, we introduce our model with AE-LSTM for WAF. In Section V, we discuss the experiments in detail. Finally, we draw our conclusions in Section VI.

II. LITERATURE REVIEW

According to recent studies, statistical characteristics of HTTP traffic [3, 4], tokenization of HTTP requests [5, 6], and packet payload based on natural language processing (NLP) can be considered as HTTP features for web application attacks detection. The packet payload analysis using discrete NLP techniques includes *n*-gram and one-hot methods. A character-based *n*-gram method is a subsequence of *n* overlapping character from given data. On the other hand, a character-based one-hot vector is a $1 \times N$ matrix (vector) used to distinguish each character in vocabulary from every other character in the vocabulary. The vocabulary includes N accessible characters. The vector consists of zeros in all cells, except for a single one in a cell used uniquely to identify the character. We can replace character by other items like word, phrase, etc. in both packet payload approaches.

Many of works for WAF based on anomaly-detection used *n*-gram for attribute construction. PayL [7], Anagram [8], McPAD [9], Spectrogram [10] used *n*-gram method to characterize the distribution of the content of the requests. With an increase of *n*, the *n*-gram becomes a very ill-posed and intractable problem, to deal with this problem, two filter-based feature selection algorithms applied in [11-13] to reduce *n*-gram attributes. Also, the deep learning approaches include stack auto-encoder (SAE) [14-16], and deep belief network (DBN) used to extract more compact features from *n*-gram attributes of HTTP payload [15].

Researchers have applied many MLs to detect WAF attacks. The Markov's methods have good results for minimizing false positives [17-19], the ontological techniques are approaches that provide semantic learning [20, 21] and, the affinity propagation is employed dynamic clustering to extract the exemplars from HTTP request [22, 23]. Besides, probability distributed model, hidden Markov model (HMM), and one-class SVM (OSVM) model are introduced in [24] for WAF. However, with the emersion of deep learning, it is possible to do feature engineering using hierarchical deep neural networks in an automated manner.

Recurrent neural network (RNN), SAE, DBN, and convolutional neural network (CNN) are various categories of deep neural networks that are used for web application firewalls. In [25] character-level CNN used for web attacks detection. The SAEs and DBNs can extract features that are more applicable and reduce the dimensionality of features vectors. Also, researchers have been applied SAE to implement WAF in [14, 26, 27] and denoising SAE in [16, 28] for web attacks detection. In addition to SAE, we used DBN model for feature extraction in [15] and then used three one-class classifiers namely OSVM, ensemble isolation forest and elliptic envelope for detection of attacks, then compared different methods with each other.

Since text data sets such as HTTP data are sequential and time-dependent, using RNN methods is the proper choice as ML algorithms. The deep-based RNN includes Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) used for WAF in [29]. This model takes uniform resource locators (URLs) in the HTTP GET requests as the input. After the URLs are tokenized by following a particular strategy, two RNNs that have learned the normal request patterns and outputs their familiarities with given URLs. Then a trained neural network decides whether given requests are anomalous or not based on the output of the RNNs. Also, ZeroWall [30] trains a self-translation machine using an encoder-decoder RNN to capture the syntax and semantic patterns of normal requests. In addition, the MLs are used to web traffic based anomaly detection. Using of C-LSTM consists of CNN and LSTM layers are presented to model spatiotemporal data contained in traffic data effectively as one-dimensional time series signal [31]. Also, CECoR-Net [32] is a character-level neural network which detected web attacks. It combines the CNN and LSTM techniques.

In this paper, **to have a good representation of features for HTTP requests, which are sequential data sets, AE-LSTM is proposed**. The AE-LSTM is a type of AEs in which instead of fully connected layer, an LSTM layer considered as encoder part of the AE. **Moreover, we will apply our model on two discrete NLP techniques: n-gram and one-hot attribute constructors**.

III. PRELIMINARIES

A. *Long Short-Term Memory networks*

Long Short-Term Memory (LSTM) network is a kind of neural network that can learn long term dependencies, i.e., it is a special variation of Recurrent Neural Network (RNN). Dependencies between various data components such as video frames, words, time steps of time series are essential information which machine learnings must extract them in the learning phase. Recurrent neural network and Recursive Neural Network (ReNN) can model long-range dependencies between data components by a chain repeating modules of neural network in their structure. RNNs are used for sequential inputs and unfold over time. While ReNNs are the generalization of RNN, where the input has to be processed hierarchically in a tree fashion [33].

Standard RNNs require to learn long-term temporal dependencies to maintain information in memory over time, but RNNs suffer from vanishing and exploding gradient problems, whereas to determine when data enters into the memory, when it gets out and finally when forgetting it, a set of gates is used. This architecture leads to learning longer-term dependencies.

For a given input sequence $X = \{x^{(1)}, x^{(2)}, \dots, x^{(N)}\}$ and output sequence $Y = \{y^{(1)}, y^{(2)}, \dots, y^{(N)}\}$, at each time step, the output of the module is controlled by a set of gates as a function of a previous hidden state h_{t-1} and the current input x_t, the forget gate f_t, the input gate i_t and the output gate o_t.

The LSTM transition function of gates are defined as follows:

$$\boldsymbol{i_t = \sigma(W_i\,[h_{t-1}, x_t] + b_i)} \quad (1)$$

$$\boldsymbol{f_t = \sigma(W_f\,[h_{t-1}, x_t] + b_f)} \quad (2)$$

$$\boldsymbol{l_t = tanh(W_l\,[h_{t-1}, x_t] + b_l)} \quad (3)$$

$$\boldsymbol{o_t = \sigma(W_o\,[h_{t-1}, x_t] + b_o)} \quad (4)$$

$$\boldsymbol{c_t = f_t \odot c_{t-1} + i_t \odot l_t} \quad (5)$$

$$\boldsymbol{h_t = o_t \odot tanh(c_t)} \quad (6)$$

Here, σ is the sigmoid function that has an output in [0, 1], $tanh$ indicates the hyperbolic tangent function that has an output in [-1, 1], and $\odot$ denotes the component-wise multiplication.

B. Auto-encoder LSTM

According to LSTM equations (1-6), LSTM is an excellent machine to learn sequential patterns in supervised learning tasks such as speech recognition and machine translation in which we have to learn long-term dependencies. We can add LSTM's ability to learn sequential dependencies to an unsupervised representation learner, encoder-decoder framework, to have both in one algorithm. The Encoder LSTM runs through a sequence of samples to come up with a compact representation. Then, this representation is decoded through another LSTM to produce the target sequence.

There are several scenarios to build encoder-decoder LSTM. Target sequence and sequence encoder output cause to vary LSTM auto-encoders. An encoder-decoder framework used for machine translation in [34] and [35] in which an RNN is responsible for encoding sequence of symbols into fixed-length representation and another network decodes the learned representation into symbols. The work in [36] is similar to auto-encoders predicted the same target sequence as the input. In these works, the encoder is a **many-to-one** LSTM and the decoder is a **one-to-many** LSTM.

The motivation of our model is similar to that of auto-encoders; we consider target similar to the input and **many-to-many** structure for encoder-decoder LSTMs. The proposed architecture is shown in Fig. 1.

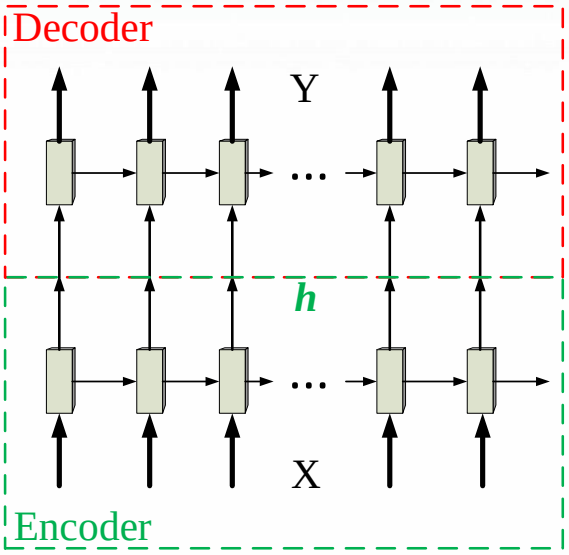

Figure 1. Auto-Encoder-LSTM structur.

Assume training examples in the form of $X = \{x^{(1)}, x^{(2)}, \dots, x^{(N)}\}$. Auto-encoder is a type of neural network in which target values are set to be the input values $Y = X$, i.e., $y^{(i)} = x^{(i)}\ for\ i = 1 \dots N$ and backpropagation used in an unsupervised manner. An auto-encoder always consists of two parts: the encoder f and the reconstruction part g as the decoder. The encoder can be defined as $h = f(x)$ and decoder is defined as $r = g(h)$. In summary, the learning process is to find a value for parameter vector θ to minimize reconstruction error as in (7):

$$\boldsymbol{J_{AE}(\theta) = \sum_i L\left(x^{(i)}, r_\theta^{(i)}\right)} \quad (7)$$

Where L is a loss function that is used to compute the error in neural networks, and one of the most commonly used loss function is the quadratic cost function, which is the square of the difference between the desired value and the prediction.

IV. PROPOSED MODELS

For designing a WAF to detect attacks against a web server that accepts HTTP requests, it is necessary to construct features from HTTP logs.

In order to extract representative features from the HTTP request, two models based on character, namely *n*-gram and binary one-hot, are applied. For statistical *n*-gram method, the size of each feature vector is equal to the number of all possible *n*-grams in the data set. The j^{th} feature of the i^{th} feature vector x_{ij} is equal to the frequency of occurrences of the j^{th} *n*-gram in the i^{th} request. In this work, we apply *n*=2 for the *n*-gram method.

After feature construction, two scenarios ReNN and RNN based on features are applied to detect malicious requests. The *n*-gram method is used as a pre-process for AE-LSTM methodology just for feature reduction like a RNN; **Error! Reference source not found.** shows this scenario. On the other hand, the binary one-

hot format is used to prepare data for AE-LSTM to feature extraction like an ReNN. Since the number of LSTM cells in this method is too large for further reduction of features, we can use AEs like SAE for feature reduction according to **Error! Reference source not found.**. After that, a one-class classifier is used as anomaly detector. For this purpose, we use the Ensemble Isolation Forest algorithm like the models in [14, 15] in which training and prediction of the algorithm are fast and accurate.

The isolation forest makes a model to 'isolate' anomalies from normal data explicitly. The idea behind the method is that since the anomalies are rare and different, they are more susceptible to isolation from normal data, and this makes it suitable for anomaly detection applications such as WAF. This method partitions data using a random tree until all instances are isolated from each other [37].

Our method examines individual HTTP requests. There is 256 ASCII code in total, but only 96 of them appear in HTTP dataset [12]. Since the 2-gram model is computed as two characters frequently, we have 96x96 items; consequently, a vector of 9216 dimensions represents each HTTP request in 2-gram model. On the other hand, for the binary one-hot model, we have a matrix of zero and one for each request, which the number of columns equal to the number of all character of that request and the size of each column is a 96-item vector.

V. EVALUATION

We conducted experiments on the CSIC 2010 dataset [38] and ECML/ PKDD 2007 dataset [39]. These datasets contain only HTTP traffic that HTTP requests in addition normal traffics include several attacks such as SQL injection, cross-site scripting (XSS), buffer overflow, XPATH injection, LDAP injection and so on. CSIC data is a publicly available labeled dataset. The training set contains 36000 normal data, and the test set contains 36000 normal and more than 25000 abnormal data. Further, the training set of ECML contains 20000 normal data and the test set contains 15000 for each normal and abnormal data. Additionally, we examined experiments on a system with the following characteristics: X86-64 GNU/ Linux, Intel ® Core ™ i7-7700k CPU @4.20 GHz, GeForce GTX 1080 Ti GPU, and 47 GB RAM.

Algorithms implemented in Python 3.6 programming language with Tensorflow and Keras libraries. These libraries are useful tools to design, build, and train deep learning models. To evaluate our deep model, some favorable metrics such as accuracy and generalization are used. A fit generalization is a good trade-off between false positive and false negative alarms [6]. We also used several measures to evaluate the performance of methods:

FP = False Positive: the total number of attacks that did not detect,

FN = False Negative: the total number of normal requests that are classified as anomalous,

TP = True Positive: the total number of attacks that are truly detected,

TN = True Negative: the total number of normal requests that are classified as normal.

And, these measures are:

$$\textbf{Accuracy (Acc.)}= \frac{TP+TN}{TP+TN+FP+FN} \times 100 \quad (8)$$

$$\textbf{Detection Rate (DR)}= \frac{TP}{TP+FN} \times 100 \quad (9)$$

$$\textbf{Precision (Pr.)}= \frac{TP}{TP+FP} \times 100 \quad (10)$$

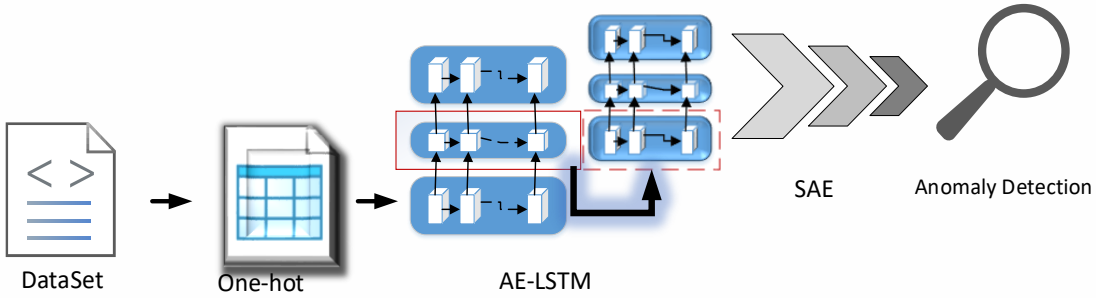

Figure 2. ReNN model for extraction one-hot features with AE-LSTM and reduction features with stack auto-encoder and then detection attacks with ensemble isolation forest Proposed input selection strategy structure.

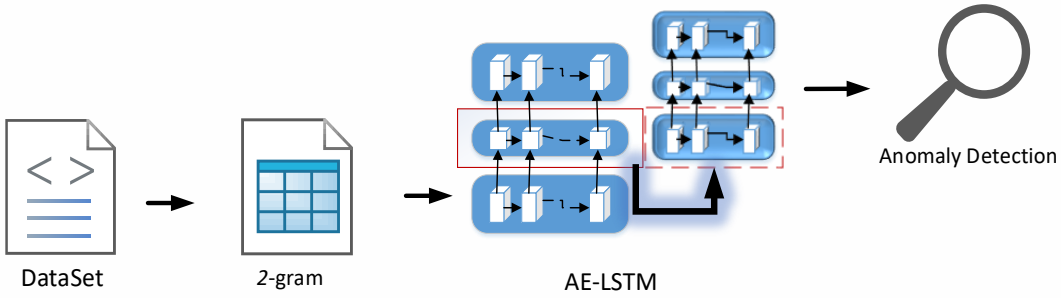

Figure 3. RNN model for reduction features of 2-gram with AE-LSTM method and then detection attacks with ensemble isolation forest Proposed input selection strategy structure.

Specificity (Spec.)= $\frac{TN}{TN+FP} \times 100$ (11)

F-score (F1)= $\frac{2 \times Pr. \times DR}{Pr. + DR} \times 100$ (12)

We also used the ROC curve that provides insights closer to FPR and TPR for different thresholds of scores.

As we mentioned, we use AE-LSTM structure for extracting features. Because when are used n-gram with a higher order of *n*, reduction of dimensions is inevitable. On the other hand, in addition to be large-scale one-hot features, it is two-dimensional raw data for each request. Moreover, since for creation one-hot attribute, it is necessary that all characters of request convert to 96-dimension vector. Thus, we consider maximum character 1500 for CSIC data and 2300 for ECML/ PKDD data to cover all characters of HTTP requests.

Furthermore, we apply four layers of AE for AE-LSTM that reduces the 96-dimensional feature vector to 50, 15, 4 and one blocks of LSTM in each layer, respectively. Thus, the number of features for the 2-gram model with AE-LSTM is 96, for one-hot with only AE-LSTM without SAE is 1500 or 2300 for CSIC or ECML/ PKDD, respectively. Because we want to compare models with others, we use SAE after AE-LSTM to reduce the dimension of the features down to 100. Also, the batch size for every SAE is 36 and the number of epochs is 30. In addition, the learning rate for SAEs is 10−4. As a result, we compare our proposed models with Naïve methods that use *n*-gram method without feature extraction. We can see presented results of different models and structures for CSIC and ECML/ PKDD data in **Error! Reference source not found.** and **Error! Reference source not found.**, respectively.

As we see in **Error! Reference source not found.**, the accuracy of AE-LSTM with SAE and one-hot attributes (the fifth row in **Error! Reference source not found.**) on CSIC data is the highest. Therefore, we can claim this method is better than other methods in terms of accuracy and generalization. We can see ROC of this model with the Area Under Curve (AUC) equal to 0.95, according to ROC curve for different models for CSIC data.. In addition, other AE-LSTM models have good results in the detection of attacks.

On the other hand, the ECML/ PKDD dataset has more non-linear relations between its features [11]; therefore, according to ROC curve for different models for ECML/PKDD data.

and **Error! Reference source not found.**, one-hot method could not get acceptable results on it. However, **Error! Reference source not found.** depicts the AE-LSTM with 2-gram model, which can extract meaningful features in order to detect anomalies based on isolation forest algorithm.

TABLE I. THE DIFFERENT OUR PROPOSED MODEL AND NAÏVE MODELS 1-GRAM AND 2-GRAM BASED ON ISOLATION FOREST ALGORITHM FOR CSIC 2010 DATA.

Model	Structure	# Features	Accuracy	DR	Pr.	Spec.	F1
1-gram	Naïve	96	75.30	48.28	71.96	89.86	57.79
2-gram	Naïve	9216	72	38.59	67.53	90	49.12
2-gram	AE-LSTM: [50-15-4-1]	96	80.33	62.46	77.02	89.96	68.98
One-hot	AE-LSTM: [50-15-4-1]	1500	77.91	55.55	74.85	89.95	63.77
One-hot	AE-LSTM: [50-15-4-1] SAE: [600-250-100]	100	87.26	82.73	81.22	89.70	81.96

TABLE II. THE DIFFERENT OUR PROPOSED MODEL AND NAÏVE MODELS 1-GRAM AND 2-GRAM BASED ON ISOLATION FOREST ALGORITHM FOR ECML/ PKDD 2007 DATA.

Model	Structure	# Features	Accuracy	DR	Pr.	Spec.	F1
1-gram	Naïve	96	73.74	55.73	87.36	91.88	68.05
2-gram	Naïve	9216	59.35	27.81	75.93	91.12	40.71
2-gram	AE-LSTM: [50-15-4-1]	96	74.62	58.68	86.36	90.67	69.88
One-hot	AE-LSTM: [50-15-4-1]	2300	55.57	19.41	70.93	91.99	30.48
One-hot	AE-LSTM: [50-15-4-1] SAE: [900-300-100]	100	56.53	21.48	72.57	91.82	33.15

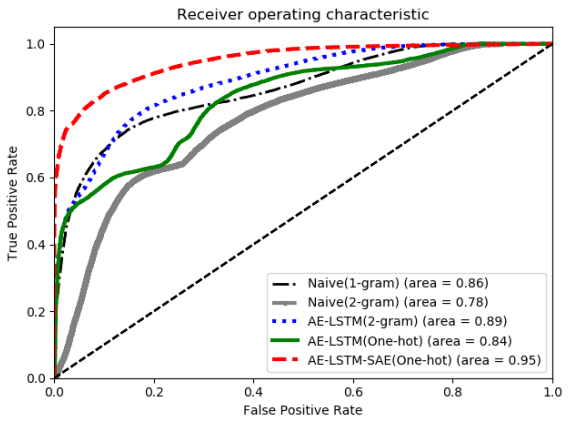

Figure 4. ROC curve for different models for CSIC data.

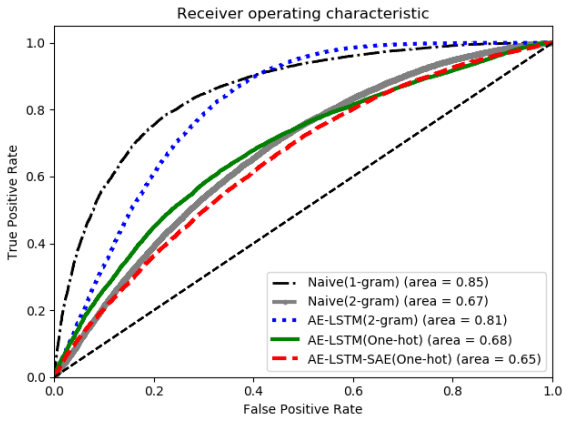

Figure 5. ROC curve for different models for ECML/PKDD data.

VI. CONCLUSION

In this study, we have leveraged DNN methods based on the auto-encoder LSTM model to extract relevant features from HTTP request logs for anomaly detection in web application firewalls. Moreover, we have used *n*-gram based method on character and binary one-hot on character for attributes construction of HTTP data and ensemble isolation forest as anomaly detection algorithm to detect attacks.

Although the one-hot method can model sequences of our text benchmark but the results depict the binary one-hot attributes is not the proper one for extracting meaningful features based on AE-LSTM model for that features which have non-linear relations between them. On the other hand, we know that HTTP data is a non-stationary structure. Therefore, HTTP requests have different length and sections of data. As a result, we cannot have invariant blocks of LSTM in order to extract features based on AE-LSTM model for one-hot method. Of course, we can consider the fusion of two feature sets in our model. For example, use the URL containing protocol, hostname, extension, and parameters of the query as the input of one-hot method and the rest of request containing the header and the body of request which are larger and more invariant as input of *n*-gram method.

REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798-1828, 2013.

[2] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, p. 1, 2015.

[3] C. Kruegel and G. Vigna, "Anomaly detection of web-based attacks," in *Proceedings of the 10th ACM conference on Computer and communications security*, 2003: ACM, 2003, pp. 251-261.

[4] C. Kruegel, G. Vigna, and W. Robertson, "A multi-model approach to the detection of web-based attacks," *Computer Networks*, vol. 48, no. 5, pp. 717-738, 2005.

[5] K. L. Ingham and H. Inoue, "Comparing anomaly detection techniques for http," in *RAID*, 2007, vol. 4637: Springer, pp. 42-62.

[6] K. L. Ingham, A. Somayaji, J. Burge, and S. Forrest, "Learning DFA representations of HTTP for protecting web applications," *Computer Networks*, vol. 51, no. 5, pp. 1239-1255, 2007.

[7] K. Wang and S. J. Stolfo, "Anomalous payload-based network intrusion detection," in *RAID*, 2004, vol. 4: Springer, 2004, pp. 203-222.

[8] K. Wang, J. Parekh, and S. Stolfo, "Anagram: A content anomaly detector resistant to mimicry attack," in *Recent Advances in Intrusion Detection*, 2006: Springer, 2006, pp. 226-248.

[9] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, "McPAD: A multiple classifier system for accurate payload-based anomaly detection," *Computer networks*, vol. 53, no. 6, pp. 864-881, 2009.

[10] Y. Song, A. D. Keromytis, and S. J. Stolfo, "Spectrogram: A Mixture-of-Markov-Chains Model for Anomaly Detection in Web Traffic," in *NDSS*, 2009, vol. 9, 2009, pp. 1-15.

[11] H. T. Nguyen, C. Torrano-Gimenez, G. Alvarez, K. Franke, and S. Petrović, "Enhancing the effectiveness of Web Application Firewalls by generic feature selection," *Logic Journal of IGPL*, vol. 21, no. 4, pp. 560-570, 2012.

[12] C. Torrano - Gimenez, H. T. Nguyen, G. Alvarez, and K. Franke, "Combining expert knowledge with automatic feature extraction for reliable web attack detection," *Security and Communication Networks*, vol. 8, no. 16, pp. 2750-2767, 2015.

[13] C. Torrano-Gimenez, H. T. Nguyen, G. Alvarez, S. Petrovic, and K. Franke, "Applying feature selection to payload-based web application firewalls," in *Security and Communication Networks (IWSCN), 2011 Third International Workshop on*, 2011: IEEE, 2011, pp. 75-81.

[14] A. M. Vartouni, S. S. Kashi, and M. Teshnehlab, "An anomaly detection method to detect web attacks using Stacked Auto-Encoder," in *Fuzzy and Intelligent Systems (CFIS), 2018 6th Iranian Joint Congress on*, 2018: IEEE, 2018, pp. 131-134.

[15] A. M. Vartouni, M. Teshnehlab, and S. S. Kashi, "Leveraging deep neural networks for anomaly-based web application firewall," *IET Information Security*, 2019.

[16] Y. Pan *et al.*, "Detecting web attacks with end-to-end deep learning," *Journal of Internet Services and Applications*, vol. 10, no. 1, pp. 1-22, 2019.

[17] J. M. Estevez-Tapiador, P. García-Teodoro, and J. E. Díaz-Verdejo, "Detection of web-based attacks through Markovian protocol parsing," in *Computers and Communications, 2005. ISCC 2005. Proceedings. 10th IEEE Symposium on*, 2005: IEEE, 2005, pp. 457-462.

[18] H. Lampesberger, P. Winter, M. Zeilinger, and E. Hermann, "An On-Line Learning Statistical Model to Detect Malicious Web Requests," in *SecureComm*, 2011: Springer, 2011, pp. 19-38.

[19] V. Relan and B. Sonawane, "Detection and mitigation of Web Services Attacks using Markov Model," *CMSC 678: Machine Learning Project*, 2009.

[20] A. Razzaq, Z. Anwar, H. F. Ahmad, K. Latif, and F. Munir, "Ontology for attack detection: An intelligent approach to web application security," *computers & security*, vol. 45, pp. 124-146, 2014.

[21] A. Razzaq, K. Latif, H. F. Ahmad, A. Hur, Z. Anwar, and P. C. Bloodsworth, "Semantic security against web

application attacks," *Information Sciences*, vol. 254, pp. 19-38, 2014.
[22] W. Wang, T. Guyet, R. Quiniou, M.-O. Cordier, F. Masseglia, and X. Zhang, "Autonomic intrusion detection: Adaptively detecting anomalies over unlabeled audit data streams in computer networks," *Knowledge-Based Systems*, vol. 70, pp. 103-117, 2014.
[23] W. Wang and X. Zhang, "High-speed web attack detection through extracting exemplars from HTTP traffic," in *Proceedings of the 2011 ACM Symposium on Applied Computing*, 2011: ACM, 2011, pp. 1538-1543.
[24] M. Zhang, S. Lu, and B. Xu, "An anomaly detection method based on multi-models to detect web attacks," in *Computational Intelligence and Design (ISCID), 2017 10th International Symposium on*, 2017, vol. 2: IEEE, pp. 404-409.
[25] M. Zhang, B. Xu, S. Bai, S. Lu, and Z. Lin, "A Deep Learning Method to Detect Web Attacks Using a Specially Designed CNN," in *International Conference on Neural Information Processing*, 2017: Springer, pp. 828-836.
[26] A. Gurina and V. Eliseev, "Anomaly-Based Method for Detecting Multiple Classes of Network Attacks," *Information*, vol. 10, no. 3, p. 84, 2019.
[27] H. Mac, D. Truong, L. Nguyen, H. Nguyen, H. A. Tran, and D. Tran, "Detecting Attacks on Web Applications using Autoencoder," in *Proceedings of the Ninth International Symposium on Information and Communication Technology*, 2018: ACM, pp. 416-421.
[28] D. Truong, D. Tran, L. Nguyen, H. Mac, H. A. Tran, and T. Bui, "Detecting Web Attacks using Stacked Denoising Autoencoder and Ensemble Learning Methods," in *Proceedings of the Tenth International Symposium on Information and Communication Technology*, 2019, pp. 267-272.
[29] J. Liang, W. Zhao, and W. Ye, "Anomaly-Based Web Attack Detection: A Deep Learning Approach," in *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, 2017: ACM, pp. 80-85.
[30] R. Tang *et al.*, "ZeroWall: Detecting Zero-Day Web Attacks through Encoder-Decoder Recurrent Neural Networks," 2020. [Online]. Available: aiops.org. aiops.org.
[31] T.-Y. Kim and S.-B. Cho, "Web traffic anomaly detection using C-LSTM neural networks," *Expert Systems with Applications*, vol. 106, pp. 66-76, 2018.
[32] X. Gong, J. Lu, Y. Wang, H. Qiu, R. He, and M. Qiu, "CECoR-Net: A Character-Level Neural Network Model for Web Attack Detection," in *2019 IEEE International Conference on Smart Cloud (SmartCloud)*, 2019: IEEE, pp. 98-103.
[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning. Book in preparation for MIT Press* (URL< http://www. deeplearningbook. org). 2016.
[34] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104-3112.
[35] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
[36] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International conference on machine learning*, 2015, pp. 843-852.
[37] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 2008: IEEE, 2008, pp. 413-422.
[38] C. Torrano-Gimnez, A. Prez-Villegas, and G. Alvarez, "The HTTP dataset CSIC 2010," ed: Instituto de Seguridad de la Información (ISI), 2010.
[39] C. Raıssi, J. Brissaud, G. Dray, P. Poncelet, M. Roche, and M. Teisseire, "Web analyzing traffic challenge: description and results," in *Proceedings of the ECML/PKDD*, 2007, 2007, pp. 47-52.

Ali Moradi Vartouni

Received his B.Sc. degree in Computer Engineering from Isfahan University, Isfahan, Iran, in Sep. 2009; the M.Sc. in Computer Science from Tabriz University, Tabriz, Iran, in Feb. 2012; and he is currently a Ph.D. candidate in the Department of Computer Engineering, at the Faculty of Computer Engineering, K.N. Toosi University of Technology, Tehran, Iran. His research interests include Artificial Intelligence, Deep Learning and Anomaly Detection.
e-mail: alimoradivartouni@ee.kntu.ac.ir

Soheil Mehralian

Received B.Sc. degree in Computer Engineering from Qazvin Islamic Azad University, Qazvin, Iran, in Sep. 2010; M.Sc. degree in Artificial Intelligence and Robotics from Isfahan University of Technology, Isfahan, Iran, in Jan. 2013; and he is currently a Ph.D. candidate in Department of Computer Engineering, Faculty of Computer Engineering, at K.N. Toosi University of Technology, Tehran, Iran. His research interests include Statistical Pattern Recognition, Deep Learning, Metric Learning, and Computer Vision.

Mohammad Teshnehlab

received the B.Sc. degree from Stony Brook University, USA, in 1980, the M.Sc. degree from Oita University, Japan, in 1990, and the Ph.D. degree from Saga University, Japan, in 1994. He is a faculty member of Electrical Eng. Department of K. N. Toosi University of Technology. Professor Teshnehlab is a member of the Industrial Control Center of Excellence and founder of Intelligent Systems Laboratory (ISLab). He is also a co-founder and member of Intelligent Systems Scientific Society of Iran (ISSSI) and a member of the editorial board of the Iranian Journal of Fuzzy Systems (IJFS), International Journal of Information & Communication Technology Research (IJICTR), and Scientific Journal of Computational Intelligence in Electrical Engineering. His research areas are Artificial Rough and Deep Neural Networks, Fuzzy Systems and Neural Nets, Optimization, and Expert Systems.

Saeed Sedighian Kashi received his Ph.D. in 2012. M.Sc. in 2005 and B.Sc. in Software Engineering, from the School of Computer Engineering, University of Science and Technology (IUST), Tehran, Iran, –in 2003. His research interests focus on Distributed Systems such as Cloud Computing, WSN and IoT.