The ISC Int'l Journal of Information Security

ISeCure

January 2020, Volume 12, Number 1 (pp. 44–54)

http://www.isecure-journal.org

Anomaly-based Web Attack Detection: The Application of Deep Neural Network Seq2Seq With Attention Mechanism

Shahriar Mohammadi [1,*], and Amin Namadchian [1]

[1] *Department of Information Technology, Faculty of Industrial Engineering, K. N. Toosi University of Technology, Iran*

ARTICLE INFO.

Article history:
Received: 23 August 2019
Revised: 23 September 2019
Accepted: 5 January 2020
Published Online: 21 January 2020

Keywords:
Deep Seq2Seq Network, Web Intrusion Detection System, Attention Mechanism, Embedding.

Abstract

Today, most activities and important data are placed on Internet websites, so attempts to intrude into these websites have grown exponentially. Intrusion detection systems (IDS) of web attacks are an approach to protect users. But, these systems are suffering from such drawbacks as low accuracy in detecting new attacks. To tackle this problem, various machine learning methods have been developed in recent years. Since attack requests differ from normal requests slightly, these anomaly detection methods have failed to exhibit good accuracy in new attack detection. In web requests and responses, a great deal of interconnected data is usually moved, and anomaly detection attempts need to consider all these connections, but this is a very complicated task. Thus, some research works on attack detection are confined to the analysis of just URL and a part of the request. This paper presents a new method for web attack detection using seq2seq networks using attention. The method is shown to successfully classify the traffic by predicting the possible responses and calculating differences from real responses of the webserver. The higher accuracy of this method versus similar methods shows that the use of the attention mechanism can cope with the challenge of analyzing long web requests and responses to a great extent. The proposed model exhibited a high result in terms of the specificity criterion when it came to such attacks as SQL Injection and XSS whose success highly depends on the server's response. This can be attributed to the inclusion of the link between the request and response in identifying web attacks in the proposed method.

1 Introduction

Today, the use of the Internet forms a key part of the modern lifestyle, and many important activities of society happen through websites. This has increased the importance of website security, particularly given the fact that efforts and investments to intrude into websites are exponentially growing. Nonetheless, the issue of security is not adequately considered during the development of web-based software. For instance, ready-to-use modules and services are mostly employed in website designing, whilst some have certain vulnerabilities. Symantec Corporation reports that web attacks account for 56 percent of all attacks [1]. Intrusion detection systems (IDSs) play a major role in inhibiting attacks to web servers by

* Corresponding author.

Email addresses: `mohammadi@kntu.ac.ir`, `amin.namadchyan@gmail.com`

ISSN: 2008-2045 © 2020 ISC. All rights reserved.

recognizing the activities of attackers and supporting proper responses to them.

An IDS monitor's software logs and/or web server requests to look for attack patterns in web requests. Due to the complexities and diversity of data in web requests, a robust machine learning method is often required to allow learning malicious or normal patterns in web requests. These algorithms usually convert data to a set of features. The selection of suitable features is important for the accuracy of learning algorithms. In recent years, the deep learning approach has performed well in different contexts, such as machine vision and mechanical translation. In this approach, the algorithms are built on the basis of neural networks, and it is attempted to use specific techniques and a hierarchy model to tackle the constraints of hidden layers [2]. Some works have attempted to exploit the learning feature of deep learning to select appropriate features for the detection of web attacks [3].

The deep approach aims to learn the hierarchical structure of features from the lower layers to the highest layer. In this approach, to learn complicated functions, features are extracted from raw data without the need for human intervention. Learning high-level features usually require a great deal of data. It is very difficult for mankind to analyze high volumes of data. But, graphical processing units (GPU) with thousands of parallel processors can readily process such a huge volume of data.

Recently, ample works have used the deep learning approach for intrusion detection. For example, Gao et al. [4] used DBN to detect intrusion on a modified version of KDD99 and reported its excellent performance. Wang [5] employed SAE to select the main features for intrusion detection on a KDD dataset. There are two approaches to detecting web attacks signature-based and anomaly-based. The former performs better in attack detection in cases that have their signature, but the latter outperforms in detecting new attacks. Some works [6, 7] have tried the combination of these two approaches.

Deep learning methods have recently been utilized for web attack detection. Saxe and Berlin [8] used deep learning-based convolutional networks. Because of the network type, they only classified the URLs and failed to detect the malicious activities in all requests and responses. The application of the deep learning method based on LSTM and GRU allows tracing the sequence of events and finding their common patterns. Common sequences are a feature of web attacks, so they are used for web attack detection.

Liang et al. [9] used GRU and LSTM networks for attack detection. [10] used BI-LSTM to detect malicious activities. Due to their limitations, LSTM and GRU networks can only check URLs and small parts of requests for attack detection.

Given the growing expansion of malicious activities, the reviewed literature has employed deep learning for automatic detection of these activities in order to settle the shortcomings of web attack detection, including the high complexity of features and the difficulty of selecting appropriate features for learning. But, today's attacks are so that one sometimes needs to check multiple requests and responses for attack detection. For example, suppose that an attacker intends to enter a malicious XSS code in the comments of a social network in order to break into the sessions of the users that observe that post. This attack is composed of a request - the malicious code - and the server's response - running the malicious code on the users' web browsers. Here, our system should check multiple requests and responses, which may be time-consuming due to the high volume of the requests and responses and their number.

The present paper aims to propose a method for checking multiple long requests and responses to identify appropriate features for attack patterns. So, the system needs to consider the relationship between the requests and responses and assume that a request is an attack if it provokes the server to give an abnormal response.

The present paper proposes a method called request-response attention that uses an attention-based seq2seq network in which sequence is the input of requests, and sequence is the output of responses. Each request is pre-processed, summarized, and converted into numerical strings by the word2vec method. Then, it is encoded in an attention-based RNN network. In addition, the value of attention is created on the basis of the requests. These values are inputted into a decoder RNN network to reflect the response, and the produced output is inputted into a fully connected layer. The difference in the output of this layer with the real response received from the webserver is used in a SoftMax classification function to classify web traffic flow.

Table 1 classifies the papers and the domains that they cover based on the challenges of web attack detection.

The contributions of the paper are as follows:

- All requests and responses are successfully considered for web traffic flow classification, thereby predicting a sound model of the normal behavior of requests and responses
- Seq2seq networks with attention are used, which

Table 1. The status of the research on the challenges considered in the present paper

Method	Long-term attention in multiple requests-responses	Short events in a few requests	Automatic high-level feature learning
Location-based deep learning [3, 6–8, 11]			✓
Time-based deep learning [9, 10]		✓	✓
Request-response attention (the present work)	✓	✓	✓

enabled our proposed model to detect long requests and responses.

- The relationship between request and response is considered, leading to better detection of most attacks, including XSS attack and SQL Injection, whose success depends on the server's response.

The proposed model was assessed on two datasets including HTTP DATASET CSIC 2010 [12] and CICIDS2017 [13]. The results revealed that the proposed method could detect new attacks while not detecting the normal traffic mistakenly by considering the responses. This is more evident in XSS attacks that depend on the server's response. As well, the model could provide better results than state of the art result.

The next parts of the paper have been organized in this way. Section 2 presents a review of the literature. Section 3 describes the proposed method with its details. In Section 4, it is described how the model was simulated on the datasets, followed with a discussion of the simulation results and its comparison with similar works in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

Web IDS can be divided into two sections: the generation of the dynamic model or static model. In the static model, offline traffic is used to create the model, and this prefabricated model is applied in the next steps. In the dynamic form, the fabricated model is always updated with new traffic. Many works have created dynamic models for network intrusion detection [14–17] . The perspective of web attacks is different from that of network attacks. Web malicious requests have a subtle difference from normal requests. The static models are unable to detect these attacks.

Ingham et al. [18] modeled normal requests with deterministic finite automata (DFA). Malicious requests were specified by comparing with DFA. Torrano et al. [19] developed a method that uses the comparison of web requests with XML files to divide income requests into normal or anomalous. Corona et al. [20] created two distribution models, including the static model and the hidden Markov model on different fields of the HTTP protocol to build models for valid web requests. Zolotukhin et al. [21] used HTTP logs to build an IDS. They first extracted the relevant features with an n-gram model. Then, an IDS was generated by three machine learning algorithms of support vector data description (SVDD), K-mean, and density-based spatial clustering of applications with noise (DBSCAN).

Choras and Kozik [22] employed dynamic programming and graph models to present a method to model normal requests. Kruegel et al. [23] presented a method in which learning models were built, drawing on different features of web traffic, including data length, character distribution across data, data arrangement, the percentage of data existing in requests and so on. These models allocated a number to each feature based on the profile representing the probability of their occurrence. The request would be considered to be an attack if this number exceeded a threshold. Some literature has tried to combine the anomaly-based method with the signature-based method to enjoy the advantages of both methods.

Mac et al. [7] used AutoEncoder for web attack detection. The method has a preprocessing step in which the URL data are replaced with ASCII codes, and they are entered into an AutoEncoder network. This network enhanced precision by 13% when combined with ModSecurity. Similarly, Tekerek et al. [6] tried to combine signature-based and anomaly-based intrusion detection systems built with a single-layer NAA to detect known attacks with the signature-based system and detect new attacks with the anomaly-based system.

An important parameter in neural networks is the selection of appropriate features for learning. Moradi et al. [3] employed SAE and DBN for the automatic selection of high-level features and performed final classification by them. Hao et al. [10] used Bi-LSTM to select the suitable features for deep learning and utilized SoftMax for final classification.

Kirchner et al. [24] employed various classification functions to analyze the features of web requests. Finally, k-nearest became a proper method that allowed the comparison of the individual requests or all normal classes existing in the feature pool in terms of individual features. Rieck et al. [25] presented a model based on the n-gram language model and used the similarity measure between a set of n-gram sequences. Duc Le [26] generated clusters for normal and anomalous data positing that different requests exhibited their own specific behavior. Accordingly, he proposed a self-organizing map (SOM) algorithm.

Tian et al. [11] used a model based on multiple resnet deep learning and word2vec for the analysis of URLs to detect attacks in Edge of Things (EOT). Saxe and Berlin [8] presented a method based on deep learning using convolutional networks to detect malicious URLs. Moh et al. [27] used a multi-stage architecture. In the first step, web logs are checked by a model-based method using Kibana. Then, Bayes Net is used for machine learning to detect SQL injection attacks by web traffic. This method uses the outputs of pattern detection to improve the results of the Bayes Net and the outputs of pattern detection to improve the pattern detection method in the context of a hybrid method. Liang et al. [9] applied two RNN networks based on GRU and LSTM for learning the URL of normal requests unsupervisedly. Then, the outputs of these networks were used by a supervised neural network to distinguish anomalous and normal requests.

3 Proposed Method

To detect web attacks, web servers usually produce certain responses to the requests of clients that vary with request status and time. In most web attacks, web servers usually send anomalous responses to the user request. We aim to detect these situations. Our main idea is to learn requests and normal responses of the webserver. Then, by measuring the similarity of the real response to what the system expects, we reach a measure of the anomaly of the request. RNN networks, such as the one presented in [4], are highly capable of learning response request processes like what happens in a web.

As depicted in Figure 1, request data are sent to the server, and its responses are received by the system. First, all information is converted into lowercase letters inpreparation of the web requests and response' phase, and the superfluous information is eliminated so that only the values of the parameters are left. Thereby, data are summarized. Then, a string of values is converted into numerical tokens by word2vec In the training phase of a DynamicRNN, the requests are first encoded, and the attention is calculated. The yielded value and the calculated attention are decoded in order to be finally revealed in a fully-connected network of responses. These responses are compared with the response received from the server, and the difference is calculated. Then, the difference is mapped in a SoftMax function to 0 or 1 in which 0 means normal traffic and 1 means attack. The model is trained by the labels of the dataset.

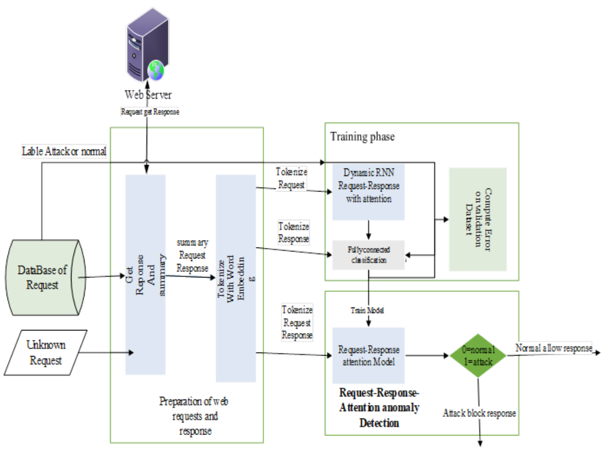

Web Server
Request get Response
Lable Attack or normal
DataBase of Request
Unknown Request
Get Reponse And summary
summary Request Response
Tokenize With Word Embedding
Preparation of web requests and response
Tokenize Request
Tokenize Response
Tokenize Request Response
Training phase
Dynamic RNN Request-Response with attention
Fullyconnected classification
Compute Error on validation Dataset
Train Model
Request-Response attention Model
0=normal 1=attack
Normal allow response
Attack block response
Request-Response-Attention anomaly Detection

Figure 1. The schematics of the proposed model for the dataset to create a classification system into normal and attack

In the detection phase, the previously developed request-response attention model is used to classify the traffic. As such, the requests are sent to the server, and the server's response is compared with the value generated by the network of DynamicRNN with attention. If the output of SoftMax is 1, it shows an attack. Then, the server's response is blocked, and the attack is reported. If the output of the SoftMax function is 0, it means that it is normal, and the response is sent to the client.

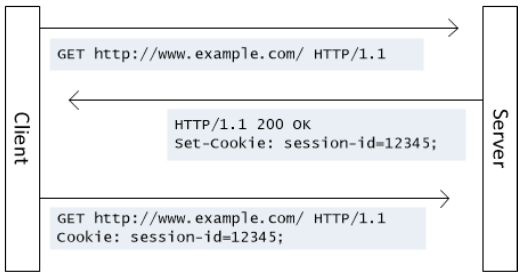
To assess the model, data were randomly divided into training, validation, and test using the k-fold methodology. The model was trained by training and validation data, and it was tested by test data.

3.1 Preparation of Web Requests and Responses

The HTTP protocol constitutes the foundation of web communications. Web server is usually composed of two sections: client and server. Clients send their requests to the server and receive the response. A web server is usually attacked by sending one or more malicious requests. So, in our IDS architecture, we check HTTP requests, sent to the server, to detect the attacks.

As depicted in Figure 2, HTTP requests consist of multiple lines containing headers and request data. For instance, when we want to enter a site, we first send our request to the webserver. In response, the server sends data such as our ID. Then, we can use this ID in our requests to show their relationship to one another.

Since the key of the parameters is less important for attack detection, but their values are more important, we create a string of values in order to summarize the requests and responses. The string is composed of the

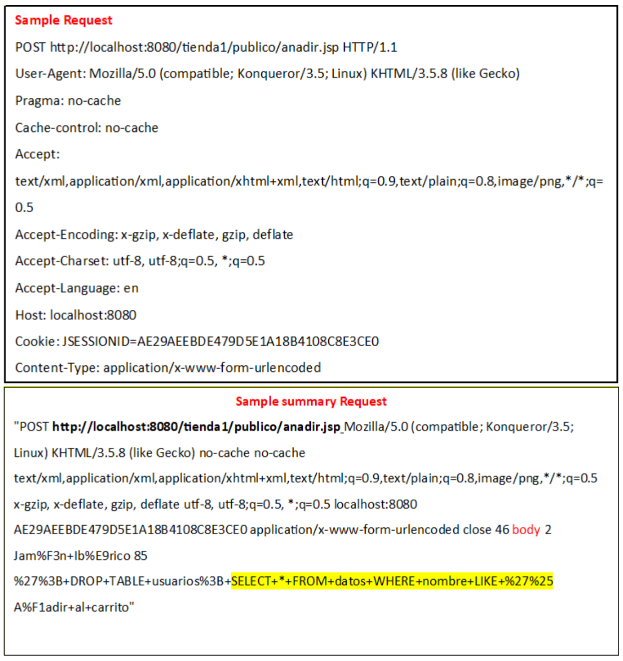


```
GET http://www.example.com/ HTTP/1.1
```

```
HTTP/1.1 200 OK
Set-Cookie: session-id=12345;
```

```
GET http://www.example.com/ HTTP/1.1
Cookie: session-id=12345;
```

Figure 2. The request process in the HTTP protocol



Sample Request

```
POST http://localhost:8080/tienda1/publico/anadir.jsp HTTP/1.1
User-Agent: Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)
Pragma: no-cache
Cache-control: no-cache
Accept:
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=
0.5
Accept-Encoding: x-gzip, x-deflate, gzip, deflate
Accept-Charset: utf-8, utf-8;q=0.5, *;q=0.5
Accept-Language: en
Host: localhost:8080
Cookie: JSESSIONID=AE29AEEBDE479D5E1A18B4108C8E3CE0
Content-Type: application/x-www-form-urlencoded
```

Sample summary Request

```
"POST http://localhost:8080/tienda1/publico/anadir.jsp Mozilla/5.0 (compatible; Konqueror/3.5;
Linux) KHTML/3.5.8 (like Gecko) no-cache no-cache
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
x-gzip, x-deflate, gzip, deflate utf-8, utf-8;q=0.5, *;q=0.5 localhost:8080
AE29AEEBDE479D5E1A18B4108C8E3CE0 application/x-www-form-urlencoded close 46 body 2
Jam%F3n+Ib%E9rico 85
%27%3B+DROP+TABLE+usuarios%3B+SELECT+*+FROM+datos+WHERE+nombre+LIKE+%27%25
A%F1adir+al+carrito"
```

Figure 3. An example request and summary request for the dataset of CSIC2010

values of the parameters separated by a space. At the beginning of the BODY section, the term 'body' is placed. Figure 3 shows an example of a request and its summary.

After generating the summary request, in order to be able to use the data in the RNN-attention model, it is necessary to convert it into a numerical string. This is performed by word embedding, as described in Section 3.3.

3.2 Request-Response Attention Network

This proposed network is illustrated in Figure 4, according to which the HTTP requests and responses are predicted in a dynamic seq2seq with an attention model. The difference of a certain response to a request with the predicted response shows if the traffic is normal and anomalous.

This model converts requests and responses to summarized strings, as described in the previous section. Our idea is to consider requests and responses like

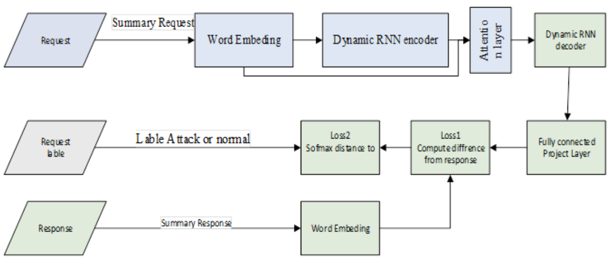


Figure 4. The proposed model architecture. The model classifies the requests into normal or attack by checking the requests and predicting the possible response and the difference with the actual response.

sentences in two different languages that should be translated into one another. We try here to use a machine translation network, which is used for automatic translation, to approximate a server's response to produce the expected response for the requests. Since normal requests are often more than anomalous requests, our system can learn the server's response by obtaining multiple requests, just like a cashing system that can store responses to similar requests of users and resend them. We guess the server's response by the responses already learned. If the produced response is far different from our guess, an attack is said to be done.

As shown in Figure 4, the requests are first encoded in an RNN network. Since our data are long, it is necessary to feed our input data into an attention layer where they are stored in a summarized form and help us in representing the response. This layer contributes to analyzing long requests. Like encoder, we have an RNN decoder that produces the features required for representing the responses and represents the responses in a fully-connected network. Then, their differences with the real answers are calculated. Finally, normal and anomalous requests are classified in a SoftMax classification function.

3.3 Word Embedding

Word embedding is used to map the words to vectors, which are the inputs to the Convolutional Neural Network (CNN, described in the next subsection). There are many branches and research groups working on word Embeddings. For example, a team at Google led by Tomas Mikolov created word2vec, a toolkit that can be used directly to generate vectors. Instead of generated by existing word embedding tools, the word's embedding vectors in our web attacks detection method are learned in the training process. In fact, there is an embedding layer joint with the CNN, and the embedding vectors are optimized through back-propagations of the whole network. We believe that the embedding vectors generated by this way are more helpful to the detection of web attacks because

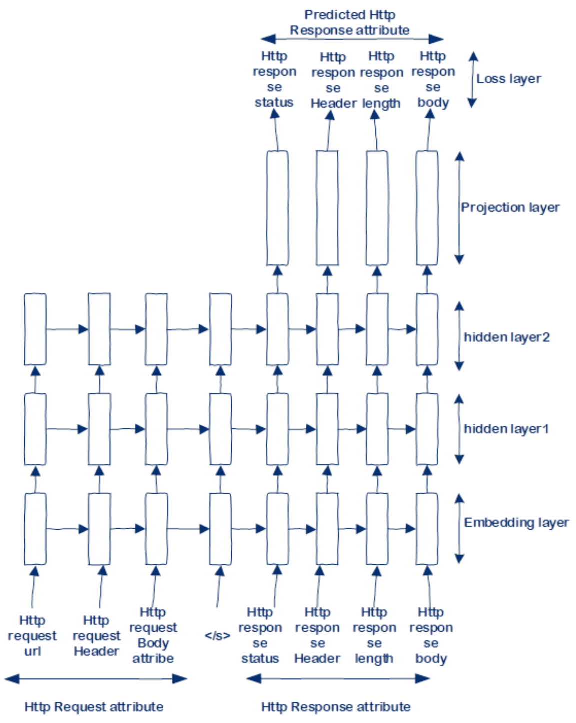


Figure 5. The seq2seq model to predict response on the basis of the HTTP requests

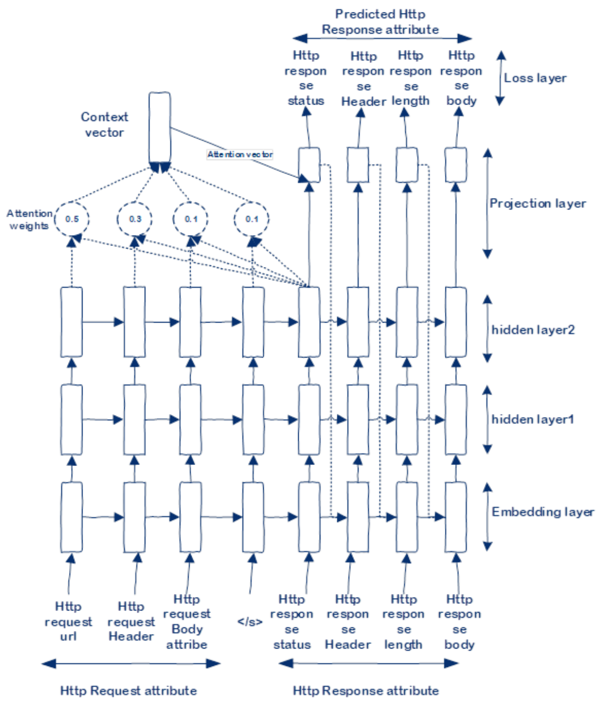


Figure 6. The use of attention model to hinder the loss of data in long requests and to give the model a high precision

they are task-specific and more reflective of their semantic meaning, whereas the vectors generated by third-party tools are relatively static.

3.4 Dynamic RNN with Attention

A summary that is expected to be produced by a server is recognized on the basis of a set of user requests, and the deviation from the real response is calculated. The higher the deviation is, the more anomalous the server response is. In this mechanism, first, each part of the information fields of HTTP-related requests and responses is embedded in a seq2seq network as tokens [28]. This network tries to use a set of GRU or LSTM to code the information of consecutive tokens of the requests to the last layer. This coded value is decoded to predict the possible responses. We employed an LSTM-based RNN in this network, as shown in Figure 5.

Since there are a lot of features in requests, we will lose a great deal of data, and the response cannot be predicted with good precision if the value coded in our requests is merely used. To tackle the problem in [29], the model related to attention is used in addition to the coded value, and all input tokens are used in response prediction. Figure 6 depicts the architecture of the method.

4 Simulation

For simulation, we employed the HTTP dataset CSIC 2010 in the present study. This dataset was created in 2010 and contains 15,000 abnormal and 56,000 normal requests [12]. In the RNN network, four LSTM layers were used for data encoding and decoding. To assess our model in real and newer environments, we simulated the proposed method on the CICIDS2017 dataset, too [13]. This dataset has been developed for the assessment of intuition detection and contains other types of attack in addition to web attacks. This dataset was used in our web traffic section (Tuesday morning, including 170,366 traffic flow), which contained brute force, XSS, and SQL Injection attacks. In order for the dataset to be used for web attacks, all requests and web responses were extracted from it.

The model was assessed by the criteria of accuracy, sensitivity, precision, specificity, F1. Accuracy reflects the percent of data detected correctly by the model, but if the model detects normal traffic mistakenly, it will be abandoned very soon. So, we use two other measures too. Sensitivity shows the detection of anomalous data in the whole existing anomalous traffic, and specificity represents the number of the detected normal data to the whole number of normal data existing in the dataset. These measures are calculated by the following equations:

$TP = TruePositive$ =detect as attack correct

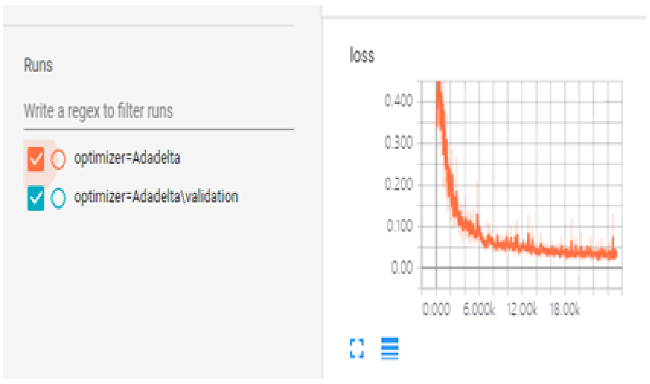


Figure 7. The loss graph pertaining to the tensor board that shows the convergence of the model in the training process

Table 2. Results of simulation for the three measures

Method	Accuracy	Sensitivity	Specificity	Precision	F1
The proposed method with attention	0.991	0.9915	0.9931	0.9755	0.9834
The proposed method without attention	0.982	0.975	0.9852	0.948	0.9613

$TN = TrueNegative$ =detect as normal correct
$FP = FalsePositive$ =detect as attack incorrect
$FN = FalseNegative$ =detect as normal incorrect

$$Accuracy = \frac{Numberofcorrectclassified}{totalnumberofdata}$$
$$Sensitivity = DR = \frac{Numberofattackdatadetectedbymodel}{Totalnumberofattacks}$$
$$Specificity = \frac{Numberofnormaldatadetectedbymodel}{Totalnumberofnormaldata}$$
$$Precision = PR = \frac{TP}{TP+FP}$$
$$F-score = F1 = \frac{2\times PR\times DR}{PR+DR}$$

The model was run in Python using the TensorFlow[1] library. The measure of loss in the proposed model is the system error based on the difference between real responses with the responses predicted by the SoftMax function. As can be observed in Figure 7, after 18,000 training cycles of the model, the accuracy moved towards convergence, and after the 6000th cycle, the loss rate of the model stopped being exponential, reflecting the convergence of our model and the correct procedure of model training.

As described earlier, to assess the model, data were randomly divided into training, validation, and test using the k-fold methodology. The model was trained by training and validation data, and it was tested by test data. The results of the model assessment by test data against three assessment measures are presented in Table 2.

To use the CICIDS2017 database, since we eliminated the null records and a part of traffic flows that were not related to the web and the dataset finally included different types of requests and responses of the web whose number is presented in Table 3, we built four classification models in terms of the attacks that were present in the dataset.

[1] tensorflow.org

Table 3. Number of type of requests and web responses that are present in the CICIDS2017 database

Web traffic type	Count
Dosslowloris	5796
Dos Slowhttp test	5499
Web attack-brute force	1507
Web attack-XSS	652
Web attack-SQL Injection	21

Table 4. The comparison of the presented method with different methods on a CSIC dataset

Method	Accuracy	Sensitivity	Specificity	Precision	F1
The proposed method with attention (this work)	0.991	0.9915	0.9931	0.9755	0.9834
The proposed method without attention (this work)	0.982	0.975	0.9852	0.948	0.9613
ModSecurity with CRS	0.7888	0.0426	0.9951	0.7059	0.0804
Without RNN [9]	0.8515	0.7403	0.9546	0.8185	0.7775
With GRU [9]	0.9788	0.9722	0.9845	0.9455	0.9587
With LSTM[9]	0.9842	0.9756	0.9912	0.9684	0.972
SAE30+1gram_IF [3]	0.8924	0.8948	0.8911	0.8158	0.8535
Regularized Deep Autoencoder [7]	0.9767	0.9462	0.9852	0.9464	0.9463
C4.5[9]	0.965	0.9914	0.8697	0.6779	0.8052
SOM[9]	0.9282	0.9497	0.9242	0.7761	0.8542
Native Bayes[9]	0.8408	0.5235	0.9286	0.6697	0.5877

5 Comparison of Results and Assessment

Table 4 compares our work with the method presented by Liang et al. [9], SAE proposed by Vartouni et al. [3] and RDA proposed by Mac et al. [7]. ModSecurity v3.0.3 powered by CRS Version v3.2.0 installed on Appache webserver was used for ModSecurity with CRS. Since the results of our simulation were comparable, we used the results presented in the literature.

As can be observed in Table 4, the presented method with attention outperformed other algorithms in terms of all measures. Only, the pattern-based sample exhibited poorer performance than ModSecurity in detecting normal traffic as it is known that the pattern-based nature outperforms learning-based methods in normal traffic detection. In all other cases, it was better than the similar methods like GRU and LSTM that are presented in [9] because it considers all request and response data and performs better in attack detection.

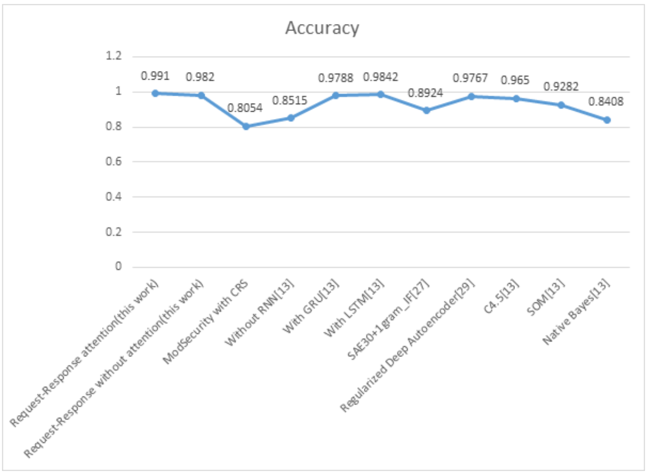


Figure 8. Model assessment in terms of accuracy

5.1 Results Analysis in Terms of Accuracy and Precision

In accuracy measure, two factors are very effective. They include new attack detection and the reduction of errors in normal traffic detection. As can be observed in 8, the worst result of the assessment was related to ModSecurity-based IDS. The drawback of these systems in detecting new attacks is the source of the problem in these methods. Our proposed method that was based on attention could yield the best result, showing an improvement of 0.0068 versus the best result in Liang et al. [9]. This improvement is caused by including other parameters of web requests in addition to URL in attack detection.

Precision shows the ratio of attacks correctly predicted by the model to total data detected as an attack by the model. C4.5 had the worst result in this criterion, and the best result was for the request-response-attention-based method proposed in this paper. As can be observed in Figure 9, It was 0.0071 more precise than the LSTM-based method [9].

5.2 Results Analysis in Terms of Sensitivity

The detection of new attacks plays a key role in the sensitivity measure. It is evident in Figure 10 that the worst result in this assessment was again related to ModSecurity-based detection systems. The inability of these systems in detecting new attacks is the main reason for this poor performance. Our proposed method that was based on attention exhibited the best result and showed an improvement of 0.0159 versus the best result reported by Liang et al. [9]. This improvement was caused by including other parameters of web requests in addition to URL in attack detection.

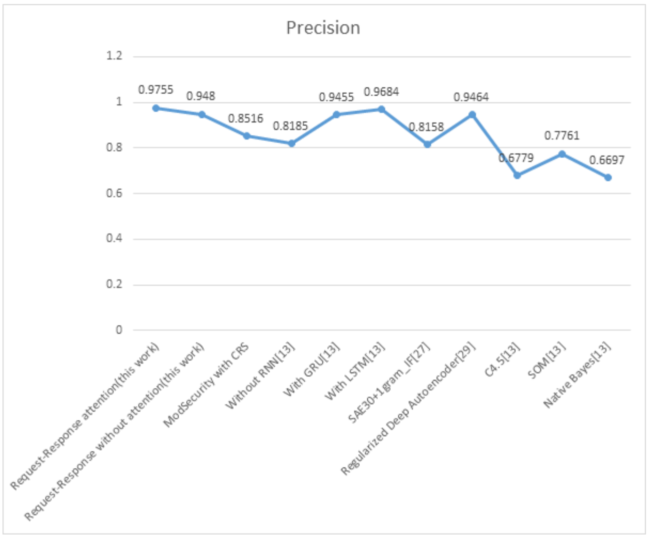


Figure 9. Model assessment in terms of precision

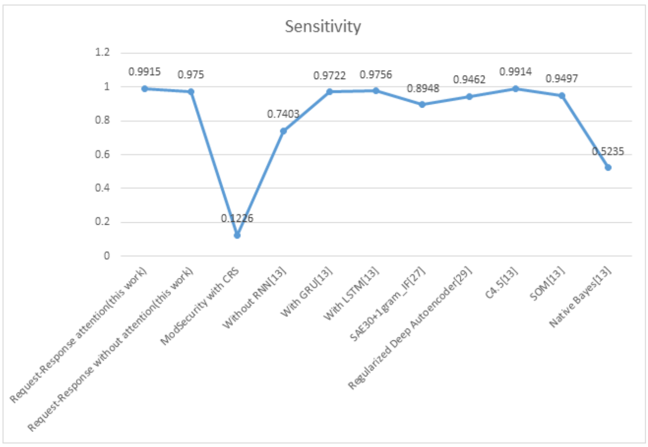


Figure 10. Model assessment in terms of sensitivity

5.3 Results Analysis in Terms of Specificity

Normal traffic detection plays a key role in specificity measure. As is seen in Figure 11, the best result in this assessment is related to the ModSecurity-based detection system. Due to the lower sensitivity of these systems to attack detection and the accuracy of the patterns, these systems can easily detect normal traffic and show lower positive error than other learning methods. But, this has caused them to fail in detecting new attacks. The worst result in this measure is related to the C4.5.

Our attention-based method exhibited the best result among learning-based methods and showed an improvement of 0.0019 when compared to the best result of Liang et al. [9]'s method. This improvement was caused by including other parameters of web requests in addition to URL that contributed to better modeling of normal traffic because most data that show the normality of the traffic may be embedded in the request body. For example, in a text editor on a

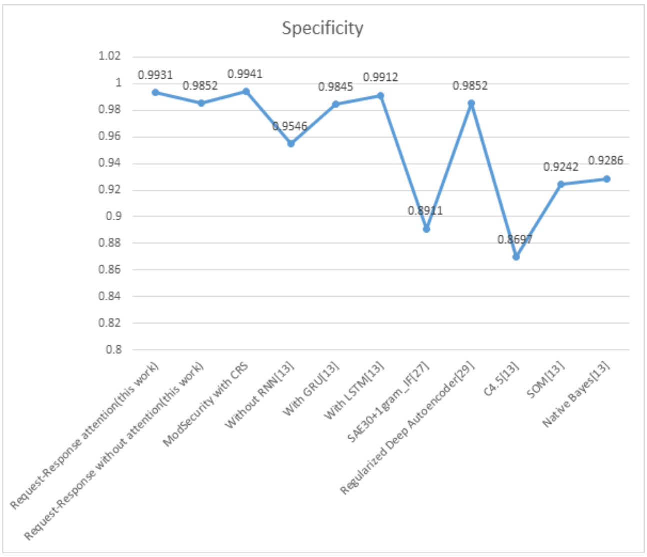


Figure 11. Model assessment in terms of specificity

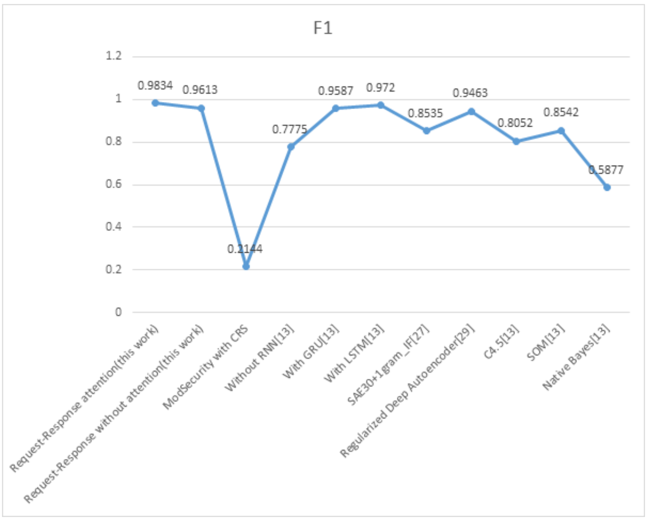


Figure 12. Model assessment in terms of F1

website, most characters may be allowed whilst these characters may be unauthorized on other pages. However, our method showed 0.001% lower performance in normal traffic detection than the ModSecurity-based method. This is the cost of detecting new attacks.

5.4 Results Analysis in Terms of F1

This measure shows the balance between precision and sensitivity. The measure aims to reflect the overall status of the model in terms of attack detection and normal traffic. The signature-based system and ModSecurity showed the worst result in this measure as they failed to detect new attacks. As can be observed in Figure 12, Our request-response attention-based method performed the best in this measure, improving it by 0.114 versus Liang et al. [9]'s LSTM-based method.

Table 5. The results of comparing the precision over the CICIDS2017 database for all attack categories

Web traffic type	Method	Accuracy	Sensitivity	Specificity	Precision	F1
Dosslowloris	The proposed method with attention (this work)	0.9007	0.8322	0.8507	0.9814	0.9069
	The proposed method without attention(this work)	0.8889	0.8188	0.8377	0.9721	0.8955
	ModSecurity with CRS	0.5204	0.69	0.8584	0.4178	0.6688
Dos slowhttp test	The proposed method with attention (this work)	0.8858	0.8178	0.8516	0.966	0.8983
	The proposed method without attention(this work)	0.8285	0.746	0.7813	0.9317	0.8426
	ModSecurity with CRS	0.4991	0.6574	0.8555	0.4023	0.6706
Web brute force	The proposed method with attention (this work)	0.6526	0.5425	0.9131	0.8189	0.9025
	The proposed method without attention(this work)	0.5988	0.4972	0.9042	0.7525	0.8872
	ModSecurity with CRS	0.2474	0.3197	0.946	0.2018	0.8628
Web attack-XSS	The proposed method with attention (this work)	0.8089	0.6871	0.9773	0.9832	0.9776
	The proposed method without attention(this work)	0.594	0.466	0.9523	0.8191	0.9459
	ModSecurity with CRS	0.298	0.3561	0.9765	0.2562	0.9416
SQL Injection	The proposed method with attention (this work)	0.8373	0.8182	0.9998	0.8572	0.9995
	The proposed method without attention(this work)	0.5085	0.3948	0.9983	0.7143	0.9979
	ModSecurity with CRS	0.6667	0.7223	0.9997	0.6191	0.9991

5.5 Results of Assessment over the CICIDS2017 Database

To better assess our model, we assessed it over the CICIDS2017 database, too, whose results are presented in Table 5. It shows the simulation results of the proposed method in two cases of having or lacking the attention layer for each class and compares them with the ModSecurity results (with the same specifications mentioned in Section 5).

As can be observed in Table 5, the proposed algorithm performed well in the case of XSS attack in terms of specificity. This can be attributed to considering the responses in addition to the requests and the use of the attention mechanism so that it could eliminate erroneous positives well and even better than ModSecurity, improving it by 0.008. Also, it performed the best in detecting SQL injection attack in terms of specificity.

6 Conclusion

This research work presented a novel method of web attack detection on the basis of seq2seq networks with attention. The presented method could model normal traffic properly by predicting the possible responses and contrasting them with real responses of the server; thereby, it could discriminate normal traffic from attack by the comparison.

The highest accuracy of the proposed method vis-a-vis the similar methods reflects the fact that the use of the attention mechanism can tackle the challenge of studying long web requests and can use more information embedded in web data to detect intrusions. Future works can focus on modifying the presented model into a dynamic model by using reinforcement learning techniques, such as policy gradient so as for the model to be updated against the feedbacks of pattern and user-based systems.

References

[1] *Symantec Internet Security Threat Report*, volume 24. 2019.

[2] Yoshua Bengio, Ian J Goodfellow, and Aaron Courville. *Deep learning*. The MIT Press, 2016.

[3] A Moradi Vartouni, M Teshnehlab, and S Sedighian Kashi. Leveraging deep neural networks for anomaly-based web application firewall. *IET Information Security*, 13(4):352–361, 2019.

[4] Ni Gao, Ling Gao, Quanli Gao, and Hai Wang. An intrusion detection model based on deep belief networks. In *Advanced Cloud and Big Data (CBD), 2014 Second International Conference on*, pages 247–252. IEEE, 2014.

[5] Zhanyi Wang. The applications of deep learning on traffic identification. *BlackHat USA*, 2015.

[6] A Tekerek and O.F. Bay. Design and implementation of an artificial intelligence-based web application firewall model. *Neural Network World*, 29(4):189–206, 2019.

[7] Hieu Mac, Dung Truong, Lam Nguyen, Hoa Nguyen, Hai Anh Tran, and Duc Tran. Detecting Attacks on Web Applications Using Autoencoder. In *Proceedings of the Ninth International Symposium on Information and Communication Technology*, SoICT 2018, pages 416–421, New York, NY, USA, 2018. ACM.

[8] Joshua Saxe and Konstantin Berlin. eXpose: {A} Character-Level Convolutional Neural Network with Embeddings For Detecting Malicious URLs, File Paths and Registry Keys. *CoRR*, abs/1702.08568, 2017.

[9] Jingxi Liang, Wen Zhao, and Wei Ye. Anomaly-Based Web Attack Detection: A Deep Learning Approach. In *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, ICNCC 2017, pages 80–85, New York, NY, USA, 2017. ACM.

[10] Saiyu Hao, Jun Long, and Yingchuan Yang. BL-IDS: Detecting Web Attacks Using Bi-LSTM Model Based on Deep Learning. In Jin Li, Zheli Liu, and Hao Peng, editors, *Security and Privacy in New Computing Environments*, pages 551–563, Cham, 2019. Springer International Publishing.

[11] Z Tian, C Luo, J Qiu, X Du, and M Guizani. A Distributed Deep Learning System for Web Attack Detection on Edge Devices. *IEEE Transactions on Industrial Informatics*, page 1, 2019.

[12] CarmenTorrano Giménez, Alejandro Pérez Villegas, Gonzalo Álvarez, and Marañón. HTTP data set CSIC 2010. 2010.

[13] Iman Sharafaldin., Arash Habibi Lashkari., and Ali A Ghorbani. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,*, pages 108–116. INSTICC, SciTePress, 2018.

[14] Bo Dong and Xue Wang. Comparison Deep Learning Method to Traditional Methods Using for Network Intrusion Detection. *8th IEEE International Conference on Communication S oftw are and N etw ork s*, 2016.

[15] Shahriar Mohammadi and Amin Namadchian. A New Deep Learning Approach for Anomaly Base IDS using Memetic Classifier. *International Journal of Computers, Communications & Control*, 12(5), 2017.

[16] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A Deep Learning Approach for Network Intrusion Detection System. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016.

[17] Jaehyun Jihyun Kim, Jaehyun Jihyun Kim, Huong Le Thi Thu, and Howon Kim. Long Short

Term Memory Recurrent Neural Network Classifier for Intrusion Detection. *2016 International Conference on Platform Technology and Service (PlatCon)*, 2016.

[18] Kenneth L Ingham, Anil Somayaji, John Burge, and Stephanie Forrest. Learning DFA Representations of HTTP for Protecting Web Applications. *Comput. Netw.*, 51(5):1239–1255, 2007.

[19] Camen Torrano-Giménez, Alejandro Perez-Villegas, and Gonzalo Alvarez Maranón. An anomaly-based approach for intrusion detection in web traffic. 2010.

[20] Igino Corona, Roberto Tronci, and Giorgio Giacinto. SuStorID: {A} multiple classifier system for the protection of web services. In *Proceedings of the 21st International Conference on Pattern Recognition, {ICPR} 2012, Tsukuba, Japan, November 11-15, 2012*, pages 2375–2378, 2012.

[21] M Zolotukhin, T Hämäläinen, T Kokkonen, and J Siltanen. Analysis of HTTP Requests for Anomaly Detection of Web Attacks. In *2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing*, pages 406–411, 2014.

[22] M Choraś and R Kozik. Machine learning techniques applied to detect cyber attacks on web applications. *Logic Journal of the IGPL*, 23(1):45–56, 2015.

[23] Christopher Kruegel, Giovanni Vigna, and William Robertson. A multi-model approach to the detection of web-based attacks. *Computer Networks*, 48(5):717–738, 2005.

[24] M Kirchner. A framework for detecting anomalies in HTTP traffic using instance-based learning and k-nearest neighbor classification. In *2010 2nd International Workshop on Security and Communication Networks (IWSCN)*, pages 1–8, 2010.

[25] Konrad Rieck and Pavel Laskov. Detecting Unknown Network Attacks Using Language Models. In *Proceedings of the Third International Conference on Detection of Intrusions and Malware & Vulnerability Assessment*, DIMVA'06, pages 74–90, Berlin, Heidelberg, 2006. Springer-Verlag.

[26] Duc Le Jr. An unsupervised learning approach for network and system analysis. 2017.

[27] Melody Moh, Santhosh Pininti, Sindhusha Doddapaneni, and Teng-Sheng Moh. Detecting web attacks using multi-stage log analysis. In *Advanced Computing (IACC), 2016 IEEE 6th International Conference on*, pages 733–738. IEEE, 2016.

[28] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA, 2014. MIT Press.

[29] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective Approaches to Attention-based Neural Machine Translation. *CoRR*, abs/1508.04025, 2015.

Shahriar Mohammadi is a former senior lecturer at the University of Derby, UK. He also used to be a Network consultant in the UK for more than fifteen years. He is currently a lecturer in the Industrial Eng. Department of the University Of K.N.Toosi, Iran. His main research interests and lectures are in the fields of networking, data security, network security, e-commerce, and e-commerce security. He has published more than eighty papers in various journals and conferences, as well as four books.

Amin Namadchian was born in Sabzevar, Iran, in 1987. He graduated in information technology engineering in 2008. He is currently a Ph.D. candidate in the Department of Industrial Eng K.N.Toosi. His main research interest is intrusion detection system, deep neural networks, evolutional algorithm, and security.