Journal of Quantum Computing
DOI:10.32604/jqc.2020.010819
Article

A Novel Intrusion Detection Algorithm Based on Long Short Term Memory Network

Xinda Hao[1], Jianmin Zhou[2,*], Xueqi Shen[1] and Yu Yang[1]

[1]School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, 100876, China
[2]College of New Media, Beijing Institute of Graphic Communication, Beijing, 102600, China
[*]Corresponding Author: Jianmin Zhou. Email: la1azjm@gmail.com

Received: 04 May 2020; Accepted: 28 August 2020

Abstract: In recent years, machine learning technology has been widely used for timely network attack detection and classification. However, due to the large number of network traffic and the complex and variable nature of malicious attacks, many challenges have arisen in the field of network intrusion detection. Aiming at the problem that massive and high-dimensional data in cloud computing networks will have a negative impact on anomaly detection, this paper proposes a Bi-LSTM method based on attention mechanism, which learns by transmitting IDS data to multiple hidden layers. Abstract information and high-dimensional feature representation in network data messages are used to improve the accuracy of intrusion detection. In the experiment, we use the public data set KDD-Cup 99 for verification. The experimental results show that the model can effectively detect unpredictable malicious behaviors under the current network environment, improve detection accuracy and reduce false positive rate compared with traditional intrusion detection methods.

Keywords: Bi-LSTM; kdd-cup99; intrusion detection; deep learning

1 Introduction

With the rapid development of cloud computing, more and more companies and individuals rely on cloud services to provide information technology support, and cyber-attacks have become one of the important issues that threaten enterprise security [1]. Intrusion detection is a proactive security protection technology that provides real-time protection against internal attacks, external attacks, and misoperations. It can respond and intercept before a hazard occurs, effectively protecting company assets from loss [2]. However, real-time detection of cyber-attacks is not a simple matter [3]. In the past few years, many models and methods based on traditional machine learning have been proposed for network intrusion detection. For example, in the thesis [4] based on the K-NearestNeighbor (KNN) algorithm, the thesis [5] uses a random forest-based algorithm, although some results have been achieved, but the effect is still not satisfactory. The traditional machine learning method belongs to shallow learning, and usually emphasizes feature engineering and selection, which cannot effectively solve the large-scale intrusion data classification problem in the actual network application environment [6]. Moreover, traditional machine learning methods also perform poorly when dealing with attack texts with contextual information, making it difficult to cope with a variety of new types of cyber-attacks such as Advanced Persistant Threat (APT), and it is difficult to cope with increasingly complex a threat. On the contrary, deep learning has excellent performance in the automatic learning and expression of features, and has a good modeling ability. The thesis [7] uses the Deep Neural Network (DNN) sequential automatic encoder framework, although it has achieved good results, but lacks memory to recall the previous network information. The Recurrent Neural Network (RNN) can resolve this limitation by maintaining a loop from the current state to the previous state for information persistence. Among them, LSTM neural network has shown excellent results in many fields such as natural language processing. The feature of this model is that it can better capture the context dependencies of the longer distance from the past to the future, so it is very suitable for modeling time series data. For example, the thesis [8] uses LSTM to achieve good results because the investigation of a single deep packet may not be sufficient to effectively detect malicious behavior. The knowledge of previous network traffic status can be used to understand the current state of the network, but in this paper data that is not considered for future use can also be used to detect attacks over a period of time. Therefore, we use Bi-LSTM modeling of bidirectional sequential long-term memory networks and introduce attention mechanisms to record attribute features that have a significant impact on malicious behavior prediction.

To improve the above problems, we use Bi-LSTM model and introduce attention mechanisms to record attribute features that have a significant impact on malicious behavior prediction. In this article, our main contributions include:

–Study the current network attack detection scheme based on machine learning.

–We propose a new attention-based network intrusion detection model that captures key information that is useful for classification and is superior to the LSTM algorithm in accelerating the classification process. The model also improves the detection accuracy for cyber threat behavior.

–We studied the performance of Naive Bayes, Logistic Regression, Support Vector Machine and KNN and other machine learning methods in the benchmark dataset KDDCup99 classification, and compared with our proposed model.

2 DataSet

2.1 KDDCup 99

In the field of network intrusion detection, public data sets (such as KDDCup 99, NSL-KDD, etc.) are commonly used to analyze the effectiveness of classical machine learning and deep neural networks of various NIDS [9]. The KDDCup 99 dataset is a network simulation environment built by Lincoln Labs to simulate the US Air Force LAN [10]. It monitors and collects simulated traffic and attack traffic in this environment. The data set consists of a total of 5 million records. It also provides a 10% training subset and test subset. Its sample category distribution table is shown in Tab. 1.

Table 1: Sample Categories in the KDDCup 99 Data Set

	TOTAL	NORMAL	PROBE	DOS	U2R	R2L
Train Set	494021	97278	4107	391458	52	1126
Test Set	311029	60593	4166	229853	228	16189

Each record of the data set records 41 attribute tags and 1 category tag. Category tags are classified into normal or abnormal. The exceptions can be classified into four types of attack types according to attack type characteristics: DoS (denial of service attack), R2L (unauthorized access from remote hosts), U2R (unauthorized local superuser privileged access) and PROBING (port monitoring or scanning). The anomaly can be subdivided into 39 different attacks, 22 of which appear in the training set, and 17 unknown types of attacks appear in the test set.

2.1 Data Preprocessing

The model we designed can only accept digital vector input, so we need to convert the character features in the original data into numeric features. In each record, we need to convert the protocol type, network service type, network connection status, and attack type to digital IDs using one-hot encoding. Take the protocol type as an example. The protocol type is a discrete type. There are three types: TCP, UDP, and ICMP. Because there are three protocol states, it is represented by 3 bits, namely: TCP: 1 0 0, UDP: 0 1 0, ICMP: 0 0 1.

Data normalization, to eliminate the dimensional influence between feature data, needs to be normalized to resolve the comparability between feature indicators. After the original data is normalized, the indicators are in the same order of magnitude, so that the optimization process of the optimal solution becomes smoother and easier to converge to the optimal solution correctly. Here we use the min-max normalization method to linearly change the raw data and linearly scale the feature data between 0 and 1. The conversion function is as follows, where x' represents the normalized value, x represents the value of the original feature, and min and max represent the minimum and maximum values of the feature in the original data set, respectively:

$$x' = \frac{x-min}{max-min} \tag{1}$$

3 Model

In this section we present the details of the Bi-LSTM model. Fig. 1 shows an overview of the structure of the proposed model. Based on the Bi-LSTM model with attention mechanism, it consists of five components: Input layer, embedded layer, Bi-LSTM layer, attention layer and output layer. Next we will detail the composition of these components.

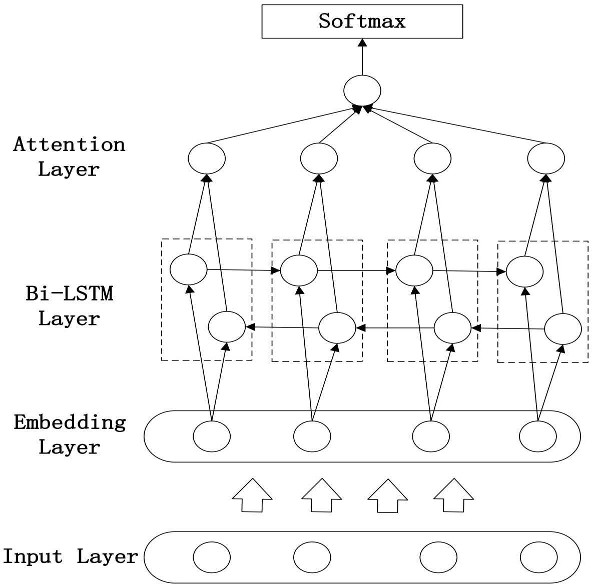

Figure 1: Bi-LSTM + Attention model

3.1 Embeddings Layer

In this paper, we describe each record in the dataset as a set of feature sequences and generate a vector $\boldsymbol{x_i}$ for each feature. For a set of input sequences **S** = $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \cdots, \boldsymbol{x_t}\}$, we can get the embedded matrix **V** = $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3}, \cdots, \boldsymbol{v_t}\}$. The eigenvector formula for each feature item in the record is as follows:

$$v_i = ReLU(W_e x_i + b_e) \tag{2}$$

where $W_e \in R^{m\times1}$ is a weight matrix, $b_e \in R^m$ is an offset matrix, and m is the dimension of the embedded matrix. The generated feature vector V is then passed as an argument to the next layer.

3.2 Bi-LSTM

Although RNN can theoretically solve the training of sequence data well, it also has the same problem as the gradient disappears like DNN. The problem is especially serious when the sequence is very long. Therefore, the above RNN model is generally not directly applicable to the application domain.

To solve the gradient disappearance problem in the cyclic neural network, LSTM introduces a gate mechanism to control the degree of historical information retained by each LSTM unit and to memorize the currently input information, retain important features, and discard unimportant features. The LSTM unit design is shown below:

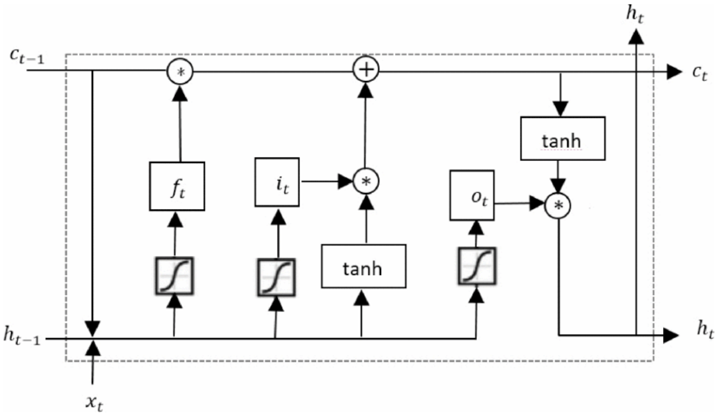

Figure 2: Design of LSTM unit

Each LSTM unit consists of three main gates that control the effect of current network traffic data on the status of the storage unit. The calculation of all gates is affected by the current input data x_t and the output value h_{t-1} of the LSTM unit at the previous moment, and is also affected by the value of the storage unit c_{t-1}.

The calculation formula for the input gate is as follow:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{3}$$

Forgotten door calculated as follow:

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{4}$$

The calculation formula for updating the cell state information is as follow:

$$c_t = f_t c_{t-1} + i_t \tanh\,(W_{xc}x_t + W_{hc}h_{t-1} + W_{cc}c_{t-1} + b_c) \tag{5}$$

The calculation formula for the output gate is as follow:

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{6}$$

$$h_t = o_t tanh\,(c_t) \tag{7}$$

where W terms represent the weight matrix, W_{hi}, W_{hf}, W_{hc}, W_{ho} represent the weight matrix of the hidden layer attached to the input gate, the forgetting gate, the unit state and the output gate; σ is the sigmoid function; i_t, f_t, c_t, and o_t represent the time t, respectively Input gates, forgetting gates, cell states and output gates.

For the task of sequence modeling, future information and historical information at each moment are equally important. The standard LSTM model does not capture future information in its order. So we use a two-way LSTM that puts two opposing LSTMs together and shares the same input and output layers. In this way, the trained data can be correlated with past and future information. Then we will use the output vector of the last time series as the feature vector, and finally perform Softmax classification. Attention is to calculate the weight of each time series first, then weight the sum of all time series vectors as the feature vector, and then perform Softmax classification.

3.3 Attention Mechanism

In the process of network intrusion detection, more attention should be paid to important attributes. The attention mechanism is mainly to imitate the function of people's attention. The principle is to mark the contribution of their hidden state to the classification result by calculating the hidden state weights generated at different times. To take advantage of information from all previously hidden states, we take attention mechanisms to capture the relationship between $h_1, h_2, \dots, h_t$, and finally obtain the final vector pair representation for classification:

$$\alpha = softmax(W^T[h_1, h_2, \dots, h_t]) \tag{8}$$

$$h^* = tanh(H\alpha^T) \tag{9}$$

where W^T is a transpose of a parameter vector obtained by training learning.

3.4 Output Layer

In this paper, we use the Softmax classifier to predict the classification label $y^{\wedge}$, h^*as the input of the hidden layer. The formula is as follows:

$$p = softmax(W_s h^* + b_s) \tag{10}$$

$$y^{\wedge} = argmax(p) \tag{11}$$

Among them W_s and b_s are parameters that need to be learned.

In this paper we use cross entropy as the loss function.

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i log(p_{i1}) + (1 - y_i)log(1 - p_{i1}) \tag{12}$$

where N represents the number of samples and p_{i1} represents the probability that the ith sample is predicted to be malicious.

4.1 Evaluation Indicators

In our model, the accuracy rate (AC) is mainly used to measure the performance indicators of the intrusion detection LSTM-AT model. In addition to the accuracy rate, we also introduced the recall rate and false positive rate. The True Positive (TP) case indicates the number of samples predicted as normal by the model in normal traffic. The Fake Positive (FP) case indicates the number of samples predicted as normal by the model in abnormal traffic. The Fake Negative (FN) case indicates the number of samples predicted as abnormal by the model in normal flow. The True Negative (TN) case indicates the number of samples predicted as abnormal by the model in abnormal traffic. Tab. 2: Confusion matrix is the definition of the confusion matrix.

Table 2: Confusion matrix

Actual Class Prediction Class	Normal	Anomaly
Normal	TP	FP
Anomaly	FN	TN

Accuracy (AC): The model predicts the correct number of samples as a percentage of the total number of samples:

$$AC = \frac{TP+FN}{TP+TN+FP+FN} \tag{13}$$

Recall Rate (TPR): The model predicts the number of correct samples in normal flow as a percentage of total samples in normal flow:

$$TPR = \frac{TP}{TP+FN} \tag{14}$$

False alarm rate (FPR): The model predicts the number of error samples in abnormal flow as a percentage of total samples in abnormal flow:

$$FPR = \frac{FP}{FP+TN} \tag{15}$$

On the one hand, from the perspective of the classifier, accuracy and detection rate are a pair of contradictory indicators. Higher accuracy means fewer false positives, but higher detection rates mean fewer false positives. For example, if more suspicious attacks are classified as attacks (extremely all records are classified as attacks), the detection rate will increase, but the accuracy will decrease, and vice versa. Therefore, a single high precision or detection rate is meaningless. On the other hand, from the perspective of intrusion detection, especially in some strict environments, the tolerance of intrusion is very low, so the recall rate is also an important indicator to consider.

4 Analysis of the Experiment and Results

To enhance the training results of the neural network model, we use hyperparameters to accomplish this task. We define hyperparameter values and apply them to the Bi-LSTM+Attention model. Using the hyperparameters shown in Tab. 3 to train the model has the best effect.

Table 3: Hyperparameter values

Hyperparameters	Value
Learning rate	0.001
Number of hidden layers	15
Number of epochs	200
Batch size	600

We use the KDD-Cup99 data set to validate our proposed solution, which is the most popular benchmark data set for cyber-attack detection. All of the following experiments were performed on a PC with GeForce® GTX TITAN X, 12 GB GDDR5 7GHz memory and Ubuntu 16.04 operating system. The LSTM and Bi-LSTM+Attention algorithms were implemented in Tensorfolw 1.4.0 using Python 3.5. For the analysis of experimental results, we compare the results with various classical machine learning classifiers such as LR, NB, KNN and SVM. The experimental results are shown in Tab. 4, and the model has been well trained and has achieved high accuracy. In terms of accuracy, recall rate and false positive rate, we notice that the effect of LSTM is better than the classical machine learning classifier, and the Bi-LSTM+Attention model proposed in this paper is superior. The original LSTM model. This is because our proposed model can better focus on the connection between important features and reduce the weak features that affect the detection rate. Fig. 3 compares the convergence of the LSTM algorithm with our proposed model. It can be seen from the figure that the experiment converges after about 22 epochs, indicating that the model can stop training at an earlier stage and the training speed is faster. Combined with previous results, our model not only has the advantage of high precision, but also has a faster convergence speed. Finally, we can conclude that using the method we designed for intrusion detection can achieve a good two-category effect.

Table 4: Experimental results

Classifier name	Accuracy (AU)	Recall rate (TPR)	False alarm rate (FPR)
LSTM	0.953	0.927	0.061
Bi-LSTM+Attention	0.983	0.935	0.034
LR	0.831	0.831	0.132
KNN	0.916	0.902	0.082
NB	0.872	0.849	0.097
SVM	0.893	0.852	0.076

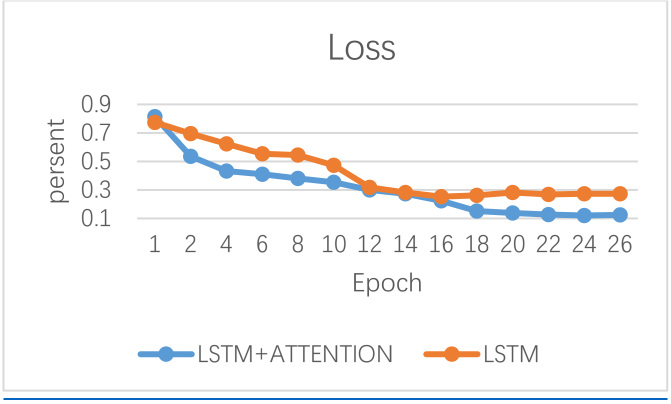

Figure 3: Loss curve

5 Conclusion

We propose a Bi-directional LSTM Neural Network intrusion detection model based on Attention mechanism, which has a timing characteristic that can comprehensively consider the analysis of network traffic. The experimental results show that the proposed model has better results than LSTM. In future research, we will further study the use of multi-GPU and distributed training models to improve intrusion detection performance.

Acknowledgement: This work is supported by the National Key R&D Program of China (2017YFB0802703), Major Scientific and Technological Special Project of Guizhou Province (20183001), Open Foundation of Guizhou Provincial Key VOLUME XX, 2019 Laboratory of Public Big Data (2018BDKFJJ014), Open Foundation of Guizhou Provincial Key Laboratory of Public Big Data (2018BDKFJJ019, 2018BDKFJJ022).

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] D. Larson, “Distributed denial of service attacks–Holding back the flood,” *Network Security*, vol. 2016, no. 3, pp. 5–7, 2016.

[2] Nadeem, Adnan and M. P. Howarth, “A survey of manet intrusion detection & prevention approaches for network layer attacks,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2027–2045, 2013.

[3] N. Sultana, N. Chilamkurti, W. Peng, R. Alhadad, “Survey on SDN based network intrusion detection system using machine learning approaches,” *Peer-to-Peer Networking and Applications*, vol. 1, no. 2, pp. 1–9, 2018.

[4] W. Li, P. Yi, Y. Wu, L. Pan and J. Li, “A new intrusion detection system based on KNN classification algorithm in wireless sensor network,” *Journal of Electrical & Computer Engineering*, vol. 2014, no. 4, pp. 345–351, 2014.

[5] N. Farnaaz and M. A. Jabbar, “Random forest modeling for network intrusion detection system,” *Procedia Computer Science*, vol. 89, no. 1, pp. 213–217, 2016.

[6] N. Farah, M. Avishek, F. Muhammad, A. Rahman and D. Md, “Application of machine learning approaches in intrusion detection system: A survey,” *International Journal of Advanced Research in Artificial Intelligence*, vol. 6, no. 3, pp. 42–46*,* 2015.

[7] R. Vinayakumar, M. Alazab, S. Kp, P. Poornachandran, A. Al-Nemrat *et al.,* “Deep learning approach for intelligent intrusion detection system,” *IEEE Access*, vol. 1, no. 1, pp. 56–59, 2019.

[8] R. C. Staudemeyer, “Applying long short-term memory recurrent neural networks to intrusion detection,”

South African Computer Journal, vol. 56, no. 1, pp. 129–134, 2015.

[9] T. Janarthanan, S. Zargari, "Feature selection in UNSW-NB15 and KDDCUP99 datasets," in *IEEE Int. Sym. on Industrial Electronics*, vol. 57, no. 1, pp. 34–39, 2017.

[10] R. C. Staudemeyer, "KDD Cup 1999 Data," 2018. [Online]. Available: https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.