Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 2632678, 13 pages
https://doi.org/10.1155/2023/2632678

Research Article

Application-Layer DDoS Attack Detection Using Explicit Duration Recurrent Network-Based Application-Layer Protocol Communication Models

Bailin Xie[1], **Yu Wang**[2], **Guogui Wen**[1] **and Xiaojun Xu**[1]

[1]*School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou, China*
[2]*Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, China*

Correspondence should be addressed to Bailin Xie; bailinxie@gdufs.edu.cn

Received 10 March 2023; Revised 3 June 2023; Accepted 7 June 2023; Published 17 June 2023

Academic Editor: Yu-an Tan

Existing application-layer distributed denial of service (AL-DDoS) attack detection methods are mainly targeted at specific attacks and cannot effectively detect other types of AL-DDoS attacks. This study presents an application-layer protocol communication model for AL-DDoS attack detection, based on the explicit duration recurrent network (EDRN). The proposed method includes model training and AL-DDoS attack detection. In the AL-DDoS attack detection phase, the output of each observation sequence is updated in real time. The observation sequences are based on application-layer protocol keywords and time intervals between adjacent protocol keywords. Protocol keywords are extracted based on their identification using regular expressions. Experiments are conducted using datasets collected from a real campus network and the CICDDoS2019 dataset. The results of the experiments show that EDRN is superior to several popular recurrent neural networks in accuracy, $F1$, recall, and loss values. The proposed model achieves an accuracy of 0.996, $F1$ of 0.992, recall of 0.993, and loss of 0.041 in detecting HTTP DDoS attacks on the CICDDoS2019 dataset. The results further show that our model can effectively detect multiple types of AL-DDoS attacks. In a comparison test, the proposed method outperforms several state-of-the-art approaches.

1. Introduction

With the progress of increasingly advanced network infrastructure and network layer defense technologies, attackers increasingly turn to Internet-based applications as their attack targets, resulting in the continuous emergence of application-layer attacks [1, 2]. These attacks are carried out using legitimate user requests and protocols at the application-layer. Therefore, the data flow of application-layer attacks at the network and transport layers is not significantly different from that generated by normal users.

Distributed denial of service (DDoS) attacks are one of the most dangerous attacks [3–6], especially application-layer DDoS (AL-DDoS) attacks, such as HTTP DDoS attacks [7, 8] and SMTP flood attacks [9]. The HTTP DDoS attacks are usually implemented by a large number of bots sending a flood of page requests to a web server at the same time, thus consuming server resources, such as database cycles, CPU cycles, or memory. In August 2022, Google encountered the largest HTTP DDoS attack in history, which tried to shut down Google's Cloud Armor customer service, with a peak of 46 million requests per second [10]. The complexity of AL-DDoS attacks is also expected to grow over time.

Existing AL-DDoS attack detection methods are mainly targeted at specific attacks but cannot effectively identify other types of AL-DDoS attacks. Therefore, to comprehensively detect AL-DDoS attacks, multiple detection methods need to be deployed in a network. However, the principles and parameter settings of each detection method are fundamentally different, which complicates network management. Moreover, deploying multiple AL-DDoS attack detection methods simultaneously will also lead to the degradation of network performance. Hence, it is necessary to design a detection method that can effectively detect various AL-DDoS attacks.

In this study, we reexamine the issue from the perspective of application-layer protocol communication. The key idea is to model the communication of the application-layer protocol through an explicit duration recurrent network (EDRN) to detect AL-DDoS attacks, taking observed application-layer protocol keywords and time intervals between adjacent protocol keywords as inputs. Application-layer protocol keywords refer to custom request commands and server response status codes, which can reflect the behavior of users when using the protocol.

Recurrent neural networks (RNNs) exploit cycles in network nodes to capture the dynamics of sequences, and they have been widely used in sequential data mining with outstanding performance results [11]. However, traditional RNNs have hidden states whose durations approximately follows a geometric or exponential distribution [12, 13]. As a result, it is difficult to use traditional RNNs to model the variable durations of hidden states.

During the communication process of the application-layer protocol, the behavior of users and time intervals between adjacent protocol keywords are determined by many factors, such as the request method, network transmission delay, and response processing time of servers. Thus, the duration of hidden states under a sequence of application-layer protocol communication may follow a relatively complex distribution, and not necessarily a geometric or exponential distribution. The EDRN is based on an extended hidden semi-Markov model (HSMM) and can describe hidden states of any duration distribution [14]. This study adopts EDRN to model application-layer protocol communication for AL-DDoS attack detection. To evaluate the model, experiments are conducted on the CICDDoS2019 dataset [15] and datasets collected in a real campus network. The experimental results show that the EDRN is superior to several popular RNNs, and the EDRN-based model can effectively detect multiple types of AL-DDoS attacks.

The main contributions of this study can be summarized as follows:

(i) We proposed an EDRN-based application-layer protocol communication model. The model uses EDRN to describe the communication process of the application-layer protocol and takes application-layer protocol keywords and time intervals between adjacent protocol keywords as inputs for the first time.

(ii) Based on the application-layer protocol communication model, we proposed an attack detection method that detects AL-DDoS attacks in real time by monitoring the application-layer protocol keywords that are used in the process of protocol communication.

(iii) We compared several RNNs based on the CICDDoS2019 dataset and a real campus network dataset, and the experimental results showed that the EDRN has the best performance. We also compared our proposed AL-DDoS attack detection method with several existing methods, and the test results confirmed the effectiveness and superiority of our method.

The remainder of this paper is organized as follows: Section 2 reviews recent studies on AL-DDoS detection. In Section 3, we describe the model for application-layer protocol communication. Section 4 presents the proposed AL-DDoS attack detection method. The experimental results are presented in Section 5 and discussed in Section 6. Section 7 concludes the paper.

2. Related Works

The detection of AL-DDoS attacks has attracted the attention of researchers [16–21]. Existing methods are mainly targeted at specific AL-DDoS attacks. For example, Xie and Yu [22] used HSMM, independent component analysis, and principal component analysis to mine web server logs to detect HTTP DDoS attacks. Wang et al. [23] used the Hellinger distance and sketch data structure to detect HTTP DDoS attacks. Zhou et al. [24] calculated the entropy of flash crowds and attacks for HTTP DDoS attack detection. Singh et al. [25] used four behavioral features and a support vector machine (SVM) to detect HTTP DDoS attacks. Praseed and Thilagam [26] used probabilistic timed automata (PTA) models to describe the behavior of legitimate users for HTTP DDoS attack detection. Lin et al. [27] used the rhythm matrix statistical model to capture the characteristics of user access trajectories to detect HTTP DDoS attacks. Zhao et al. [28] used URL access entropy to identify HTTP DDoS attacks. Praseed and Thilagam [29] used signatures based on HTTP request patterns to detect HTTP DDoS attacks. Raja Sree and Mary Saira Bhanu [30] used fuzzy bat clustering to analyze web server logs for HTTP DDoS attack detection in the cloud.

In terms of SMTP flood attack detection, Tudosi et al. [31] analyzed the traffic of SMTP flood attacks and used Snort (open source intrusion prevention system) to detect SMTP flood attacks. Schneider et al. [32] used the statistical characteristics of attack flows to detect SMTP flood attacks. Aziz and Okamura [33] adopted deep learning algorithms to detect SMTP flood attacks on software-defined networking (SDN) platforms. Gurusamy and Msk [34] detected SMTP flood attacks by monitoring all ports' traffic statistics in the SDN.

In addition, Kasim [35] used the convolutional neural network (CNN) and long short-term memory (LSTM) to detect DNS flood attacks. Trejo et al. [36] used a visual platform and K-nearest neighbor (KNN) classification algorithm to detect DNS flood attacks. Datta et al. [37] detected DNS flood attacks by monitoring the DNS query per second in IoT networks. Bushart and Rossow [38] used an anomaly-based low-pass filter to detect DNS flood attacks.

Existing methods are mainly targeted at specific AL-DDoS attacks and do not consider the characteristics of application-layer protocol communication. In this study, we adopt EDRN to describe the communication process of the application-layer protocol, which can capture the suddenness, randomness, and volume of protocol communication, and then present an EDRN-based application-layer protocol communication model for AL-DDoS attack detection. This model can effectively detect multiple types of AL-DDoS attacks.

3. Application-Layer Protocol Communication Models

From the perspective of application-layer protocols, when using an application-layer protocol, user behavior over a period of time is reflected in the application-layer protocol; that is, the interaction between a series of application-layer protocol keywords. Application-layer protocol keywords refer to custom request commands and server response status codes, which can reflect the behavior of users when using the application-layer protocol. For example, HTTP protocol keywords include request commands "POST," "GET," and "HEAD," and server response status codes "100," "200," "304," and "404," while SMTP protocol keywords are composed of "MAIL FROM," "HELO," "RCPT TO," "VRFY," "QUIT," "REST," "DATA," "EXPN," "HELP," and "NOOP" and server response codes, such as "250" and "334."

3.1. Application-Layer Protocol Communication Process. When regular users are using an application-layer protocol, the statistical characteristics of the protocol keywords and the time intervals between adjacent protocol keywords are quite different from those of AL-DDoS attacks. For example, when regular users are using the HTTP protocol, their speed of clicking pages, time taken to, and the process of browsing pages have certain stability. However, in the application-layer protocol keyword sequences generated by HTTP DDoS attacks, the protocol keyword "GET" appears very frequently, while other protocol keywords appear less frequently, and the time intervals between adjacent protocol keywords are small. Therefore, the application-layer protocol keywords and the time intervals between adjacent protocol keywords can be used as observations to describe the communication process of the application-layer protocol and enable the detection of AL-DDoS attacks.

Figure 1 shows the communication process between users and a web server represented by a sequence of HTTP protocol keywords, wherein the HTTP protocol keyword sequence representing users' behavior is as follows: "GET," "POST," "200," "HEAD," "304," . . ., "200," and "GET."

3.2. Application-Layer Protocol Keyword Extraction. We first identify the application-layer protocol based on regular expressions, and then extract the protocol keywords. In this way, the number of protocol keywords to be matched each subsequent time can be reduced, thereby improving the speed of the protocol keyword extraction process. When identifying a TCP-based application-layer protocol, the first few data packets of each TCP connection are cached, then the application-layer data of the data packets are reassembled, and finally the protocol regular expression [39] is matched against the reassembled application-layer data. When identifying a UDP-based application-layer protocol, we use regular expressions to match the payload of each data packet. The identification process of TCP-based application-layer protocols is shown in Figure 2. This method can identify application-layer protocols in real time.

3.3. Protocol Communication Modeling. At the gateway of a network, we can obtain application-layer protocol keywords and their arrival times using the protocol keyword extraction method described in Section 3.2. Assuming that the application-layer protocol has W keywords, which can be digitized as: $1, 2, ..., W$. When users are using the application-layer protocol, the communication process can be described as a series of observations thus: $I_1^t = \{I_1, I_2, \ldots, I_t\}$, where I_t $(t \geq 2)$ is the observation at the t^{th} time that the protocol keyword arrives the gateway. The value of I_t is based on the protocol keyword and the time interval between adjacent protocol keywords arriving at the gateway; that is, $I_t = (i_t^{(1)}, i_t^{(2)})$, where $i_t^{(1)}$ is the digitized label of the t^{th} protocol keyword arriving at the gateway, and $i_t^{(2)}$ is expressed by equation (1). In equation (1), R_t denotes the time the t^{th} protocol keyword arrives the gateway, and R_{t-1} denotes the time the $(t-1)^{\text{th}}$ protocol keyword arrives the gateway. In this study, the unit of time measurement is chosen as seconds; I_1 is the observation generated by the first protocol keyword arriving at the gateway, where $i_1^{(1)}$ is the digitized label of the first protocol keyword arriving at the gateway, and $i_1^{(2)} = 0$.

$$i_t^{(2)} = -\lg\left(R_t - R_{t-1}\right). \tag{1}$$

When using the application-layer protocol, users' behaviors may change. For example, users may use the HTTP protocol for varied purposes, including browsing web pages, watching movies online, and shoping online. Therefore, the protocol keywords and time intervals between adjacent protocol keywords arriving at the gateway will change over time. Therefore, the durations of hidden states in the observation sequences of an application-layer protocol communication process may follow a relatively complex distribution.

We used the EDRN to model the communication process of the application-layer protocol. The EDRN-based application-layer protocol communication model is shown in Figure 1, where x_t is the next possible states' predicted probabilities at the $(t+1)^{\text{th}}$ time and y_t is the all possible states' probabilities at the t^{th} time. The unfolded unit structure of EDRN is presented in Figure 3, where tan$h$ denotes the hyperbolic tangent function and σ denotes the sigmoid function. $\mathbf{Z}_{\text{input}}, \mathbf{Z}_{\text{forget}}, \mathbf{Z}_{\text{output}}$, and $\mathbf{Z}_{\tan h}$ denote input, forget, output, and tanh gates, respectively.

Assuming that the communication process of the application-layer protocol has K macrostates, and each macrostate has L substates. $\mathbf{Z}_{\text{forget}}, \mathbf{Z}_{\text{input}}, \mathbf{Z}_{\tan h}$, and $\mathbf{Z}_{\text{output}}$ are calculated using the following equations:

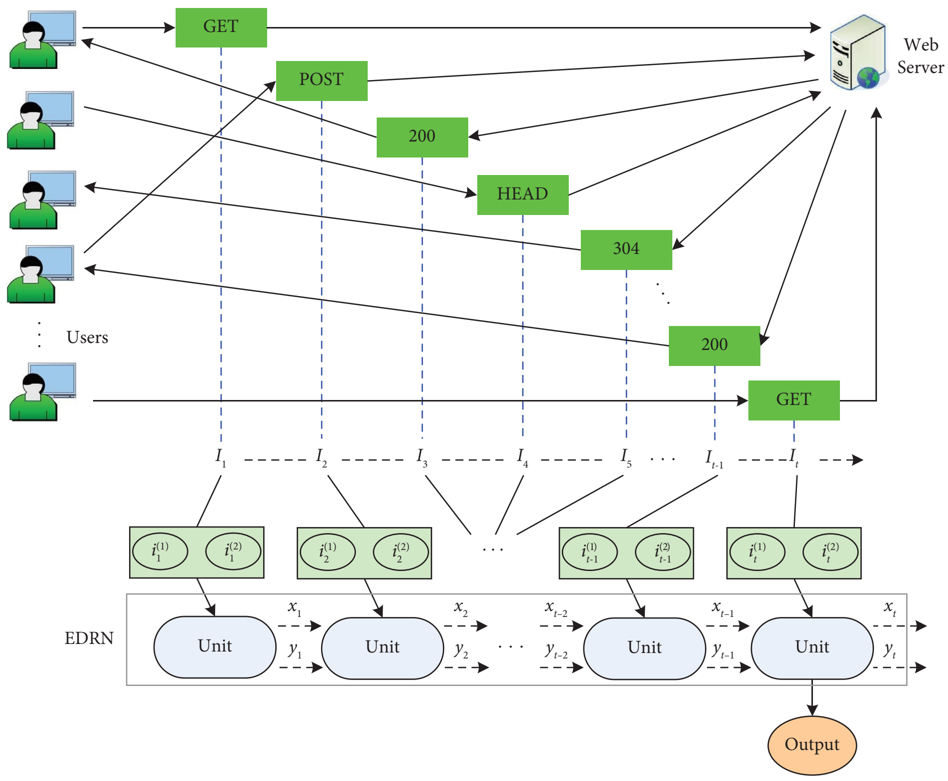

GET
POST
200
HEAD
304
200
GET
Web Server
Users
I_1 I_2 I_3 I_4 I_5 I_{t-1} I_t
$i_1^{(1)}$ $i_1^{(2)}$ $i_2^{(1)}$ $i_2^{(2)}$ $i_{t-1}^{(1)}$ $i_{t-1}^{(2)}$ $i_t^{(1)}$ $i_t^{(2)}$
EDRN
Unit
x_1 y_1 x_2 y_2 x_{t-2} y_{t-2} x_{t-1} y_{t-1} x_t y_t
Output

Figure 1: Application-layer protocol communication model.

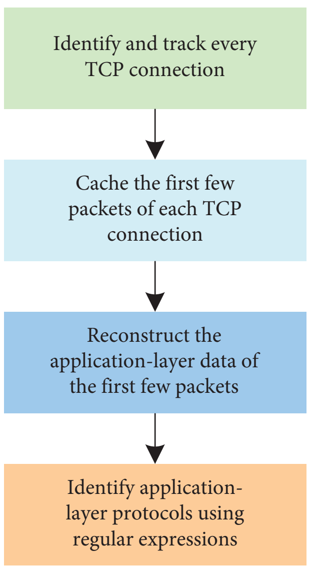

Identify and track every TCP connection
Cache the first few packets of each TCP connection
Reconstruct the application-layer data of the first few packets
Identify application-layer protocols using regular expressions

Figure 2: TCP-based application-layer protocol identification.

$$\mathbf{Z}_{\text{forget}} = \sigma\left(x_{t-1}\mathbf{A}^{(1)} + I_t\mathbf{B}^{(1)} + b^{(1)}\right), \quad (2)$$

$$\mathbf{Z}_{\text{input}} = \sigma\left(x_{t-1}\mathbf{A}^{(2)} + I_t\mathbf{B}^{(2)} + b^{(2)}\right), \quad (3)$$

$$\mathbf{Z}_{\tan h} = \tan h\left(y_{t-1}\mathbf{A}^{(*)} + y_{t-1}(:L)\mathbf{A}^{(3)} + I_t\mathbf{B}^{(3)} + b^{(3)}\right), \quad (4)$$

$$\mathbf{Z}_{\text{output}} = \sigma\left(y_t(:L)\mathbf{A}^{(4)} + I_t\mathbf{B}^{(4)} + b^{(4)}\right). \quad (5)$$

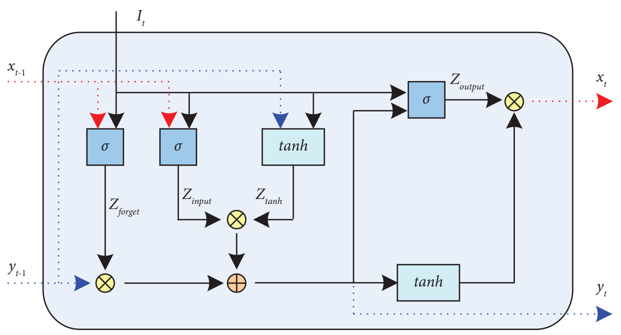


Figure 3: Unfolded unit structure.

In the above equations, $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$, $\mathbf{A}^{(3)}$, and $\mathbf{A}^{(4)}$ denote probability matrixes of state transition; $b^{(1)}$, $b^{(2)}$, $b^{(3)}$, and $b^{(4)}$ are bias parameters following a marginal distribution; $\mathbf{B}^{(1)}$, $\mathbf{B}^{(2)}$, $\mathbf{B}^{(3)}$, and $\mathbf{B}^{(4)}$ denote probability matrixes of observations; $\mathbf{A}^{(*)}$ is the probability matrix of substate transition. In equations (4) and (5), ":" symbolizes all the states.

Similar to LSTM, each unit finally returns (x_t and y_t) to the next unit. The x_t and y_t are calculated using equations (6) and (7), respectively, where "*" represents the element-wise production as follows:

$$x_t = \tan h\left(y_t\right) * \mathbf{Z}_{\text{output}}, \quad (6)$$

$$y_t = y_{t-1} * \mathbf{Z}_{\text{forget}} + \mathbf{Z}_{\text{input}} * \mathbf{Z}_{\tan h}. \quad (7)$$

4. AL-DDoS Attack Detection

The AL-DDoS detection method proposed in this study involves two phases. In the first phase, we train the EDRN-based protocol communication model. In the second phase, every application-layer protocol communication process is monitored in real time. Once a protocol keyword arrives at the network gateway, the corresponding observation sequence I_1^t will be updated, where t denotes the number of observations. Then, we calculate the output η using following equation:

$$\eta = \log \frac{\left[\sum_v \sum_\tau y_t(v,\tau)\right]}{t}. \tag{8}$$

In equation (8), $y_t\ (v,\tau)$ denotes the probability of p_t under I_1^t and $p_t = (v,\tau)$ denotes that $\tau\,(1 \le \tau \le L)$ protocol keywords will appear in state $v\,(1 \le v \le K)$. The $y_t\ (v,\tau)$ is defined and expressed as follows:

$$y_t(v,\tau) \equiv \Pr\left[p_t = (v,\tau) \middle| I_1^t\right], \tag{9}$$

$$y_t(v,\tau) = \xi_{v,\tau} y_{t-1}(v,\tau) \chi_{v,\tau}^{(2)}(I_t) + \left(\sum_{l<\tau} y_{t-1}(v,l) a_{(v,l),(v,\tau)} + \sum_{k\neq v} y_{t-1}(v,L) a_{(k,L),(v,\tau)}\right) \chi_{v,\tau}^{(1)}(I_t). \tag{10}$$

In equation (10), $a_{(k,l),(v,\tau)}$ denotes the interstate transition probability from $p_t = (k,l)$ to $p_{t+1} = (v,\tau)$ and is defined by following equation:.

$$a_{(k,l),(v,\tau)} \equiv \Pr\left[p_t = (k,l), p_{t+1} = (v,\tau)\right]. \tag{11}$$

In equation (10), $\xi_{v,\tau}$ is the probability of $p_t = (v,\tau)$ and defined by following equation:

$$\xi_{v,\tau} \equiv \Pr\left[p_t = (v,\tau)\right]. \tag{12}$$

In equation (10), $\chi_{v,\tau}^{(1)}(I_t)$ and $\chi_{v,\tau}^{(2)}(I_t)$ are defined by equations (13) and (14), where $\Pr\left[I_t | I_1^{t-1}\right]$ is the scaling factor as follows:

$$\chi_{v,\tau}^{(1)}(I_t) \equiv \frac{\Pr\left[I_t \middle| p_t = (v,\tau), p_{t-1} \neq p_t, I_1^{t-1}\right]}{\Pr\left[I_t \middle| I_1^{t-1}\right]}, \tag{13}$$

$$\chi_{v,\tau}^{(2)}(I_t) \equiv \frac{\Pr\left[I_t \middle| p_t = (v,\tau), p_{t-1} = p_t, I_1^{t-1}\right]}{\Pr\left[I_t \middle| I_1^{t-1}\right]}. \tag{14}$$

If η is larger than a predefined threshold, the network is considered as normal. Otherwise, we consider that there is an AL-DDoS attack related to this protocol in the network. The detection architecture of our method is shown in Figure 4. Our method can detect AL-DDoS attacks in real time.

5. Evaluation

In this section, we test our proposed AL-DDoS attack detection method using multiple datasets to evaluate the detection performance against HTTP DDoS and SMTP flood attacks.

5.1. Datasets

5.1.1. HTTP Datasets. At the gateway of the campus network, we collected the data generated by a large number of normal users when using the HTTP protocol. In addition, we adopted the method described in [16] and DDoS generators to generate three different types of HTTP DDoS attacks, namely, single-page, random-page, and top-five--page HTTP DDoS attacks. A single-page HTTP DDoS attack targets a specific page of a website, usually one that is frequently visited by users, while a random-page HTTP DDoS attack targets a random page from all potentially visited pages of a website. A top-five-page HTTP DDoS attack targets the top five most visited pages from a resource site. Subsequently, we extracted observation sequences from the collected data for training and testing. The time length of each observation sequence was 60 seconds. The HTTP datasets are summarized in Table 1.

5.1.2. SMTP Dataset. Similar to HTTP data collection, we collected data generated by a large number of normal users when using the SMTP protocol. We adopted the method described in [18] to generate SMTP flood attacks. After that, we extracted observation sequences. The time length of each observation sequence was equally 60 seconds. The SMTP dataset is summarized in Table 2.

5.1.3. CICDDoS2019 Dataset. The CICDDoS2019 dataset is a public dataset developed by the Canadian Institute for Network Security (CIC) in 2019 [15]. This dataset is one of the popular datasets and is widely used in the field of DDoS detection. The dataset contains 11 kinds of DDoS attacks, among which the AL-DDoS attack is HTTP DDoS attack. The packet in the CICDDoS2019 dataset contains the application-layer payload. We use this dataset to test the performance of our method against HTTP DDoS attacks.

5.2. Estimation Criteria. In the recurrent neural network training phase of our proposed AL-DDoS detection method, we use accuracy and loss as evaluation metrics, while in the

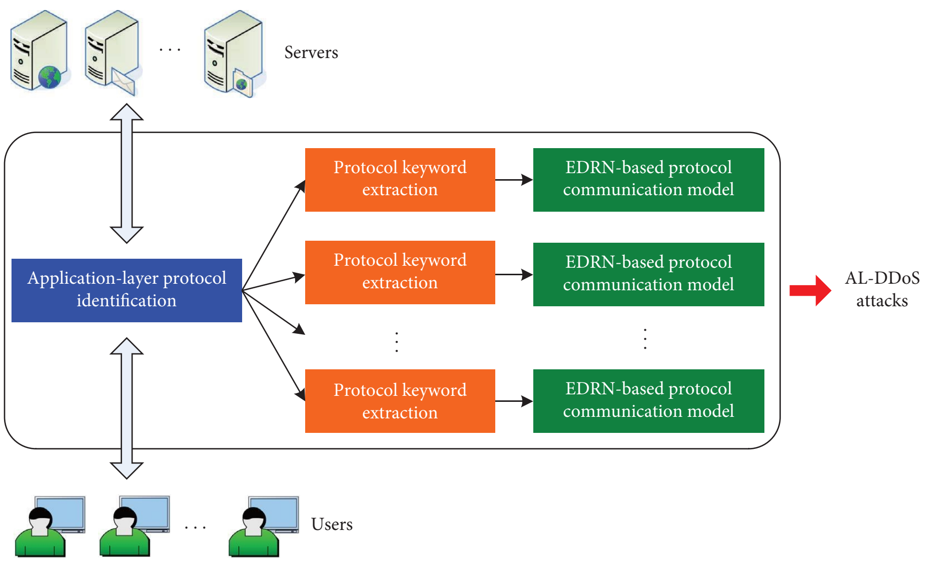

Servers
Application-layer protocol identification
Protocol keyword extraction
EDRN-based protocol communication model
AL-DDoS attacks
Users

FIGURE 4: AL-DDoS attack detection architecture.

TABLE 1: HTTP datasets.

Dataset	Data	# training sequences	# testing sequences
D_1	Normal	3500	1723
	Single-page HTTP DDoS attacks	3500	1648
D_2	Normal	3500	1765
	Random-page HTTP DDoS attacks	3500	1711
D_3	Normal	3500	1836
	Top-five-page HTTP DDoS attacks	3500	1659

AL-DDoS attack detection phase, we use accuracy, $F1$, recall, and loss as evaluation metrics. In the comparison experiment with other methods, we use accuracy, $F1$ and recall as evaluation metrics. The loss is calculated using following equation:

$$\text{loss} = -\sum_t \left[\vartheta_t \times \ln(\lambda_t) + (\vartheta_t - 1) \times \ln(1 - \lambda_t)\right]. \quad (15)$$

In equation (15), ϑ_t is the label value of the sample and λ_t is the predicted value of the recurrent neural network.

5.3. AL-DDoS Attack Detection Results. In this section, experiments are carried out on a computer with 64 bit Ubuntu OS (version: 20.04.1), TensorFlow (version: 1.14.0), Python (version: 3.6.2), and Keras (version: 2.2.5). To prove that the EDRN can better model application-layer protocol communication, we compared it with other RNNs, including LSTM [12], GRU [40], PLSTM [13], IndRNN [41], and DSTP-RNN [42]. In the recurrent neural network training phase of our AL-DDoS detection method, the maximum value of the epoch was set to 100.

5.3.1. Detection Results on HTTP Datasets. The training results of different RNNs on D_1, D_2 and D_3 datasets, as the epoch changes, are shown in Figures 5(a) and 5(b), 6(a) and 6(b), 7(a) and 7(b), respectively. The EDRN had a higher training accuracy and lower training loss on the D_1 dataset

TABLE 2: SMTP dataset.

Dataset	Data	# training sequences	# testing sequences
D_4	Normal	3500	1824
	SMTP flood attacks	3500	1737

than the other RNNs. When the epoch reached 100, the training accuracy rates of LSTM, GRU, PLSTM, IndRNN, DSTP-RNN, and the EDRN were 0.9921, 0.9929, 0.9914, 0.9923, 0.9926, and 0.9981, respectively, while their training loss rates were 0.0274, 0.0211, 0.0247, 0.0230, 0.0206, and 0.0105, respectively.

On the D_2 dataset, the EDRN had the lowest training loss and highest training accuracy. At the end of training, the training accuracy rates of LSTM, GRU, PLSTM, IndRNN, DSTP-RNN, and the EDRN were 0.9929, 0.9939, 0.9933, 0.9936, 0.9941, and 0.9979, respectively, while their training loss rates were 0.0207, 0.0174, 0.0202, 0.0188, 0.0156, and 0.0050, respectively.

On the D_3 dataset, the training accuracy of EDRN was the highest. When the epoch was 100, the training accuracy rates of LSTM, GRU, PLSTM, IndRNN, DSTP-RNN, and the EDRN were 0.9928, 0.9940, 0.9933, 0.9938, 0.9943, and 0.9976, respectively. Conversely, the training loss of the EDRN was the lowest. At the end of training, the training loss rates of LSTM, GRU, PLSTM, IndRNN, DSTP-RNN,

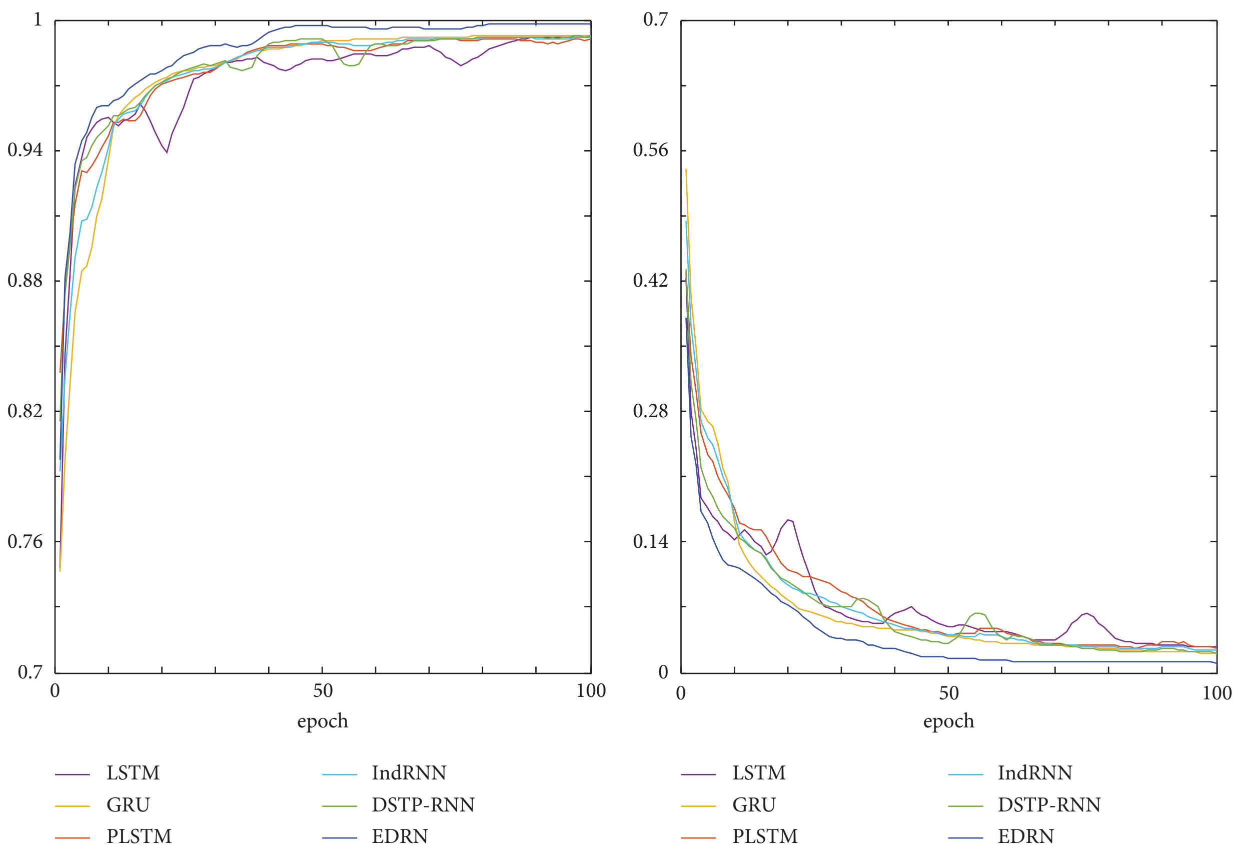


Figure 5: Training results on D_1 dataset. (a) Training accuracy. (b) Training loss.

and the EDRN were 0.0225, 0.0184, 0.0209, 0.0197, 0.0168, and 0.0065, respectively.

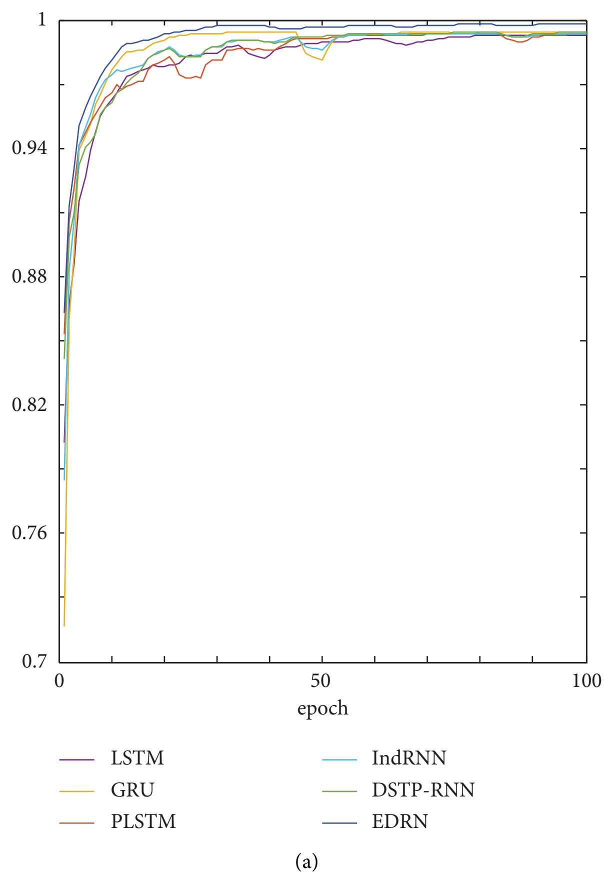
The lower the training loss and the higher the training accuracy, the better the performance of the recurrent neural network is. Therefore, the EDRN performed best on D_1, D_2, and D_3 datasets in the training phase. On the HTTP datasets, the average training accuracy and loss of the EDRN were 0.9978 and 0.0073.

After training, we used the corresponding testing sets to evaluate the EDRN and other RNNs. The test results are listed in Table 3, and as shown, the EDRN had the highest accuracy, $F1$, and recall, and the lowest loss on D_1, D_2, and D_3 datasets. Hence, the EDRN had the best performance in the HTTP DDoS attack detection phase. On the HTTP datasets, the average test accuracy, $F1$, recall and loss of the EDRN were 0.995, 0.991, 0.992, and 0.042, respectively.

5.3.2. Detection Results on SMTP Dataset. The training results on the SMTP dataset are shown in Figures 8(a) and 8(b). At the end of training, the training accuracy rates of LSTM, GRU, PLSTM, IndRNN, DSTP-RNN, and the EDRN were 0.9926, 0.9937, 0.9931, 0.9936, 0.9941, and 0.9986, respectively; the training loss rates of LSTM, GRU, PLSTM, IndRNN, DSTP-RNN, and the EDRN were 0.0224, 0.0183, 0.0203, 0.0193, 0.0165, and 0.0062, respectively. In the training phase, the EDRN attained lowest training loss and the highest training accuracy. That is, EDRN achieved the best performance in the training phase. A comparison of test results is shown in Table 4. Compared with other RNNs, EDRN had a better performance in detecting SMTP flood attacks.

5.3.3. Detection Results on CICDDoS2019 Dataset. We compared the EDRN with other RNNs on the CICDDoS2019 dataset. A comparison of the test results is shown in Table 5 and show that the EDRN achieved the best performance in detecting HTTP DDoS attacks on the CICDDoS2019 dataset.

5.3.4. Comparison with Existing Approaches. In this section, we compare our proposed AL-DDoS attack detection method with several existing state-of-the-art approaches. Accuracy, $F1$ and recall are adopted as evaluation metrics.



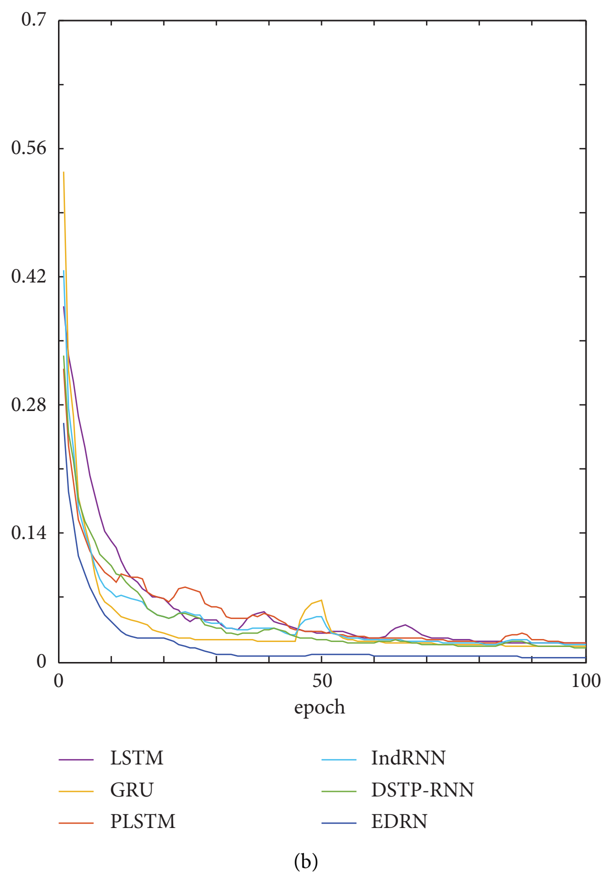


Figure 6: Training results on D_2 dataset. (a) Training accuracy. (b) Training loss.

The comparison results based on HTTP, SMTP, and CICDDoS datasets are presented in Tables 6–8, respectively.

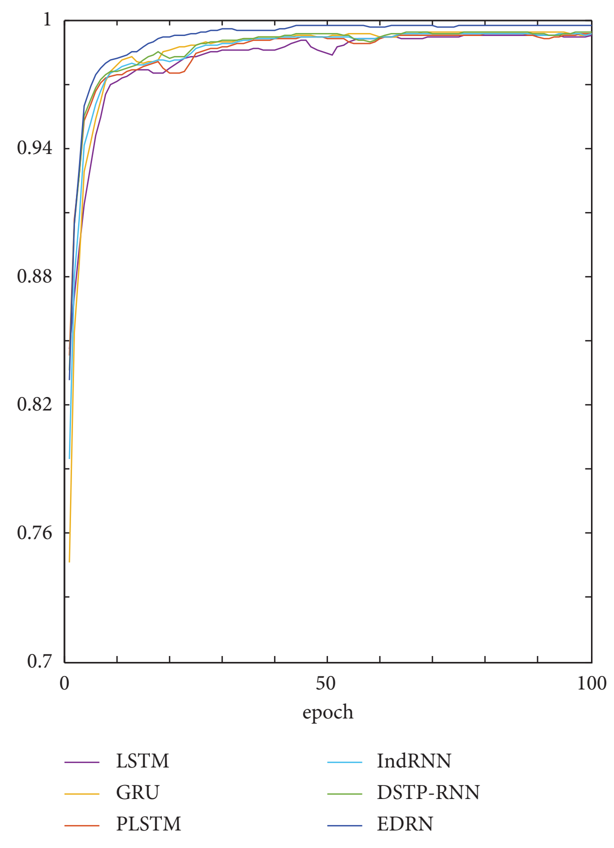
Existing approaches use traditional statistical analysis or machine learning algorithms to detect HTTP DDoS attacks, while this study uses EDRN to construct an application-layer protocol communication model to detect HTTP DDoS attacks. The EDRN is a novel recurrent neural network that has better performance than traditional statistical analysis and machine learning algorithms in sequence data mining. Therefore, our method outperforms existing approaches in detecting HTTP DDoS attacks.

Existing approaches do not consider the characteristics of application-layer protocol communication when detecting SMTP flood or HTTP DDoS attacks. However, the communication process of the application-layer protocol can better reflect the users' behavior. This study uses EDRN to describe the communication process of the application-layer protocol, which can capture the suddenness, randomness, and volume of protocol communication. Therefore, our method has better performance than existing approaches in detecting SMTP flood attacks.

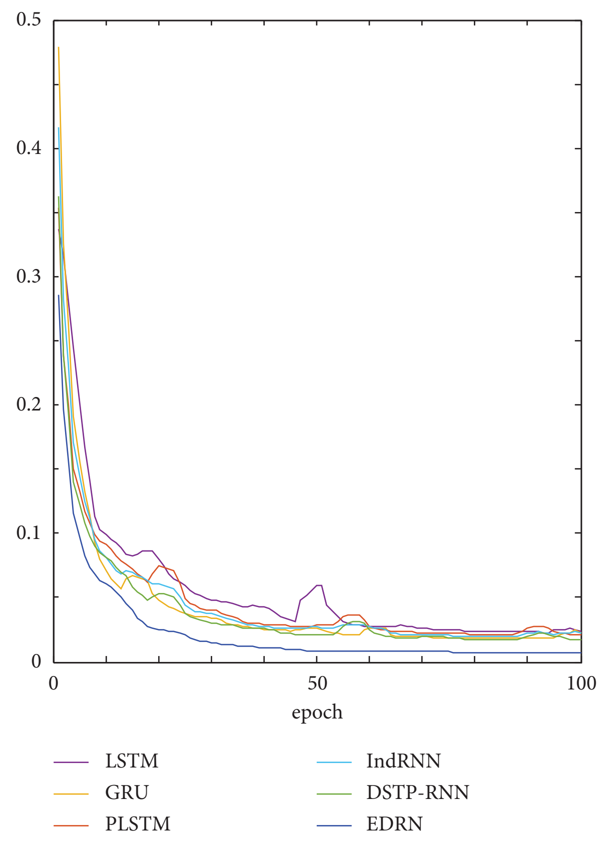
6. Discussion

The accuracy of the application-layer protocol identification method has a great influence on the performance of the proposed AL-DDoS attack detection method. We conducted an online test on the application-layer protocol identification method at the gateway of a real campus network, shown in Figure 9. The duration of the test experiment was five hours, and accuracy and recall were selected as evaluation indicators. Table 9 presents the identification results of some common application-layer protocols. The test results show that for common application-layer protocols, the accuracy and recall of the method were both above 0.998. Therefore, the application-layer protocol identification method can meet the needs of AL-DDoS attack detection.

To improve the accuracy of the EDRN-based protocol communication model, we update the model parameters online. Specifically, we collect training observation sequences of normal and AL-DDoS attacks online, and then train the model parameters at regular intervals, as shown in Figure 10.



(a)



(b)

FIGURE 7: Training results on D_3 dataset. (a) Training accuracy. (b) Training loss.

TABLE 3: Test results on D_1, D_2, and D_3 datasets.

Dataset	Model	Accuracy	$F1$	Recall	Loss
D_1	LSTM	0.979	0.968	0.973	**0.093**
	GRU	0.986	0.981	0.982	0.064
	PLSTM	0.982	0.977	0.976	0.073
	IndRNN	0.989	0.983	0.984	0.051
	DSTP-RNN	0.994	0.986	0.987	0.045
	EDRN	**0.997**	**0.992**	**0.994**	0.037
D_2	LSTM	0.967	0.960	0.962	**0.114**
	GRU	0.978	0.971	0.969	0.081
	PLSTM	0.975	0.968	0.964	0.092
	IndRNN	0.980	0.974	0.972	0.073
	DSTP-RNN	0.987	0.978	0.980	0.059
	EDRN	**0.993**	**0.990**	**0.991**	0.048
D_3	LSTM	0.972	0.964	0.967	**0.108**
	GRU	0.981	0.974	0.972	0.072
	PLSTM	0.979	0.971	0.969	0.083
	IndRNN	0.983	0.978	0.975	0.066
	DSTP-RNN	0.990	0.982	0.983	0.054
	EDRN	**0.995**	**0.991**	**0.992**	0.042

The bold values indicate the maximum values.

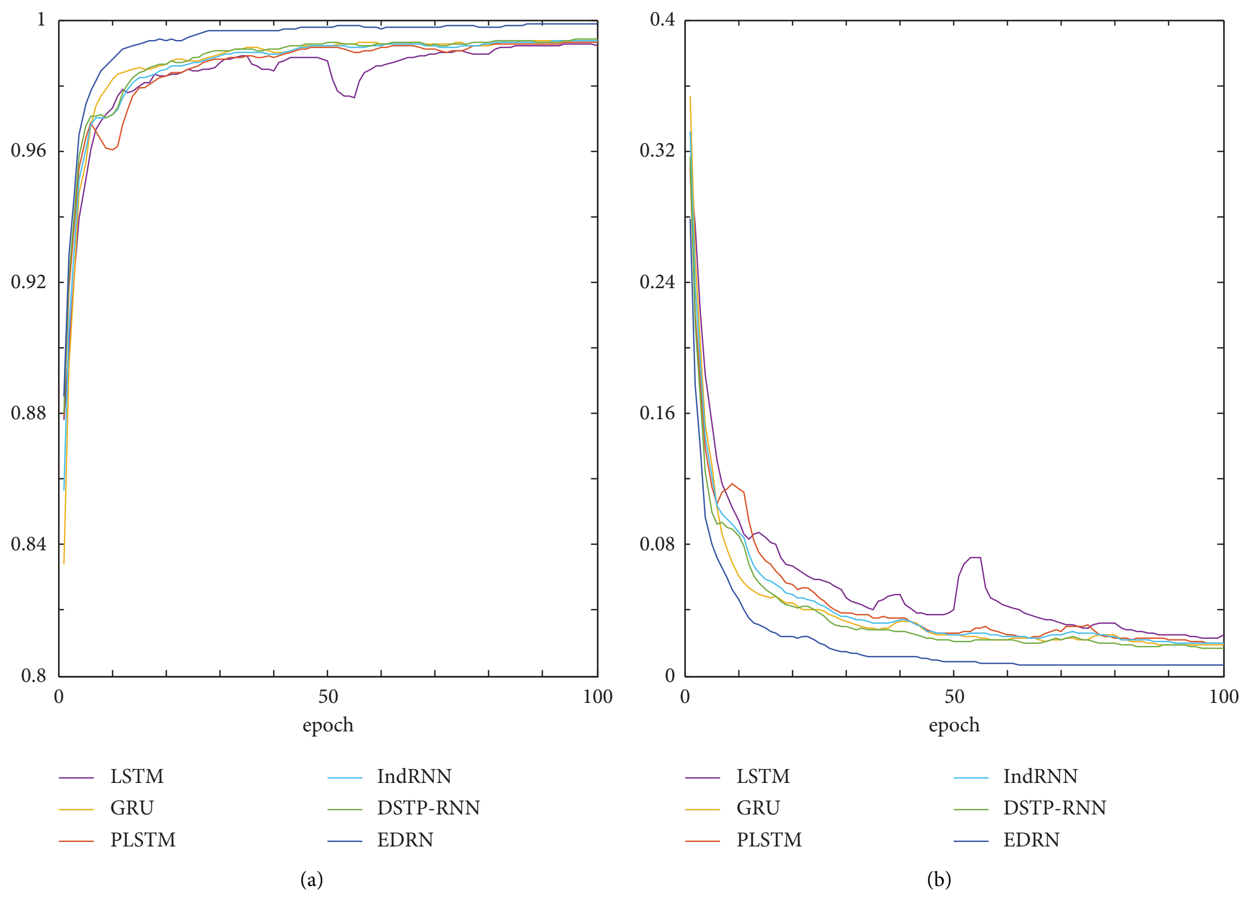


Figure 8: Training results on D_4 dataset. (a) Training accuracy. (b) Training loss.

Table 4: Test results on SMTP dataset.

Dataset	Model	Accuracy	*F*1	Recall	Loss
D_4	LSTM	0.970	0.962	0.965	**0.111**
	GRU	0.981	0.973	0.972	0.078
	PLSTM	0.977	0.969	0.968	0.089
	IndRNN	0.982	0.975	0.974	0.071
	DSTP-RNN	0.989	0.979	0.983	0.056
	EDRN	**0.994**	**0.991**	**0.990**	0.045

The bold values indicate the maximum values.

Table 5: Test results on CICDDoS2019 dataset.

Dataset	Model	Accuracy	*F*1	Recall	Loss
CICDDoS2019	LSTM	0.971	0.963	0.966	**0.109**
	GRU	0.983	0.976	0.972	0.078
	PLSTM	0.979	0.973	0.968	0.086
	IndRNN	0.985	0.979	0.976	0.067
	DSTP-RNN	0.991	0.982	0.983	0.055
	EDRN	**0.996**	**0.992**	**0.993**	0.041

The bold values indicate the maximum values.

Table 6: Comparison of results on HTTP datasets.

Dataset	Method	Accuracy	$F1$	Recall	Year
D_1	Wang et al. [23]	0.989	0.982	0.985	2018
	Singh et al. [25]	0.967	0.959	0.963	2018
	Zhao et al. [28]	0.954	0.943	0.947	2018
	Lin et al. [27]	0.986	0.978	0.982	2019
	Praseed and Thilagam [26]	0.991	0.984	0.988	2021
	Proposed method	**0.997**	**0.992**	**0.994**	**Present**
D_2	Wang et al. [23]	0.983	0.974	0.978	2018
	Singh et al. [25]	0.958	0.950	0.955	2018
	Zhao et al. [28]	0.945	0.936	0.939	2018
	Lin et al. [27]	0.976	0.965	0.971	2019
	Praseed and Thilagam [26]	0.984	0.977	0.979	2021
	Proposed method	**0.993**	**0.990**	**0.991**	**Present**
D_3	Wang et al. [23]	0.986	0.978	0.981	2018
	Singh et al. [25]	0.962	0.954	0.959	2018
	Zhao et al. [28]	0.949	0.938	0.944	2018
	Lin et al. [27]	0.982	0.973	0.977	2019
	Praseed and Thilagam [26]	0.988	0.980	0.983	2021
	Proposed method	**0.995**	**0.991**	**0.992**	**Present**

The bold values indicate the maximum values.

Table 7: Comparison of results on SMTP dataset.

Dataset	Method	Accuracy	$F1$	Recall	Year
D_4	Aziz and Okamura [33]	0.872	0.865	0.869	2017
	Gurusamyand Msk [34]	0.916	0.911	0.908	2019
	Feng et al. [18]	0.955	0.947	0.949	2020
	Schneider et al. [32]	0.937	0.932	0.926	2020
	Proposed method	**0.994**	**0.991**	**0.990**	**Present**

The bold values indicate the maximum values.

Table 8: Comparison of results on CICDDoS dataset.

Dataset	Method	Accuracy	$F1$	Recall	Year
CICDDoS2019	Wang et al. [23]	0.988	0.979	0.982	2018
	Singh et al. [25]	0.965	0.957	0.961	2018
	Zhao et al. [28]	0.952	0.941	0.946	2018
	Lin et al. [27]	0.985	0.976	0.979	2019
	Praseed and Thilagam [26]	0.990	0.983	0.986	2021
	Proposed method	**0.996**	**0.992**	**0.993**	**Present**

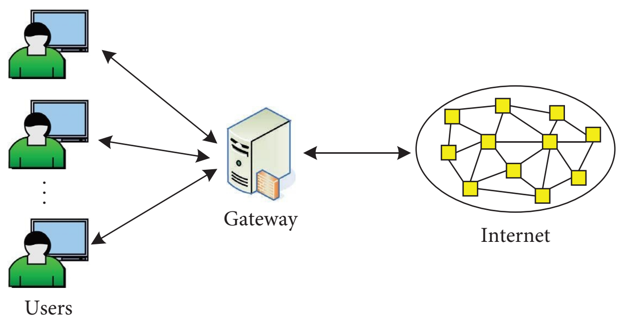


Figure 9: Experimental topology of a real campus network.

Table 9: Application-layer protocol identification results.

Application-layer protocols	Accuracy	Recall
HTTP	0.9993	0.9991
SMTP	0.9995	0.9992
Telnet	0.9996	0.9994
FTP	0.9997	0.9996
DNS	0.9994	0.9990
RTP	0.9992	0.9985

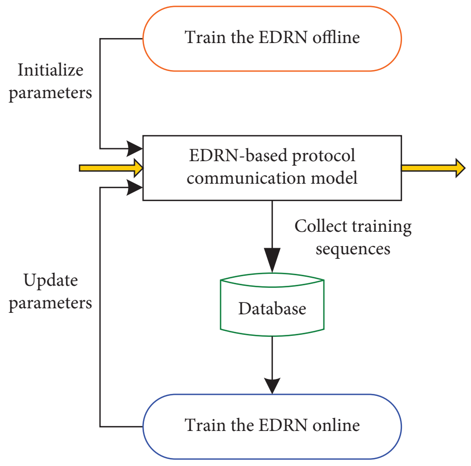


Figure 10: Model parameter online update.

7. Conclusion

This study investigated the issue of AL-DDoS attack detection. An application-layer protocol communication model is proposed based on the EDRN. The model takes as input application-layer protocol keywords and time intervals between adjacent protocol keywords. Based on the application-layer protocol communication model, a new method is proposed for AL-DDoS attacks detection. We evaluated the EDRN-based model and compared it with other RNNs. The experimental results show that the EDRN outperforms traditional RNNs, and our model can effectively detect multiple types of AL-DDoS attacks. For the datasets collected from a real campus network, our model achieves an overall accuracy of 0.995, *F*1 of 0.991, recall of 0.992, and loss of 0.043. For the CICDDoS2019 dataset, our model can effectively detect HTTP DDoS attacks, with an accuracy of 0.996, *F*1 of 0.992, recall of 0.993, and loss of 0.041. Our model can be used to detect AL-DDoS attacks in multiple networks, including the Internet of Vehicles, the Internet of Things, and software-defined networks.

However, it is difficult to define the protocol keywords of emerging application-layer protocols. Therefore, our model cannot effectively detect AL-DDoS attacks based on emerging application-layer protocols. In future work, we aim to automatically analyze emerging application-layer protocols and define their protocol keywords.

Data Availability

The CICDDoS2019 dataset used to support the findings of this study is a public dataset developed by the Canadian Institute for Network Security (CIC) in 2019. The CICDDoS2019 data can be downloaded from https://www.unb.ca/cic/datasets/ddos-2019.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Guangdong Basic and Applied Basic Research Foundation (grant no. 2018A0303130045) and the Science and Technology Program of Guangzhou (grant no. 201904010334).

References

[1] N. Tripathi and N. Hubballi, “Application layer denial-of-service attacks and defense mechanisms: a survey,” *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–33, 2021.
[2] S. Black and Y. Kim, “An overview on detection and prevention of application layer DDoS attacks,” in *Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0791–0800, IEEE, Singapore, January 2022.
[3] L. D. Tsobdjou, S. Pierre, and A. Quintero, “An online entropy-based DDoS flooding attack detection system with dynamic threshold,” *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1679–1689, 2022.
[4] M. Hajimaghsoodi and R. Jalili, “RAD: a statistical mechanism based on behavioral analysis for DDoS attack countermeasure,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2732–2745, 2022.
[5] M. Mittal, K. Kumar, and S. Behal, “Deep learning approaches for detecting DDoS attacks: a systematic review,” *Soft Computing*, vol. 8, pp. 1–37, 2022.
[6] L. Zhou, Y. Zhu, T. Zong, and Y. Xiang, “A feature selection-based method for DDoS attack flow classification,” *Future Generation Computer Systems*, vol. 132, pp. 67–79, 2022.
[7] G. A. Jaafar, S. M. Abdullah, and S. Ismail, “Review of recent detection methods for HTTP DDoS attack,” *Journal of Computer Networks and Communications*, vol. 2019, Article ID 1283472, 10 pages, 2019.
[8] K. Singh, P. Singh, and K. Kumar, “Application layer HTTP-GET flood DDoS attacks: research landscape and challenges,” *Computers and Security*, vol. 65, pp. 344–372, 2017.
[9] W. Z. Khan, M. K. Khan, F. T. Bin Muhaya, M. Y. Aalsalem, and H. C. Chao, “A comprehensive study of email spam botnet detection,” *IEEE Communications Surveys and Tutorials*, vol. 17, no. 4, pp. 2271–2295, 2015.
[10] G. Digitaltrends, “Just thwarted the largest HTTPS DDoS attack in history,” 2022, https://www.digitaltrends.com/computing/google-just-thwarted-the-largest-https-ddos-attack-in-history/.
[11] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A critical review of recurrent neural networks for sequence learning,” 2015, https://arxiv.org/abs/1506.00019.
[12] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[13] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition,” 2014, https://arxiv.org/abs/1402.1128.

[14] S. Z. Yu, "Explicit duration recurrent networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 7, pp. 3120–3130, 2022.

[15] I. Sharafaldin, A. H. Lashkari, and S. Hakak, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST)*, pp. 1–8, IEEE, Chennai, India, October 2019.

[16] M. M. Najafabadi, T. M. Khoshgoftaar, and C. Calvert, "A text mining approach for anomaly detection in application layer DDoS attacks," in *Proceedings of the 13th International Florida Artificial Intelligence Research Society Conference*, pp. 312–317, Jensen Beach, FL, USA, January 2017.

[17] A. Praseed and P. S. Thilagam, "DDoS attacks at the application layer: challenges and research perspectives for safeguarding web applications," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 661–685, 2019.

[18] Y. Feng, J. Li, and T. Nguyen, "Application-layer DDoS defense with reinforcement learning," in *Proceedings of the 2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, pp. 1–10, IEEE, Hangzhou, China, June 2020.

[19] N. M. Yungaicela-Naula, C. Vargas-Rosales, and J. A. Perez-Diaz, "SDN-based architecture for transport and application layer DDoS attack detection by using machine and deep learning," *IEEE Access*, vol. 9, pp. 108495–108512, 2021.

[20] H. Beitollahi, D. M. Sharif, and M. Fazeli, "Application layer DDoS attack detection using cuckoo search algorithm-trained radial basis function," *IEEE Access*, vol. 10, pp. 63844–63854, 2022.

[21] M. K. Kareem, O. D. Aborisade, S. A. Onashoga, T. Sutikno, and O. M. Olayiwola, "Efficient model for detecting application layer distributed denial of service attacks," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 1, pp. 441–450, 2023.

[22] Y. Xie and S. Z. Yu, "Monitoring the application-layer DDoS attacks for popular websites," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 15–25, 2008.

[23] C. Wang, T. T. N. Miu, X. Luo, and J. Wang, "SkyShield: a sketch-based defense system against application layer DDoS attacks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 3, pp. 559–573, 2018.

[24] W. Zhou, W. Jia, S. Wen, Y. Xiang, and W. Zhou, "Detection and defense of application-layer DDoS attacks in backbone web traffic," *Future Generation Computer Systems*, vol. 38, pp. 36–46, 2014.

[25] K. Singh, P. Singh, and K. Kumar, "User behavior analytics-based classification of application layer HTTP-GET flood attacks," *Journal of Network and Computer Applications*, vol. 112, pp. 97–114, 2018.

[26] A. Praseed and P. S. Thilagam, "Modelling behavioural dynamics for asymmetric application layer DDoS detection," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 617–626, 2021.

[27] H. Lin, S. Cao, J. Wu, Z. Cao, and F. Wang, "Identifying application-layer DDoS attacks based on request rhythm matrices," *IEEE Access*, vol. 7, pp. 164480–164491, 2019.

[28] Y. Zhao, W. Zhang, Y. Feng, and B. Yu, "A classification detection algorithm based on joint entropy vector against application-layer DDoS attack," *Security and Communication Networks*, vol. 2018, Article ID 9463653, 8 pages, 2018.

[29] A. Praseed and P. S. Thilagam, "HTTP request pattern based signatures for early application layer DDoS detection: a firewall agnostic approach," *Journal of Information Security and Applications*, vol. 65, Article ID 103090, 2022.

[30] T. Raja Sree and S. Mary Saira Bhanu, "Detection of HTTP flooding attacks in cloud using fuzzy bat clustering," *Neural Computing and Applications*, vol. 32, no. 13, pp. 9603–9619, 2020.

[31] A. D. Tudosi, D. G. Balan, and A. D. Potorac, "New Snort rule for detection and prevention of SMTP e-mail bomb attacks," in *Proceedings of the 2022 international conference on development and application systems (DAS)*, pp. 78–84, IEEE, Suceava, Romania, May 2022.

[32] M. Schneider, H. Shulman, and M. Waidner, "Blocking email bombs with email glass," in *Proceedings of the GLOBECOM 2020-2020 IEEE Global Communications Conference*, pp. 1–6, IEEE, Taipei, Taiwan, December 2020.

[33] M. Z. A. Aziz and K. Okamura, "Leveraging SDN for detection and mitigation SMTP flood attack through deep learning analysis techniques," *International Journal of Computer Science and Information Security*, vol. 17, no. 10, p. 166, 2017.

[34] U. M. Gurusamy and M. Msk, "Detection and mitigation of UDP flooding attack in a multicontroller software defined network using secure flow management model," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 20, Article ID e5326, 2019.

[35] Ö. Kasim, "A Robust DNS flood attack detection with a hybrid deeper learning model," *Computers and Electrical Engineering*, vol. 100, Article ID 107883, 2022.

[36] L. A. Trejo, V. Ferman, M. A. Medina-Pérez, F. M. Arredondo Giacinti, R. Monroy, and J. E. Ramirez-Marquez, "DNS-ADVP: a machine learning anomaly detection and visual platform to protect top-level domain name servers against DDoS attacks," *IEEE Access*, vol. 7, pp. 116358–116369, 2019.

[37] S. Datta, A. Kotha, K. Manohar, and U. Venkanna, "DNSguard: a raspberry pi-based DDoS mitigation on DNS server in IoT networks," *IEEE Networking Letters*, vol. 4, no. 4, pp. 212–216, 2022.

[38] J. Bushart and C. Rossow, "Anomaly-based filtering of application-layer DDoS against DNS authoritatives," in *Proceedings of the 8th IEEE European Symposium on Security and Privacy*, pp. 1–18, Taipei, Taiwan, February 2023.

[39] C. Xu, S. Chen, J. Su, S. M. Yiu, and L. C. K. Hui, "A survey on regular expression matching for deep packet inspection: applications, algorithms, and hardware platforms," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 4, pp. 2991–3029, 2016.

[40] K. Cho, B. Van Merriënboer, and C. Gulcehre, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, https://arxiv.org/abs/1406.1078.

[41] S. Li, W. Li, and C. Cook, "Independently recurrent neural network (indrnn): building a longer and deeper rnn," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5457–5466, Washington DC, USA, June 2018.

[42] Y. Liu, C. Gong, L. Yang, and Y. Chen, "DSTP-RNN: a dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction," *Expert Systems with Applications*, vol. 143, Article ID 113082, 2020.