Hindawi
Security and Communication Networks
Volume 2021, Article ID 9994127, 13 pages
https://doi.org/10.1155/2021/9994127

Research Article

Malicious URL Detection Based on Improved Multilayer Recurrent Convolutional Neural Network Model

Zuguo Chen, Yanglong Liu, Chaoyang Chen, Ming Lu, and Xuzhuo Zhang

School of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, Hunan, China

Correspondence should be addressed to Yanglong Liu; 1804060104@mail.hnust.edu.cn

Received 13 March 2021; Accepted 14 May 2021; Published 27 May 2021

Academic Editor: Entao Luo

The traditional malicious uniform resource locator (URL) detection method excessively relies on the matching rules formulated by the network security personnel, which is hard to fully express the text information of the URL. Thus, an improved multilayer recurrent convolutional neural network model based on the YOLO algorithm is proposed to detect malicious URL in this paper. First, single characters are mapped to dense vectors using word embedding, and the dense vectors are participated in the training process of the whole model according to the structural characteristics of the URL in the method. Then, the CSPDarknet neural network model based on the improved YOLO algorithm is proposed to extract features of the URL. Finally, the extracted features are used to evaluate malicious URL by the bidirectional LSTM recurrent neural network algorithm. In order to verify the validity of the algorithm, a total of 200,000 URLs are collected, including 100,000 normal URLs labeled "good" and 100,000 malicious URLs labeled "bad". The experimental results show that the method detects malicious URLs more quickly and effectively and has high accuracy, high recall rate, and high accuracy compared with Text-RCNN, BRNN, and other models.

1. Introduction

With the rapid development of Internet technology, network crime is becoming more and more serious, which brings heavy losses for personal network privacy and property security [1]. However, mixing well-known URLs with malicious URLs to cause user confusion and achieve intrusion attacks on the host is one of the most common attack methods. At present, malicious URLs are detected by using rule matching and the black-and-white list [2, 3]. But these methods are excessively dependent on the knowledge breadth of security personnel, which increases the possibility of false blocking of malicious URLs. Moreover, when these detection models are used to detect fishing URL of unknown attack types, there will be a great probability of false blocking or missed blocking.

To solve these problems, scholars at home and abroad have done a lot of research studies. Anwar et al. [2] and Li et al. [4] proposed a method combining linear and nonlinear spatial transformation for URL recognition and detection. The method significantly improves the accuracy of URL recognition and detection using a support vector machine and neural network. Vu et al. [5] proposed a new cost-sensitive classifier to detect malicious URLs in large enterprise networks. The method classifies the input URL into benign, unknown, and malicious and uses the cost matrix to select the most relevant features and to control the model misclassification. It can effectively reduce the false detection rate of the malicious URL. Yang et al. [6] proposed a URL feature representation method based on malicious keywords. The method uses a convolutional gated recurrent unit (GRU) neural network to replace the feature collection of the original pooling layer in the time dimension, which obtains a high-precision result. Yuan et al. [7] proposed a parallel neural joint model algorithm for analyzing and detecting malicious URL. The semantic features and text features are combined by merging parallel joint neural network and independent recurrent neural network in the algorithm, which can improve the detection accuracy of the fishing URL of unknown attack types. Yang et al. [8] proposed a multidimensional feature detection method for malicious URLs based on deep reinforcement learning. The method first extracts the sequence features of a given URL, classifies them quickly

through deep learning, and fuses the statistical features, the web page code features, and the web page text features into multidimensional features for detection, which can obtain higher detection accuracy of the malicious URL. Wang et al. [9] proposed a bidirectional LSTM algorithm based on convolutional neural network and independent recurrent neural network. The algorithm extracts the feature information of malicious URL binary file and uses the Word2Vec algorithm to train URL word vector characteristics and extract URL static vocabulary features, which can improve the detection accuracy of the malicious URLs. Chen et al. [10] proposed a multifeature information fusion-based identification algorithm in a complex environment, which achieves a better effect.

These methods are difficult to find the appropriate vector space to represent a single character in the process of URL numerical expression because of the randomness of the composition characters of URL strings, which results in low recognition accuracy of the malicious URL. A malicious URL detection model based on the combination of multilayer convolutional neural network and bidirectional recurrent neural network is proposed in the paper. First, the separated characters and the special characters are filtered by the model according to the structural characteristics of the URL. Then, the model unifies the lengths of all URL characters, intercepting the longer URL and filling the shorter URL with zero. By participating in the training of the neural network model through the word embedding layer, each character in the URL can be mapped into a dense vector in the embedding space, so each URL can be represented as a two-dimensional tensor. Next, the improved continuous multilayer convolutional neural network, which is based on the CSPDarknet neural network in the YOLO model, is used for feature extraction. The convolution layer in the network uses a one-dimensional convolutional neural network to extract the local context in the sequence. Finally, the results of feature extraction are input into the bidirectional recurrent neural network, and the network detects malicious URLs in positive and negative directions.

The main contributions of the paper are summarized as follows:

(1) CSPDarknet network model of the YOLO algorithm is improved by the one-dimensional convolutional neural network, which is used for feature extraction of URL sequence

(2) bidirectional recurrent neural network is used to process the URL sequence after feature extraction

(3) The translation invariance of the one-dimensional convolutional neural network is combined with the sequence sensitivity of RNN

The remainder of the paper is organized as follows. Section 2 introduces the character statistics and encoding of URL. Section 3 introduces how to combine convolutional neural networks and recurrent neural networks and proposes an improved multilayer convolutional recurrent neural network model based on the YOLO algorithm. Section 4 carries out simulation experiments and data analysis and verifies the advancement of this method through comparative experiments. Section 5 summarizes the conclusions.

2. URL Data Preprocessing

2.1. URL String Preprocessing. The research in this article mainly uses the Windows system, and it is unnecessary to match a case for any URL, so the uppercase letters in the URL can be transformed into lowercase. The smallest granularity, character, is selected as the smallest processing unit. Based on the statistics of the frequency of various characters in a large number of positive and negative datasets, this article deletes the low-frequency special characters to ensure that each URL provides the most useful information as much as possible while reducing the complexity of the URLs. This paper collects more than 400,000 URLs and counts the frequency of characters in each URL. The results are shown in Figure 1. The abscissa represents the index of the character, and the ordinate represents the frequency of character occurrences. It can be seen from Figure 1 that the occurrence frequency of characters after the 45th index is very low, so the characters after the 45th index can be deleted from the URL, and the influence on the feature information in the URL can be ignored. At this point, the length of each URL is mostly inconsistent, so the length of each URL needs to be standardized. This article counts the length of each URL in the dataset and finds that the average length is 48 characters. Therefore, all URLs in the dataset are uniformly processed into 48 characters in length. The long part is truncated, and the short part is filled with zero to ensure that each URL has the same length.

2.2. Character Encoding. URL can be regarded as a series of text sequences, but as most machine learning algorithms, the deep learning model cannot directly receive the original text sequence as input, and it can only process numerical tensor. Therefore, how to encode the information contained in the URL as a numerical expression is an important prerequisite for model recognition and detection. At present, the common coding methods are based on N-gram [11], one-hot [12], and word embedding methods [13]. N-gram is a word segmentation method that does not preserve text order. It is often processed in shallow natural language, while URL detection depends on text order. One-hot encoding usually maps the text sequence to a high-dimensional sparse tensor. This method cannot calculate the similarity between tensors, and it is easy to fall into high-dimensional disaster in the actual neural network training. However, there is a certain similarity among malicious URLs in many cases. In order to solve these problems, this paper uses the word embedding method, which can integrate more information into lower dimensions. This paper compares the one-hot word vector with the word embedding vector, as shown in Figure 2. In Figure 2, each row of squares represents a character vector, and each square with different colors represents different values. For instance, the one-hot word vector associates each character with a unique integer index i and converts this integer index into a binary character vector with length M.

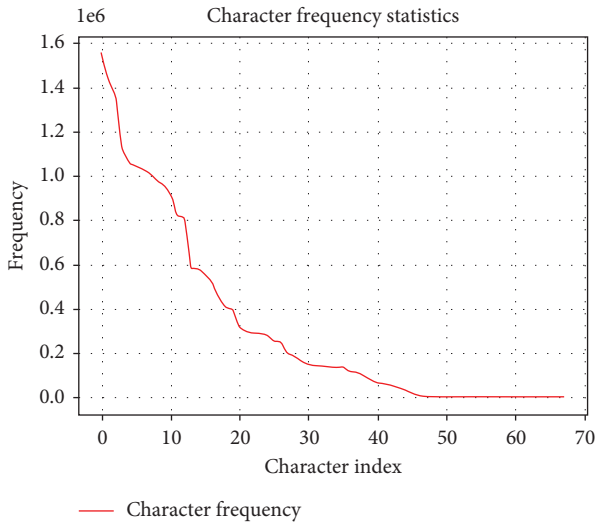

Figure 1: Character frequency statistics.

This character vector only has the value of the i-th index of 1, and the rest are 0. In the word embedding vector, each square represents different values. Therefore, the word embedding vector can fully express more information in lower dimensions than the one-hot word vector.

In order to obtain the embedding space model that maps characters to dense vectors, this paper uses the embedding layer to learn word embedding and to participate in the training process of the entire neural network model. In the process of model training, the weight parameters of the embedding layer are adjusted through the reverse transmission of the entire network. With the gradual convergence of the whole network model, the embedding space model of character mapping to vector will also tend to be stable, so that the obtained embedding space structure can facilitate the use of downstream neural network models.

3. Malicious URL Detection Model Based on Improved Multilayer Recurrent Convolutional Neural Network

3.1. Convolutional Neural Network. In recent years, the neural network has achieved a major breakthrough in the field of machine vision. One of the important reasons is the emergence of the convolutional neural network, which enables the neural network to perform convolution operation on images and extract information features from part of images [14]. At the same time, one-dimensional convolutional neural networks also perform extremely well in sequence processing, such as speech recognition and machine translation. This paper will realize malicious URL detection, which also depends on the character sequence. Therefore, this paper uses one-dimensional convolutional neural network to process the character vector of URL so as to achieve feature extraction. The working principle of one-dimensional convolutional neural networks is shown in Figure 3.

Figure 3 is a two-dimensional numerical tensor formed by character-level vectorization of a single URL. The window sliding on this tensor will extract the surrounding feature blocks at the location, and then each block does a tensor product with the same weight matrix (or called convolution kernel). Reusing multiple different convolution kernels will form multiple sets of vectors, and the one-dimensional convolution operation is completed by spatial reorganization of all vectors.

3.2. Bidirectional Recurrent Neural Network. A recurrent neutral network (RNN) has an additional information memory function in its hidden layer compared with the full connection layer. The input of the hidden layer at each time step includes not only the input of the current time step but also the output of the previous time step hidden layer [15]. This is conducive to information interaction between neural units in the same layer and realizes the memory function of past information. The method of RNN processing sequence is to traverse every element of all sequences and save a state, respectively. The output of the current event step is used as the state input of the next time step to form a neural network with an internal loop. The RNN-specific network structure is shown in Figure 4.

In Figure 4, X_t represents the input layer at the t-th time step, O_t represents the output value at the t-th time step, and S_t represents the state value at the t-th time step. U, W, and V represent weight matrices. The output value O_t is calculated in equation (1), and the state value S_t is calculated in equation (2).

$$O_t = g(VS_t), \tag{1}$$

$$S_t = f(UX_t + WS_{t-1}), \tag{2}$$

where g represents the activation function of neurons in the output layer and f represents the activation function of neurons in the hidden layer.

Theoretically, RNN can remember all the information that it traversed many time steps before. But practically, it is impossible to learn such long-term dependence because of the gradient disappearance problem. Therefore, this paper uses long short-term memory (LSTM) to build the neural network model. In essence, LSTM is a variant network of RNN. It adds a method to carry information across multiple time steps. To be specific, it allows information traversed by past time steps to reenter the network at future time steps, so as to solve the problem of gradient disappearance [16]. The unit structure of the LSTM neural network is shown in Figure 5, which consists of forgetting gate, LSTM unit state, input gate, and output gate.

In Figure 5, x_t represents the input vector at time t, h_t represents the hidden state at time t, and c_t represents the LSTM unit state at time t. The forgetting gate receives the hidden state h_{t-1} at the previous time and the input vector x_t at the current time and transmits them to the Sigmoid function. The range of the output value f_t is [0, 1]. If the output value is close to 0, it means information forgotten, and if it is close to 1, it means information retention. Therefore, the forgetting gate determines the abandonment

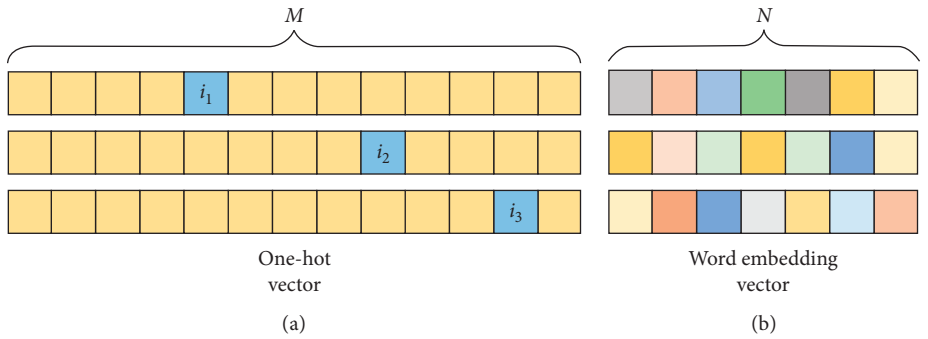

Figure 2: Comparison between one-hot word vector and word embedding vector.

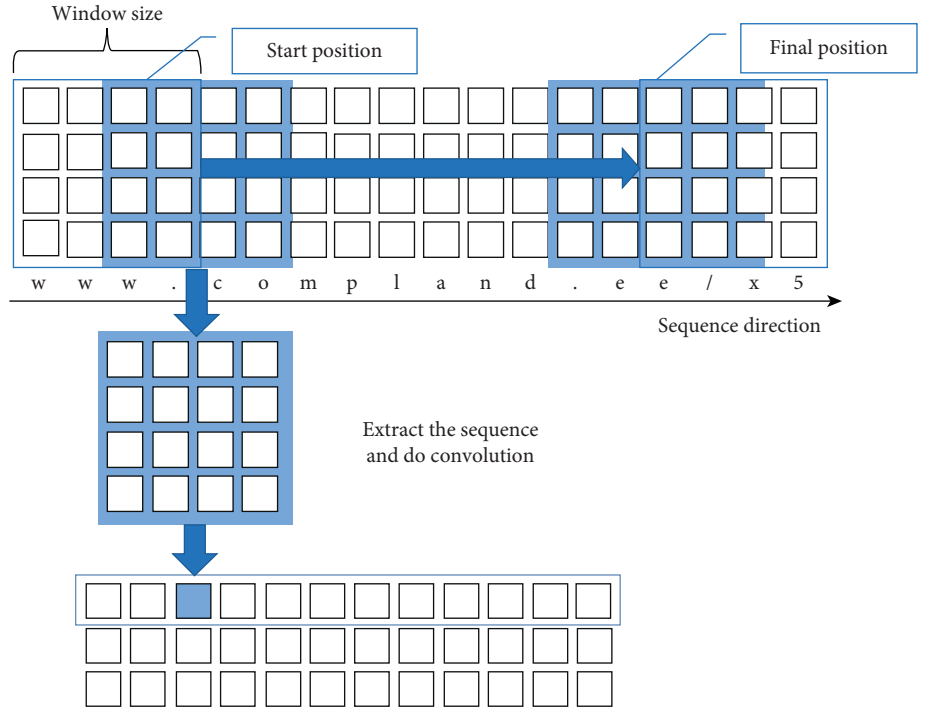

Figure 3: One-dimensional convolution operation principle diagram.

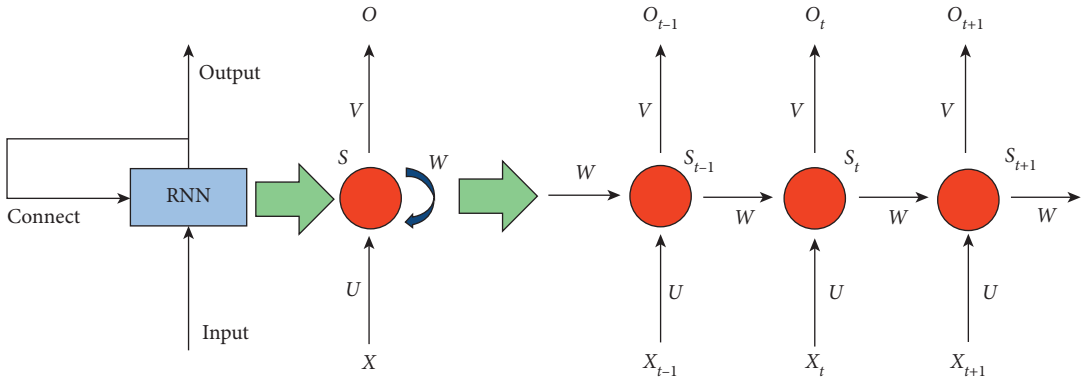

Figure 4: The structure of the RNN.

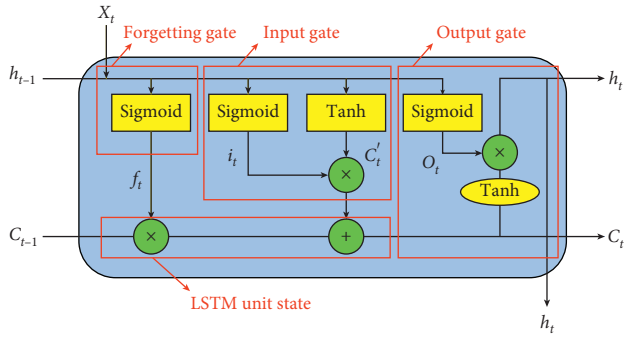

Figure 5: LSTM neural network unit.

and retention of information. The calculation process of the output value f_t of the forgetting gate is as follows:

$$f_t = \text{Sigmoid}\left(W_f \cdot [h_{t-1}, x_t] + b_f\right), \tag{3}$$

where W_f and b_f, respectively, represent the weight and bias in the forgetting gate.

The input gate receives the hidden state h_{t-1} at the previous moment and the input vector x_t at the current moment. They are transmitted to the Sigmoid function and the Tanh function simultaneously. The range of output value i_t of the Sigmoid function is [0, 1]. The closer the output value is to 0, the less important the information is, and the closer the output value is to 1, the more important the information is. The range of output value c_t' of the Tanh function is [–1, 1], which is used to output a new candidate vector. Then, the output values of the Sigmoid function and the Tanh function are multiplied, so that the output value of the Sigmoid function can determine which information is important in the candidate vector output by the Tanh function and can be saved. The calculation processes of the Sigmoid function output value i_t and the Tanh function output value c_t' are shown in equations (4) and (5), respectively:

$$i_t = \text{Sigmoid}\left(W_i \cdot [h_{t-1}, x_t] + b_i\right), \tag{4}$$

$$c_t' = \text{Tan}\,h\left(W_c \cdot [h_{t-1}, x_t] + b_c\right), \tag{5}$$

where W_i and b_i, respectively, represent the weight and bias of the Sigmoid function in the input gate and W_c and b_c represent the weight and bias of the Tanh function in the input gate, respectively.

The LSTM unit state receives the unit state c_{t-1} at the previous time, multiplies it with the output value f_t of the forgetting gate, and then adds the output value of the input gate to get the cell state c_t at the current time, so as to update the unit state in the LSTM neural network. The calculation formula of c_t is as follows:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot c_t'. \tag{6}$$

The output gate receives the hidden state h_{t-1} at the previous moment and the input vector x_t at the current moment and transfers them to the Sigmoid function. At the same time, the obtained LSTM unit state c_t is transferred to the Tanh function. Then, the output value of the Sigmoid function is multiplied by the output value of the Tanh function to obtain the hidden state h_t at the current moment. The calculation process of h_t is as follows:

$$h_t = \text{Sigmoid}\left(W_s \cdot [h_{t-1}, x_t] + b_s\right) \times \text{Tanh}\left(c_t\right), \tag{7}$$

where W_s and b_s, respectively, represent the weight and bias of the Sigmoid function in the output gate.

Malicious URL detection is strictly dependent on character order, and LSTM recurrent neural networks are processed in a single forward sequence. Therefore, this article, in order to further explore the relationship between the future state and the past state, adopts the bidirectional recurrent neural network, which is processed in a front-to-back and back-to-front direction, respectively [17]. Finally, the processing results of the two are combined to achieve more comprehensive data mining. The implementation structure of bidirectional recurrent neural network is shown in Figure 6, which is mainly composed of the input layer, hidden layer, and output layer.

In Figure 6, the first column represents the input layer of the bidirectional recurrent neural network, and the middle two columns represent the forward hidden state and the reverse hidden state, respectively. Their calculation formulas are shown as follows:

$$F_t = \phi\left(X_t W_{xh}^F + F_{t-1} W_{hh}^F + b_h^F\right), \tag{8}$$

$$B_t = \phi\left(X_t W_{xh}^B + B_{t+1} W_{hh}^B + b_h^B\right), \tag{9}$$

where ϕ represents the activation function of the hidden layer, X_t represents the input at time t, h represents the number of positive and negative hidden units, W_{xh}^F and W_{hh}^F represent positive weights, W_{xh}^B and W_{hh}^B represent negative weights, and b_h^F and b_h^B represent positive bias and negative bias, respectively. Then, the forward hidden state F_t is connected with the reverse hidden state B_t obtained above so as to obtain the hidden state H, and then the hidden state H is input to the output layer O_t. The calculation process is shown as follows:

$$O_t = H_t W_{hq} + b_q, \tag{10}$$

where H_t represents the hidden state at time t, q represents the number of output units, W_{hq} represents the weight from hidden unit to output, and b_q represents the output bias.

3.3. The Establishment of Malicious URL Detection Network Model. Inspired by the YOLO algorithm, this paper applies the CSPDarknet neural network model to extract the feature of character vectors. The YOLO algorithm is used for image target detection. It realizes high-precision and high-efficiency real-time detection. This article is based on the YOLOv4 algorithm to realize the malicious URL detection of the multilayer convolutional recurrent neural network model. The YOLOv4 algorithm is mainly composed of three network components: Backbone, Neck, and Head [18]. The Neck network component is mainly used to generate feature pyramids in image target detection and identify targets of

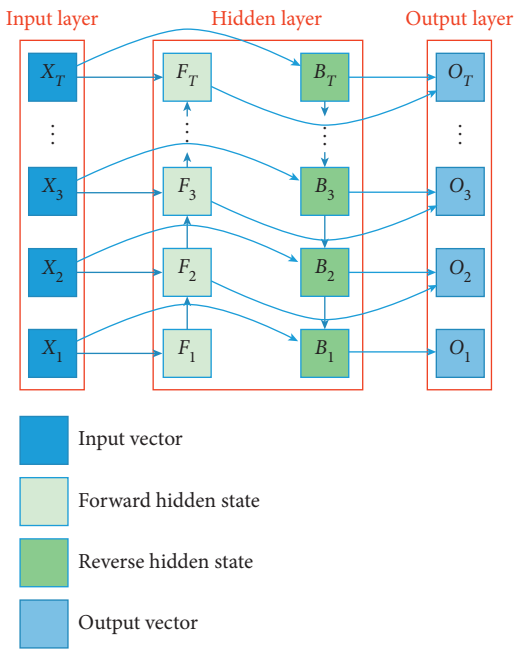

Figure 6: Structure diagram of bidirectional recurrent neural network.

different sizes by detecting zoom tensors of different scales. The Head network component is mainly used for the final anchoring of image target detection while generating target category probability, target score, and the position vector of the bounding box. CSPDarknet is mainly used to extract rich feature information from the image. It integrates the feature information into the feature map from top to bottom and gradually reduces the size. When using the CSPDarknet network to extract features of the URL numerical tensor, it can downsample the high-dimensional URL tensor to a low-dimensional space, which improves the detection speed of the model. Moreover, this paper uses a one-dimensional convolutional neural network to process sequence tensor in the CSPDarknet network. When the CSPDarknet network learns a certain local feature on the URL sequence, because a one-dimensional convolutional neural network has translation invariance, it can identify this local feature at any position on any URL. When dealing with each URL, this paper first uses word embedding to carry out numerical vectorization with characters as the smallest unit to obtain a two-dimensional tensor. Then, the CSPDarknet network can be improved and applied to the preprocessing step of recurrent neural network processing sequence front-end, so as to integrate the information after feature extraction of the convolutional neural network, and the bidirectional LSTM is used to identify malicious URLs. The whole network structure is shown in Figure 7.

In Figure 7, the model first receives the URL through word embedding processing for character-level vectorization to form a two-dimensional digital tensor. Each square in the tensor represents a character vector. The two dimensions of this two-dimensional tensor, respectively, represent the character length of the URL and the space vector of each character. However, when processing multiple URLs, a dimension is added to represent the number of URLs. Therefore, word embedding can convert the URLs into a three-dimensional tensor. Then, the CSPDarknet network framework, which is mainly composed of CBM, ResUnit, and CSPn, is used to extract the features of this three-dimensional tensor. The CBM component is composed of a one-dimensional convolutional neural network, batch normalization, and Mish activation function. The one-dimensional convolutional neural network can process the three-dimensional tensor obtained by the URL after word embedding. Moreover, the size of the convolution kernel of this one-dimensional convolutional neural network is 3, and the stride is 1. Batch standardization is to standardize not only the input layer but also the input (before activation function) of each intermediate layer. It is conducive to gradient propagation [19]. The Mish activation function is shown in equation (11), and its shape is similar to the ReLU activation function. However, the Mish activation function also allows relatively small negative gradient inflow in the case of negative values. The Mish activation function ensures that the positive value is unbounded and avoids the phenomenon of saturation.

$$\text{Mish}(x) = x \cdot \text{Tan}\, h\left(\ln\left(1 + e^{x}\right)\right). \tag{11}$$

Network component ResUnit indicates residual connection after two CBM operations. The residual connection solves the problem of gradient disappearance [20]. Its principle is to take the output of the previous layer as the input of the latter layer, so as to create a shortcut to let information directly enter the deep network, which effectively avoids the problems of gradient disappearance and gradient explosion. The calculation of CSPn components has two processing directions. The first processing direction is to

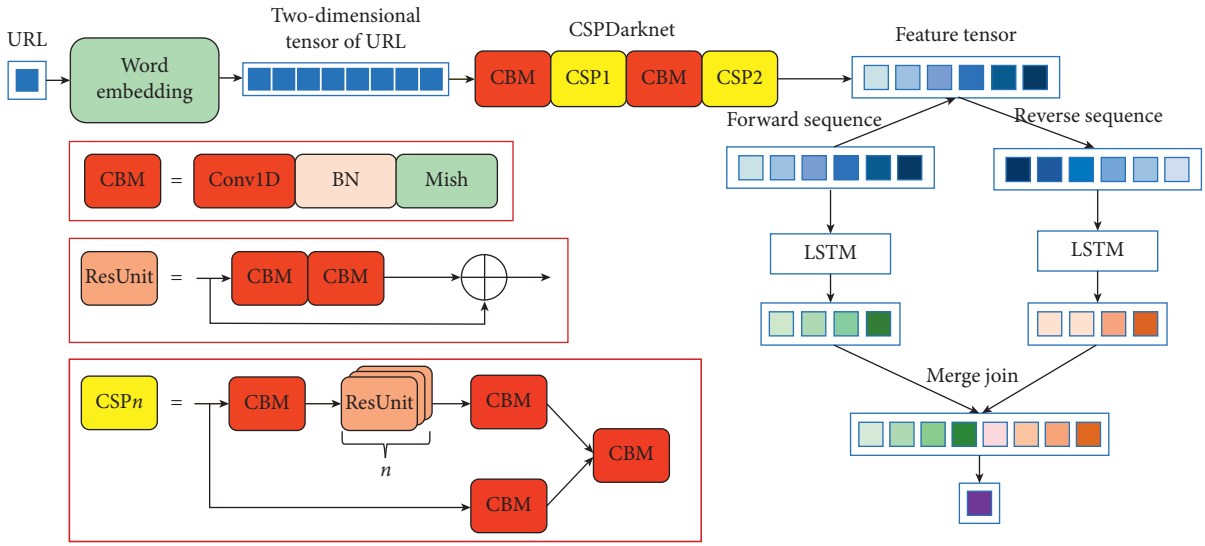

Figure 7: Malicious URL detection network model.

calculate the input value through the CBM network. The calculated results are connected by n times residuals, and the connected results are then calculated through the CBM network. The second processing direction is to calculate the input value through the CBM network. But the calculated results will be connected to the results of the first processing direction and output. After extracting feature information by the multilayer convolutional neural network, the obtained feature sequence tensor is processed by a bidirectional LSTM recurrent neural network, and the detection is performed in two directions: front to back and back to front. Then, the two tensors processed in two directions are spliced into a three-dimensional tensor, which is then expanded to a one-dimensional tensor. Finally, the output of the whole model is realized through the dense layer and the Sigmoid activation function. This paper stipulates that the closer the output of the model is to "1," the URL is labeled as "good", the closer the output of the model is to "0," the URL is labeled as "bad".

4. Experiment Results and Analysis

This article collected 200,000 URLs, including 100,000 normal URLs labeled as "good" and 100,000 malicious URLs labeled as "bad". This article randomly selected 90,000 normal URLs and 90,000 malicious URLs as training datasets, and the remaining 10,000 normal URLs and 10,000 malicious URLs are treated as test datasets. Twenty percentage of the training dataset is selected as the validation dataset in the process of model training.

In this paper, we use the embedding layer to learn character embedding and participate in the training process of the whole neural network model. Therefore, each character of URL can learn the unique spatial vector representation in the convergence process of the whole network model. After experiments, in order to analyze the character vector more clearly, the URL character vector is reduced to a three-dimensional space through the PAC algorithm, as shown in Figure 8. Each color in the figure represents a different character. Although the dimensionality is reduced, it can still be seen that the number charactered from "0" to "9" are more concentrated, the alphabetic characters from "a" to "z" are more concentrated, and the division between digital characters and alphabetic characters is more obvious. It can also be concluded from the actual URL that most of the feature information in the URL is represented by letters, and numbers are usually used to represent parameters. Therefore, it can be considered that the character vector obtained by model training has a good effect and provides a good foundation for the following feature extraction and malicious detection.

This article uses 25 URLs for training in small batches per round, with a maximum of 20 rounds. At the same time, in order to avoid the occurrence of overfitting, a callback function is added when training the model, and the checkpoint of the model is set and terminated early. If the target indicators monitored during the training process are no longer improved within the specified 20 rounds, the training can be terminated in advance, and the model weight can be saved. The accuracy and loss values of the model are shown in Figures 9(a) and 9(b), respectively. The red curve represents the accuracy and loss values of the training dataset, and the blue curve represents the accuracy and loss values of the validation dataset. It can be seen from the figures that, with the increase of training iterations, the training accuracy and loss values tend to converge, and the accuracy and loss values of the validation dataset are consistent with the training data. It indicates that the model can effectively identify malicious URLs.

In order to prove the advancement of the malicious URL identification method proposed in this paper, comparative experiments are carried out among the Darknet network model based on YOLOv3, the traditional bidirectional recurrent neural network, the traditional recurrent neural network, the RCNN neural network, and the neural network based on the full connection layer. Figures 10(a) and 10(b)

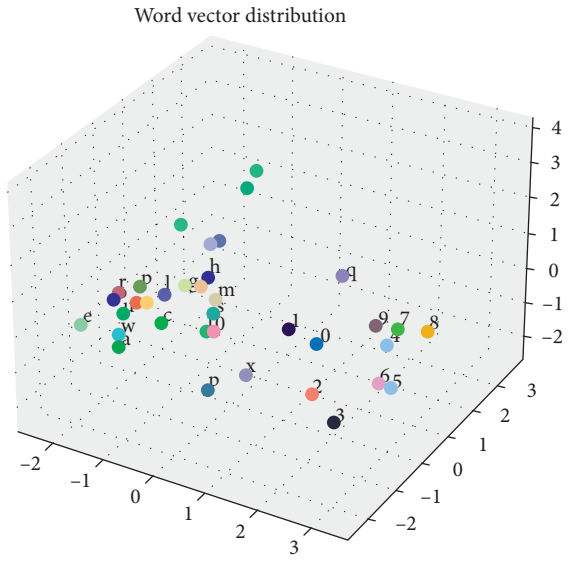

Figure 8: Three-dimensional mapping of character space vector.

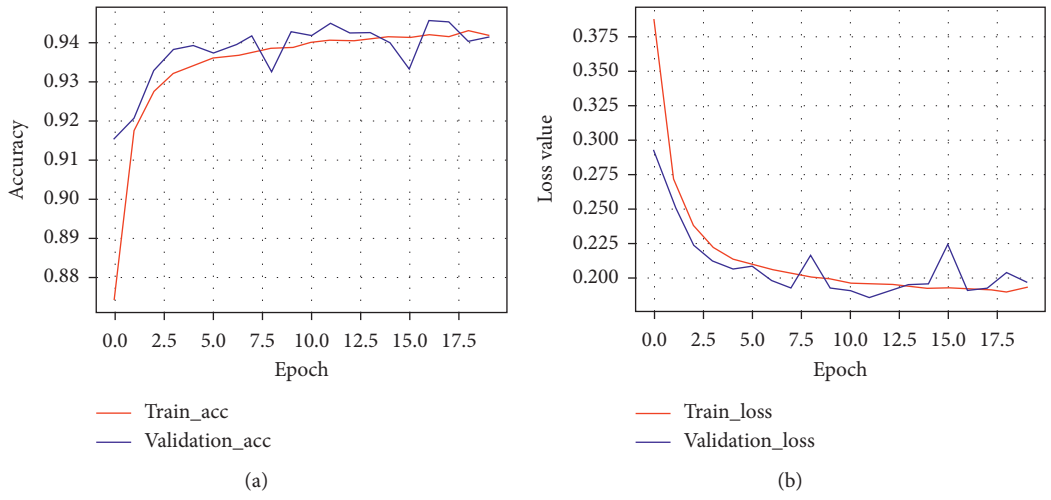

Figure 9: Training process of CSPDarknet model: (a) accuracy of model training and (b) loss values of model training.

are the training process of the Darknet network model based on YOLOv3. With the increase of the number of training iterations, the accuracy of the training dataset and the validation dataset are gradually increased. At the same time, the loss value of the training dataset and the validation dataset are gradually reduced. The final accuracy can be stabilized at about 94%, and the loss value can be stabilized at about 0.19. It can be seen that the training results of the Darknet network model based on YOLOv3 and the CSPDarknet network model based on YOLOv4 are similar.

Figures 11(a) and 11(b) are based on the traditional bidirectional recurrent neural network training process. It can be seen that the accuracy of the validation dataset is higher than that of the training dataset at the beginning stage, and the loss value of the validation dataset is lower than that of the training dataset. However, with the increase of iterations, the accuracy of the validation dataset is gradually lower than that of the training dataset, and the loss value of the validation dataset is gradually higher than that of the training dataset. These indicate that overfitting occurred in the later stage of the model training, and the generalization ability of the model is reduced.

Figures 12(a) and 12(b) are neural network model training process based on RCNN. It can be seen from the two figures that the whole model no longer improves in the ninth iteration of training, that is, it tends to be convergent and stable. The accuracy of the validation dataset is stable at about 92%, and the loss value is stable at about 0.22. It can be also concluded that the RCNN model begins to appear overfitting in the fourth iteration.

Figures 13(a) and 13(b) are neural network model training process based on RNN. It can be seen from these two figures that while the accuracy and loss values of the training dataset gradually converge, the accuracy and loss values of the validation dataset also gradually converge, but the model is unstable during the convergence process. It can be concluded from the convergence results that the accuracy

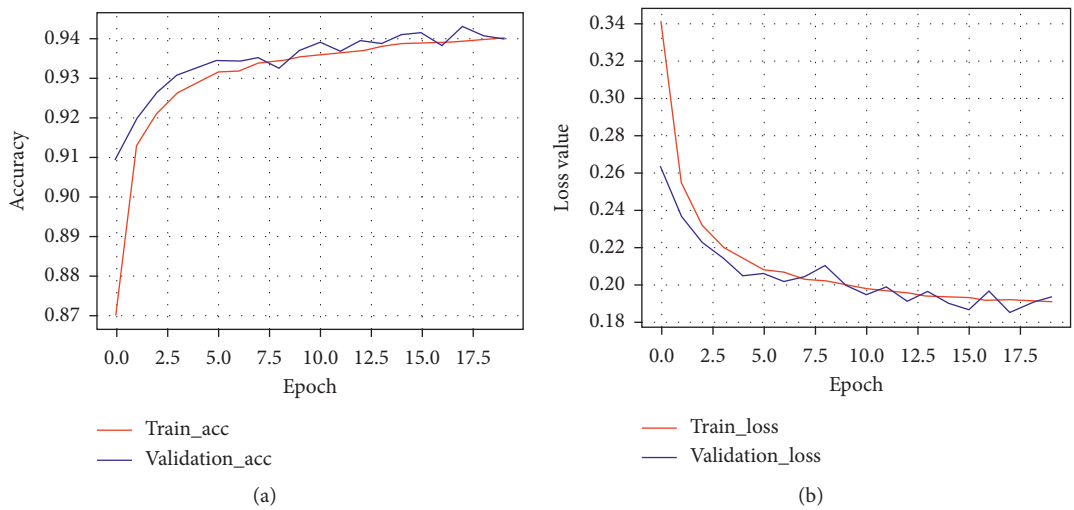

Figure 10: Training process of Darknet model: (a) accuracy of model training and (b) loss values of model training.

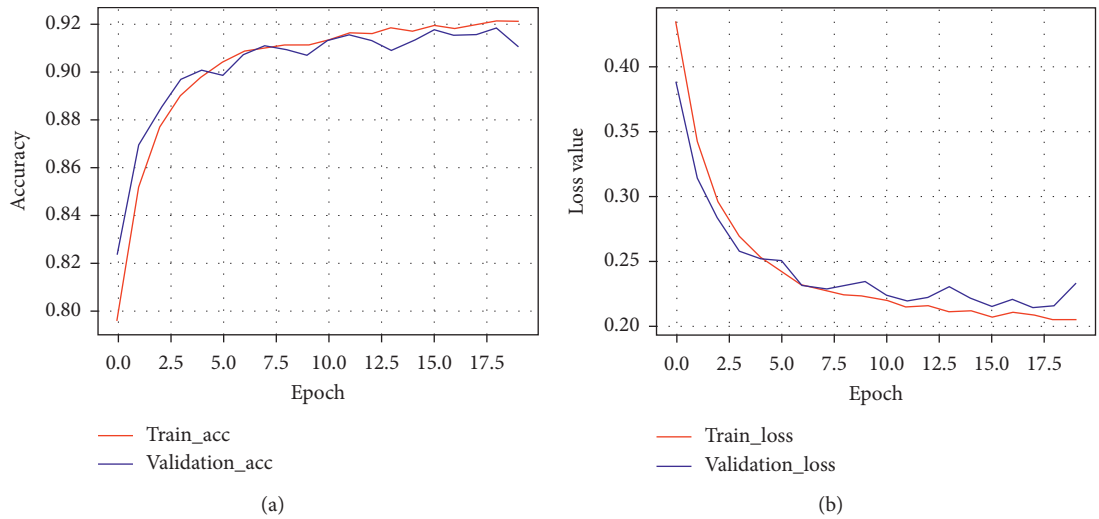

Figure 11: Training process of BRNN model: (a) accuracy of model training and (b) loss values of model training.

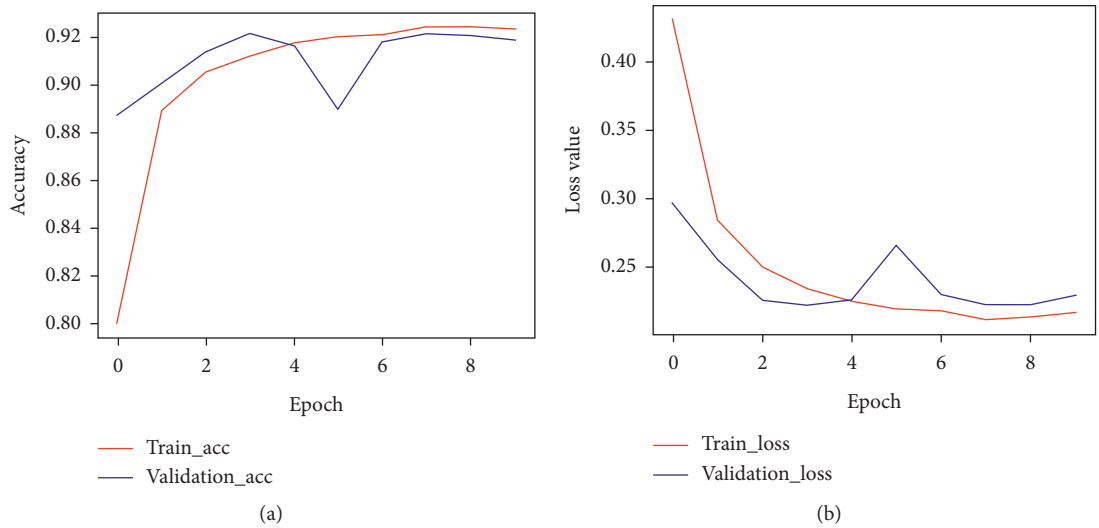

Figure 12: Training process of RCNN model: (a) accuracy of model training and (b) loss values of model training.

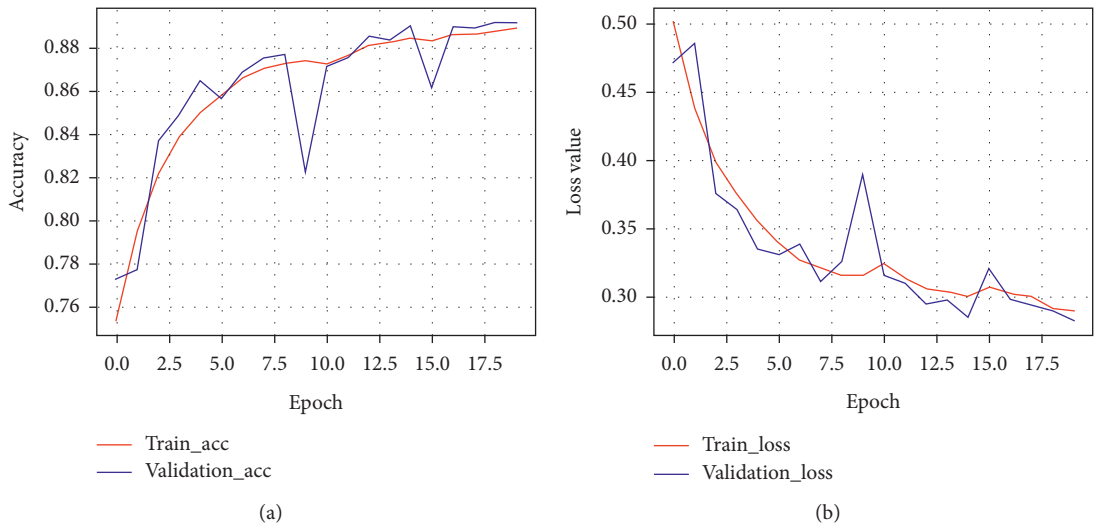

Figure 13: Training process of the RNN model: (a) accuracy of model training and (b) loss values of model training.

of the validation dataset is no more than 90%, and the minimum loss value is no less than 0.32.

Figures 14(a) and 14(b) are neural network models based on fully connected layers. It can be seen from the two figures that the training effect of the model no longer improves at the tenth iteration. In addition, the model is serious overfitting. The accuracy of the validation data set is about 86%, and the loss value is about 0.33. However, it is still at an unstable state. The effect of the model is extremely worse than the other five types of models.

It can be seen from the comparative analysis of Figures 9 and 14 that, in the whole training process, if we take the loss value reduced to 0.2 as the standard, the loss value of the two models, namely, the improved model based on CSPDarknet network in YOLOv4 and the improved model based on Darknet network in YOLOv3, will drop roughly the same. But if we take the accuracy increased to 94% as the standard, the improved model based on CSPDarknet has been completed in the 10th iteration, and the improved model based on Darknet is completed after the 17th iteration. Therefore, the improved model based on CSPDarknet has a faster convergence rate, and the accuracy is slightly higher than the improved model based on Darknet. The traditional bidirectional recurrent neural network model, the RCNN model, and the network model based on the full connection layer exist overfitting phenomena, and the severity of overfitting increases sequentially. Although there is no overfitting phenomenon based on the traditional recurrent neural network model, the recognition accuracy of the model is low, and there is a large fluctuation in the convergence process. If the six models are sorted according to the accuracy of the validation data set, then they are ranked from high to low as follows: the CSPDarknet network model based on YOLOv4, the Darknet network model based on YOLOv3, the traditional bidirectional recurrent neural network, the RCNN neural network, the traditional recurrent neural network, and the neural network based on the full connection layer.

The abovementioned trained models are used in the test dataset. The test dataset contains 10,000 normal URLs and 10,000 malicious URLs. The evaluation results are shown in Figure 15.

Figure 15 shows the performance of the six models in a dataset of 20,000 URLs. Obviously, the improved model based on CSPDarknet and the improved model based on Darknet have higher recognition accuracy and lower loss value. At the same time, since the activation function used in the output layer of the models is sigmoid, the difference in the loss value function is not large, and the accuracy of the improved model based on CSPDarknet is greater than that of the improved model based on Darknet. It is considered that the improved model based on CSPDarknet is slightly better than that based on Darknet. Finally, it can be seen that the accuracy and loss values of the other four models on the test dataset are worse than those of the improved multilayer convolutional recurrent neural network model proposed in this paper.

At the same time, if the RNN model is compared with the network model based on the full connection layer, we can conclude that the detection of malicious URL is dependent on the character sequence, and the relationship between the context characters in the URL can be found based on RNN, thus improving the accuracy of the model. If the RNN model is compared with the BRNN model, we can conclude that the bidirectional recurrent neural network can process the URL sequence in two directions: front to back and back to front. By combining the relationship between the future sequence and the past sequence at the current time step, the accuracy of the model is further improved. If the RCNN model is compared with the BRNN model, we can conclude that when the one-dimensional convolutional neural network is combined with the bidirectional recurrent neural network, the convolutional neural network can first extract the feature information in the URL and then hand it to the bidirectional recurrent neural network, which can effectively improve the recognition accuracy.

In order to better evaluate the superiority of each model in identifying malicious URLs, this paper selects precision,

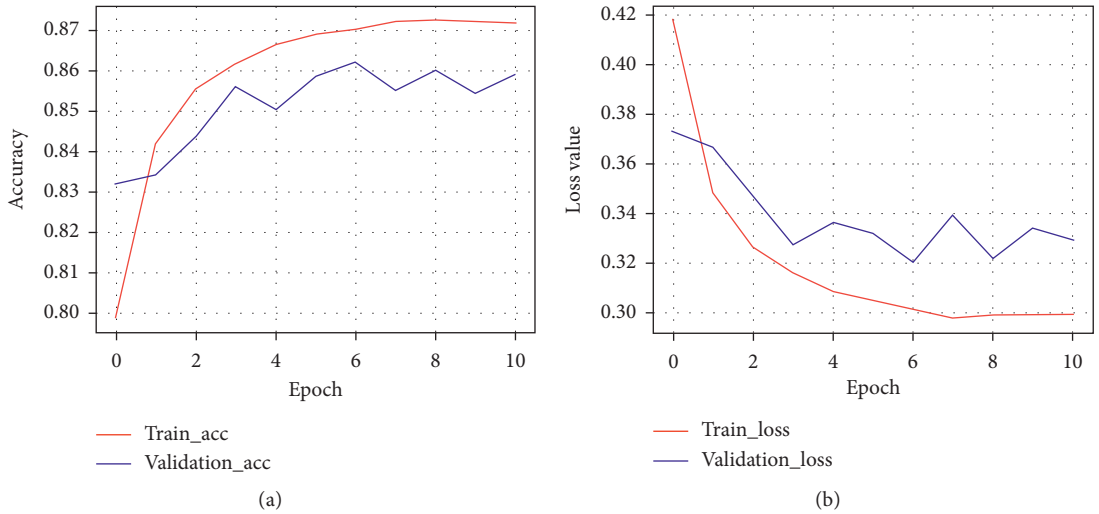

Figure 14: Training process of full connection layer network model: (a) accuracy of model training and (b) loss values of model training.

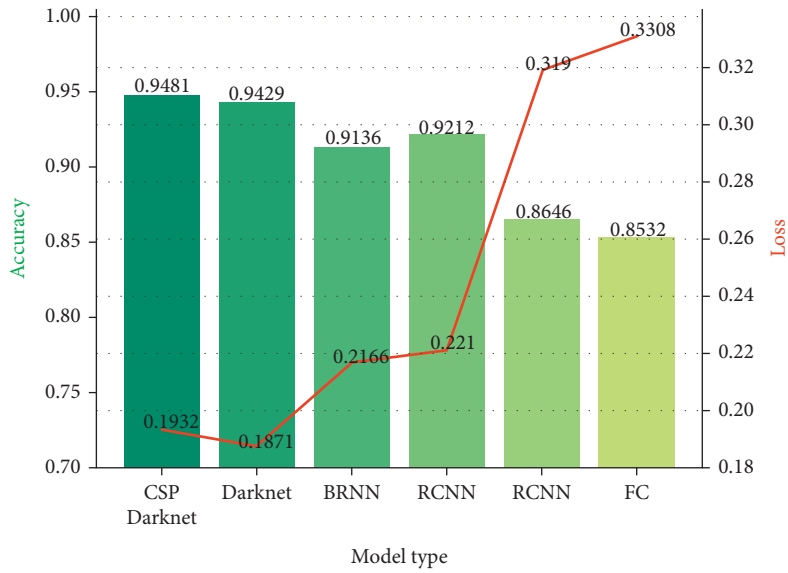

Figure 15: The accuracy and loss of the test set.

recall, *F*1-value, and AUC value as the evaluation parameters of the model. For the convenience of the following description, it is now assumed that the positive samples represent normal URLs and the negative samples represent malicious URLs. The precision represents the probability of actually being a positive sample in all predicted positive samples. The recall rate represents the probability of being predicted as a positive sample in the actual positive sample. The *F*1 value takes both precision and recall rate into account, allowing both to reach the maximum of the equilibrium. As shown in Figures 16(a) and 16(b), the precision-recall curve (P-R curve) and ROC curve based on the improved multilayer convolution recurrent neural network model are given, respectively.

As it is shown in Figure 16(a), the abscissa of the P-R curve represents the recall rate, and the ordinate represents the accuracy. With the increase of the recall rate of the model, more and more actual positive samples will be predicted as positive samples, while the model still has high accuracy. As it is shown in Figure 16(b), the abscissa of the ROC curve represents the proportion of false-positive samples to actual negative samples, and the ordinate represents the recall rate. When the proportion of false-positive samples predicted by the model decreases gradually, the model still has a high recall rate. It can be concluded that the improved multilayer convolutional recurrent neural network model proposed in this paper has the best classification effect and recognition ability. Finally, we, respectively, calculate the accuracy, recall rate, precision, *F*1-value, and AUC value of the six models under the test dataset. The results are shown in Figure 17.

It can be seen from Figure 17 that the improved multilayer convolutional recurrent neural network model is superior to the other five recognition models in accuracy,

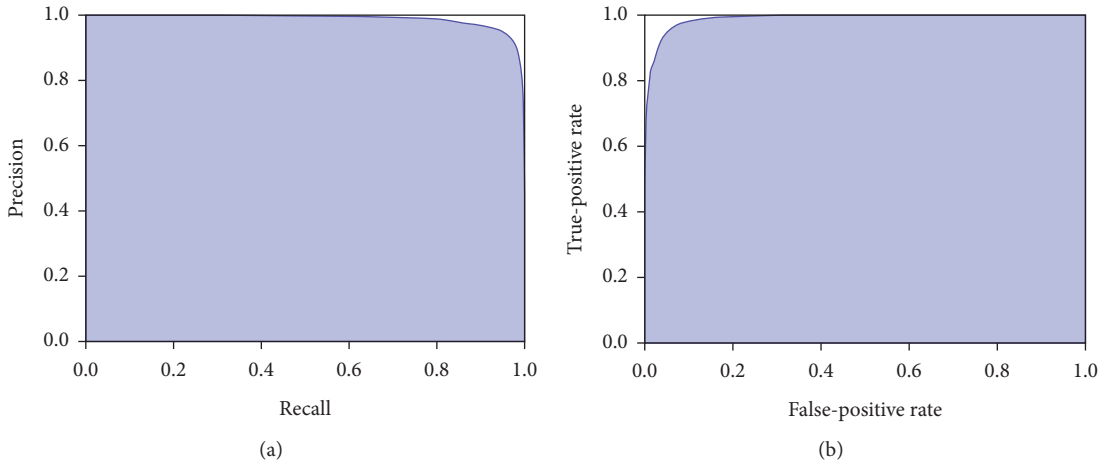

Figure 16: Model evaluation: (a) P-R curve and (b) ROC curve.

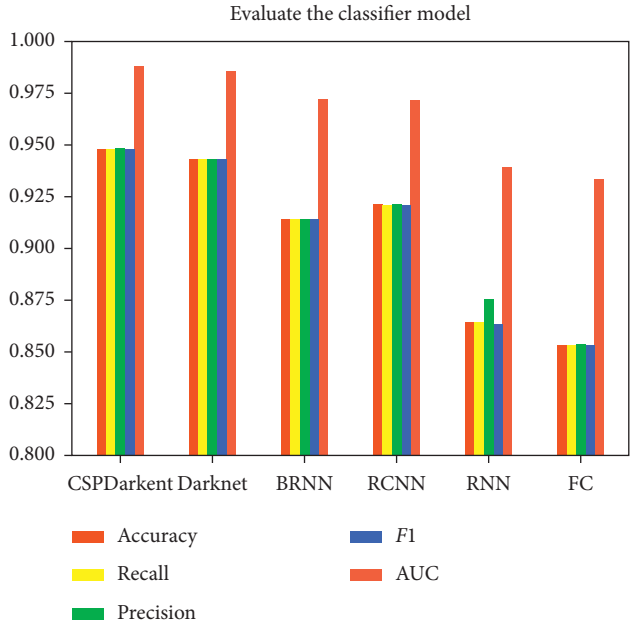

Figure 17: Model evaluation and comparison.

recall rate, precision, $F1$ value, and AUC value. Therefore, it can be concluded that the method proposed in this paper has higher recognition accuracy and better generalization ability than other existing malicious URL recognition models.

5. Conclusions

Malicious URL identification and detection are two of the important maintenance methods in maintaining network information security. The traditional malicious URL detection methods unduly rely on similarity matching rules, and the information of URL text is lost after numerical vectorization, which together makes it difficult to identify the context relationship of URL, and there are misjudgments and omissions. Therefore, this paper proposes an improved multilayer convolutional recurrent neural network model to detect malicious URLs. First, the model uses word embedding to participate in the training process of the entire neural network model, and the obtained word embedding space can vectorize the input URL text at the character level. Then, a single URL can be transformed into a two-dimensional tensor, and the feature extraction is carried out based on the multilayer convolutional neural network model improved by the YOLO algorithm. Finally, the extracted feature tensor is input into the bidirectional LSTM neural network for malicious URL recognition and detection, and the discriminant results are output. The experimental results show that when the embedding layer is used to participate in the training process of the entire model, the obtained embedding space has a relatively close relationship between the character vectors and at the same time has good representation ability. The CSPDarknet network improved based on the YOLO algorithm can effectively extract the features of the URL two-dimensional numerical tensor. At the same time, through the multilayer convolutional neural network, it can reduce the dimensionality of the URL numerical tensor and reduce the complexity of the model. The bidirectional LSTM recurrent neural network is used to process the convolution numerical tensor, which can effectively find the relationship between the contexts in the URL and further improve the detection accuracy of malicious URL. Through the evaluation of the model, it can be concluded that the improved multilayer convolution recurrent neural network model proposed in this paper can effectively improve the detection efficiency and recognition accuracy of malicious URLs by network security personnel and ensure the information security of network users. It also has a good prospect in the field of network security maintenance. However, since this paper adopts the truncated method to standardize the length of all URLs, it is inevitable to lose some information when facing a longer URL. Therefore, in the process of URL text vectorization, how to avoid missing information still has certain research value.

Data Availability

The data used to support the findings of this study are available at https://github.com/faizann24/Using-machine-learning-to-detect-malicious-URLs.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research has been partially supported by grants from the National Natural Science Foundation of China (Ref. 61903137). The work was also supported by grants from the National Key R&D Program of China for International S&T Cooperation Projects (2019YFE0118700) and the Natural Science Foundation of Hunan Province (Ref. 2020JJ5201).

References

[1] M. Sameen, K. Han, and S. O. Hwang, "PhishHaven-an efficient real-time AI phishing URLs detection system," *IEEE Access*, vol. 8, pp. 83425–83443, 2020.

[2] S. Anwar, F. Al-Obeidat, A. Tubaishat et al., "Countering malicious URLs in internet of things using a knowledge-based approach and a simulated expert," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4497–4504, 2020.

[3] X. Yan, Y. Xu, B. Cui, S. Zhang, T. Guo, and C. Li, "Learning URL embedding for malicious website detection," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6673–6681, 2020.

[4] T. Li, G. Kou, and Y. Peng, "Improving malicious URLs detection via feature engineering: linear and nonlinear space transformation methods," *Information Systems*, vol. 91, pp. 1–17, 2020.

[5] L. Vu, P. Nguyen, and D. Turaga, "Firstfilter: a cost-sensitive approach to malicious URL detection in large-scale enterprise networks," *IBM Journal of Research and Development*, vol. 60, no. 4, pp. 4:1–4:10, 2016.

[6] W. Yang, W. Zuo, and B. Cui, "Detecting malicious URLs via a keyword-based convolutional gated-recurrent-unit neural network," *IEEE Access*, vol. 7, pp. 29891–29900, 2019.

[7] J. Yuan, G. Chen, S. Tian, and X. Pei, "Malicious URL detection based on a parallel neural joint model," *IEEE Access*, vol. 9, pp. 9464–9472, 2021.

[8] P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning," *IEEE Access*, vol. 7, pp. 15196–15209, 2019.

[9] H.-H. Wang, L. Yu, S.-W. Tian, Y.-F. Peng, and X.-J. Pei, "Bidirectional LSTM malicious webpages detection algorithm based on convolutional neural network and independent recurrent neural network," *Applied Intelligence*, vol. 49, no. 8, pp. 3016–3026, 2019.

[10] Z. Chen, M. Lu, Y. Zhou, and C. Chen, "Information synergy entropy based multi-feature information fusion for the operating condition identification in aluminium electrolysis," *Information Sciences*, vol. 548, pp. 275–294, 2021.

[11] R. Liao, R. Zhang, J. Guan, and S. Zhou, "A new unsupervised binning approach for metagenomic sequences based on *N*-grams and automatic feature weighting," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 11, no. 1, pp. 42–54, 2014.

[12] F. Jafarzadehpour, A. Sabbagh Molahosseini, A. A. Emrani Zarandi, and L. Sousa, "Efficient modular adder designs based on thermometer and one-hot coding," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 9, pp. 2142–2155, 2019.

[13] Z. Li, F. Yang, and Y. Luo, "Context embedding based on Bi-LSTM in semi-supervised biomedical word sense disambiguation," *IEEE Access*, vol. 7, pp. 72928–72935, 2019.

[14] L. Lu, Y. Yi, F. Huang, K. Wang, and Q. Wang, "Integrating local CNN and global CNN for script identification in natural scene images," *IEEE Access*, vol. 7, pp. 52669–52679, 2019.

[15] S. Wang, R. Yao, T. A. Tsiftsis, N. I. Miridakis, and N. Qi, "Signal detection in uplink time-varying OFDM systems using RNN with bidirectional LSTM," *IEEE Wireless Communications Letters*, vol. 9, no. 11, pp. 1947–1951, 2020.

[16] J. Ma, H. Liu, C. Peng, and T. Qiu, "Unauthorized broadcasting identification: a deep LSTM recurrent learning approach," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 9, pp. 5981–5983, 2020.

[17] T. Dai, L. Zhu, Y. Wang, and K. M. Carley, "Attentive stacked denoising autoencoder with bi-LSTM for personalized context-aware citation recommendation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 553–568, 2020.

[18] S. Albahli, N. Nida, A. Irtaza, M. H. Yousaf, and M. T. Mahmood, "Melanoma lesion detection and segmentation using YOLOv4-DarkNet and active contour," *IEEE Access*, vol. 8, pp. 198403–198414, 2020.

[19] M. M. Kalayeh and M. Shah, "Training faster by separating modes of variation in batch-normalized models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 6, pp. 1483–1500, 2020.

[20] J. Naranjo-Alcazar, S. Perez-Castanos, I. Martín-Morató, P. Zuccarello, F. J. Ferri, and M. Cobos, "A comparative analysis of residual block alternatives for end-to-end audio classification," *IEEE Access*, vol. 8, pp. 188875–188882, 2020.