Hindawi
Scientific Programming
Volume 2022, Article ID 7777211, 11 pages
https://doi.org/10.1155/2022/7777211

Research Article

An Intrusion Detection Method Based on Fully Connected Recurrent Neural Network

Yuhong Wu[1] **and Xiangdong Hu**[2]

[1]*College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, China*
[2]*School of Automation, Chongqing University of Posts and Telecommunications, Chongqing, China*

Correspondence should be addressed to Xiangdong Hu; huxd@cqupt.edu.cn

Received 11 July 2022; Revised 23 August 2022; Accepted 1 September 2022; Published 26 September 2022

Academic Editor: Lianhui Li

Now, the use of deep learning technology to solve the problems of the low multiclassification task detection accuracy and complex feature engineering existing in traditional intrusion detection technology has become a research hotspot. In all kinds of deep learning, recurrent neural networks (RNN) are very important. The RNN processes 41 feature attributes and maps them to a 122-dimensional high-dimensional feature space. To detect multiclassification tasks, this study proposes an intrusion detection method based on fully connected recurrent neural networks and compares its performance with previous machine learning methods on benchmark datasets. The research results show that the intrusion detection system (IDS) model based on fully connected recurrent neural network is very suitable for classification of intrusion detection. Classification methods, especially in multiclassification tasks, have high detection accuracy, significantly improve the detection performance of detection attacks and DoS attacks, and it provides a new research direction for the future attempts of intrusion detection methods for industrial control systems.

1. Introduction

With the achievements of deep learning in image recognition, speech recognition, etc, it also provides a new method and idea for researchers in the field of intrusion detection to carry out related work.

Recurrent neural networks (RNN) were important. In 2007, Rachid et al. applied RNN and standard neural network to the field of intrusion detection and conducted experiments on the constructed small sample dataset. The experimental results showed that the detection and classification performance based on the RNN was relatively general on the small sample dataset, which was lower than the neural network under the same conditions. In 2010, Mansour et al., considering the complexity of the fully connected neural network structure, constructed a partially connected RNN, and the features between groups were constructed into a fully connected recurrent neural network, and there was no information between the features between groups [1–3]. Contact feedback to reduce model training time and achieve better detection results on the datasets you build. Although the locally connected recurrent neural network structure shortens the training time, the features were artificially classified, and the connections and roles between different groups of features were not considered.

After in-depth analysis, this study proposes an intrusion detection model based on fully connected recurrent neural network. Forty one feature attributes were processed and mapped into a 122-dimensional high-dimensional feature space, and features were no longer grouped and classified. Considering the relationship between features, this study investigates the detection ability of fully connected recurrent neural networks under multiclassification tasks.

2. Recurrent Neural Network (RNN)

Currently, recurrent neural networks (RNN) are mainly used to solve dynamic system problems involving time series of events. Structurally, a RNN includes a hidden layer, an input layer, and an output layer [4–6]. The current output is related to the previous output, and the nodes of each hidden layer are no longer disconnected. Therefore, the main work is achieved through the loop of the hidden layer itself. Essentially, an RNN is a unidirectional information flow from the input layer to the hidden layer, combined with a unidirectional information flow from the last sequential hidden layer to the current hidden layer. Figure 1 visually compares the differences between traditional neural networks and RNN.

Figure 2 shows the structure of the RNN after expansion. All states before time series *t* will be represented as outputs at time series *t*-1 and affect the time series *t*. Therefore, a RNN is a learning model with a dynamic deep structure. If the hidden unit is regarded as the storage space of the entire network, when the RNN is expanded according to the time series, it can be considered that the RNN has memorized all the information so far, which is a typical end-to-end learning method. Theoretically, a RNN can learn arbitrarily long sequence information and can remember end-to-end information, reflecting the "depth" of deep learning.

Obviously, in training, the training of RNN includes forward pass and backward pass. Similar to the traditional neural network training algorithm, the forward pass is output according to the time sequence, and the reverse pass is to pass the accumulated residuals of the previous period back through the RNN. During forward propagation, the hidden layer output (h_t) is

$$h_t = \sigma\left(W x_t + U h_{t-1} + b_h\right), \tag{1}$$

where σ is the activation function, x_t is the input vector of the time series t, h_t is the output of the hidden layer, W is the weight matrix, U is the self-circulating weight matrix, and b_h is hidden layer bias.

2.1. Intrusion Detection System (IDS) Based on Fully Connected Recurrent Neural Network. The overall framework of the intrusion detection model is shown in Figure 3, which mainly includes five steps [7–10].

Obviously, the training of the FCRNN-IDS has two aspects: forward propagation and weight fine-tuning. The forward propagation was responsible for the operation of the output data, and the fine-tuning of the weights was to update the weights by passing the accumulated residuals [11], which was no different from ordinary neural network training. The training was divided into two steps: first, the forward propagation algorithm was used to calculate its output value for each training sample of the input model. Then, using the weight fine-tuning algorithm, the entire model parameters were fine-tuned through backpropagation, and finally, a complete fully connected RNN classification model was obtained.

According to Figure 2, Algorithm 1 is a forward propagation algorithm, and Algorithm 2 is a weight update algorithm, respectively. Calculate the output $\hat{y}_i$ of each instance x_i using the forward propagation algorithm.

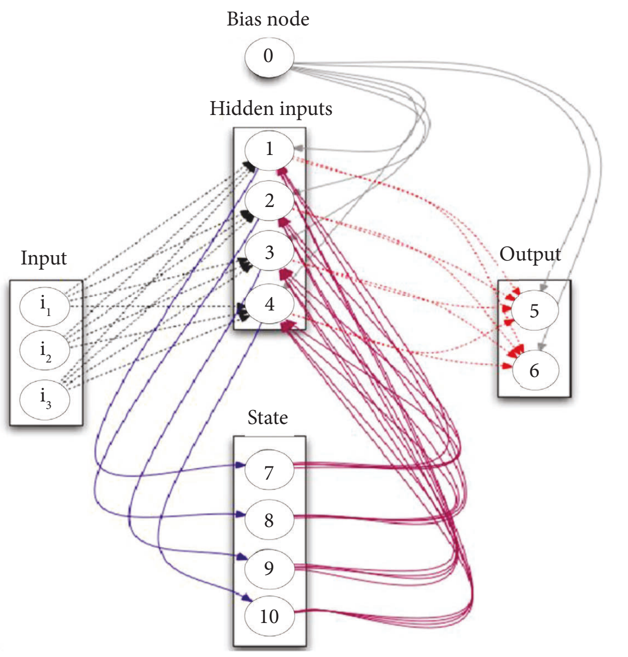


Figure 1: RNN structure.

3. Experiment

3.1. Data Sources. The dataset used in the experiment is a new benchmark dataset NSL-KDD [12–15]. This dataset is widely used.

3.2. Data Feature Extraction and Selection. Each connection record in the NSL-KDD contains 41 feature attributes [16–19]. Among them, 41 features can be divided into 4 categories:

(i) Basic features (9 types in total, numbered 1 to 9)

(ii) Content features (13 types in total, serial numbers 10 to 22)

(iii) Time-based network traffic statistics (9 types in total, serial numbers 23~31)

(iv) Host-based network traffic statistics (10 types in total, serial numbers 32~41)

3.3. Data Preprocessing. Using the NSL-KDD dataset, each connection record consists of 41 feature attributes, including 3 non-numeric feature attributes. Data preprocessing mainly includes two parts: numericalization of nonnumerical feature attributes. After one-hot encoding, the attribute feature 'protocol_type' corresponds to the binary feature vectors (1, 0, 0), (0, 1, 0), and (0, 0, 1). The other two nonnumerical attribute properties "service" and "flag" have 70 and 11 values, respectively. After such digital processing, the original 41-dimensional feature vector was converted into a 122-dimensional high-dimensional feature vector. On the one hand, one-hot encoding solves the problem of non-numerical data conversion, making the calculation of "distance" between features more reasonable.

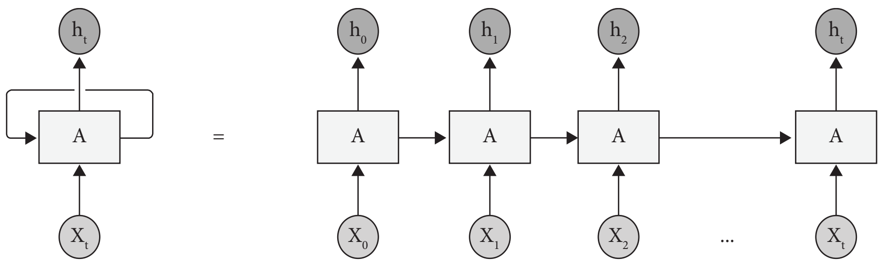


Figure 2: Unrolled RNN.

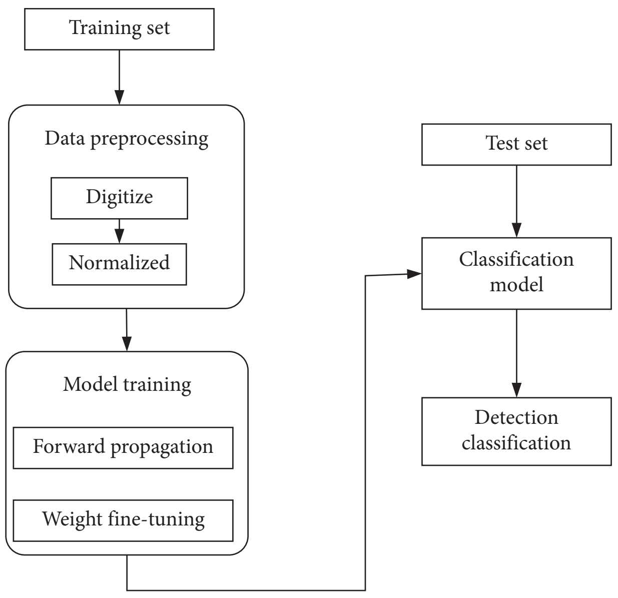


Figure 3: Framework of intrusion detection system based on fully connected recurrent neural network (IDS-FCRNN).

```
    Input: the training sample was $x_i$ ($i = l$, 2, m), the weight matrix was $W_{hx}$, $W_{hh}$, and $W_{yh}$, the bias was $b_h$ and $b_y$, the activation function $e$ uses the sigmoid function, and the classification function g uses the SoftMax function.
    Output: the output value $\hat{y}_i$ corresponding to the training sample $X_i$
(1) for xi from 1 to $m$ do
(2) $t_i = W_{hx}x_i + W_{hh}h_{i-1} + b_h$
(3)     $h_i = sigmoid(t_i)$
(4)     $s_i = W_{yh}h_i + b_y$
(5) $\hat{y}_i = SoftMax(s_i)$
(6) End for
```

Algorithm 1: Forward propagation algorithm.

```
    Input: the training sample was $(x_1, y_1)$ ($i = 1, 2, \ldots, m$).
    Initialization: the initialization model parameter was $\theta = \{W_{hx}, W_{hh}, W_{yh}, b_h, b_y\}$
    Output: the fine-tuned model parameter was $\theta = \{W_{hx}, W_{hh}, W_{yh}, b_h, b_y\}$
(1) For each sample $x_i$, input a fully connected RNN, the output $\hat{y}_i$ of $x_i$ was calculated by Algorithm 2.1
(2) Calculate the cross-entropy $L(y:\hat{y}_i)$ between the output value of each sample and the label value:
    $L(y\colon \hat{y}_i) \leftarrow -\sum_i \sum_i y_{ij} \log(\hat{y}_{ij}) + (1 - y_{ij}) \log(1 - \hat{y}_{ij})$
(3) For each network model parameter $\theta_i$ in $\theta$, calculate the partial derivative $\delta_i$: $\delta_i \leftarrow dL/d\theta_i$
(4) Make the error propagate back along the network and update each network model parameter $\theta_i$ in $\theta$:
    $\theta_i \leftarrow \theta_i + \eta\delta_i$
(5) If $t = k$, save the model parameters and the algorithm ends
(6) If $t < k$, then $t = t + 1$, turn to 1.
```

Algorithm 2: Weight fine-tuning algorithm.

The first step was that the data value space was too large. Feature attributes such as “duration[0, 58329],” “dst_bytes [0,1.3 × 109],” and “src_bytes[0,1.3 × 109],” which were correspondingly scaled by the logarithmic correction method as “duration[0, 4.77],” “dst_bytes[0, 9.11],” and “src_bytes[0, 9.11];” then, make each instance lie on the same order of magnitude on this feature. The second step was to normalize the data to the [0, 1] value range according to the following formula:

$$x_i = \frac{x_i - \mathrm{Min}}{\mathrm{Max} - \mathrm{Min}}, \tag{2}$$

where x_i was the attribute eigenvalue, Min was the minimum value, and Max was the maximum value.

3.4. Test Plan. This study adopts the comparative test method to detect the accuracy of FCRNN-IDS.

3.4.1. Comparative Experiment 1: Comparison with Traditional Machine Learning Methods. Using the same dataset NSL-KDD, the detection accuracy of seven traditional machine learning algorithms [20–33] such as decision tree, Naive Bayes, Naive Bayes tree, random tree, random forest, support vector machine, multilayer perceptron, and the detection accuracy of the FCRNN-IDS model in the case of 2-class (normal, abnormal) and 5-class (normal, probe, Dod, R2L and U2R) were studied and compared.

In [5], the authors investigated the anomaly detection performance of the above 7 classification algorithm models on the NSL-KDD benchmark dataset using Weka machine learning and data mining tools. Under the 2-class task, Figures 4 and 5 show the detection accuracy of KDDTest+ and KDDTest-21 by seven traditional machine learning methods, respectively. This study takes the research results of [5] as one of the comparative experiments; under the 2-class task, it was compared with the detection model based on fully connected recurrent neural network.

3.4.2. Comparative Experiment 2: Comparison with Recent Similar Literature. Wang and Cai [8] studied the performance of artificial intrusion detection systems under two and five types of tasks based on the same benchmark dataset NSL-KDD. The experimental results show that, under the dataset KDDTest+, the highest detection rate of the model was 81.2% under the 2-class classification; the highest detection rate of the model was 79.9% under the 5-class classification.

Deng et al. [9] proposed three-layer partially connected recurrent neural network architecture with 41 features as input and 4 intrusion categories and normal category as output. Taking the KDD99 dataset as the benchmark, some connection records were selected as the training set and the test set, respectively. The results show that the highest detection accuracy of the model was 94.1%. On the test set, the training time was set at 1383 seconds.

4. Results’ Analysis

4.1. Experimental Results of 2-Class Tasks. As mentioned earlier, the 41-dimensional feature vector was mapped to a 122-dimensional feature vector, so in the 2-class experiment. Figure 6 shows the detection accuracy of the FCRNN-IDS model on the training set with different structures and Learningrates.

As shown in Figure 7, it shows the detection accuracy of the FCRNN-IDS model on the test set KDDTest+ with different structures and Learningrates. As can be seen from Figure 7, when the Learningrate was 0.1 and the number of HiddenNodes was 80, the detection accuracy of the model on the test set KDDTest+ was 83.28%.

Figure 8 shows the detection accuracy of the FCRNN-IDS model on KDDTest-21 with different structures and Learningrates. As can be seen from Figure 8, when the Learningrate was 0.1 and the hidden node was 80, the model has the highest detection accuracy on KDDTest-21, which was 68.55%.

The experimental results were as follows.

As shown in Table 1, the number of HiddenNodes was 80 and the Learningrate was 0.1, which obtains high detection accuracy. Figure 9 details the variation in detection accuracy of the FCRNN-IDS model iteratively trained on the KDDTrain+, KDDTest+, and KDDTest-21.

Ashfaq e al. [5] studied the detection accuracy of classification algorithms such as J48, Multilayer Perception, Naive Bayes, Support Vector Machine, and Random Forest. The results are shown in Figures 4 and 5; the artificial neural network algorithm has the highest detection accuracy on the test set KDDTest+ in the 2-class task, reaching 81.2%, which was the latest literature on the application of related algorithms. The above model experimental results were all based on the dataset NSL-KDD, so they had similar comparison conditions.

As shown in Figure 10, the three algorithm models showed good classification and detection performance, especially the Naive Bayesian tree on the test sets, KDDTest+ and KDDTest-21. Better classification and detection performance: the detection accuracy of this method was high, 82.02% and 66.16%, respectively.

Compared with the detection method based on artificial neural network proposed in [8], FCRNN-IDS has the highest detection accuracy under the 2-class task, which was 81.2%, and the detection accuracy under the 2-class task was also higher. Table 2 shows the confusion matrix of the ANN-based detection model on the test set KDDTest + when performing the 2-class task. Table 3 presents the confusion matrix of FCRNN-IDS on KDDTest + under the 2-class task.

Therefore, when performing 2-class tasks, FCRNN-IDS further improved the detection ability of attack behavior, improved the accuracy, and reduced the false positive rate.

4.2. Experimental Results of Multiclassification Tasks. Figure 11 shows the detection accuracy of FCRNN-IDS on the training set with different structures and

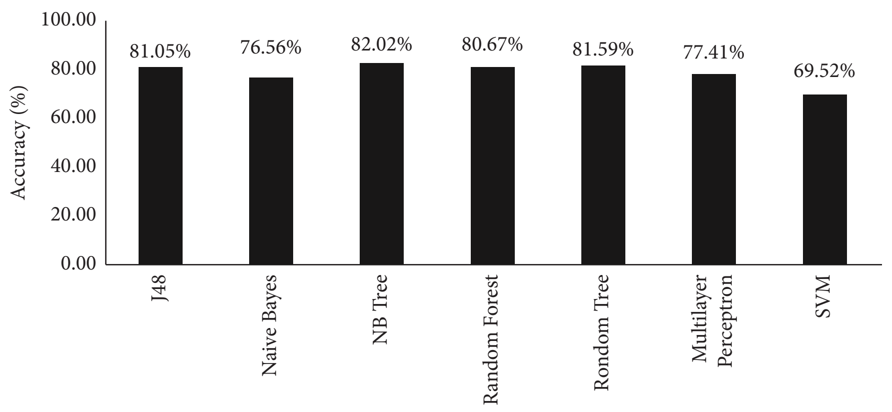


Figure 4: Detection accuracy of traditional machine learning methods in the test set KDDTest+ (2-class).

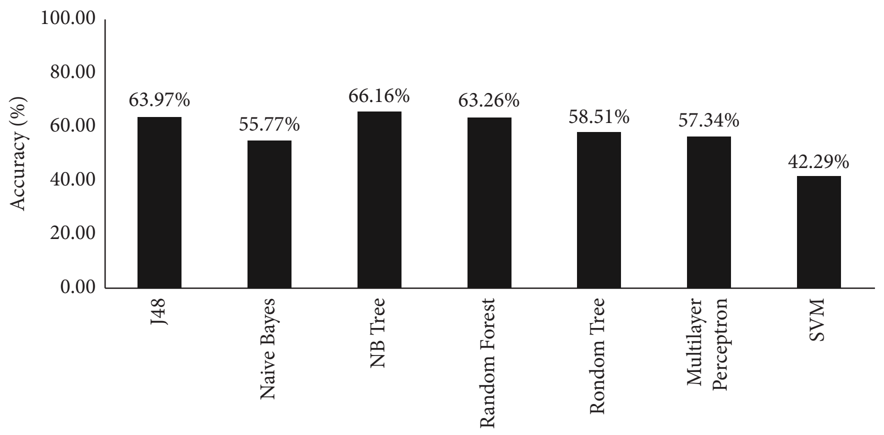


Figure 5: Detection accuracy of traditional machine learning methods on the test set KDDTest-21 (2-class).

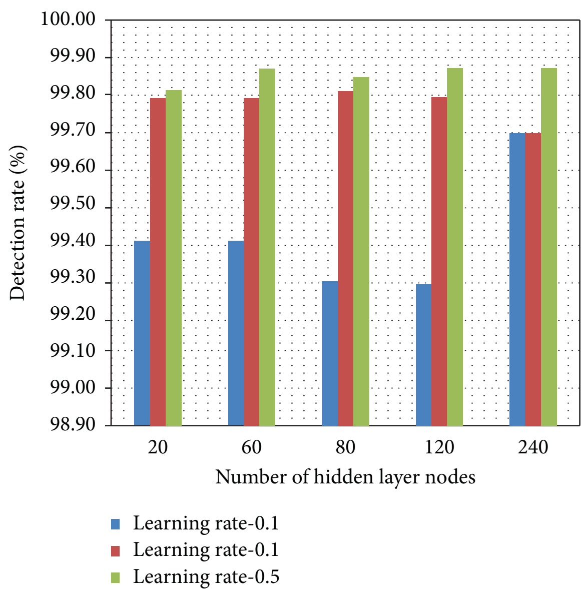


Figure 6: The accuracy of the model on the training set under different structures and Learningrates (2-class).

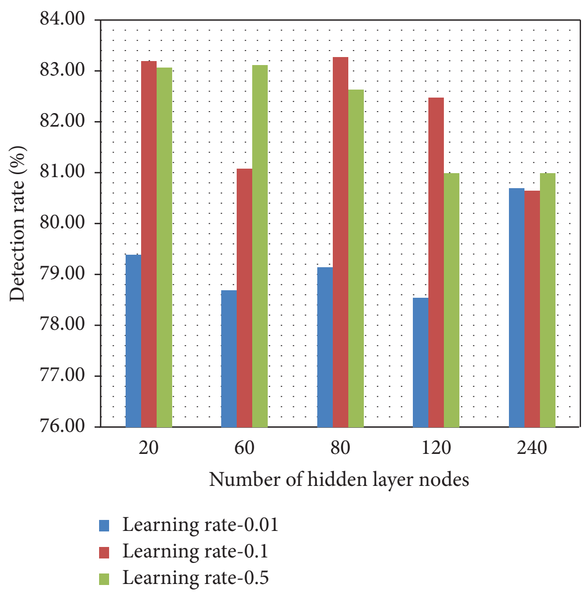


Figure 7: Detection rate of the model on KDDTest+ with different structures and Learningrates (2-class).

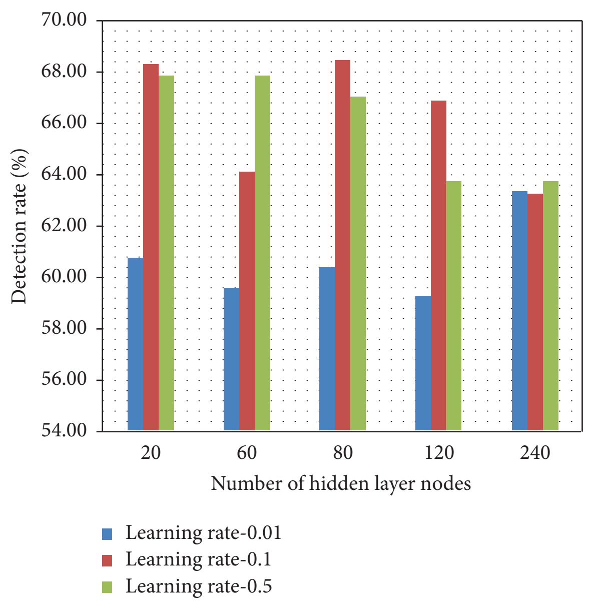


Figure 8: Detection rate of the model on KDDTest-21 under different structures and Learningrates (2-class).

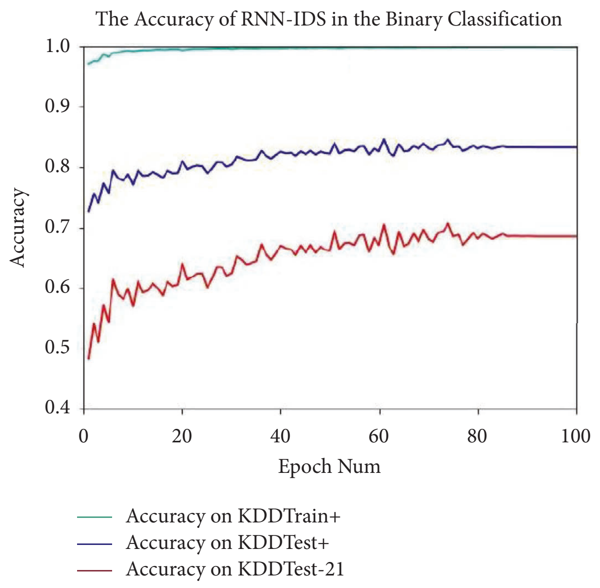


Figure 9: Detection performance of the model on the dataset (2-class).

Table 1: Model detection accuracy (2-class) under different structures and Learningrates.

	KDDTrain$^+$	KDDTest$^+$	KDDTest^{-21} (%)
HiddenNodes = 20, Learningrate = 0.01	99.40%	79.37%	60.76
HiddenNodes = 20, Learningrate = 0.1 rate = 0.1	99.79%	83.18%	68.23
HiddenNodes = 20, Learningrate = 0.5	99.81%	83.09%	67.84
HiddenNodes = 60, Learningrate = 0.01	99.39%	78.72%	59.54
HiddenNodes = 60, Learningrate = 0.1	99.79%	81.06%	64.08
HiddenNodes = 60, Learningrate = 0.5	99.87%	83.11%	67.82
HiddenNodes = 80, Learningrate = 0.01	99.29%	79.16%	60.34
HiddenNodes = 80, Learningrate = 0.1	99.81%	83.28%	68.55
HiddenNodes = 80, Learningrate = 0.5	99.85%	82.66%	66.99
HiddenNodes = 120, Learningrate = 0.01	99.28%	78.55%	59.25
HiddenNodes = 120, Learningrate = 0.1	99.79%	82.48%	66.83
HiddenNodes = 120, Learningrate = 0.5	99.87%	80.97%	63.69
HiddenNodes = 240, Learningrate = 0.01	99.69%	80.69%	63.28
HiddenNodes = 240, Learningrate = 0.1	99.69%	80.67%	63.28
HiddenNodes = 240, Learningrate = 0.5	99.87%	80.97%	63.69

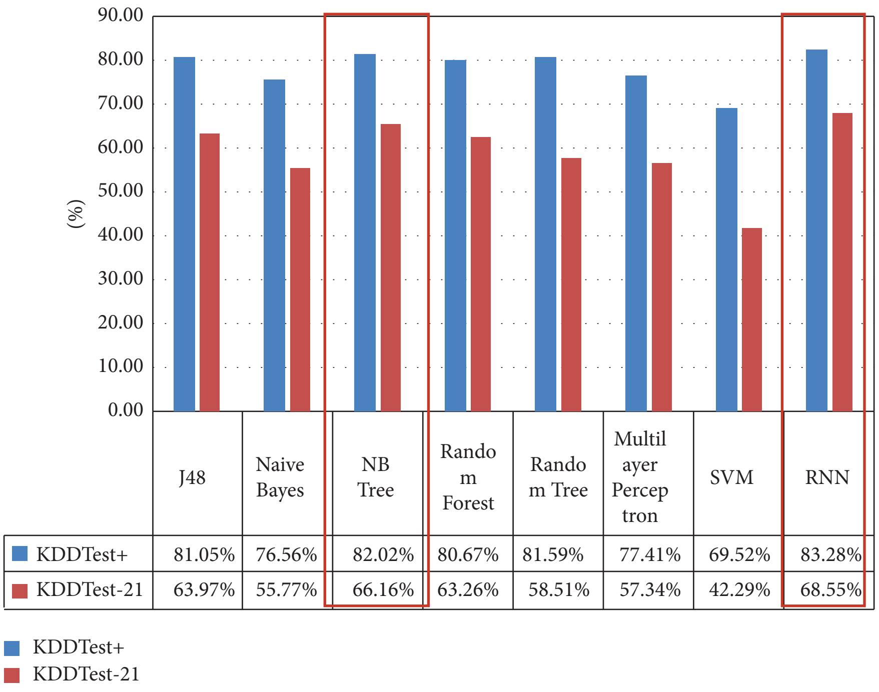
Learningrates. As can be seen from the figure, when the Learningrate was 0.5 and the hidden node was 60, the model has the highest detection accuracy on the training set, which was 99.87%.

As shown in Figure 12, from the detection accuracy of FCRNN-IDS on the test set KDDTest + under different structures and Learningrates, it can be seen that the Learningrate was 0.5 and the hidden node was 80; the model has the highest detection accuracy in the test set KDDTest+, which was 81.29%.

As shown in Figure 13, from the detection accuracy of FCRNN-IDS on the test set KDDTest-21 under different structures and Learningrates, the Learningrate was 0.5 and the number of HiddenNodes was 80; the model has the highest detection accuracy on the test set KDDTest-21, which was 64.67%.

Table 4 shows the detection accuracy of FCRNN-IDS on the training set and 2 test sets when performing multiclass detection tasks with different network structures and different Learningrates. Obviously, the experimental results on multiclassification tasks show that different network structures and Learningrates can affect the detection ability of the FCRNN-IDS. As shown in Table 4, when the hidden layer of the FCRNN-IDS was set to 80 nodes and the Learningrate was set to 0.5, the model has higher detection accuracy on the KDDTest-21 and KDDTest + test sets, which was 81.29% and 64.67%.

In order to compare the detection accuracy of various algorithms, similar to the 2-type task experiment, J48, Naive Bayes, Random Forest, and multilevel models were established through data mining software Weka and open source machine learning. Using 10 layers of cross-validation in the training set KDDTrain+, model training was performed using 7 machine learning algorithm models including layer perceptrons and support vector machines, and then, the model detection accuracy was tested in the



	J48	Naive Bayes	NB Tree	Random Forest	Random Tree	Multilayer Perceptron	SVM	RNN
🟦 KDDTest+	81.05%	76.56%	82.02%	80.67%	81.59%	77.41%	69.52%	83.28%
🟥 KDDTest-21	63.97%	55.77%	66.16%	63.26%	58.51%	57.34%	42.29%	68.55%

Figure 10: Detection accuracy of different models under 2-class tasks.

Table 2: Confusion matrix of the detection model based on artificial neural network on KDDTest$^+$ (2-class).

Actualclass	Predictedclass	
	Abnormal	Normal
Abnormal	8900	3933
Normal	314	9397

Table 3: Confusion matrix of model on KDDTest$^+$ (2-class).

Actualclass	Predictedclass	
	Abnormal	Normal
Abnormal	9362(↑)	3471
Normal	298	9413(↑)

test set. The experimental results are shown in Figure 14. Compared with the previous 2-class task, the detection accuracy of the traditional classification model generally drops under the multiclass task. Multilayer perception has the highest detection accuracy on the test sets KDDTest+ and KDDTest-21, 78.10% and 58.40%, respectively.

Under the same conditions, the neural network-based classification model achieves a detection accuracy of 79.9% when performing multiclassification tasks. Obviously, FCRNN-IDS performs better than other neural network-based detection models when performing multiclassification tasks.

Tables 5 and 6 show the confusion matrices of the neural network-based detection model and the fully connected recurrent neural network-based detection model on the test set KDDTest+, respectively.

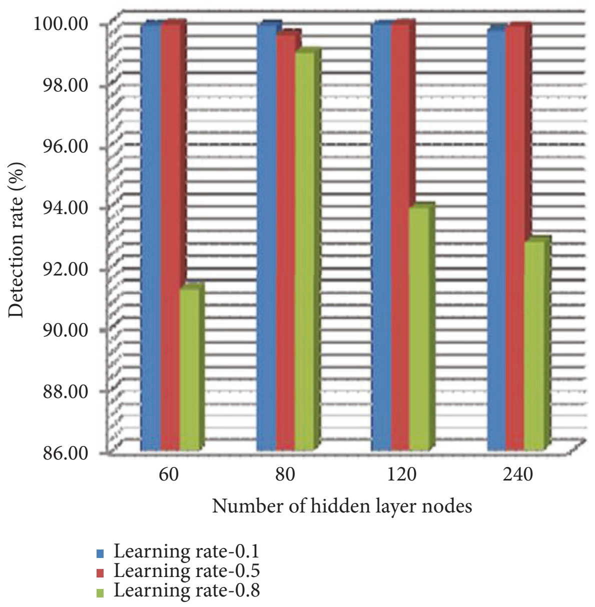


Figure 11: Detection rate of the model on the training set (multiclass).

Comparing the detection results in Tables 5 and 6, it can be seen that, in terms of correctly detecting DoS attacks, detection attacks, and U2R attacks, the detection model based on the RNN was fully connected to correctly detect more than 429 and 165 detection models based on neural networks, respectively, 2 contact records. Of course, in terms of correctly detecting normal connection records and R2L attack categories, the detection model

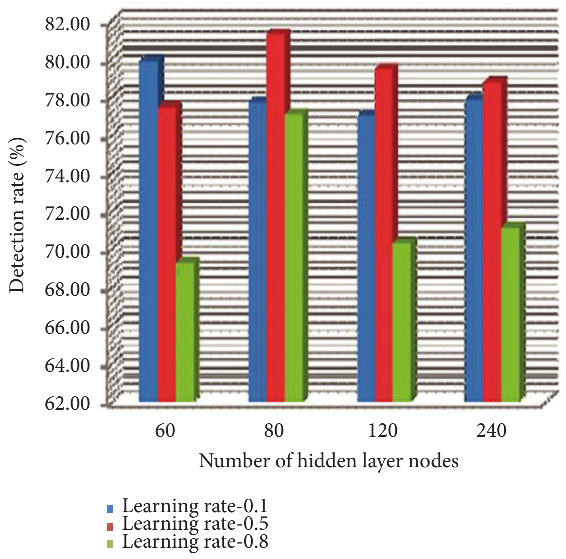


FIGURE 12: The detection rate of the model on the test set KDDTest^{+} (multiclassification).

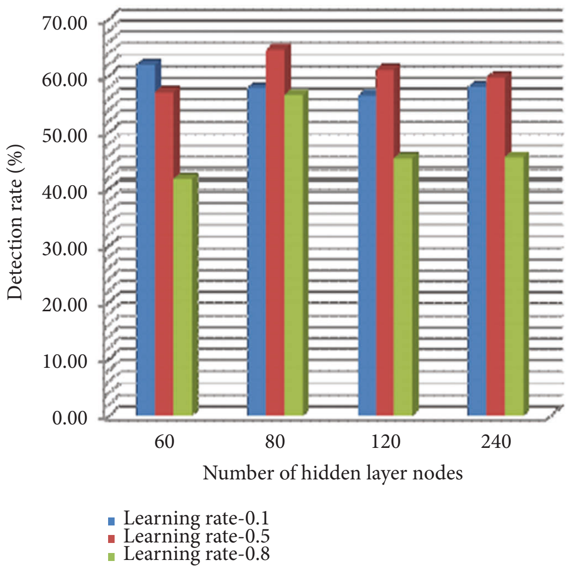


FIGURE 13: Detection rate of the model on the test set KDDTest^{-21} (multiclassification).

based on the fully connected RNN correctly detected 20 and 272 fewer connection records than the neural network, respectively.

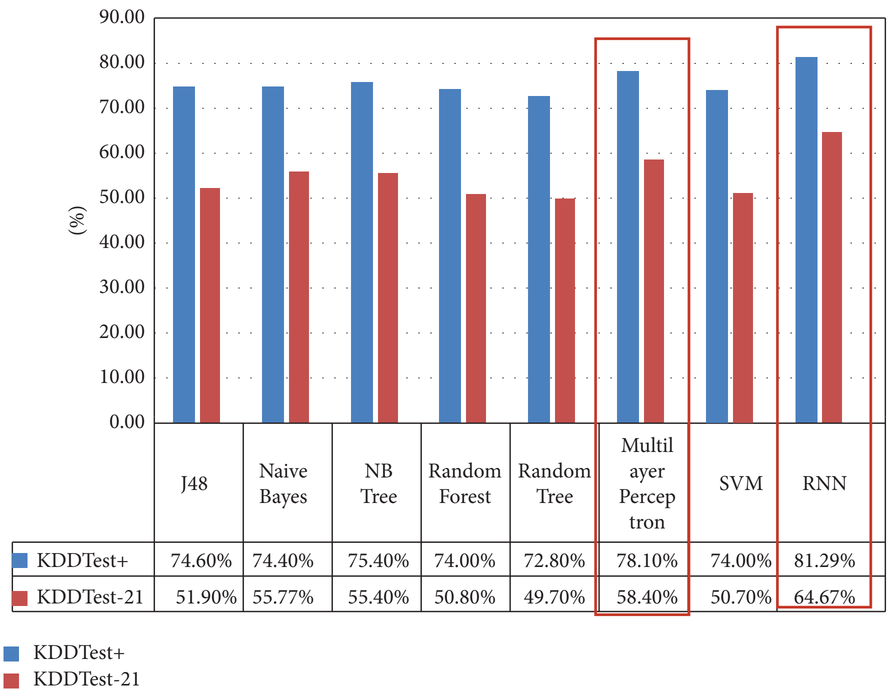
The confusion matrices of the four attack types detected by the model on the test set KDDTest+ are shown in Tables 7–10. Table 10 shows that the model false positives and recalls vary according to the type of attack. Table 11 shows the recall and false positive rates for different attack types.

In order to compare the detection performance of the fully connected neural network model and the partially connected RNN model proposed in [9] for intrusion detection, the training set and test set were constructed according to the method mentioned in [9], as shown in Table 12. In the experiments, the model was set to 20 HiddenNodes, the Learningrate was 0.1, and the training epoch was 50 times. The detection accuracy of the trained model on the test dataset reaches 97.09%, which was higher than 94.1% in the literature.

As shown in the experimental results above, the fully connected model proposed in this study has stronger feature space modeling ability and higher accuracy. Of course, without GPU acceleration, the model training time based on the fully connected RNN was 1765 seconds, which was higher than the training time of the model based on the partially connected RNN at 1383 seconds.

Table 4: Accuracy of models with different structures and Learningrates (multiclassification).

	$KDDTrain^+$	$KDDTest^+$	$KDDTest^{-21}$ (%)
HiddenNodes = 60, Learningrate = 0.1	99.84%	79.87%	61.98
HiddenNodes = 60, Learningrate = 0.5	99.87%	77.46%	57.18
HiddenNodes = 60, Learningrate = 0.8	91.23%	69.29%	41.85
HiddenNodes = 80, Learningrate = 0.1	99.82%	77.73%	57.84
HiddenNodes = 80, Learningrate = 0.5	99.53%	81.29%	64.67
HiddenNodes = 80, Learningrate = 0.8	98.97%	77.09%	56.64
HiddenNodes = 120, Learningrate = 0.1	99.85%	77.02%	56.55
HiddenNodes = 120, Learningrate = 0.5	99.87%	79.44%	61.11
HiddenNodes = 120, Learningrate = 0.8	93.90%	70.32%	45.49
HiddenNodes = 160, Learningrate = 0.1	99.68%	77.85%	58.05
HiddenNodes = 160, Learningrate = 0.5	99.80%	78.73%	59.71
HiddenNodes = 160, Learningrate = 0.8	92.79%	71.15%	45.64



	J48	Naive Bayes	NB Tree	Random Forest	Random Tree	Multilayer Perceptron	SVM	RNN
🟦 KDDTest+	74.60%	74.40%	75.40%	74.00%	72.80%	78.10%	74.00%	81.29%
🟥 KDDTest-21	51.90%	55.77%	55.40%	50.80%	49.70%	58.40%	50.70%	64.67%

Figure 14: Detection accuracy of different classification models under 5-class tasks.

Table 5: Confusion matrix of the neural network-based detection model on the test set $KDDTest^+$ for multiclassification tasks.

Actual class	Predicted class				
	Normal	DoS	R2L	U2R	Probe
Normal	9397	65	23	6	220
DoS	1515	5798	0	0	145
R2L	1789	2	952	6	5
U2R	144	6	10	21	19
Probe	403	164	0	0	1854

Table 6: Confusion matrix of the model on the test set KDDTest$^+$ for multiclassification tasks.

Actual class	Predicted class				
	Normal	DoS	R2L	U2R	Probe
Normal	9377 (1)	88	2	6	238
DoS	1011	6227 (f)	125	0	95
R2L	2058	0	680 (1)	6	10
U2R	149	0	11	23 (f)	17
Probe	231	166	5	0	2019 (f)

Table 7: DoS type confusion matrix.

Actual class	Predicted class	
	DoS	Others
DoS	6227	1231
Others	254	12099

Table 8: R2L type confusion matrix.

Actual class	Predicted class	
	R2L	Others
R2L	680	2074
Others	143	17646

Table 9: U2R type confusion matrix.

Actual class	Predicted class	
	U2L	Others
U2R	23	177
Others	12	18303

Table 10: Probe type confusion matrix.

Actual class	Predicted class	
	Probe	Others
Probe	2019	402
Others	360	16307

Table 11: Recall and false positive rates for different attack types.

Attack type	False positive rate (%)	Recall rate (%)
DoS	2.16	83.49
U2R	0.17	11.50
R2L	0.81	24.69
Probe	2.17	83.40

Table 12: Dataset composition.

Class	Number of training set samples	Number of test set samples
Normal	19454	12118
DoS	78290	45970
Probe	822	834
U2R	12	48
R2L	226	3238

5. Conclusion

Compared with traditional machine learning classification models, fully connected recurrent neural network, as a deep learning method, has stronger feature representation ability, can more comprehensively map high-dimensional feature space into low-dimensional feature representation, and has the ability to express complex functions. Therefore, the detection model based on the fully connected RNN can detect a large number of abnormal attack records in binary and multiclassification tasks, improve the accuracy of intrusion detection, and reduce the false positive rate. For example, in terms of correctly classifying abnormal records, the fully connected recurrent neural network-based detection model correctly detected 462 more records than the neural network-based detection model when performing a 2-class task [34].

The main contributions of this study are as follows:

(1) A new intrusion detection system based on fully connected recurrent neural network (FCRNN-IDS) was proposed, the training method of the model was studied, and the detection rate of models with different structures and different Learningrates was studied.

(2) Using the dataset NSL-KDD, the detection performance of seven traditional machine learning methods in 2-class and multiclass tasks was studied, respectively. It lays a foundation for FCRNN-IDS with traditional learning methods.

(3) The detection accuracy of FCRNN-IDS in 2-class and multiclassification tasks was studied, the performance of FCRNN-IDS in detecting various types of attacks was deeply analyzed, and the performance of FCRNN-IDS in detecting various types of attacks was compared.

Data Availability

The dataset can be obtained from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] P. Torres, W. Lu, C. Catania, S. Garcia, and C. G. Garino, "An analysis of recurrent neural networks for botnet detection behavior," in *Proceedings of the 2016 IEEE Biennial Congress of Argentina (ARGENCON)*, pp. 1–6, IEEE, Buenos, Aires, Argentina, June 2016.

[2] S. Kudugunta and E. Ferrara, "Deep neural networks for bot detection," *Information Science*, vol. 467, pp. 312–322, 2018.

[3] M. Tavallaee and S. Garcia, "A Detailed Analysis of the KDD CUP 99 Data set," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6, IEEE, Ottawa, ON, Canada, July 2009.

[4] P. S. Bhattacharjee, A. K. Fujail, and S. A. Begum, "Intrusion detection system for NSL-KDD data set using victories fitness

function in genetic algorithm," *Advances in Computational Sciences and Technology*, vol. 10, no. 2, pp. 235–246, 2017.

[5] R. A. R. Ashfaq, X. Z. Wang, J. Z. Huang, H. Abbas, and Y. L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Information Sciences*, vol. 468, pp. 470–486, 2019.

[6] B. Ingre and A. Yadav, "Performance Analysis of NSL-KDD Dataset Using ANN," in *Proceedings of the 2015 International Conference on Signal Processing and Communication Engineering Systems (SPACES)*, pp. 93–95, IEEE, Guntur, India, January 2015.

[7] M. Sheikhan, Z. Jadidi, and A Farrokhi, "Intrusion detection using reduced-size RNN based on feature grouping," *Neural Computing & Applications*, vol. 21, no. 6, pp. 1185–1190, 2012.

[8] Y. Wang and W.. Cai, "A deep learning approach for detecting malicious JavaScript code," *Security and Communication Networks*, vol. 9, no. 11, pp. 1520–1534, 2016.

[9] L. Deng, D. Li, and X. Yao, "Mobile Network Intrusion Detection for IotSystem Based on Transfer Learning Algorithm," *Cluster Computing*, vol. 22, no. 2, pp. 1–16, 2020.

[10] Z. Li and W. Ou, "Network communication intervention strategy for probabilistic model detection," *Small Micro Computer System*, vol. 38, no. 6, pp. 1175–1180, 2019.

[11] S. Wang, "Research on computer maintenance methods based on virus prevention and control[J]," *Electronic Components and Information Technology*, vol. 3, no. 08, pp. 108–111, 2019.

[12] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: taxonomy and open challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 369–392, 2014.

[13] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J Han, "When intrusion detection meets blockchain technology: a review," *IEEE Access*, vol. 6, pp. 10179–10188, 2018.

[14] K. Wang, M. Du, S. Maharjan, and Y. Sun, "Strategic honeypot game model for distributed denial of service attacks in the smart grid," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2474–2482, 2017.

[15] X. Liu, P. Zhu, Y. Zhang, and K. Chen, "A collaborative intrusion detection mechanism against false data injection attack in advanced metering infrastructure," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2435–2443, 2015.

[16] Y. Xiao, *Research and Analysis of Virus Propagation Model and Immune Strategy in Complex Network*, Southwest University, 2016.

[17] C. Network, *Based Application Recognition*, 2017, https://www.cwasco.com/c/en/us/products/ios-nx-os-softwwere/network-based-application-recognition-nbar/index.html?dtid=osscdc000283 (MBAR)[EB/OL].

[18] P. Ponmurugan, C. Venkatesh, M. D. Priyadharshini, and S. Balamurugan, "Intrusion detection strategies in smart grid," in *Design and Analysis of Security Protocol for Communication*, pp. 211–233, Scrivener Publishing, 2020.

[19] K. Song, P. Kim, V. Tyagi, and S. Rajasekaran, "Artificial Immune System (AIS) Based Intrusion Detection System (IDS) for Smart Grid Advanced Metering Infrastructure (AMI) Networks," *CS4624: Multimedia, Hypertext, and Information Access*, Virginia Tech, 2018.

[20] Y. Cui, "Modeling of Ideological and Political Education System in Colleges and Universities Based on Naive Bayes-BP Neural Network in the Era of Big Data," *Mobile Information Systems*, vol. 2022, Article ID 7609697, 9 pages, 2022.

[21] M. Chen, J. Cheng, G. Ma, L. Tian, X. Li, and Q. Shi, "Service Composition Recommendation Method Based on Recurrent Neural Network and Naive Bayes," *Scientific Programming*, vol. 2021, Article ID 1013682, 9 pages, 2021.

[22] L. Li, B. Lei, and C. Mao, "Digital twin in smart manufacturing," *Journal of Industrial Information Integration*, vol. 26, no. 9, Article ID 100289, 2022.

[23] L. Li, T. Qu, Y. Liu et al., "Sustainability assessment of intelligent manufacturing supported by digital twin," *IEEE Access*, vol. 8, pp. 174988–175008, 2020.

[24] Z. Zhang and S. Zhang, "Application of internet of things and naive Bayes in public health environmental management of government institutions in China," *Journal of Healthcare Engineering*, vol. 2021, p. 7, 2021.

[25] J. Yang, Y. Huang, R. Zhang, F. Huang, Q. Meng, and S. Feng, "Study on PPG Biometric Recognition Based on Multifeature Extraction and Naive Bayes Classifier," *Scientific Programming*, vol. 2021, Article ID 5597624, 12 pages, 2021.

[26] Y. Xiong, M. Ye, and C. Wu, "Cancer classification with a cost-sensitive naive Bayes stacking ensemble," *Computational and Mathematical Methods in Medicine*, vol. 2021, p. 12, 2021.

[27] L. Li, C. Mao, H. Sun, Y. Yuan, and B. Lei, "Digital twin driven green performance evaluation methodology of intelligent manufacturing: hybrid model based on fuzzy rough-sets AHP, multistage weight synthesis, and PROMETHEE II," *Complexity*, vol. 2020, no. 6, Article ID 3853925, 24 pages, 2020.

[28] P. Tao, H. Shen, Y. Zhang, P. Ren, J. Zhao, and Y. Jia, "Status forecast and fault classification of smart meters using LightGBM algorithm improved by random forest," *Wireless Communications and Mobile Computing*, vol. 2022, p. 11, 2022.

[29] R. Li, W. Zhang, S. Shen et al., "An intelligent heartbeat classification system based on attributable features with AdaBoost+Random forest algorithm," *Journal of Healthcare Engineering*, vol. 2021, Article ID 9913127, 19 pages, 2021.

[30] L. Li and C. Mao, "Big data supported PSS evaluation decision in service-oriented manufacturing," *IEEE Access*, vol. 8, no. 99, pp. 154663–154670, 2020.

[31] P. Fan, "Random Forest Algorithm Based on Speech for Early Identification of Parkinson's Disease," *Computational Intelligence and Neuroscience*, vol. 2022, Article ID 3287068, 6 pages, 2022.

[32] H. Alla, L. Moumoun, and Y. Balouki, "A Multilayer Perceptron Neural Network with Selective-Data Training for Flight Arrival Delay Prediction," *Scientific Programming*, vol. 2021, Article ID 5558918, 12 pages, 2021.

[33] F. Sayyahi, S. Farzin, and H. Karami, "Forecasting daily and monthly reference evapotranspiration in the aidoghmoush basin using multilayer perceptron coupled with water wave optimization," *Complexity*, vol. 2021, Article ID 6683759, 12 pages, 2021.

[34] J. Woodbridge, H. S. Anderson, A. Ahuja, and D. Grant, "Predicting domain generation algorithms with long short-term memory networks," 2019, https://arxiv.org/abs/1611.00791.