Design and Implementation of Intrusion Detection System Based on Neural Network

Zengyu Cai, Jingchao Wang, Jianwei Zhang(✉), and Xi Chen

Zhengzhou University of Light Industry, Zhengzhou 450000, China

Abstract. With the continuous emergence of cyber-attacks, traditional intrusion detection methods become increasingly limited. In the field of network security, new intrusion detection methods are needed to ensure network security. To solve the problems, relevant knowledge involved was first introduced. Then, an intrusion detection system based on neural network was designed according to the general intrusion detection framework, and the design of the event collector and analyzer in the system was described in detail. Experiments were conducted with the weight initialization method of the neural network model, the selection of the activation function, and the selection of the optimizer. Finally, the most suitable hyperparameters were determined and the optimal neural network model was trained. The test results show that the application of neural network to the intrusion detection system can greatly improve the accuracy of intrusion detection, thereby improving the security of computer networks.

Keywords: Intrusion detection system · Intrusion detection technology · Neural network

1 Introduction

Intrusion detection system is one of the commonly used equipment to prevent cyber-attacks. The intrusion detection system is a system used to detect the security status of a computer network. It can detect malicious intrusions and attacks that cannot be detected by the firewall [1]. At present, the intrusion detection system is not only used in the internal network of enterprises and institutions, but also widely used in various fields such as smart home systems, smart grid, and smart cities [2–5]. However, traditional intrusion detection systems require a large number of normal or abnormal behavior characteristics to detect whether intrusions have occurred. Its detection ability largely depends on the size of the rule base, and the rule base requires a lot of manpower and material resources to maintain, and the detection effect of emerging and unknown cyber-attacks is not satisfactory.

In recent years, artificial intelligence technology has been born under the mutual influence of other science and technology, and has been applied to various fields, and network security defense has also been greatly affected. W. Alhakami [6] stated that integrating machine learning into an intrusion detection system can greatly improve the

J. Zeng et al. (Eds.): ICPCSEE 2021, CCIS 1452, pp. 179–189, 2021.
https://doi.org/10.1007/978-981-16-5943-0_15

efficiency of intrusion detection. X. K. Li and A. Khraisat [7, 8] proposed random forest or decision tree algorithm in machine learning algorithm as an intrusion detection algorithm. Its detection speed is fast, but the detection accuracy is low. I. Ahmad, W. L. Al-Yaseen and J. X. Wei [9–11] proposed support vector machines as an intrusion detection algorithm, it has better performance than the former, but the accuracy of classification is still not ideal. G. J. Liu and H. Wang [12, 13] designed an intrusion detection system based on convolutional neural network. Convolutional neural network has great advantages in image processing and also shows good results in the field of intrusion detection. In summary, a variety of machine learning algorithms have been applied to the field of intrusion detection, but different machine learning algorithms have shown great differences in practical applications, and the selection and processing of data sets also have a great impact on detection effects. In this paper, the neural network with better effect is selected as the core detection algorithm of intrusion detection, and the NSLKDD data set which has been improved on the KDDCUP99 data set is selected, and the data set is processed by one-hot encoding. Finally, after adjusting the hyperparameters, the expected intrusion detection classifier is obtained and the whole system is realized.

2 Related Technologies

2.1 Intrusion Detection System

Intrusion Detection System (IDS) is a network security device that monitors the data flowing in the network in real time, and issues an alarm or takes active defense actions when suspicious behavior is found. It monitors the operating status of the network and computer in real time, continuously collects information and analyzes the collected information, thereby discovering signs of system attacks or intrusions. When the system detects suspicious behavior or signs of abnormality, it will generate a detection log and issue an alarm, and take active defense measures when necessary [14, 15].

2.2 BP Neural Network

BP (Back Propagation) neural network algorithm is a kind of machine learning algorithm. The main idea is to perform back propagation through error calculation and adjust the parameters of the neural network model to achieve a more accurate neural network model [16, 17]. The BP neural network is composed of a stack of multi-layer neuron models. After multiple iterations of training and weight updates, a model that can be used for prediction or classification is finally obtained. Each iteration consists of three steps: First, forward calculation, send the training input to the neural network to get the model output; second, back propagation, calculate the difference between the model output and the corresponding real output to obtain the output error of the model. Finally, the weight is updated. The partial derivative of each weight is obtained for the output error, the gradient of the output error is obtained, and the gradient is multiplied by a ratio and the inverse is added to the current weight. At this point, one iteration is completed. After many iterations in the neural network training process, the difference between the model output and the real output is getting smaller and smaller, which means that the model output is getting closer and closer to the real output, and the accuracy of the model is getting higher and higher.

3 Design of Intrusion Detection System Based on Neural Network

3.1 System Framework

The design of the intrusion detection system based on neural network is based on the general intrusion detection framework, which mainly includes four modules, event collector, event analyzer, system monitoring platform and database system, as shown in Fig. 1.

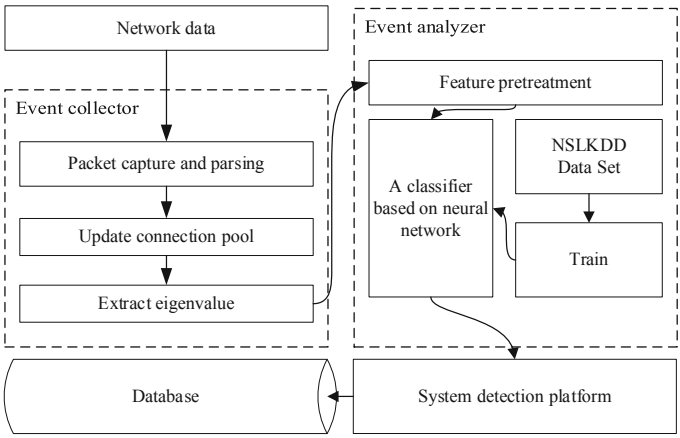

Fig. 1. System diagram of Intrusion Detection System based on Neural Network

Event collector: The information exchange on the network is carried out in the form of data packets, so a large number of data packets can be captured from the network through a specific technology. However, the captured data packets are in binary form, which requires further analysis of the captured data packets, and converts the binary value into the meaning it expresses according to the structure of the data packet. However, it is impossible to judge whether the current network status is safe by only relying on the captured data packets. The data packets need to be further sorted to extract the characteristic information required by the system, such as the number of simultaneous TCP connections and whether the data packets contain sensitive permission information, etc.

Event analyzer: The event analyzer first needs to load the trained model, and then receives the event feature values transmitted from the event collector. After preprocessing the event feature values, input a series of feature values into the neural network-based Classifier for classification. According to the classification results, determine whether there is intrusion behavior in the network environment where the system is located, and then send the analysis results to the system monitoring platform.

System monitoring platform: The system monitoring platform accepts messages from the event analyzer, organizes the messages and displays them to users and stores them in the database. If the system receives a report of intrusion behavior sent by the event analysis platform, it will send an alert and intrusion behavior information to the system administrator.

Database: The database can be used to store the results of system analysis and captured packet information, so that the administrator can use the stored information to further analyze the security status of the system.

3.2 Event Collector

Packet Capture and Parsing. The capture and analysis of data packets are the most basic part of the entire system, and network data packets are the information source of the system. This system uses Python language to develop event collector based on PyCharm. Scapy is a Python class library, which encapsulates the interface of WinPcap, allowing python to call the interface of WinPcap more conveniently to achieve the purpose you want. In the event collector, this article mainly uses Scapy's sniff function to capture data packets. The scapy package can be installed in the Python environment through the command "pip install scapy". After the packet is captured, the packet needs to be parsed. According to the different structure of different types of data packets, the protocol type, service type source address, destination address and other fields are parsed from it. For this system, only need to parse and capture IP, TCP and UDP data packets.

Connection Pool Management. In order to extract the information required for system classification from a large number of data packets, a connection pool needs to be established within the system to record the current connection status of the system. But only the TCP protocol will establish a link in the process of transmitting information, and UDP is a connectionless transmission protocol, which requires corresponding adjustments to the definition of the link in the connection pool. The information to be recorded for each link in the connection pool is shown in Table 1.

Table 1. Variables in the connection pool

Name	Explain	Name	Explain
src_ip	Source IP	flag	Connect flag
dst_ip	Destination IP	src_port	Source port
protocol	Protocol type	dst_port	Dextination port
time	Connect update time	is_fin	Connect status
s_time	Connect start time	num_urg	Emergency packets count
src_bytes	Source traffic count	num_wro	Error packets count
dst_bytes	Destination traffic count		

For UDP data packets, each UDP data packet will be regarded as a connection, the protocol is set to UDP, time and s_time are set to the time when the data packet is received, and the flag field is set to the normal state by default.

For TCP, the connection status can be obtained through TCP data packets. For example, when the SYN flag bit in the TCP data packet is 1, it means that a host sends a request

to create a link to another host. At this time, a new connection information is created in the connection pool, the protocol is set to TCP, and the s_time and time fields are set to the current time and flag to S0, which means that only the connection establishment request is received, but the link is not established. Set the service field according to the port number. When a data packet with the same source address and port and destination address and port is received, update the connection information, update the time field and the connection information is placed at the top of the entire connection pool, and at the same time according to the data packet content updates flag, num_wro, num_urg, is_fin, src_bytes, dst_bytes and other fields.

Every time the connection pool is updated, the system will scan the entire connection pool. When the number of connections in the connection pool exceeds 100, the connection information that ends in more than two seconds and is after the first 100 connections will be deleted to ensure a low cost of system operation.

Extract Characteristic Values. After the necessary conditions for extracting feature values are prepared, the feature values required by the intrusion detection classifier can be counted from the connection pool. Because the classifier of this system is trained by using the NSLKDD data set, it is necessary to count the feature value types in NSLKDD when counting the feature values. Each sample in the data set has 43 values, including 42 feature values and 1 label value. The process of extracting feature values is to extract and count the captured data packet information and the information in the connection pool.

Extraction of Basic Features. There are 9 types of eigenvalues in this part. Duration represents the duration of the connection, which can be obtained by subtracting s_time from the time in the connection information. The protocol_type can be obtained from the protocol field of the connection information. The service can be obtained according to the destination port number. The flag, src_bytes, dst_bytes, wrong_fragmenta, urgent can be obtained directly from the connection information, but due to the lack of detailed descriptions of some states and feature extraction methods in the official documents of the NSLKDD data set, it is temporarily defined as 4 states, and SF means normal Status, S0 means that only SYN packets are received without SYN/ACK packets, that is, a SYN error occurs, an RST flag is a REJ error, and OTH means other connection error conditions. The land feature value indicates whether the address and port of the source host and the address and port of the destination host are the same, which can be obtained by comparing the corresponding fields in the connection information.

Extraction of Content Features. There are 13 types of feature values in this part, namely hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, is_hot_login, is_guest_login. The extraction of the feature value in this part requires analysis of the load of the data packet, but there is less information currently implemented, which is difficult to achieve in this article, so it is basically filled with 0, so these features can be ignored in the feature extraction stage.

Extraction of Statistical Characteristics of Network Traffic Based on Time and Host. This is part of a total of 19 kinds of characteristic values, respectively, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate,dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate,dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate. The characteristic value of this part needs to be statistically obtained on the connection information. In the connection pool, the characteristic value of this part can be obtained through the statistics of the connection information of the corresponding condition.

3.3 Event Analyzer

Character Feature Numerical Processing. Among the 41 feature values extracted, three feature values are expressed in the form of strings, which requires processing them and converting them into numbers. The three characteristic values are the protocol type, the service type of the target host, and the normal or error status of the connection. There are two ways to digitize. As far as the protocol type is concerned, you can replace TCP with 1, UDP with 2, and ICMP with 3. However, this method will not avoid the size relationship of the number itself, but for the protocol type, there is no size relationship between them, so it is inappropriate to use this method. Therefore, this system adopts one-hot encoding method, which represents a one-dimensional vector of bits, such as TCP-[1,0,0], UDP-[0,1,0], ICMP-[0,0,1]. In this way, the size relationship among the numbers themselves is avoided, making the feature value more scientific. After numerical processing, the feature value of each event has changed from 41 to 122.

Classifier Design Based on Neural Network. This system uses a neural network containing two hidden layers and an output layer as the classifier for intrusion detection, and uses the NSLKDD data set as the training set and test set of the neural network model. In the NSLKDD data set, each piece of data consists of 42 numbers, including 41 feature values and a label value. The feature values in the training set also need to be pre-processed to change the 41-dimensional feature values into 122-dimensional feature values. For the label values in the training set, one-hot encoding is also required. There are two types of label values. One is normal, which means that no intrusion has occurred on the network, and the other is abnormal, which means that intrusion has occurred. After processing with one-hot encoding, the label value becomes a two-dimensional vector.

The classifier model is divided into three layers. Input the processed feature values into the first layer to get the result, and then use the output of the first layer as the input of the second layer, and then use the output of the second layer as the input of the output layer. Finally, the output layer outputs a two-dimensional vector as the result. Tensorflow commonly used activation functions include relu, sigmoid, tanh, etc. The commonly used weight initialization methods include initializing weights with random numbers generated by uniform distribution, initializing weights with random numbers generated

by normal distribution, and initializing weights with random constants. Commonly used types of optimizers include SGD, Adam, Adamax, etc.

The choice of the three hyperparameters needs to be further determined by analyzing the following experimental results. Since the final classification result is a two-dimensional vector, the output layer must be composed of two neurons. However, for different event eigenvalue models, the output values may differ greatly. Therefore, the softmax activation function needs to be used for the output layer to map the sum of the output two-dimensional vectors to 1. When the first value in the two-dimensional vector is greater than 0.5, the second value must be less than 0.5, and vice versa. According to the one-hot encoding rule, for events with the second digit value greater than or equal to 0.5, the system determines that there is an intrusion in the network and issues an alarm.

Classifier Training Based on Neural Network. In the training process of the neural network model, it is necessary to use the same training set for multiple iteration training until the minimum loss value is obtained. At the beginning of training, the weights in the model need to be initialized, and then the data set is sent to the neural network model, and the model calculates the corresponding output value according to the received data set. After obtaining the model output value, it is necessary to calculate the degree of difference between the output value and the true value.

This article chooses to use the Euclidean distance to measure the degree of difference between the two, and then calculate the gradient of the loss function, and according to the given learning Rate to update the weight and bias of the model, and then send it

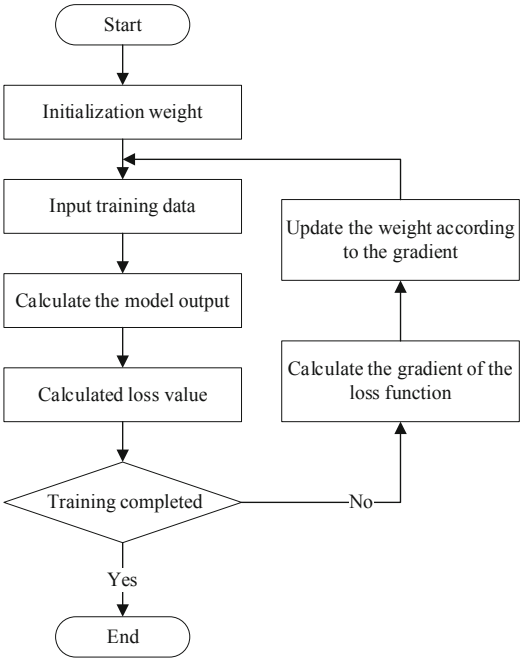

Fig. 2. Classifier training process

to the training set again and repeat the above process, until the size of the loss value no longer drops or the decline is small, you can stop the training, at this time a trained neural network model is obtained which is also called a classifier. The flow of the entire training process is shown in Fig. 2.

4 Intrusion Detection Experiment Based on Neural Network

4.1 Experimental Environment

This article mainly uses the Python programming language to implement the intrusion detection system. The database used is SQL Server 2019. The Python packages used are Scapy 2.4.3, Scikit-learn 0.23.1, Tensorflow 2.1.0, Numpy 1.18.5, and WinPcap 4.1.2 is required for packet capture.

The main workload in the subject includes: using Scapy to call the WinPcap interface for packet capture and analysis; using Numpy to read and process the original data set; using Scikit-learn, Tensorflow to train and test the neural network model of the event analyzer; optimize the neural network model for indicators such as accuracy rate, false alarm rate, and false alarm rate.

The neural network model used in this article is a three-layer fully connected neural network. The first and second layers have 128 neurons respectively. The third layer, the output layer, has 2 neurons. BATCH_SIZE (the number of samples selected for one training) is set to 100, EPOCHS (the number of cycles of the entire data set) is set to 100, and LEARN_RATE (learning rate) is set to 0.01.

4.2 Experimental Results and Analysis

There are many hyperparameters in the training of neural network models, and the choice of hyperparameters has an important influence on the final detection effect of the model. This article mainly conducts experiments on three hyperparameters, including weight initialization method, activation function, and optimizer, and selects the optimal parameters. When experimenting on a certain parameter, other parameters remain unchanged.

Neural Network Weight Initialization Method. In this experiment, the main task is to compare the three weight initialization methods and select the most suitable weight initialization method. The activation function uses sigmoid, and the optimizer uses Adam. Use the prepared data set for testing, and the test results obtained are shown in Table 2.

According to Table 2, in terms of weight initialization, when using the random number generated by the normal distribution to initialize the weight, the accuracy is 0.9128, which is significantly higher than the random number generated by the uniform distribution and the accuracy of using the random constant to initialize the weight. This paper uses random numbers generated from a normal distribution to initialize the weights.

Table 2. Comparison of experimental results with different weight initialization methods

	Random normal intializer	Random uniform initializer	Constant initializer
Accuracy	0.9128	0.4654	0.8594
Non-response rates	0.0273	0	0.0158
False positives rates	0.0597	0.5345	0.1246

Activation Function of Neural Network. In this experiment, the main task is to compare the three activation functions such as relu, sigmoid, and tanh to select the most appropriate activation function. The weight initialization method has been determined by the first experiment to initialize the random number generated by the normal distribution. For weight, the optimizer uses Adam. Use the prepared data set for testing, and the test results obtained are shown in Table 3.

Table 3. Comparison of experimental results with different activation functions

	sigmiod	relu	tanh
Accuracy	0.9128	0.4654	0.9166
Non-response rates	0.0273	0.0	0.0265
False positives rates	0.0597	0.5345	0.0567

According to Table 3, among the three activation functions, the tanh activation function performs the best among the three activation functions. Although the effect of using the sigmoid activation function is close to that of using the tanh activation function, it has the accuracy rate, false negative rate and the false alarm rate in three aspects is worse than using the tanh activation function, so this article finally uses the tanh activation function to apply to the neural network.

Neural Network Optimizer. In this experiment, the main task is to compare three optimizers such as SGD, Adam, and Adamax to select the most suitable optimizer. The weight initialization method has been determined by the first experiment to initialize the random number generated by the normal distribution. The activation function is determined by the second experiment and use the relu function as the activation function of the neural network model. Use the prepared data set for testing, and the test results obtained are shown in Table 4.

It can be seen from Table 4 that for the optimizer, the use of the Adam optimizer and the use of the Adamax optimizer have similar effects, but the effect of using the Adam optimizer in the false negative rate is obviously better than using the Adamax optimizer, so the final decision to use the Adam optimizer is for the training of neural networks.

Table 4. Comparison of experimental results using different optimizers

	Adam optimizer	Adamax optimizer	SGD optimizer
Accuracy	0.9166	0.9134	0.9094
Non-response rates	0.0265	0.0324	0.0210
False positives rates	0.0567	0.0541	0.0694

In summary, after multiple rounds of testing, the optimized neural network model on the NSL-KDD dataset is obtained: the weights of the neural network are initialized with random numbers generated by the normal distribution, the activation function uses the tanh function, and the optimizer uses Adam optimizer.

5 Conclusions

This paper takes the current significant problems in the field of network security as the demand, and uses the current popular neural network algorithm in the field of artificial intelligence to evaluate the engineering application value of the field of network security. The neural network model is used as the core of intrusion detection algorithm based on network data packets to identify intrusions in the network. Multiple hyperparameters of the neural network are tuned and built on the basis of Tensorflow open source machine learning architecture. Good results are achieved in actual tests.

Acknowledgements. This work is supported by National Natural Science Foundation of China (62072416), Henan Province Science and Technology Research Project (202102210176), Zhongyuan Science and Technology Innovation Leader of Zhongyuan Talent Project (214200510026), Henan Province Science and Technology Project (212102210429), The fourth batch of innovative leading talents of Zhihui Zhengzhou 1125 talent gathering plan (ZhengZheng[2019] No. 21).

References

1. Yin, C., Xia, L., Zhang, S., Sun, R., Wang, J.: Improved clustering algorithm based on high-speed network data stream. Soft. Comput. **22**(13), 4185–4195 (2017). https://doi.org/10.1007/s00500-017-2708-2
2. Zhang, R., et al.: An intrusion detection scheme based on repeated game in smart home. Mob. Inf. Syst. **2020**, 9 (2020). Article ID. 8844116
3. Zou, X., Cao, J.H., Guo, Q., Wen, T.: A novel network security algorithm based on improved support vector machine from smart city perspective. Comput. Electr. Eng. **65**, 67–78 (2018)
4. Jow, J., Xiao, Y., Han, W.L.: A survey of intrusion detection systems in smart grid. Int. J. Sensor Netw. **23**(3), 170–186 (2017)
5. Zuo, X., Chen, Z., Dong, L., Chang, J., Hou, B.: Power information network intrusion detection based on data mining algorithm. J. Supercomput. **76**(7), 5521–5539 (2019). https://doi.org/10.1007/s11227-019-02899-2

6. Alhakami, W.: Alerts clustering for intrusion detection systems: overview and machine learning perspectives. Int. J. Adv. Comput. Sci. Appl. **10**(5), 1–10 (2019)
7. Li, X.K., Chen, W., Zhang, Q.R., Wu, L.F.: Building auto-encoder intrusion detection system based on random forest feature selection. Comput. Secur. **95** (2020). Article ID. 101851
8. Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J., Alazaab, A.: Hybrid intrusion detection system based on the stacking ensemble of C5 decision tree classifier and one class support vector machine. Electronics **9**, 173 (2020)
9. Ahmad, I., Basheri, M., Iqbal, M.J., Rahim, A.: Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. IEEE Access **6**, 33789–33795 (2018)
10. Al-Yaseen, W.L., Othman, Z.A., Nazri, M.Z.A.: Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. Expert Syst. Appl. **67**, 296–303 (2017)
11. Wei, J.X., Long, C., Li, J.W., Zhao, J.: An intrusion detection algorithm based on bag representation with ensemble support vector machine in cloud computing. Concurrency Comput. Pract. Exp. **32**(24), e5922 (2020)
12. Liu, G.J., Zhang, J.B.: CNID: research of network intrusion detection based on convolutional neural network. Discrete Dyn. Nat. Soc. **2020**, 11 (2020)
13. Wang, H., Cao, Z.J., Hong, B.: A network intrusion detection system based on convolutional neural network. J. Intell. Fuzzy Syst. **38**(6), 7623–7637 (2020)
14. Li, P., et al.: Intrusion detection methods based on incomplete RFID traces. Chin. J. Electron. **26**(4), 675–680 (2017)
15. Keegan, N., Ji, S.-Y., Chaudhary, A., Concolato, C., Yu, B., Jeong, D.H.: A survey of cloud-based network intrusion detection analysis. HCIS **6**(1), 1–16 (2016). https://doi.org/10.1186/s13673-016-0076-z
16. Khan, R., Al-Shehri, M.: A futuristic analysis approach of neural network for intrusion detection system. Dilemas Contemporaneos-Educacion Politica Y Valores **6**(3), 1–16 (2019)
17. Liu, Y., Zhu, L.: A new intrusion detection and alarm correlation technology based on neural network. EURASIP J. Wirel. Commun. Netw. **2019**(1), 1 (2019). https://doi.org/10.1186/s13638-019-1419-z