Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.035919
Article

RRCNN: Request Response-Based Convolutional Neural Network for ICS Network Traffic Anomaly Detection

Yan Du[1,2], Shibin Zhang[1,2,*], Guogen Wan[1,2], Daohua Zhou[3], Jiazhong Lu[1,2], Yuanyuan Huang[1,2], Xiaoman Cheng[4], Yi Zhang[4] and Peilin He[5]

[1]Country School of Cybersecurity, Chengdu University of Information Technology, Chengdu, 610225, China
[2]Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu, 610225, China
[3]DAQSOFT CO., LTD, Chengdu, 610213, China
[4]Communication and Information Technology Center, Petro China Southwest Oil and Gas Company, Chengdu, 610057, China
[5]School of Computing and Information, University of Pittsburgh, Pittsburgh, 152607, PA, USA
*Corresponding Author: Shibin Zhang. Email: cuitzsb@cuit.edu.cn
Received: 09 September 2022; Accepted: 19 November 2022

Abstract: Nowadays, industrial control system (ICS) has begun to integrate with the Internet. While the Internet has brought convenience to ICS, it has also brought severe security concerns. Traditional ICS network traffic anomaly detection methods rely on statistical features manually extracted using the experience of network security experts. They are not aimed at the original network data, nor can they capture the potential characteristics of network packets. Therefore, the following improvements were made in this study: (1) A dataset that can be used to evaluate anomaly detection algorithms is produced, which provides raw network data. (2) A request response-based convolutional neural network named RRCNN is proposed, which can be used for anomaly detection of ICS network traffic. Instead of using statistical features manually extracted by security experts, this method uses the byte sequences of the original network packets directly, which can extract potential features of the network packets in greater depth. It regards the request packet and response packet in a session as a Request-Response Pair (RRP). The feature of RRP is extracted using a one-dimensional convolutional neural network, and then the RRP is judged to be normal or abnormal based on the extracted feature. Experimental results demonstrate that this model is better than several other machine learning and neural network models, with F1, accuracy, precision, and recall above 99%.

Keywords: Industrial control system (ICS); dataset; network traffic; anomaly detection

1 Introduction

The network environment of early industrial control systems is relatively isolated, and attackers on the Internet could not interfere with it directly [1,2]. In order to ensure the stable operation of the system, the communication protocols used in ICS generally do not add additional security mechanisms or adopt high-efficiency but still flawed security mechanisms [3]. But in recent years, ICS has changed significantly [4]. In order to further improve industrial production efficiency and industrial upgrading, the majority of ICS are already connected to the Internet. While the Internet has brought convenience to ICS, it has also brought severe security problems [5]. In addition, vulnerabilities in the firmware and software of some programmable logic controllers (PLC) are also an important cause of security issues [6].

In the past few years, ICS has often suffered from serious network attacks. In 2010, Stuxnet virus swept the global industry and was able to carry out targeted attacks on infrastructure, of which Iran suffered the most serious attack [7]. In 2014, Dragonfly used the malware Havex to break into specific industrial control systems, with the main victims being energy companies in Europe and the United States [8]. In 2015, the Ukrainian power department was compromised by malicious code called BlackEnergy, causing multiple substations to malfunction [9]. In 2017, the ransomware NotPetya attacked many government agencies and companies in Ukraine, ultimately wreaking havoc worldwide [10].

Network attacks on ICS will result in disruption of control over these critical infrastructures, causing severe physical damage to plants, the environment, and humans. To address this new set of threats, it is necessary to strengthen ICS's ability to respond to network attacks.

The main contributions of this work are: (1) An automatic pumping system based on Modbus TCP protocol was constructed and then several types of network attacks were performed on the system. The raw network traffic generated by the system is captured and can be used in the research of ICS security. (2) Based on one-dimensional convolutional neural network, a unique network traffic anomaly detection algorithm is proposed. The F1, accuracy, precision, and recall obtained by this algorithm are all above 99%.

The remainder of the paper is structured in the following manner: Section 2 overviews related work. Section 3 introduces the dataset we produced, including the environment of the automatic pumping system built in the laboratory, the attack methods, and the production process of the dataset. Section 4 introduces our proposed anomaly detection model: RRCNN. To measure the performance of RRCNN and better conduct comparative experiments, other anomaly detection models are introduced in Section 5. In Section 6, the experimental results are presented and analyzed. Finally, our conclusions and future work are presented in Section 7.

2 Related Work

To ensure the security of ICS, anomaly detection is necessary and has attracted attention from industry and academia [11–13]. Shang et al. [14] designed a particle swarm optimization algorithm (PSO) for parameter optimization PSO-SVM algorithm through the frequency of the function code to identify the abnormal traffic of Modbus TCP. However, this method only considers the role of function code and ignores the correspondence between register address and function code. Based on Snort software, Morris et al. [15] improved the serial-based ICS to enable the system to detect abnormal packets, but on the premise that fairly strict Snort rules were formulated.

Artificial intelligence has achieved considerable achievements in the past few years [16,17], and some scholars have gradually started to combine artificial intelligence with anomaly detection. Based

on the method of reducing error pruning tree, Ponomarev et al. [18] proposed a network anomaly detection algorithm for ICS. But the generalization ability of the decision tree is poor. The detection algorithm proposed by Liang et al. [19] is based on multi-feature data clustering optimization. The algorithm first needs to determine a node with a high safety factor, and then use the node as the center to cluster the surrounding multi-feature data.

It is worth noting that, as a branch of artificial intelligence [20], deep learning has been applied to network anomaly detection by many scholars. Yang et al. [21] proposed an intrusion detection algorithm that uses CNN to extract features of network traffic. Combining deep neural networks with naive Bayes, Wang [22] proposed an intrusion detection algorithm. Using one-dimensional CNN, Kravchik et al. [23] proposed an algorithm to detect abnormal behavior. Thirty-one of the 36 network attacks in the Swat dataset can be detected by their algorithm. Hao et al. [24] proposed an attention based Bi-LSTM intrusion detection algorithm. The algorithm can effectively detect the attack behavior in the current network environment. Teixeira et al. [25] presented a publicly available labeled dataset and proposed a flow-based intrusion detection algorithm using artificial neural networks (ANNs). This method achieves outstanding performance. However, this algorithm first requires the use of Argus tool to extract statistical features of the traffic, which is time consuming.

However, these algorithms above rely on statistical properties that are manually extracted using the experience of security experts. Feature engineering not only requires a lot of effort and time, but also the performance of the algorithm is largely limited by the quality of the extracted features. To solve this problem, we propose an ICS network traffic anomaly detection method based on one-dimensional CNN, which uses raw data to extract the joint features of request packet and response packet.

3 Dataset

Compared with traditional network security, the field of ICS network security is even more lacking in datasets. Many manufacturers are reluctant to open up data from their production systems to the public. Lemay et al. [26] proposed a SCADA network dataset, but all attacks were implemented through off-the-shelf tools (i.e., Metasploit) without exploiting the vulnerabilities of the Modbus protocol.

In this work, in addition to off-the-shelf penetration testing tools, we have designed some unique attacks for the flaws in the Modbus TCP protocol. The generated dataset is more representative of the current complex ICS network environment and can be used to evaluate the performance of ICS network anomaly detection algorithms.

3.1 Automatic Pumping System

We have built an automatic pumping system in our laboratory, which is a type of industrial control system. As shown in Fig. 1, the system consists of an operator station, an engineer station, a Schneider PLC, two water level sensors, a water tower, a reservoir, a water pump, an inlet valve, and an outlet valve.

The main function of the automatic pumping system is to maintain the water level in the water towers and reservoirs at a specific value, and the operator can change this value in real time. The water tower has an outlet valve, and when it is opened, the water in the tower decreases. When the water in the water tower is less than the preset value, the pump is turned on and then the water from the reservoir will be pumped to the water tower. When the water in the reservoir is less than the preset value, the inlet valve is opened so that the water in the reservoir is increased.

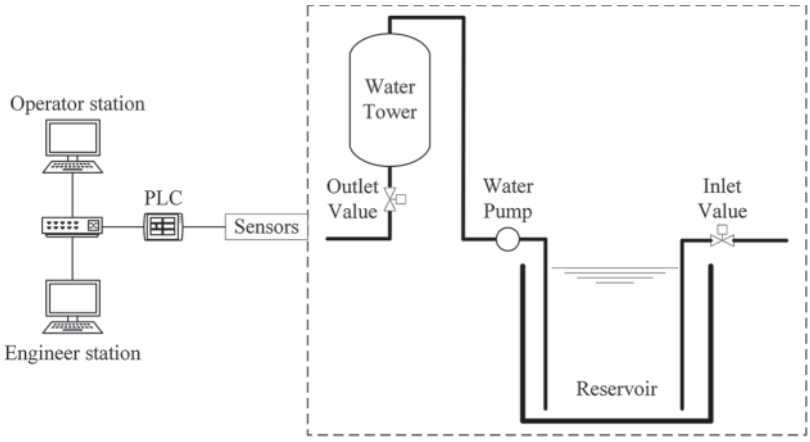

Figure 1: System topology

SCADA software including the Master Terminal Unit (MTU) and Human Machine Interface (HMI) is installed on the operator and engineering stations. In this system, the MTU sends a Modbus read packet to the PLC at regular intervals to read the latest status of the field devices, such as the water level of the water tower. The PLC returns the requested result to the MTU, and then updates it on the HMI in real time. In addition, the operator can change the state of the field device through the HMI, such as turning on the water pump.

3.2 Attack Scenarios and Data Capture

We conducted multiple Denial of Service (DoS) attacks not only on the above automatic pumping system, but also multiple Man-In-The-Middle (MITM) attacks, as shown in Fig. 2. In a MITM attack, an attacker can arbitrarily tamper with the payload of a packet. Therefore, an attacker can launch command injection attacks or response injection attacks. The difference between these two attacks is the difference in the sender of the packets.

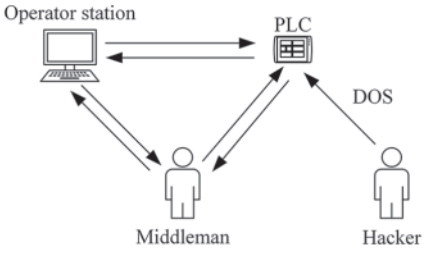

Figure 2: MITM and DoS attacks in ICS

Command Injection: When the MTU sends a control command to the PLC, the middleman captures the request packet and tampers with it. Through command injection, the original intention of the operator can be altered and dangerous commands can be sent to the PLC. For example, a close valve control command in a request packet is tampered with as an open valve control command.

Response Injection: When the PLC receives a control command from the MTU, it sends the result of the command execution to the MTU in the form of a response packet. The middleman captures the packet and tampers with it. Response injection allows hiding the true state of the field device from the

operator. For example, the information contained in the untampered response packet sent from the PLC to the MTU indicates that the valve is closed. After the middleman tampered with the packet, the HMI showed that the valve was open.

Denial of service (DoS): Smod is a modular framework with multiple diagnostic and attack capabilities for penetration testing the Modbus protocol. We conducted multiple DoS attacks on the above industrial control system using Smod, and the total duration of the attacks was 21 min. The specific modules of Smod and the corresponding presentations are shown in Table 1.

Table 1: DoS attack modules in Smod

Module name	Description
GalilRIO	DoS galil RIO-47100
WriteAllCoils	DoS with write all coils
WriteAllRegister	DoS with write all register function
WriteSingleCoil	DoS with write single coil function
WriteSingleRegister	DoS write single register function

We captured the network traffic of the automatic pumping system during normal operation and when it was under attack. The details of the captured pcap files are shown in Table 2.

Table 2: Details of the captured pcap files

File name (*.pcap)	Description	Number of packets
ResponseInjection1	**Response injection.** The duration is 3 min. The tampering value is fixed.	8,197
ResponseInjection2	**Response injection.** The duration is 20 min. The tampering value is random.	40,298
CommandInjection1	**Command injection.** The duration is 3 min. Tampering with the stable water level.	9,430
CommandInjection2	**Command injection.** The duration is 10 min. Tampering with valve status.	30,773
DoS	**DoS attack.** The duration is 21 min.	526,277
Normal	**No attack.**	171,851
	Total	786,826

3.3 Dataset Generation

As shown in Fig. 3, for comparative experiments, we made three new datasets using the captured raw pcap files.

(1) Data conversion is performed on the original data packets to make a dataset for the deep learning algorithm. This dataset is called dataset A.
(2) Use CICFlowMeter to extract features of Modbus TCP sessions and make a dataset for machine learning algorithms such as decision tree. This dataset is called dataset B.
(3) To compare with the detection algorithm used by Teixeira et al., we use argus-clients to extract features of Modbus TCP sessions to make a dataset. This dataset is called dataset C.

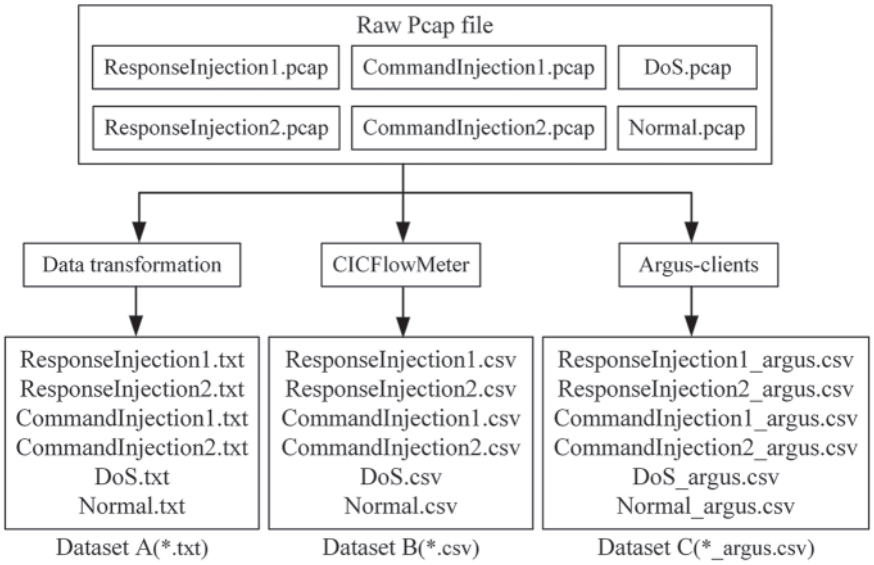

Figure 3: Three new datasets were made with captured raw pcap files

3.3.1 Perform Data Conversion

A Modbus TCP communication usually includes three stages: connection establishment, data transmission, and disconnection. Application layer data only exists in the data transmission phase, not the connection and disconnection phases. These data are called Modbus application Data Units (ADUs). In this paper, the data packets in the data transmission phase are called Modbus-ADU packets. Because response injection and command injection attacks are both tampering with application layer data, these two attacks are both at the application layer level. Therefore, in our dataset, the packets in the connection establishment and disconnection phases in each Modbus TCP communication are discarded, and only the request packet and response packet in the data transmission phase is retained, i.e., only request-response pair (RRP) are retained.

First, the original Modbus-ADU packet is parsed to get its byte sequence. Then the data conversion is performed on the byte sequence of the Modbus-ADU packet, in order to be able to mine its features using deep learning algorithms. Specifically, the data conversion consists of the following three steps:

(1) Sets the address and port number of the Modbus-ADU packet to zero. The model may only need to distinguish malicious samples from normal samples based on IP address, MAC address

or port numbers. To avoid this limitation of the model, the previously mentioned addresses and port numbers should be replaced with zero bytes of equal length. For example, replace the IP address 192.168.1.1 with 0.0.0.0, and replace the MAC address 12:34:56:76:54:32 with 00:00:00:00:00:00.

(2) Unified Modbus-ADU packet size in bytes. Since the deep learning model requires a fixed and uniform sample size, further sample processing is required. By filling zero bytes or truncate, unified the Modbus-ADU packet size for N bytes.

(3) Encode Modbus-ADU packet byte sequence into numeric vector. A byte can be represented as an integer between 0 and 255. Correspondingly, a packet byte sequence can be represented by a numeric vector.

The whole process of data conversion for an original Modbus-ADU packet can be seen in Fig. 4. After data conversion is performed on a pair of request and response Modbus-ADU packets, they can be used as a sample of a deep learning algorithm, named RRP (Request-Response Pairs), with a size of (2, N). After data conversion, dataset A will be formed. The number of samples is shown in Table 3.

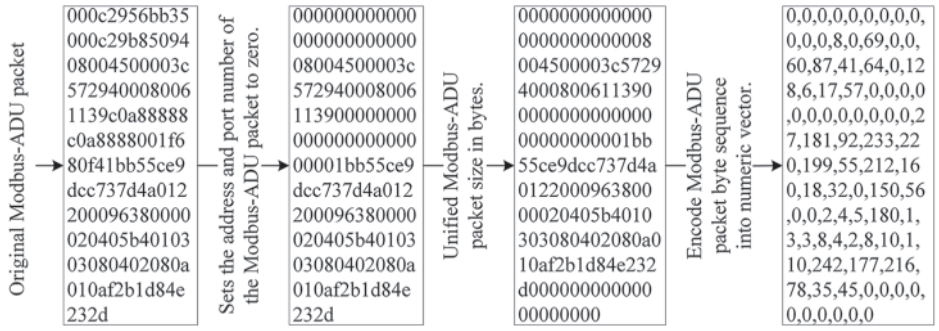

Figure 4: The whole process of data conversion

Table 3: Dataset A (*.txt) and dataset B (*.csv) and dataset C (*_argus.csv)

Dataset name	Normal	Abnormal	Total
ResponseInjection1.txt ResponseInjection1.csv ResponseInjection1_argus.csv	705	114	819
ResponseInjection2.txt ResponseInjection2.csv ResponseInjection2_argus.csv	1803	2222	4025
CommandInjection1.txt CommandInjection1.csv CommandInjection1_argus.csv	862	80	942
CommandInjection2.txt CommandInjection2.csv CommandInjection2_argus.csv	3048	25	3073
DoS.txt	5846	44094	49940
DoS.csv DoS_argus.csv	5847	44095	49942

(Continued)

Table 3: Continued

Dataset name	Normal	Abnormal	Total
Normal.txt Normal.csv Normal_argus.csv	17164	0	17164
Total	29428	46535	75963

3.3.2 Use CICFlowMeter to Extract Features of Network Flow

Using the pcap file as the input file for CICFlowMeter [27,28], the resulting CSV file contains 82-dimensional features. Similarly, to avoid model limitations, it is necessary to remove the IP address and port number features from the CSV file. In this paper, the z-Score method is used to standardize the data. The function is given by Eq. (1). After the data is normalized, dataset B will be formed. The number of samples is shown in Table 3.

$$x^* = \frac{x - \mu}{\sigma} \tag{1}$$

3.3.3 Use Argus-Clients to Extract Features of Network Flow

Using argus-clients [25], 126 statistical features of Modbus TCP sessions can be extracted and saved as CSV files. Actually, in dataset C, each CSV file has only 120 features. In the paper by Teixeira et al., 19 of these features are adopted (1–19 in Table 4). There are some useless features in the 120 features, because whether the samples are abnormal or normal, the values of the useless features are either empty or the same for all samples, which is meaningless for model training. Therefore, to better conduct comparative experiments, we screened the original features and retained 31 of them (1–15 and 20–35 in Table 4). The data were normalized using the z-Score method as before. After the data is normalized, the sample size of the formed dataset C is shown in Table 3.

Table 4: Features in the dataset captured using Argus

Number	Features	Number	Features	Number	Features
1	Mean	13	Rate	25	Max
2	Sport	14	SrcRate	26	sTtl
3	Dport	15	DstRate	27	dTtl
4	TotPkts	16	Loss	28	SrcTCPBase
5	SrcPkts	17	SrcLoss	29	DstTCPBase
6	DstPkts	18	DstLoss	30	TcpRtt
7	TotBytes	19	pLoss	31	SynAck
8	SrcBytes	20	Dur	32	AckDat
9	DstBytes	21	RunTime	33	Offset
10	Load	22	IdleTime	34	sMeanPktSz
11	SrcLoad	23	Sum	35	dMeanPktSz
12	DstLoad	24	Min		

3.4 Dataset Labeling

When labeling samples, an RRP (or flow) should be labeled as anomalous whenever at least one packet in the RRP (or flow) has been maliciously tampered with by the attacker (or sent directly by the attacker).

4 Methodology

This section describes in detail the specific structure of our proposed model RRCNN and the anomaly detection process.

An RRP comprises two Modbus-ADU packets, and a Modbus-ADU packet consists of several bytes. The relationship between RRP, Modbus-ADU packet, and byte is similar to the relationship between article, sentence, and phrase in natural language processing [29]. Inspired by text classification [30], this paper proposes to take the request packet and the corresponding response packet as a whole and use one-dimensional CNN to extract their joint abstract features to design an anomaly detection algorithm, which is mainly for the Modbus TCP protocol.

The model has two steps. First, the feature map of the RRP is calculated using a one-dimensional convolution operation. Then, the second step uses the feature map computed in the first step as the input of the multilayer perceptron (MLP) and then classifies the RRP. The overall structure of RRCNN is shown in Fig. 5.

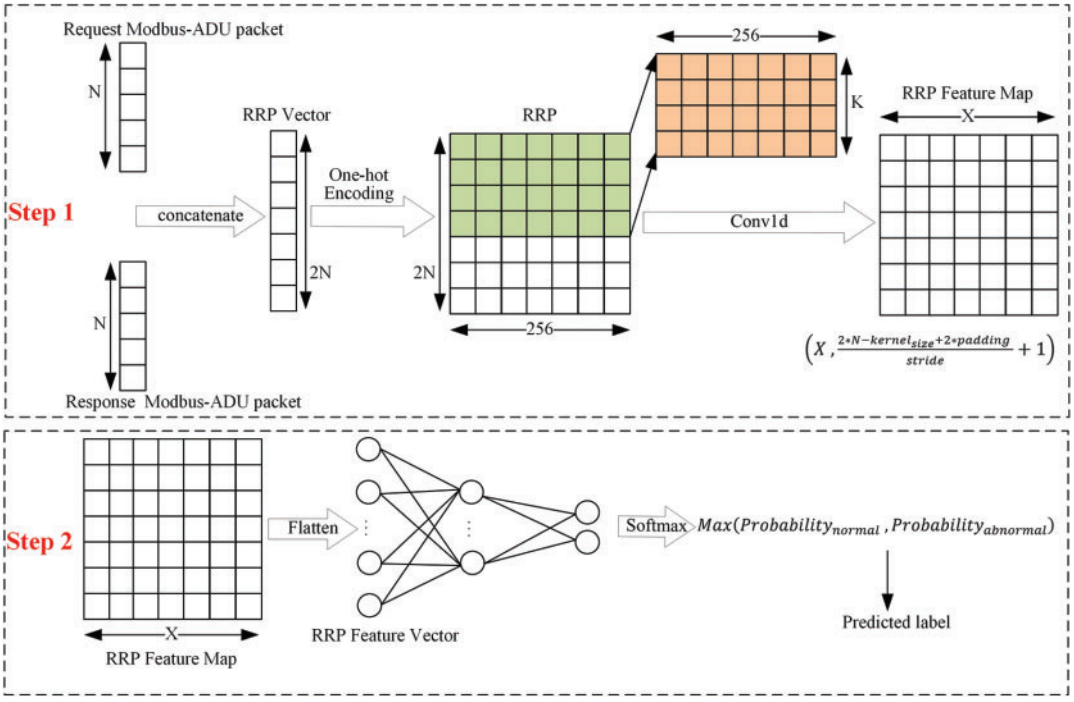

Figure 5: The overall structure of RRCNN

4.1 Determining the Parameter N

Before extracting the features of the RRP, the value of N must be determined first. That is Modbus-ADU packet size in bytes. The length of the Modbus-ADU packet depends on the control command sent by the MTU to the PLC. When the request packet sent by the MTU requires more

addresses to be read, the length of the corresponding response packet is longer. This means that the length of the Modbus-ADU packet depends on the specific industrial control system. As shown in Fig. 6, in the automatic pumping system we built based on the Modbus TCP protocol, the maximum length of the Modbus ADU packet is 83 bytes, and the minimum is 75 bytes. To more fully utilize the information of the raw packet, we set N to 84.

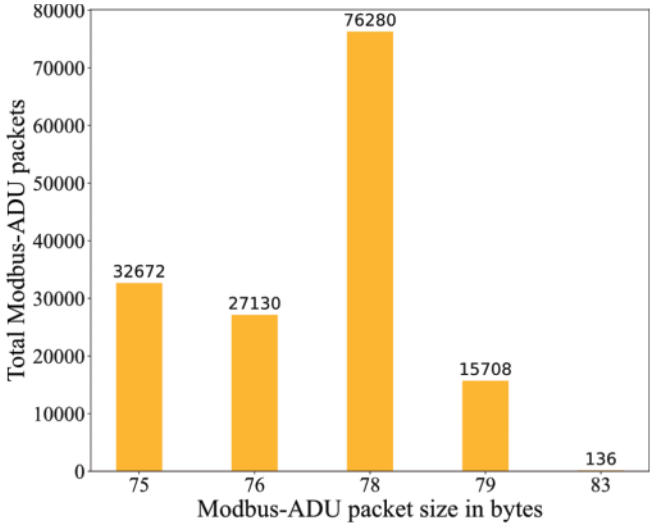

Figure 6: The number of bytes in the packet and the number of packets

4.2 Step 1

As shown in Eqs. (2) and (3), both the request Modbus-ADU packet and the response Modbus-ADU packet can be represented as a sequence of bytes of length N. Concatenating a request Modbus-ADU packet and a response Modbus-ADU packet can form a byte sequence of length 2 N, which is called RRP.

$$Request\ Modbus\ ADU\ Packet = \{byte_1, byte_2, \ldots, byte_N\} \tag{2}$$

$$Response\ Modbus\ ADU\ Packet = \{byte_1, byte_2, \ldots, byte_N\} \tag{3}$$

$$RRP = \{byte_1, byte_2, \ldots, byte_{2*N}\} \tag{4}$$

Because a byte can be encoded as an integer between 0 and 255, the RRP (containing two Modbus-ADU packets) can be represented as a vector of 2 N integers. After one-hot encoding, the RRP can be converted into a matrix of size (256, 2N).

We use one-dimensional convolution to extract features of RRP, using X convolution kernels with a size of (256, K), each convolution kernel can extract K bytes of spatial features. The size of the finally obtained RRP Feature Map was related to specific parameters of the convolution kernel. Specifically, the size of the feature map is given by Eq. (5).

$$RRP\ Feature\ Map = \left(X\ ,\ \frac{2 * N - kernel_{size} + 2 * padding}{stride} + 1\right) \tag{5}$$

4.3 Step 2

Flatten the RRP feature map obtained in step 1, and the RRP feature vector can be obtained. The output layer of MLP has 2 neurons, but the number of neurons in its input layer depends on the length of the RRP feature vector. Softmax is added after the output layer of MLP to indicate normal or abnormal network traffic probability value. Taking RRP feature vector as the input of MLP, the probability of RRP belonging to normal data and the probability of belonging to abnormal data will be output, and the category with the most considerable probability value will be used as the predicted label of the model.

4.4 Parameter Details of RRCNN

The parameter details of RRCNN are shown in Fig. 7. The number of sample (RRP) involved in training the RRCNN model at the same time is Batchsize. When N is set to 84, the RRP is represented as an integer vector of length 168. Batchsize RRPs will be used as input to the model, which is a matrix of size (Batchsize, 168). The RRP feature map is calculated after Block 1, and the size is (Batchsize, 32, 4). In Block 1, three one-dimensional convolution operations and two maximum pooling operations are performed. After each convolution operation, the ReLU is adopted as the activation function. In Block 2, the RRP Feature Map is first flattened into a one-dimensional vector and connected to the MLP. The last layer of MLP has 2 nodes, and after Softmax operation, the probability of the category (normal or abnormal) to which the sample belongs is output.

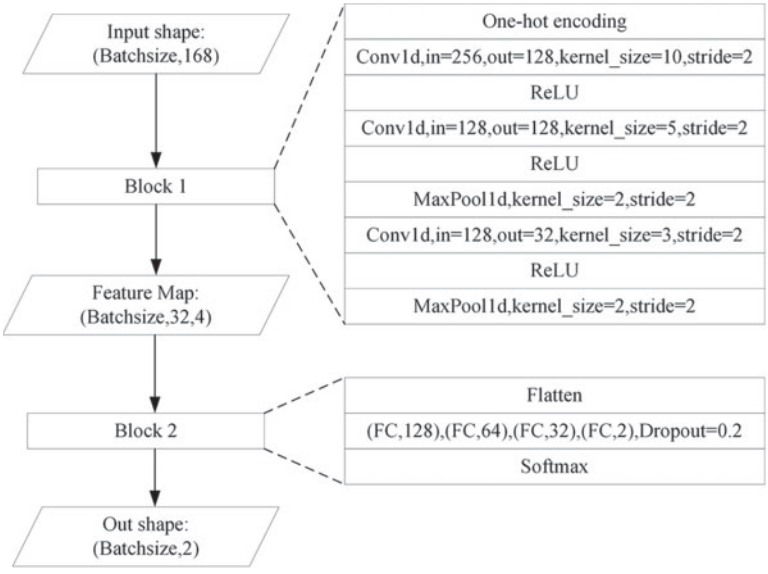

Figure 7: The parameter details of RRCNN

5 Other Anomaly Detection Models Based on Deep Learning

For comparison experiments, we designed a model based on a primary convolutional neural network (hereafter referred to as CNN model), which can also be used to detect anomalous RRPs. In addition, we conducted experiments using the artificial neural networks used by Teixeira et al. [25], including Teixeira_ANN and Teixeira_ANN_31. Specifically, Teixeira_ANN used 19 features, in line with the paper [25], while Teixeira_ANN_31 selected 31 features.

5.1 CNN Model

In the CNN model we designed, N is set to 100. Both the request Modbus-ADU packet and the response Modbus-ADU packet are represented as a matrix of size (10, 10), so RRP can be represented by an image of size (2, 10, 10) containing two channels. The feature map of RRP can be extracted using two two-dimensional convolution operations, the size is (32, 2, 2). After flattening the RRP feature map, a one-dimensional vector with a length of 128 is obtained, which is then used as the input of the multilayer perceptron. The input layer of the multilayer perceptron contains 128 nodes, the hidden layer has 32 nodes, and the output layer has 2 nodes. The probability of the sample category is output after the Softmax operation. The specific structure of the CNN we designed are shown in Fig. 8.

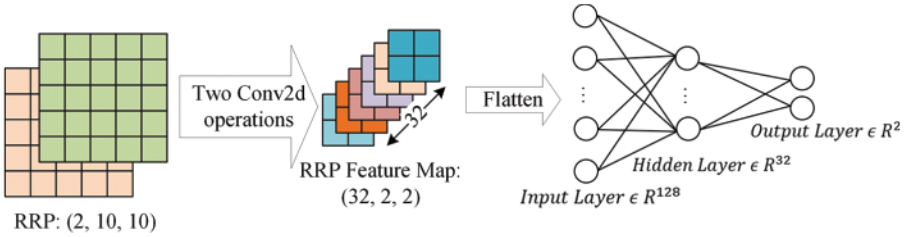

Figure 8: The parameter details of RRCNN

5.2 Teixeira_ANN and Teixeira_ ANN_31

The only difference between Teixeira_ANN and Teixeira_ANN_31 is the number of selected features, i.e., the number of nodes in the input layer of the neural network.

The input layer of Teixeira_ANN has 19 nodes (Teixeira_ANN_31 has 31 nodes), both hidden layers have 9 nodes, and the output layer contains one node. The activation functions used are ReLU and Sigmoid. After the Sigmoid operation, the probability of the category (normal or abnormal) of the RRP will be obtained. The specific structure of Teixeira_ANN and Teixeira_ANN_31 is shown in Fig. 9.

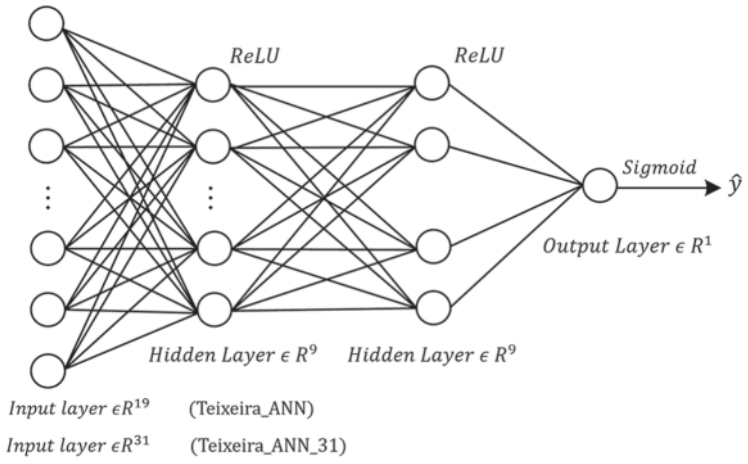

Figure 9: The structure of Teixeira_ANN and Teixeira_ANN_31

6 Experiments and Analysis

6.1 Experiment Environment, Metrics and Dataset

(1) Experiment environment: The experiments were performed using the following hardware and software platforms: Intel (R) Core (TM) i7-10700 CPU, Windows 10 Professional (64 bits), NVIDIA GeForce GTX 2080 Ti, NVIDIA CUDA 11.2.136, Python 3.8.8, Pytorch 1.8.0.

(2) Metrics: The proposed model is evaluated using recall, precision, F1 score, accuracy. TP, TN, FP, FN represent true positive, true negative, false positive and false negative, respectively.

$$Recall = \frac{TP}{TP + FN} \cdot 100\% \tag{6}$$

$$Precision = \frac{TP}{TP + FP} \cdot 100\% \tag{7}$$

$$F1 = 2 \cdot \frac{Recall \cdot Precision}{Recall + Precision} \cdot 100\% \tag{8}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \cdot 100\% \tag{9}$$

(3) Dataset: Take 80% of the dataset as the training set and the rest as the validation set. The division results of datasets A, B and C are shown in Table 5.

Table 5: Training set and validation set (Dataset A, B, C)

Categories	Training set	Validation set	Total
Normal	23542	5886	29428
Abnormal	37228	9307	46535
Total	60770	15193	75963

6.2 Results Analysis

This section presents the performance comparison results between RRCNN and Teixeira_ANN, Teixeira_ANN_31, CNN, and four machine learning algorithms (Decision Tree, SVM, Logistic Regression, KNN) and analyzes the results.

The performance comparison between RRCNN and several other algorithms is shown in Fig. 10. And the detailed results obtained by the above algorithms can be found in Table 6.

Among the four machine learning algorithms, the decision tree algorithm ranks first in terms of comprehensive performance, with various metrics ranging from 0.982 to 0.985. The KNN algorithm achieves a slightly inferior metric to the decision tree algorithm, between 0.973 and 0.981. Although the SVM algorithm achieved high accuracy, it performed the worst in terms of recall. The worst performer was the logistic regression algorithm, which achieved a terrible recall rate. A low recall rate means that there are many false negatives. This means that many abnormal samples are judged to be normal, which is fatal to the anomaly detection system.

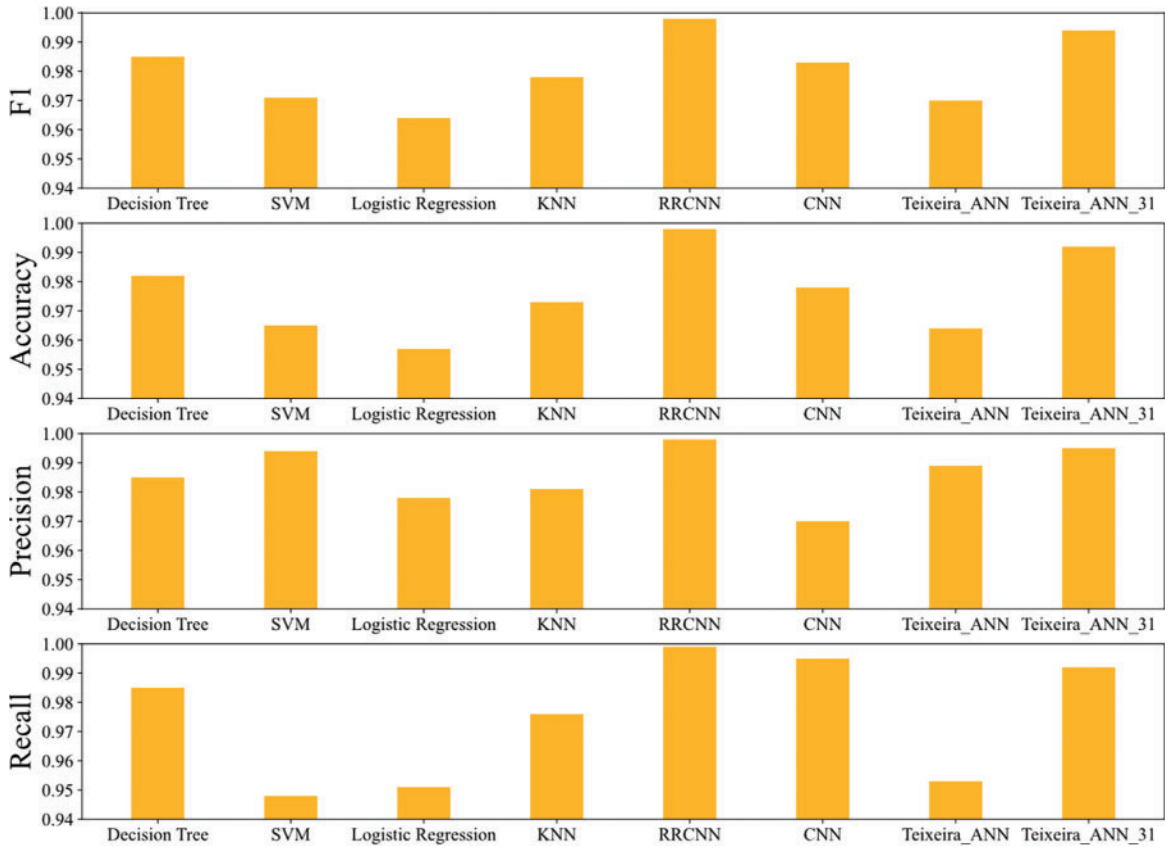

Figure 10: Performance comparison of RRCNN algorithm and other algorithms

Table 6: Performance comparison of RRCNN algorithm and other algorithms

Algorithms	F1	Accuracy	Precision	Recall
Decision Tree	0.985	0.982	0.985	0.985
SVM	0.971	0.965	0.994	0.948
Logistic Regression	0.964	0.957	0.978	0.951
KNN	0.978	0.973	0.981	0.976
CNN	0.983	0.978	0.97	0.995
Teixeira_ANN	0.97	0.964	0.989	0.953
Teixeira_ANN_31	0.994	0.992	0.995	0.992
RRCNN	**0.998**	**0.998**	**0.998**	**0.999**

Among the four deep learning-based algorithms, RRCNN has the best overall performance. As shown in Fig. 10, RRCNN achieves the best results in all metrics. CNN performed poorly in terms of accuracy, but was second only to RRCNN in terms of recall. This indicates that although the false positive rate obtained by CNN is low, the false positive rate is high. There are many normal samples that are considered abnormal. The overall performance of Teixeira_ANN_31 is second only

to RRCNN, with all metrics above 0.99. In addition, the metrics obtained by Teixeira_ANN are all lower than those of Teixeira_ANN_31. This phenomenon should be attributed to the fact that Teixeira_ANN_31 selects more features, while Teixeira_ANN ignores some essential features, such as sMeanPktSz, dMeanPktSz, etc. For each of the above deep learning-based algorithms, we trained 40 epochs. After completing one training with the training set, the model was tested with the validation set. The changes of loss during training and accuracy during validation are shown in Fig. 11.

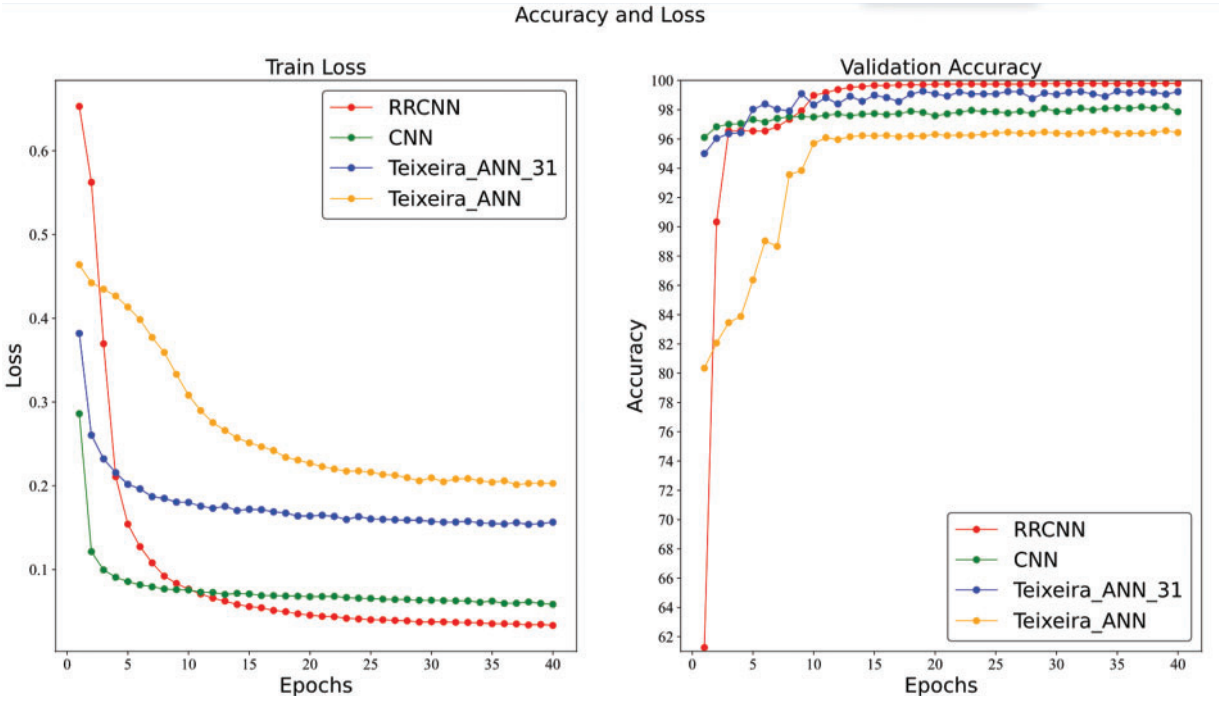

Figure 11: The loss and accuracy of four deep learning algorithms

As shown in Fig. 11, after the RRCNN algorithm completes the first training, the loss value of the model is large, and the accuracy rate on the validation set is the lowest among all algorithms. Regarding the accuracy of the validation set, while the other three deep learning-based algorithms can achieve higher results after the first training, the RRCNN can achieve higher accuracy after just 15 rounds of training.

7 Conclusions and Future Work

In this work, we built an industrial control system in the laboratory and collected a traffic dataset containing multiple network attack types. This dataset can be used to evaluate ICS network traffic anomaly detection methods. Meanwhile, this paper proposes a one-dimensional convolutional neural network-based anomaly detection method for ICS network traffic, which can make full use of the raw data of network traffic. The experimental results show that our proposed method has better performance than the conventional studies. Specifically, the F1 score, accuracy rate, precision rate and recall rate exceeded 99%.

The ICS presented in this work uses the Modbus TCP protocol, which is one of the most widely used protocols in industry worldwide. However, other protocols are also used in industry, such as S7

and DNP3. Our future research work is to build an ICS with multiple communication protocols and use more diverse attacks when acquiring data. In addition, we plan to study an ICS network anomaly detection system with better performance and higher compatibility.

Funding Statement: This work is supported by the National Natural Science Foundation of China (No.62076042, No. 62102049), the Key Research and Development Project of Sichuan Province (No.2021YFSY0012, No. 2020YFG0307, No.2021YFG0332), the Science and Technology Innovation Project of Sichuan (No. 2020017), the Key Research and Development Project of Chengdu (No. 2019-YF05-02028-GX), the Innovation Team of Quantum Security Communication of Sichuan Province (No.17TD0009), the Academic and Technical Leaders Training Funding Support Projects of Sichuan Province (No. 2016120080102643).

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

References

[1] J. Ling, Z. Zhu, Y. Luo and H. Wang, "An intrusion detection method for industrial control systems based on bidirectional simple recurrent unit," *Computers & Electrical Engineering*, vol. 91, no. 107049, 2021.

[2] Q. Wang, H. Chen, Y. Li and B. Vucetic, "Recent advances in machine learning-based anomaly detection for industrial control networks," in *Proc. 1st Int. Conf. on Industrial Artificial Intelligence*, Shenyang, Liaoning, China, pp. 1–6, 2019.

[3] D. Dzung, M. Naedele, T. P. Von Hoff and M. Crevatin, "Security for industrial communication systems," *Proc. of the IEEE*, vol. 93, no. 6, pp. 1152–1177, 2005.

[4] E. Sisinni, A. Saifullah, S. Han, U. Jennehag and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[5] C. Feng, T. T. Li and D. Chana, "Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks," in *Proc. 47th Annual IEEE/IFIP Int. Conf. on Dependable Systems and Networks*, Denver, CO, USA, pp. 261–272, 2017.

[6] X. Pan, Z. Wang and Y. Sun, "Review of PLC security issues in industrial control system," *Journal of Cybersecurity*, vol. 2, no. 2, pp. 69–83, 2020.

[7] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[8] J. Liu, W. Zhang, T. Ma, Z. Tang, Y. Xie *et al.,* "Toward security monitoring of industrial cyber-physical systems via hierarchically distributed intrusion detection," *Expert Systems with Applications*, vol. 158, no. 113578, 2020.

[9] R. Khan, P. Maynard, K. McLaughlin, D. Laverty and S. Sezer, "Threat analysis of blackenergy malware for synchrophasor based real-time control and monitoring in smart grid," in *Proc. 4th Int. Symp. for ICS & SCADA Cyber Security Research*, Queen's University Belfast, UK, pp. 53–63, 2016.

[10] T. Alladi, V. Chamola and S. Zeadally, "Industrial control systems: Cyberattack trends and countermeasures," *Computer Communications*, vol. 155, pp. 1–8, 2020.

[11] N. Z. Jhanjhi, M. Humayun and S. N. Almuayqil, "Cyber security and privacy issues in industrial internet of things," *Computer Systems Science and Engineering*, vol. 37, no. 3, pp. 361–380, 2021.

[12] Z. K. Maseer, R. Yusof, S. A. Mostafa, N. Bahaman, O. Musa *et al.,* "Deepiot. IDS: Hybrid deep learning for enhancing IoT network intrusion detection," *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3945–3966, 2021.

[13] A. H. Mohammad, "Intrusion detection using a new hybrid feature selection model," *Intelligent Automation & Soft Computing*, vol. 30, no. 1, pp. 65–80, 2021.

[14] W. L. Shang, S. S. Zhang and M. Wan, "Modbus/TCP communication anomaly detection based on PSO-SVM," *Applied Mechanics and Materials*, vol. 490, pp. 1745–1753, 2014.
[15] T. Morris, R. Vaughn and Y. Dandass, "A retrofit network intrusion detection system for MODBUS RTU and ASCII industrial control systems," in *Proc. 45th Hawaii Int. Conf. on System Sciences*, Maui, Hawaii USA, pp. 2338–2345, 2012.
[16] Y. Y. Huang, J. Z. Lu, H. Z. Tang and X. L. Liu, "A hybrid association rule-based method to detect and classify botnets," *Security and Communication Networks*, vol. 2021, no. 1028878, pp. 1–9, 2021.
[17] D. Kwon, H. Kim and J. Kim, S. C. Suh, I. Kim *et al.,* "A survey of deep learning-based network anomaly detection," *Cluster Computing*, vol. 22, no. 1, pp. 949–961, 2019.
[18] S. Ponomarev and T. Atkison, "Industrial control system network intrusion detection by telemetry analysis," *Transactions on Dependable and Secure Computing*, vol. 13, no. 2, pp. 252–260, 2011.
[19] W. Liang, K. Li, J. Long, X. Kui and A. Y. Zomaya, "An industrial network intrusion detection algorithm based on multifeature data clustering optimization model," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2020.
[20] Y. Y. Huang, D. Wang, Y. Sun and B. Hang, "A fast intra coding algorithm for HEVC by jointly utilizing naive Bayesian and SVM," *Multimedia Tools and Applications*, vol. 79, no. 45, pp. 33957–33971, 2020.
[21] H. Yang, L. Cheng and M. C. Chuah, "Deep-learning-based network intrusion detection for SCADA systems," in *Proc. IEEE Conf. on Communications and Network Security*, Washington, D.C., USA, pp. 1–7, 2019.
[22] K. Wang, "Network data management model based on naïve Bayes classifier and deep neural networks in heterogeneous wireless networks," *Computers & Electrical Engineering*, vol. 75, pp. 135–145, 2019.
[23] M. Kravchik and A. Shabtai, "Detecting cyber attacks in industrial control systems using convolutional neural networks," in *Proc. 2018 Workshop on Cyber-Physical Systems Security and Privacy*, Toronto, ON, Canada, pp. 72–83, 2018.
[24] X. Hao, J. Zhou, X. Shen and Y. Yang, "A novel intrusion detection algorithm based on long short term memory network," *Journal of Quantum Computing*, vol. 2, no. 2, pp. 97, 2020.
[25] M. A. Teixeira, M. Zolanvari, K. M. Khan, R. Jain and N. Meskin, "Flow-based intrusion detection algorithm for supervisory control and data acquisition systems: A real-time approach," *IET Cyber-Physical Systems: Theory & Applications*, vol. 6, no. 3, pp. 178–191, 2021.
[26] A. Lemay and J. M. Fernandez, "Providing {SCADA} network data sets for intrusion detection research," in *Proc. 9th Workshop on Cyber Security Experimentation and Test*, Austin, TX, United States, 2016.
[27] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proc. ICISSP*, Italy, pp. 407–414, 2016.
[28] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *Proc. ICISSP*, Porto, Portugal, pp. 253–262, 2017.
[29] L. Yu, J. Dong and L. Chen, "PBCNN: Packet bytes-based convolutional neural network for network intrusion detection," *Computer Networks*, vol. 194, no. 108117, 2021.
[30] J. Cai, J. Li, W. Li and J. Wang, "Deeplearning model used in text classification," in *Proc. 15th International Computer Conference on Wavelet Active Media Technology and Information Processing*, Chengdu, China, pp. 123–126, 2018.