applied sciences

MDPI

Article

Intrusion Detection for Industrial Control Systems Based on Improved Contrastive Learning SimCLR

Chengcheng Li [1,2,3,4], Fei Li [5,*], Liyan Zhang [6], Aimin Yang [1,2,3,4,7], Zhibin Hu [1,2,3,4] and Ming He [1,2,3,7]

1 Hebei Key Laboratory of Data Science and Application, North China University of Science and Technology, Tangshan 063210, China; aimin@ncst.edu.cn (A.Y.)
2 The Key Laboratory of Engineering Computing in Tangshan City, Tangshan 063210, China
3 College of Science, North China University of Science and Technology, Tangshan 063210, China
4 Hebei Engineering Research Center for the Intelligentization of Iron Ore Optimization and Ironmaking Raw Materials Preparation Processes, North China University of Science and Technology, Tangshan 063210, China
5 Shanxi Jianlong Industrial Co., Ltd., Yuncheng 044000, China
6 College of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
7 Tangshan Intelligent Industry and Image Processing Technology Innovation Center, North China University of Science and Technology, Tangshan 063210, China
* Correspondence: lifei@ejianlong.com

Abstract: Since supervised learning intrusion detection models rely on manually labeled data, the process often requires a lot of time and effort. To make full use of unlabeled network traffic data and improve intrusion detection, this paper proposes an intrusion detection method for industrial control systems based on improved comparative learning SimCLR. Firstly, a feature extraction network is trained on SimCLR using unlabeled data; a linear classification layer is added to the trained feature extraction network model; and a small amount of labeled data is used for supervised training and fine-tuning of the model parameters. The trained model is simulated on the Secure Water Treatment (SWaT) dataset and the publicly available industrial control dataset from Mississippi State University, and the results show that the method has better results in all evaluation metrics compared with the deep learning algorithm using supervised learning directly, and the comparative learning has research value in industrial control system intrusion detection.

Keywords: contrast learning; residual networks; industrial control systems; intrusion detection

Citation: Li, C.; Li, F.; Zhang, L.; Yang, A.; Hu, Z.; He, M. Intrusion Detection for Industrial Control Systems Based on Improved Contrastive Learning SimCLR. *Appl. Sci.* **2023**, *13*, 9227. https://doi.org/10.3390/app13169227

Academic Editor: Vincent A. Cicirello

Received: 12 June 2023
Revised: 1 August 2023
Accepted: 8 August 2023
Published: 14 August 2023

1. Introduction

The Industrial Control System (ICS) serves as the backbone of the Industrial Internet, which in turn is the product of the combination of new-generation technologies such as communication and automation with traditional industrial networks in the 21st century [1]. Industrial control systems are an important part of industrial development, controlling the operation of different working systems in different industries and building a digital, networked industrial chain through the connection of field devices, operators, and other facilities. It is an important cornerstone of Industry 4.0 [2]. Network security has always been a key concern, and the security of industrial control systems needs to be given the same high priority [3]. The Purdue model, which is now the reference standard for industrial control system security, demonstrates the interdependence of all the components of a typical ICS and is an important reference point for starting to build a typical modern ICS architecture. The Purdue model is shown in Figure 1. The intrusion detection system is a very important part of the computer network system; its purpose is to collect key useful information from different parts of the network system and, by analyzing the collected data, determine whether there are insecure behaviors in the current network system that cause damage to the network. Compared with traditional network defense mechanisms such as firewalls, VNPs, access control, etc., network intrusion detection systems can detect some unknown means of attack [4,5]. At the same time, intrusion detection systems can

detect attacks without affecting network performance and achieve real-time protection of network systems [6,7].

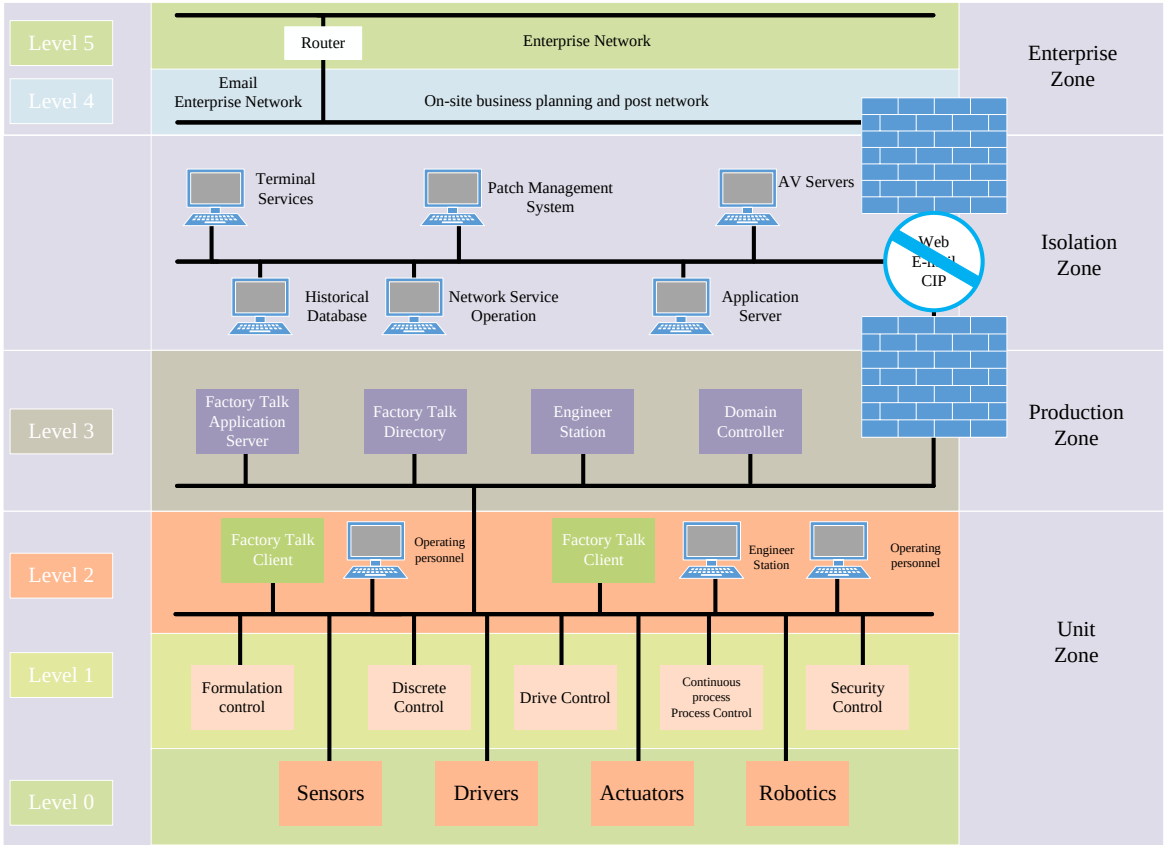

Figure 1. Purdue Model.

Intrusion detection systems commonly use two analysis methods, anomaly detection and misuse detection, to detect and analyze abnormal behavior. As deep learning technology advances, progresses, and evolves, people are gradually combining deep learning algorithms and industrial control system intrusion detection to find more suitable methods to protect the system.

Compared with traditional machine learning algorithms, deep learning can mine more advanced and important features in massive data. Akashdeep Bhardwaj et al. [8] introduced a pattern recognition algorithm named "Capture the Invisible (CTI)". Used to find hidden processes in industrial control device logs and detect behavior-based attacks executed in real-time. Yongle Chen et al. [9] developed a method to improve the information transfer link in adversarial domain adaptation (DA). This approach is capable of training the anomaly detection depth using unbalanced data. Experiments have high detection accuracy on SCADA network layer-based data. Khan, M.A. [10] creating a deep learning-based hybrid intrusion detection framework by convolutional recurrent neural network (CRNN) using convolutional neural network (CNN) to capture local features and recurrent neural network (RNN) to capture temporal features to improve the performance of the intrusion detection system, the effectiveness of the model is validated on the CSE-CIC-DS2018 dataset. Yan Hu et al. [11] proposed a new alignment entropy-based method to detect stealth attacks on ICSs, by which the non-randomness contained in the residuals can be characterized, thus effectively distinguishing the residuals from random sequences during stealth attacks. The experiments were synthesized in the Matlab-Simulink environment, and the results verified excellent detection capabilities. Jie Ling et al. [12] proposed an intrusion detection method based on bi-directional simple recursive units (BiSRU). It was also validated on two standard industrial datasets at Mississippi State University, and

the results showed that the proposed method is more accurate and requires less training time. Chao Wang et al. [13] proposed a self-encoder-based intrusion detection method for industrial control systems that simultaneously predicts and reconstructs the input data, thus overcoming the drawback of using each data individually. Using the errors obtained from the model, a rate of change is proposed to locate the most likely suspect devices under attack. Experiments were conducted on the SWaT dataset to verify the high validity of the method.

The above research is based on the deep learning method of supervised learning; however, supervised learning requires labeled data to train the model, so a large amount of data is needed to support the training of the model [14]. In industrial control systems, communication data is the object of study, and normal or abnormal traffic occurs in the process of system operation, in which there may be uneven data distribution and unknown data traffic. Most of these data need to be manually labeled to distinguish, and if these unlabeled data cannot be used well, it will bring trouble to the intrusion detection system. Contrast learning belongs to self-supervised learning, which in turn is the category of unsupervised learning [15]. Contrast learning enables the use of unlabeled data to assist in training a feature extraction network and improve the accuracy of subsequent classification detection tasks. The classical algorithms for contrast learning include SimCLR, Momentum Contrast for Unsupervised Visual Representation Learning (MoCo), Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning (BYOL), Exploring Simple Siamese Representation Learning (SimSiam), Swapping Assignments between Multiple Views of the Same Image (SWaV) [16–20], etc.

In this paper, we propose an intrusion detection model for industrial control systems based on the improved comparative learning model SimCLR by combining the improved comparative learning model SimCLR with industrial control system intrusion detection. Firstly, we need to perform data pre-processing on the obtained industrial control traffic data, use normalization to map the data to the 0-1 interval to eliminate the adverse effects caused by odd sample data, and use the principal component analysis (PCA) algorithm for dimensionality reduction on the normalized data, which can effectively reduce the data dimensionality, eliminate redundant information, and improve the efficiency of data processing and data quality. Unlike the supervised model, the contrast learning model does not require the use of labeled data to train the network model. It is a self-supervised learning method that allows the model to learn which data points are similar or which data points are different to learn the general characteristics of the data set without the data being labeled. The contrast learning model has four main phases: data augmentation, feature extraction, feature projection, and the calculation of contrast loss. The trained contrast learning feature extraction network is transferred to the supervised learning training, and the model is fine-tuned by using only 10% of the labeled data in the simulated experimental dataset. Finally, the trained model is tested on the test data.

The innovative points of this paper are as follows:

(1) An intrusion detection model for industrial control systems based on improved comparative learning SimCLR is proposed, and the data enhancement is improved by adding random noise, sequence inversion, and random sampling of the Synthetic Minority Over-sampling Technique (SMOTE) algorithm to the original industrial control traffic data. The other one uses only the SMOTE algorithm. The other one uses only the SMOTE algorithm to replace the original data with the same multiplicity of sampling.

(2) The asymmetric network structure is adopted on top of the original model, which enables different networks to perform feature extraction for different types of data.

(3) The feature projection structure is improved by using feature cross-fusion to cross-fuse two feature vectors and using a jump join between the first and last linear layer to add the two vectors before and after the projection, which increases the similarity between positive and negative examples and makes the similarity between positive and negative examples more distant.

The article is structured as follows: Section 2 introduces the SMOTE algorithm and the deep residual network, ResNet. Section 3 presents the proposed intrusion detection model based on contrast learning for industrial control systems. Section 4 presents the datasets used in the experiments, the model evaluation metrics, and the experimental results. Section 5 summarizes the work of this paper and provides an outlook for the future.

2. Theoretical Basis

2.1. SMOTE Algorithm

The SMOTE algorithm is a commonly used oversampling technique, using which the number of some specified samples can be increased to solve the problems caused by the raw data in the experiment, thus achieving the purpose of data enhancement [21]. The steps of its implementation are shown below:

Step 1. A sample x is selected from a small number of classes of samples; the number of k-nearest neighbors is set; and this sample is compared with its k-nearest neighbor samples by using the Euclidean distance calculation method.

Step 2. A sample $\widetilde{x}$ is randomly selected from among the obtained k-nearest neighbor samples.

Step 3. A randomly selected point between the first selected minority class sample x and the k-nearest neighbor selected sample $\widetilde{x}$ is the newly generated

The calculation is shown in Equation (1):

$$x_{\text{new}} = x + \text{rand}(0,1) \times (\tilde{x} - x) \tag{1}$$

where x_{new} is the sample newly generated by sampling, x is the original sample, $\text{rand}(0,1)$ is the randomly generated number between 0 and 1 , and $\tilde{x}$ is the k-nearest neighbor sample calculated with Euclidean distance.

2.2. Residual Network

In the early days, it was widely believed that as the depth of a neural network increased, the performance of the network got better. For example, in the earlier deep learning model VGG, the number of network layers was 19, but in the later emergence of the network model GoogleNet, the network depth reached 22 layers. Later, as the depth gradually increases, problems such as model overfitting and gradient disappearance and explosion appear, and the experimental results are often not as good as the shallow layer of the network. In 2015, four scholars from Microsoft Asia Research proposed a new neural depth network model, the ResNet residual network [22], which solved the problem of model degradation due to the increase in network depth. The residual network is composed of one residual block, and each residual block can be connected with each other by using jump connections. The architecture of the deep residual network is illustrated in the accompanying Figure 2:

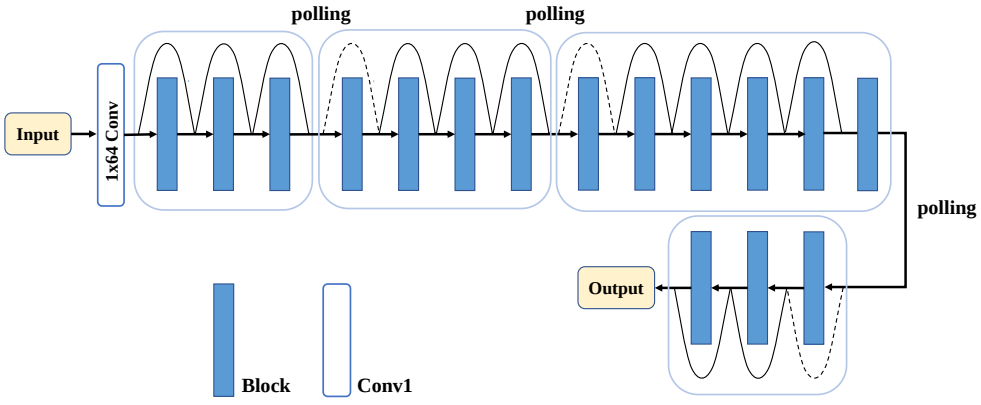

Figure 2. Deep residual network.

The structure of each residual block is shown in Figure 3:

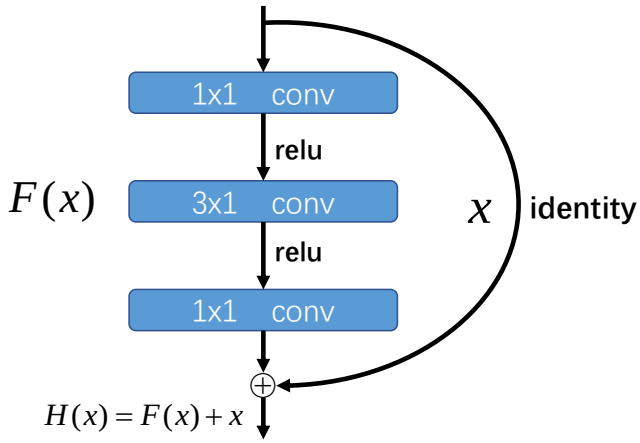

Figure 3. Residuals Module.

The output of a residual block can be calculated using in Equation (2):

$$H(x) = F(x) + x \tag{2}$$

where x is the input term of the residual network, $F(x)$ denotes the residual term, and $H(x)$ is the output. When the residual term $F(x)$ is 0 , the output result is equal to the input x, which constitutes a constant mapping $H(x) = x$.

Conv is the convolution operation, and the calculation procedure is shown in Equation (3):

$$y_c = \sum_{L}^{l=0} \sum_{M}^{m=0} w_{l,m} \cdot x_{l,m} + b \tag{3}$$

where $x_{l,m}$ is the input matrix; $w_{l,m}$ is the parameter of the convolution kernel; l and m are the width and height of the input matrix and the convolution kernel; b is the bias vector; and y_c is the output of the convolution.

2.3. LayerNorm

LayerNorm is a normalization process. It normalizes different time steps in one-dimensional data, which largely avoids the gradient disappearance and explosion problem, and the hidden state transmission is more stable. The layerNorm calculation process is as follows.

(1) Calculate the expectation μ and standard deviation σ for each stratum. The calculation is shown in Equations (4) and (5):

$$\mu^l = \frac{1}{H} \sum_{i=1}^{H} a_j^l \tag{4}$$

$$\sigma^l = \sqrt{\frac{1}{H} \sum_{i=1}^{H} \left(a_i^H - \mu^l\right)^2} \tag{5}$$

where l denotes the l th hidden layer, H denotes the number of nodes in that layer, and a denotes the value of a particular node before activation.

(2) The standardized calculation is shown in Equation (6):

$$a_i^{-l} = \frac{g^l}{\sigma^l} \cdot \left(a_i^l - \mu^l\right) + b \tag{6}$$

where g and b denote the gain and bias parameters, respectively, which can be included in the training samples to be trained together.

(3) The output is obtained by adding the activation function, as shown in Equation (7):

$$h = \text{Relu}\left(a_i^{-l}\right) \tag{7}$$

where Relu is the activation function and the mathematical expression is shown in Equation (8):

$$\text{Relu}(x) = \begin{cases} x, x > 0 \\ 0, x \leq 0 \end{cases} \tag{8}$$

3. Contrastive Learning-Based Intrusion Detection Model for Industrial Control Systems

We improved the original comparative learning model SimCLR to fit our industrial control system intrusion detection traffic data, and the differences between the two are shown in Table 1.

Table 1. Improved comparative learning based on the difference between the intrusion detection model and the original model for industrial control systems.

Model	Data Type	Data Enhancement Methods	Feature Extraction Network	Feature Projection	Loss Function
Our Model	One-dimensional sequences	Add Gaussian noise, sequence inversion, SMOTE sampling	ResNet	LayerNorm	InfoNCE
Original Models	Two-dimensional images	Random cropping, random color transformation, etc.	ResNet	BatchNorm	InfoNCE

3.1. Data Pre-Processing

3.1.1. Normalization

The data used in this experiment contains some null values and some infinite data, for which the processing method is to delete the rows containing these data. After the outliers are processed, normalization is also required. The data normalization process limits the data to a range, which can eliminate the adverse effects caused by odd sample data, speed up the training of the model, and improve the accuracy of the model in some models. The calculation of data normalization is shown in Equation (9).

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{9}$$

where x' is the normalized data, $x_{\max}$ and $x_{\min}$ are the maximum and minimum values of the same feature attribute in all data, respectively.

3.1.2. PCA Downscaling

Principal Components Analysis (PCA) is often used for data dimensionality reduction. By using PCA data dimensionality reduction, the data is reduced to lower dimensions, which can reduce the storage space needed for the data, save time for model training, remove redundant attributes, and improve the accuracy of detection.

3.2. Training Contrast Learning Models

3.2.1. Data Enhancement

Data enhancement plays a crucial role in contrast learning, and the use of different data enhancement methods can have different effects on feature extraction. Common data enhancement methods in the image domain include image flipping and rotation, cropping and scaling, color dithering, etc. When dealing with sequence data, common data enhancement methods include dimensional flipping, scaling, window adjustment, window

cropping, adding noise, etc. And with the development of deep learning, the use of deep learning algorithms for data enhancement has become more and more widespread, including image enhancement using generative adversarial networks and image enhancement using self-encoders.

In this paper, a new data enhancement approach is proposed for the acquired data of industrial control intrusion detection. Given the original input set $X = \{x1, x2, x3, \ldots, xn\}$ and n denotes the number of samples, the SMOTE algorithm is used to sample the acquired data using the same multiplicity to get $X1 = \{x_1^1, x_1^2, x_1^3, \ldots, x_1^n\}$. Also add Gaussian noise to the data to get the new data $X2 = \{x_2^1, x_2^2, x_2^3, \ldots, x_2^n\}$, and use the sequence inversion method to invert the sequence of each data xn in the original data to get the new data set $X3 = \{x_3^1, x_3^2, x_3^3, \ldots, x_3^n\}$, and fuse the same in the three newly generated data sets The final enhanced data $X_{new} = \{[x_1^1, x_2^1, x_3^1], [x_1^2, x_2^2, x_3^2], [x_1^3, x_2^3, x_3^3], \ldots, [x_1^n, x_2^n, x_3^n]\}$ is obtained. where $X1$ and X_{new} are used as inputs to the two feature extraction networks for contrast learning.

3.2.2. Feature Extraction

Following data augmentation, the next step involves feature extraction from the augmented data, for which a deep residual network is employed. To account for the different channels in the two sets of data post-feature enhancement, an asymmetric feature extraction structure is utilized. Specifically, a one-dimensional deep residual network with input channel 1 is used to extract features from the data after SMOTE sampling, while a one-dimensional deep residual network with input channel 3 is employed to extract features from the data after three rounds of feature enhancement and channel fusion.

This experiment uses a one-dimensional deep residual network, ResNet50, to extract features from the enhanced data, and the network structure consists of a Conv1 and four Block_Layer layers, each with 3, 4, 6, and 3 blocks, respectively, and finally connected to the feature mapping layer using full connectivity.

3.2.3. Feature Projection

The SimCLR model adds a feature projection layer after feature extraction and then does a nonlinear transformation to reduce the loss of feature information, and this structure has been widely used in the subsequent comparison learning. After accepting the vector after feature extraction, this structure first does a linear transformation, followed by batch normalization using BatchNorm, nonlinear mapping using the activation function, and finally another linear transformation, as shown in Figure 4:

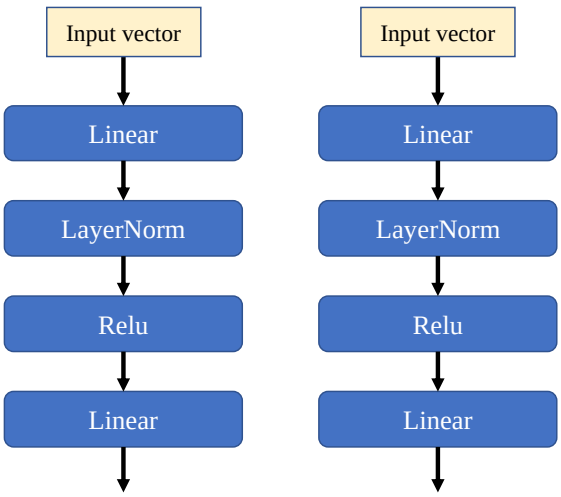

Figure 4. Feature projection structure.

In this experiment, by improving the original feature projection structure, the two vectors g_1 and g_2 after feature extraction is added by using the cross-fusion jump connec-

tion, and the vector after the summation is used as the new input vector of the feature projection layer, while this vector is added to the output of the last layer, which makes it possible to guarantee the loss of important features when calculating the contrast loss. Since BatchNorm calculates normalized statistics according to the number of samples and LayerNorm is used to normalize different time steps in one-dimensional data, the gradient disappearance and explosion problem is largely avoided, and the transmission of hidden states is more stable. The structure of the improved feature projection layer is shown in Figure 5:

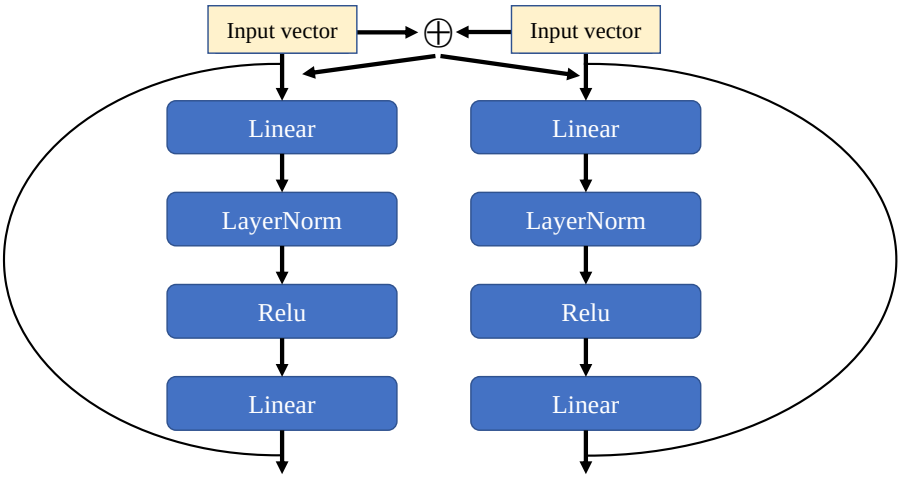

Figure 5. Improved feature projection network.

3.2.4. Contrast Loss Function

The contrast loss function is a loss function with the self-discovery property of difficult negative samples, which is essential for learning high-quality self-supervised representations, and a loss function without this property can significantly deteriorate the performance of self-supervised learning.

The loss function used is InfoNCE Loss, which is calculated as shown in Equation (10):

$$L_i = -\log\left(\exp\left(S\left(z_i, z_i^+\right)/\tau\right)/\sum_{j=0}^{K}\exp\left(S\left(z_i, z_j\right)/\tau\right)\right) \tag{10}$$

where $S\left(z_i, z_i^+\right)$denotes the similarity of the feature vectors of the same sample after data enhancement, that is, the degree of similarity between positive examples. $S\left(z_i, z_j\right)$ denotes the similarity of the feature vectors of one data-enhanced sample to the feature vectors of other enhanced samples. τ denotes the temperature coefficient, which is a hyperparameter that controls how well the model discriminates between negative samples. A small temperature coefficient will make samples that are more similar to this sample separate, that is, be able to be classified more evenly. S is the calculation of similarity, which is usually calculated by using the dot product operation after the L2 norm or by using cosine similarity. The calculation formula is shown in Equation (11):

$$S\left(z_i, z_i^+\right) = z_i^T z_i^+ / \left(\|z_i\|_2 \left\|z_i^+\right\|_2\right) \tag{11}$$

3.3. Supervised Training of Fine-Tuned Models

After training the contrast learning model using unlabeled data, the feature extraction network is obtained and retrained using supervised learning to fine-tune the network parameters. This is completed by using labeled data from the dataset, feeding the data into the feature extraction network for feature extraction, and then adding a linear classification layer to classify the data, i.e., to distinguish between normal data and attack data. Thus,

a comparative learning-based intrusion detection model for industrial control systems is constructed.

The flowchart of the intrusion detection model for industrial control systems based on comparative learning SimCLR is shown in Figure 6. The symbols that appear in the model and their descriptions are shown in Table 2.

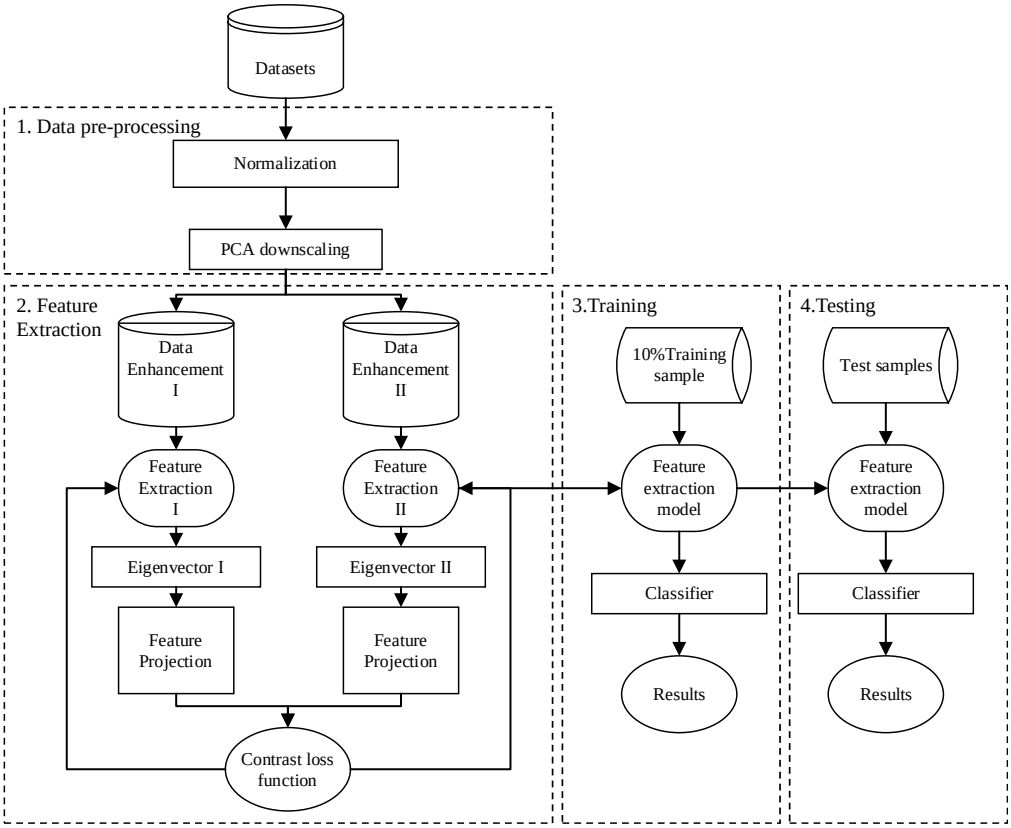

Figure 6. Overall process of the model.

Table 2. Symbols used in the model.

Symbols	Description
x'	Data after normalization
x	Un-normalized data
x_{min}	The minimum value in a piece of data
x_{max}	The maximum value in a piece of data
X	Raw data
$X1$	Data after SMOTE algorithm enhancement
$X2$	Add Gaussian noise to the data
$X3$	Data after sequence inversion
X_{new}	New data after merging
g_1	Feature extraction vector 1
g_2	Feature extraction vector 2
L_i	Contrast loss values
log	Logarithmic operations
exp	Exponential arithmetic
S	Calculate similarity
z_i	Data without data enhancement
z_i^+	Positive sample after data enhancement
K	Number of samples
z_j	Negative samples after data enhancement
τ	Temperature coefficient: used to adjust the discrimination of difficult negative samples

4. Experiments and Results Analysis

The experiments divide the data into unlabeled data for training the feature extraction network in the contrast learning model and labeled training and test data for supervised training of the fine-tuned model. Firstly, comparison experiments between different temperature coefficients are conducted to select the best temperature coefficients for contrast learning. The effectiveness of this method is verified by comparing the experimental results between different data enhancement methods and this method. A contrast learning feature extraction network trained with unlabeled data is added with a linear classification layer, and a small amount of labeled training data is used to compare the classification results with other deep learning models through model fine-tuning to verify the effectiveness of this method. Finally, the SWaT dataset was replaced and tested using the present model to further validate the applicability of the model.

4.1. Experimental Data Set

The intrusion detection dataset for the industrial control system used in this experiment was obtained from Mississippi State University. The researchers examined and captured the natural gas pipeline control system traffic data through a network data logger and obtained 97,019 experimental datasets. The data contains pre-processed network transaction data with the underlying transport data (TCP, MAC, etc.) removed. Each data entry contains 26 traffic attributes and one attack category, where the attack category has seven attack types and one normal type. The attack types and label descriptions are shown in Table 3. The distribution of sample size is shown in Figure 7.

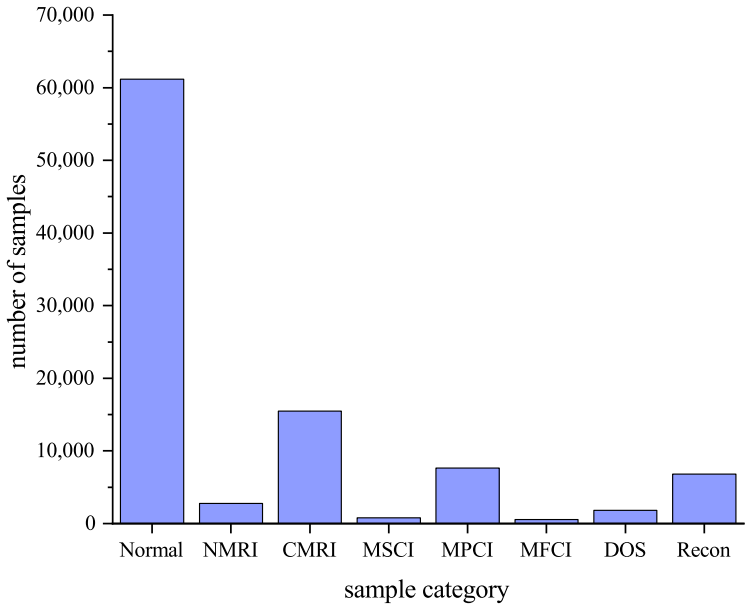

Figure 7. Distribution of the number of each category in the dataset.

Table 3. Data types in the dataset.

Type of Attacks	Abbreviation
Normal	Normal
Naïve Malicious Response Injection	NMRI
Complex Malicious Response Injection	CMRI
Malicious State Command Injection	MSCI
Malicious Parameter Command Injection	MPCI
Malicious Function Code Injection	MFCI
Denial of Service	DoS
Reconnaissance	Recon

4.2. Model Evaluation Metrics

In this experiment, four metrics were adopted to evaluate the model: Accuracy, Precision, Recall, and F1, in which several basic concepts need to be introduced:

TP (True Positive): The prediction is positive, and the actual case is positive.

FP (False Positive): Predicted positive case, actual negative case.

FN (False Negative): Predicted negative case, actual positive case.

TN (True Negative): The predicted case is negative, and the actual case is negative.

Accuracy rate: It indicates the percentage of all correctly classified samples in the total number of samples. The specific calculation formula is shown in Equation (12):

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

Precision rate: indicates the percentage of correct predictions that are positive among all predictions that are positive. The specific calculation formula is shown in Equation (13):

$$\text{Precision } = \frac{TP}{TP + FP} \tag{13}$$

Recall: indicates the percentage of correct predictions that are positive in all actual positive. The specific calculation formula is shown in Equation (14):

$$\text{Recall } = \frac{TP}{TP + FN} \tag{14}$$

F_1: The calculation formula is shown in Equation (15):

$$F_1 = \frac{2}{1/\text{ precison } + 1/\text{ recall}} \tag{15}$$

4.3. Parameter Setting

The system environment used for this experiment is the Windows 10 operating system; the CPU model of the computer is I7-11800H; the GPU model is Geforce GTX 2080Ti; the system running memory is 16 Gb; and the model is built using the deep learning framework Pytorch 1.10. The experimental part of the model needs to select the appropriate Batch_size, learning rate, and optimizer, and the most appropriate learning rate, optimizer, and Batch_size are selected by experimental comparison, and the experimental results are shown in Figures 8, 9 and 10, respectively.

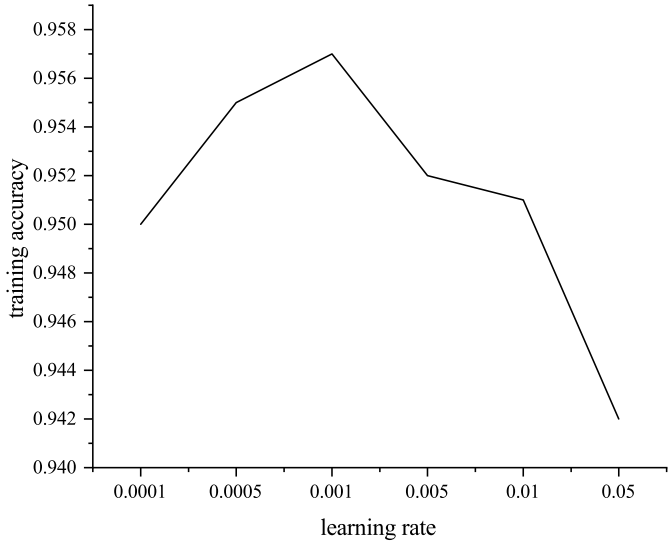

Figure 8. Comparison of model training accuracy under six different learning rates.

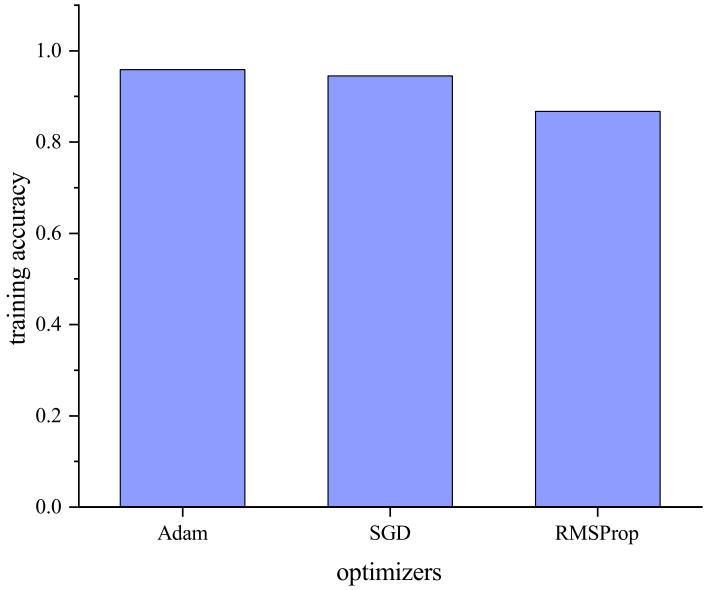

Figure 9. The training accuracy of the model obtained by three different optimizers.

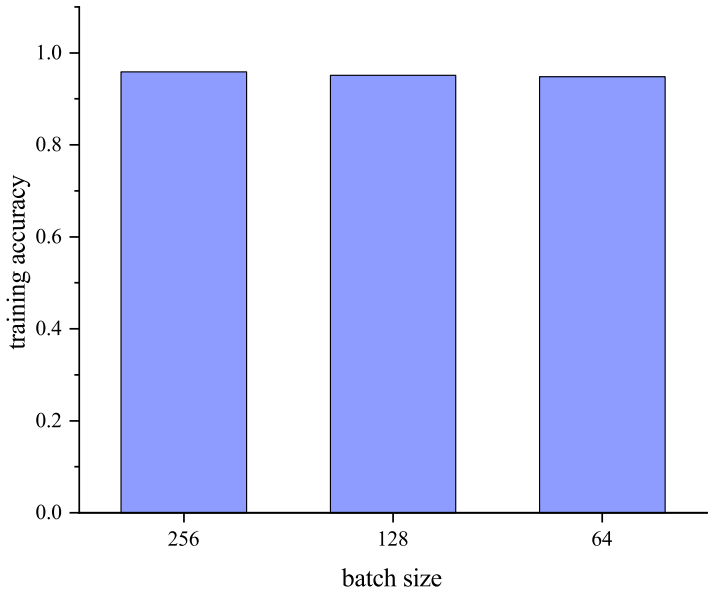

Figure 10. The training accuracy of the model obtained from three different batch sizes.

The experiments use training accuracy as an evaluation metric, and the best learning rate is 0.001, the best optimizer is Adam, and the best batch size is 256, as derived from Figures 8–10.

4.4. Experimental Results and Analysis

4.4.1. Comparison of Experimental Results with Different Temperature Coefficients

The temperature coefficient plays an important role in the process of comparison loss calculation, and by changing the size of the temperature coefficient, the degree of attention to difficult samples can be adjusted. In general, the smaller the temperature coefficient, the more attention is paid to separating positive samples from other samples and, thus, whether better classification can be achieved. The temperature coefficients of 0.5, 0.2, and 0.07 were used to derive the change curves of contrast loss values with an increasing number of iterations, as shown in Figure 11.

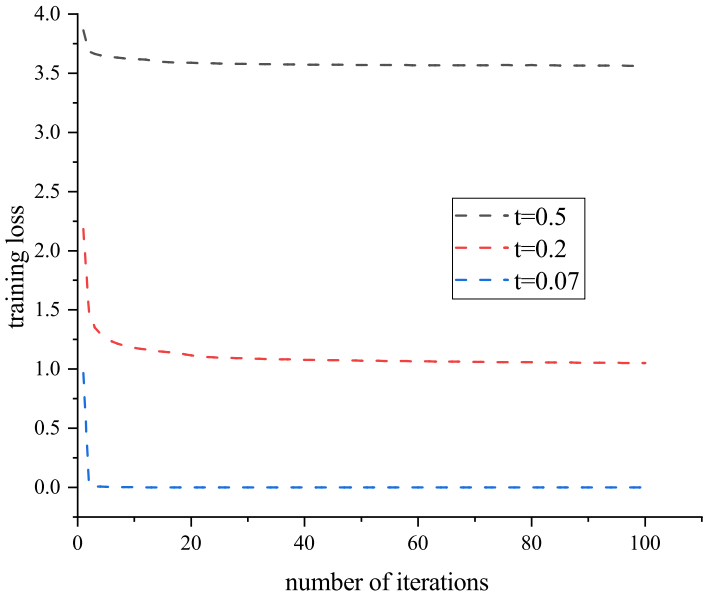

Figure 11. Contrast loss value curves with increasing number of iterations obtained by using three different temperature coefficients in the training phase of contrast learning.

As can be seen from Figure 11, the contrast loss is minimized when using a temperature coefficient of 0.07 and maximized when the temperature coefficient is 0.5. The smaller the temperature coefficient, the stronger the contrast learning effect, and the more you can distinguish between positive and negative samples.

Then the feature extraction network was trained using the temperature coefficients of 0.5, 0.2, and 0.07, respectively, and the accuracy of the model training for 100 batches is shown in Figure 12 by training the classification behind the network by adding a fully connected layer.

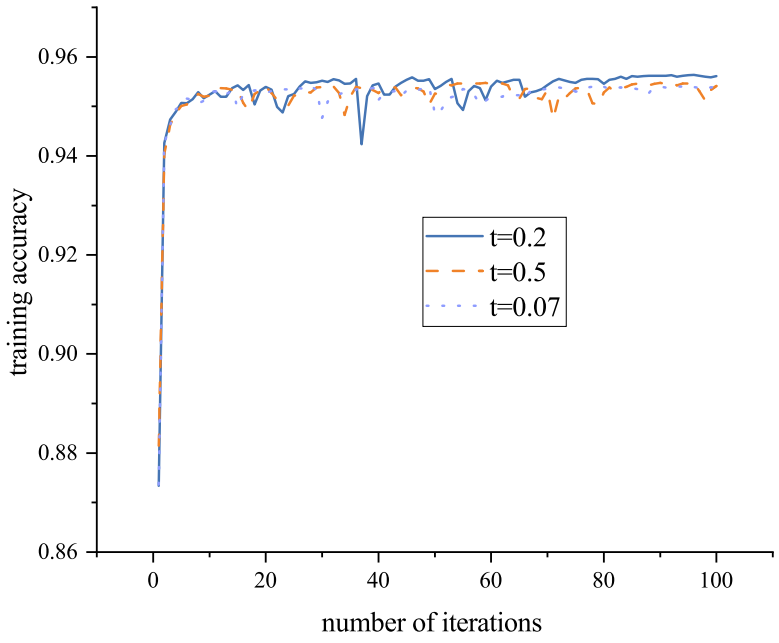

Figure 12. Training accuracy for three different temperature coefficients as the number of iterations increases.

As can be seen from Figure 12, the highest accuracy and best training results are achieved when the temperature coefficient is chosen to be 0.2. In this case, the negative

samples that are extremely similar to the positive samples are often likely to be potential positive samples, and the larger the temperature coefficient, the more there is no difference for the comparison between positive and negative samples, which tend to be treated equally and do not pay too much attention to the more difficult negative samples, so too large or too small temperature coefficients are not conducive to the calculation of contrast loss.

4.4.2. Comparison of Experimental Results of Different Data Enhancement Methods

To validate the efficacy of the data augmentation technique proposed in this paper, the SMOTE algorithm sampling, adding Gaussian noise, sequence inversion, and the original data were selected for comparison learning experiments with the data channel fusion method in this paper, respectively. Firstly, different data enhancements were performed on the basis of obtaining the original industrial control data traffic, and the sequence distribution after using different data enhancements is shown in Figure 13.

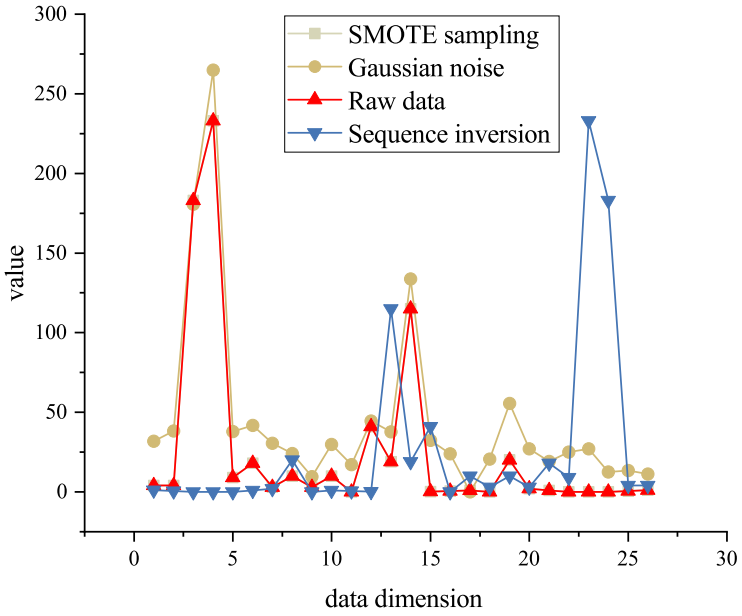

Figure 13. Sequence distribution of the three data enhancements for a single sample versus the original data.

It can be seen from Figure 13 that the feature distribution of the data with Gaussian noise added and the data sampled with the SMOTE algorithm is not much different from the original data. The feature distribution of the data after the sequence inversion is opposite to the original data.

The experimental accuracy of the comparison learning model training after using different data enhancements is shown in Figure 14.

The experimental results show that the data with channel merging are better than those with SMOTE sampling, Gaussian noise addition, sequence inversion, and the original data. The data with sequence inversion differed from the original data in dimensional order, and the experimental results were worse than the other three. The data after channel merging incorporates data enhanced by different types of data and has higher-order features of multiple data types, which can obtain better features after subsequent feature extraction and improve the classification detection effect.

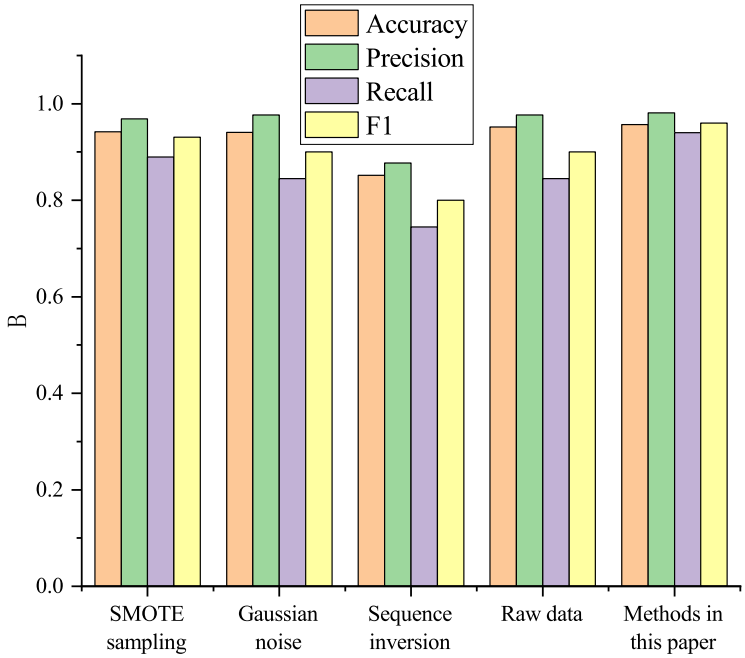

Figure 14. Comparison of the four evaluation metrics of the improved data enhancement approach with the other three enhancement approaches on the test set.

4.4.3. Comparison of Experimental Results of Different Classification Models

On top of the trained contrast learning model, the feature extraction network is obtained, a linear classification layer is added and retrained using a small amount of labeled data, and the classification experiment is completed by fine-tuning the model parameters. CNN and LSTM are commonly used deep learning algorithms. The results of this model are compared with three algorithms after improvement: ResNet, CNN-LSTM, and Attention-LSTM, which are deep learning methods without using comparison learning models for training assistance. The loss values and accuracy curves of the models in the training phase are shown in Figures 15 and 16, the confusion matrix of the test set results is shown in Figure 17, and the evaluation metrics of the four models are shown in Table 4.

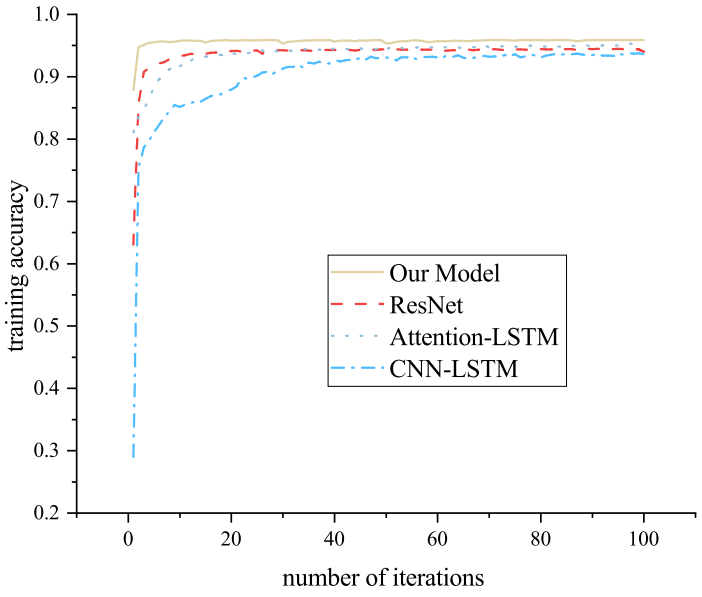

Figure 15. Accuracy curves of the four models on the training set as the number of iterations increases.

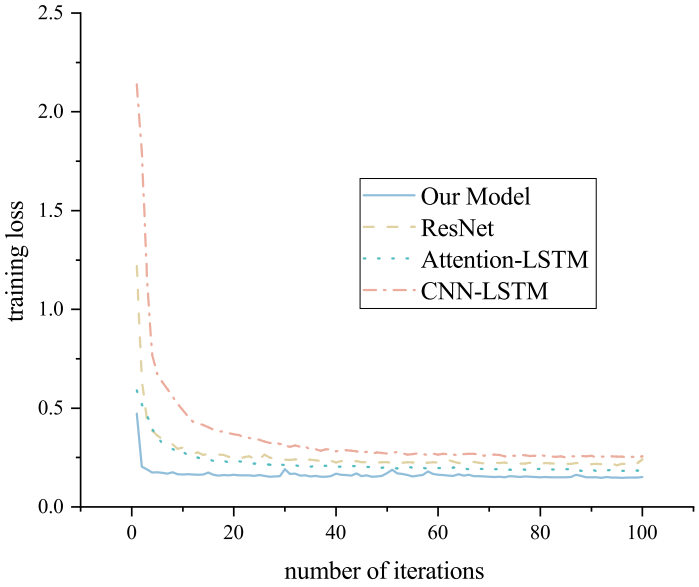

Figure 16. Loss curve with increasing number of iterations.

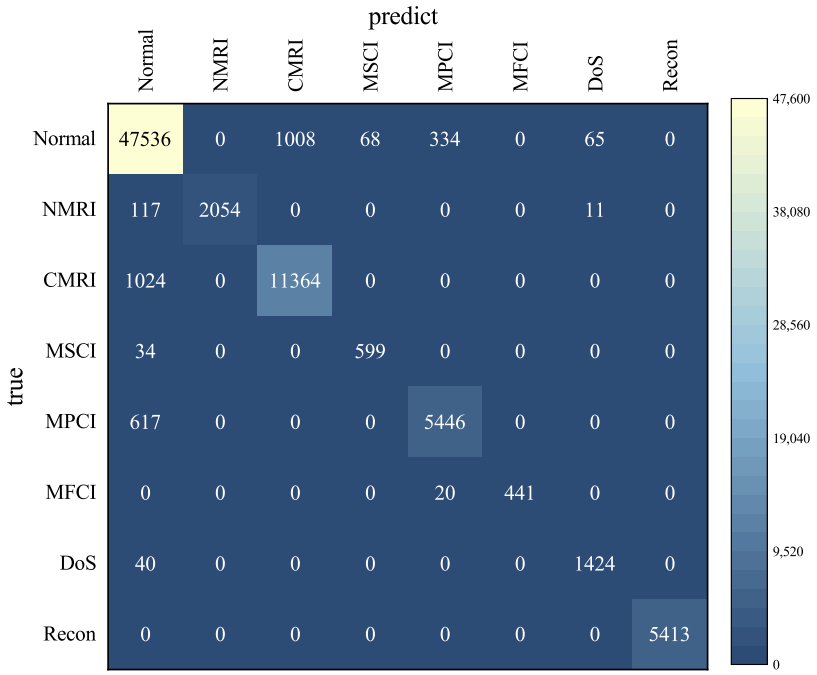

Figure 17. Natural Gas Pipeline Control System Dataset Confusion Matrix.

Table 4. Four evaluation indicators for the natural gas pipeline control system dataset on four different models.

Model	Acc	Precision	Recall	F1
Our Model	0.957	0.981	0.940	0.960
ResNet	0.932	0.947	0.915	0.930
Attention-LSTM	0.947	0.953	0.923	0.937
CNN-LSTM	0.928	0.939	0.896	0.916

From Figures 15 and 16, it can be seen that the model training on the training set, as the number of training iterations increases, has the highest accuracy and the smallest loss value. From the confusion matrix in Figure 17; it can be seen that most of the results are located

on the diagonal of the matrix, and a small number of results are located on both sides of the diagonal, indicating that the model is able to detect most of the results. As can be seen from Table 4, using the test set in the trained model for model testing, the method in this paper achieves 95.7%, 98.1%, 94.0%, and 96.0% in the four indexes of accuracy, precision, recall, and F1, respectively. The results are better than ResNet, CNN-LSTM, and Attention-LSTM. The model training time is shown in Table 5:

Table 5. Model training time.

Model	Training Time
Our Model	196.32 s
ResNet	477.73 s
Attention-LSTM	210.83 s
CNN-LSTM	472.13 s

4.4.4. Experimental Results for Different Datasets

To verify the applicability of the model, this section further validates its performance by replacing it with the SWaT dataset, using the same experimental approach as described above. Again, the experimental results are compared with three common deep learning algorithms, and the confusion matrix for the experiments as well as the model evaluation metrics are shown in Figure 18 and Table 6 below.

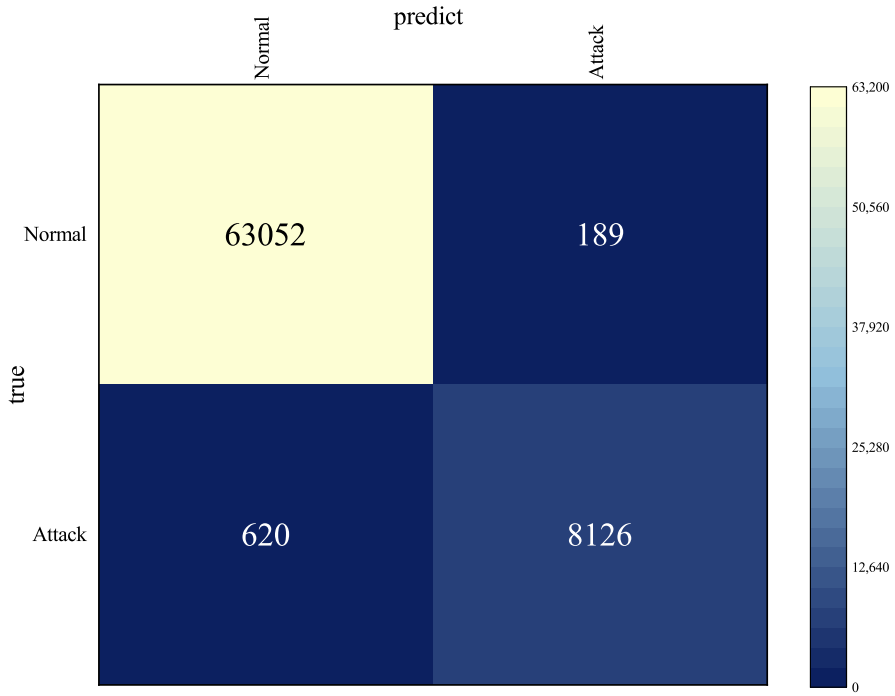

Figure 18. SWaT Dataset Confusion Matrix.

Table 6. Four evaluation indicators for the SWaT dataset on four different models.

Model	Acc	Precision	Recall	F1
Our Model	0.989	0.984	0.963	0.973
ResNet	0.978	0.965	0.960	0.962
Attention-LSTM	0.980	0.976	0.961	0.968
CNN-LSTM	0.967	0.962	0.958	0.959

From the confusion matrix Figure 18, we can see that most of the detected results are located on the diagonal of the matrix, which indicates that this model has a better

classification effect, and from Table 6, we can see that the four indexes of accuracy, precision, recall, and F1 in this paper reach 98.9%, 98.4%, 96.3%, and 97.3%, respectively, compared with the other three common deep learning algorithms.

5. Conclusions

Traditional supervised learning methods are unable to train deep learning models using unlabeled data and are ineffective at intrusion detection. We propose a contrast learning SimCLR-based intrusion detection model for industrial control systems, using the contrast learning model SimCLR to assist in training a feature extraction network from a large amount of unlabeled industrial control system traffic data, adding a linear classification layer after the trained feature extraction network, and retraining with a small number of labeled samples to be able to detect normal and attack samples. Our improved data augmentation allows for more useful information to be obtained after feature extraction, improving the accuracy of subsequent classification and detection tasks, and the cross-jump connection used in the feature-projection stage allows for more useful feature information to be retained when calculating losses. The experiments are conducted on two publicly available datasets collected on industrial control systems for model training and testing, and the results show that the proposed model is effective in detecting both normal and attack samples, as well as better detection compared with other supervised learning algorithms that do not use a contrast learning model to assist in training. In future work, the cost of time will be considered so that the model can perform detection quickly and accurately in real-time.

Author Contributions: Conceptualization, A.Y. and F.L.; Formal analysis, C.L.; Investigation, Z.H.; Methodology, C.L.; Project administration, C.L.; Supervision, M.H.; Validation, L.Z.; Writing—original draft, C.L.; Writing—review & editing, C.L. and F.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (52074126); Hebei Provincial Natural Science Foundation of China (E2022209110); Scientific Basic Research Projects (Natural Sciences) (JQN2021027); and Hebei Natural Science Foundation Project (E2021209024).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset address used in this paper https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets. https://itrust.sutd.edu.sg/itrust-labs_datasets/(accessed on 10 February 2023).

Acknowledgments: Support by colleagues and the university is acknowledged.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alanen, J.; Linnosmaa, J.; Malm, T.; Papakonstantinou, N.; Ahonen, T.; Heikkilä, E.; Tiusanen, R. Hybrid ontology for safety, security, and dependability risk assessments and Security Threat Analysis (STA) method for industrial control systems. *Reliab. Eng. Syst. Saf.* **2022**, *220*, 108270. [CrossRef]
2. Anthi, E.; Williams, L.; Rhode, M.; Burnap, P.; Wedgbury, A. Adversarial attacks on machine learning cybersecurity defences in industrial control systems. *J. Inf. Secur. Appl.* **2021**, *58*, 102717. [CrossRef]
3. Zhang, D.; Wang, Q.G.; Feng, G.; Shi, Y.; Vasilakos, A.V. A survey on attack detection, estimation and control of industrial cyber–physical systems. *ISA Trans.* **2021**, *116*, 1–16. [CrossRef] [PubMed]
4. Corallo, A.; Lazoi, M.; Lezzi, M.; Luperto, A. Cybersecurity awareness in the context of the Industrial Internet of Things: A systematic literature review. *Comput. Ind.* **2022**, *137*, 103614. [CrossRef]
5. Li, B.; Wu, Y.; Song, J.; Lu, R.; Li, T.; Zhao, L. DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems. *IEEE Trans. Ind. Inform.* **2020**, *17*, 5615–5624. [CrossRef]
6. Roy, S.; Li, J.; Choi, B.J.; Bai, Y. A lightweight supervised intrusion detection mechanism for IoT networks. *Future Gener. Comput. Syst.* **2022**, *127*, 76–285. [CrossRef]

7. Tsimenidis, S.; Lagkas, T.; Rantos, K. Deep learning in IoT intrusion detection. Journal of network and systems management. *J. Netw. Syst. Manag.* **2022**, *30*, 1–40. [CrossRef]
8. Bhardwaj, A.; Al-Turjman, F.; Kumar, M.; Stephan, T.; Mostarda, L. Capturing-the-invisible (CTI): Behavior-based attacks recognition in IoT-oriented industrial control systems. *IEEE Access* **2020**, *8*, 104956–104966. [CrossRef]
9. Chen, Y.; Su, S.; Yu, D.; He, H.; Wang, X.; Ma, Y.; Guo, H. Cross-Domain Industrial Intrusion Detection Deep Model Trained With Imbalanced Data. *IEEE Internet Things J.* **2022**, *10*, 584–596. [CrossRef]
10. Khan, M.A. HCRNNIDS: Hybrid Convolutional Recurrent Neural Network-Based Network Intrusion Detection System. *Processes* **2021**, *9*, 834. [CrossRef]
11. Hu, Y.; Li, H.; Luan, T.H.; Yang, A.; Sun, L.; Wang, Z.; Wang, R. Detecting stealthy attacks on industrial control systems using a permutation entropy-based method. *Future Gener. Comput. Syst.* **2020**, *108*, 1230–1240. [CrossRef]
12. Ling, J.; Zhu, Z.; Luo, Y.; Wang, H. An intrusion detection method for industrial control systems based on bidirectional simple recurrent unit. *Comput. Electr. Eng.* **2021**, *91*, 107049. [CrossRef]
13. Wang, C.; Wang, B.; Liu, H.; Qu, H. Anomaly detection for industrial control system based on autoencoder neural network. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8897926.
14. Umer, M.A.; Junejo, K.N.; Jilani, M.T.; Mathur, A.P. Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations. *Int. J. Crit. Infrastruct. Prot.* **2022**, *38*, 100516. [CrossRef]
15. Liu, X.; Zhang, F.; Hou, Z.; Mian, L.; Wang, Z.; Zhang, J.; Tang, J. Self-supervised learning: Generative or contrastive. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 857–876.
16. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. *Proc. Int. Conf. Mach. Learn.* **2020**, 119, 1597–1607.
17. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9729–9738.
18. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.; Buchatskaya, E.; Doersch, C.; Avila Pires, B.; Guo, Z.; Gheshlaghi Azar, M.; et al. Bootstrap your own latent-a new approach to self-supervised learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21271–21284.
19. Chen, X.; He, K. Exploring simple siamese representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; Volume 2021, pp. 15750–15758.
20. Caron, M.; Misra, I.; Mairal, J.; Goyal, P.; Bojanowski, P.; Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9912–9924.
21. Wang, S.; Dai, Y.; Shen, J.; Xuan, J. Research on expansion and classification of imbalanced data based on SMOTE algorithm. *Sci. Rep.* **2021**, *11*, 24039. [CrossRef] [PubMed]
22. Shafiq, M.; Gu, Z. Deep Residual Learning for Image Recognition: A Survey. *Appl. Sci.* **2022**, *12*, 8972. [CrossRef]