Hindawi
Security and Communication Networks
Volume 2019, Article ID 8124254, 11 pages
https://doi.org/10.1155/2019/8124254

Research Article

Industrial Anomaly Detection and Attack Classification Method Based on Convolutional Neural Network

Yingxu Lai,[1] **Jingwen Zhang**,[1] **and Zenghui Liu**[2]

[1]*College of Computer Science, Faculty of Information, Beijing University of Technology, Beijing 100124, China*
[2]*Institute of Electromechanical Engineering, Beijing Polytechnic, Beijing 100176, China*

Correspondence should be addressed to Yingxu Lai; laiyingxu@bjut.edu.cn and Zenghui Liu; zenghuiliu@yeah.net

Received 26 December 2018; Revised 10 July 2019; Accepted 19 August 2019; Published 29 September 2019

Academic Editor: Mamoun Alazab

The massive use of information technology has brought certain security risks to the industrial production process. In recent years, cyber-physical attacks against industrial control systems have occurred frequently. Anomaly detection technology is an essential technical means to ensure the safety of industrial control systems. Considering the shortcomings of traditional methods and to facilitate the timely analysis and location of anomalies, this study proposes a solution based on the deep learning method for industrial traffic anomaly detection and attack classification. We use a convolutional neural network deep learning representation model as the detection model. The original one-dimensional data are mapped using the feature mapping method to make them suitable for model processing. The deep learning method can automatically extract critical features and achieve accurate attack classification. We performed a model evaluation using real network attack data from a supervisory control and data acquisition (SCADA) system. The experimental results showed that the proposed method met the anomaly detection and attack classification needs of a SCADA system. The proposed method also promotes the application of deep learning methods in industrial anomaly detection.

1. Introduction

Industrial control systems (ICSs) play an important role in critical infrastructure sectors such as railway, petrochemical, and electricity. Although the application of information technology has significantly improved production efficiency, it has also resulted in many cyber-attacks to ICSs that may cause damage to the national infrastructure and bring about major economic losses [1]. In 2010, an Iranian nuclear power station was attacked by Stuxnet using operating system vulnerabilities and led to the infection of many PLCs connected to centrifuges [2, 3]. This was the first discovered destructive malware explicitly designed for ICSs. In 2012, the Flame virus was discovered in several Middle Eastern countries. This malware can collect sensitive information from various fields such as individuals, private companies, government agencies, and academic institutions. In 2014, hackers attacked the ICSs in the energy field of Europe and the United States using the Havex malicious software to specifically target supervisory control and data acquisition (SCADA) systems [4]. In addition to these well-known industrial security incidents, hundreds of attacks against ICSs appear every year. Although there are fewer attacks on industrial systems than on the Internet, the damage caused by cyber-physical attacks cannot be ignored.

ICSs were initially designed with insufficient consideration for network security because of resource constraints and system isolation [5]. With the development of modern ICSs and information technology, potential security issues have been gradually exposed. To ensure an industrial process runs steadily and reliably, an ICS needs to be protected. Traditional information system solutions have been applied to ICSs as an early defense method. However, these measures cannot adequately detect cyber-physical attacks under real-time and resource-constrained conditions [6]. In recent years, the anomaly detection and safety protection of ICSs have been researched extensively worldwide to identify malicious patterns by measuring the deviation of current

behavior with normal behavior as the standard. This study focuses on improving the performance of the industrial anomaly detection method. In addition, it is not enough to treat abnormal behavior detection as a binary classification problem. To quickly locate the source when a network is attacked and to achieve the mitigation and recovery of the control system state, it is necessary to observe ICS abnormal patterns in more detail.

Motivated by the research objectives above, this study attempted to apply deep learning to SCADA anomaly detection and attack classification. Deep learning is a type of machine learning method that has developed rapidly in recent years. Deep neural networks can capture deep relationships that are not easily obtained by ordinary means. The deep learning method can automatically learn from original data with no need to manually select features [7]. The deep neural network model we selected was the convolutional neural network (CNN) that has been widely used for image classification and recognition. We used this method to classify ICS abnormal traffic. A real gas pipeline data set proposed by the Mississippi State University was used in this study.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 provides the theoretical basis of the method, and Section 4 introduces the methodology of the proposed method based on a CNN. Section 5 introduces the data sets used in this study and shows the relevant experimental results. Finally, Section 6 summarizes the results.

2. Related Work

2.1. Anomaly Detection. As a subset of intrusion detection, anomaly detection plays a significant role in the active defense process of ICSs. It is also a key technology for discovering abnormal behavior. To summarize existing research work, the anomaly detection approaches of ICSs include the following types.

2.1.1. Knowledge Based. This method analyzes the system state, behavior pattern, or protocol specification under normal conditions and establishes a detection rule to detect attacks that do not conform to the specification. For example, the industrial intrusion detection system proposed by Khalili and Sami [8] used the Apriori algorithm to generate candidate key states of the ICS iteratively. During each iteration, the key state of the industrial process is decided based on expert experience. Carcano et al. [9] proposed a state-based SCADA intrusion detection system using single-signature and state analysis techniques to analyze Modbus data packets and proposed a rule language to describe Modbus signatures and field device status. Experimental results showed that the proposed IDS detected potential threats to a SCADA system. Yang et al. [3] designed multiattribute IDS for SCADA systems in smart grids. The IDS was constructed using the access control whitelist and the protocol whitelist method. At the same time, normal system behavior rules were used as additional methods. For the high periodicity of an industrial control system flow, Goldenberg and Wool [10] used a deterministic finite automaton (DFA) for each HMI-PLC channel's modeling to construct an accurate flow detection model. However, the traditional cyclic DFA model could not handle the burst traffic in an industrial control network. Markman et al. [11] constructed a new burst-DFA model based on semantic analysis that successfully solved the above problem. The shortcoming of the anomaly detection technique based on knowledge is that it cannot detect potential attack operations that exploit system vulnerabilities or that conform to protocol specifications.

2.1.2. Statistics Based. The statistical anomaly detection technique often uses analytical and statistical correlation methods to analyze the parameters of ICSs and establish the normal behavioral contours of a system. Nasr and Varjani [12] analyzed the alarm attributes of a SCADA system and used a statistical method called SADM to detect the abnormality of a smart grid. The CUSUM algorithm is a commonly used statistical algorithm. Do et al. [13] proposed an improved algorithm named VTWL CUSUM to capture transient attacks in SCADA systems and optimized the algorithm by the finite moving average method. The inadequacy of statistical anomaly detection methods is that they are insensitive to the sequence of events and internal connections. At the same time, intruders can train detection systems to treat intrusions as normal behavior.

2.1.3. Machine Learning Based. Machine learning methods have been extensively used in data mining, speech recognition, target detection, and other fields. Researchers have tried to apply related technologies in the field of anomaly detection in ICSs. Shang et al. [14] used the Modbus/TCP protocol as their research goal and trained an OCSVM model using the protocol function code sequence. Furthermore, the particle swarm optimization algorithm was used to optimize the model parameters. Zhou et al. [15] comprehensively analyzed the multidomain knowledge of the field-control layer in industrial process automation and extracted multimodal data based on domain knowledge. An intelligent classifier based on the hidden Markov model (HMM) was constructed to realize the intrusion detection of industrial process data. Nader et al. [16] researched two approaches of one-class classification regarding support vector machines and applied these two methods to SCADA anomaly detection. A heuristic algorithm was also proposed to optimize the machine learning parameters.

To a certain extent, intrusion detection based on machine learning can improve the accuracy of abnormal behavior detection in the industrial control environment, and it is of great significance in establishing an intelligent and efficient intrusion detection model.

2.2. Deep Learning. We were inspired by the findings of Nader et al. [16]. They proposed that a multiclass classification of anomalies can be achieved based on the completed work so that the types of attacks on a SCADA system can be directly determined. This proposal is very valuable, and we

attempted to implement anomaly detection and attack classification for ICSs and also to consider algorithm performance. Deep learning methods seem to be a good choice. Compared with traditional machine learning methods, the most significant advantage of deep learning is the ability to learn features directly from the original data automatically, and it has excellent performance [17]. The successful application of deep learning methods in the field of image classification and speech recognition has fully proved this point, and there have been some related studies in the security field [18].

Javaid et al. built a network intrusion detection system using the self-taught learning method [19]. The authors used the proposed approach to perform anomaly detection on an NSL-KDD data set and further improved the classifier performance using the NB tree, random tree, and J48. The metrics for two-class and five-class problems were calculated to demonstrate the effectiveness of the self-taught learning method. Thing [20] analyzed threats that may exist in IEEE 802.11 networks and used the stacked automatic encoder (SAE) to implement anomaly detection and attack classification. The authors used different activation functions to enhance the classification performance of the model. The experimental results showed that the PReLU model based on the two-hidden-layer and the three-hidden-layer architecture had a relatively balanced performance in the anomaly detection and attack classification for IEEE 802.11 networks. At the same time, the SAE of the two-hidden-layer architecture was superior to that of the three-hidden-layer architecture. The experimental results showed that the proposed method had a higher overall accuracy of 98.6688% compared with the most advanced method. Yan and Han [21] used an SAE model to reduce the original sample data and then added sparse constraints to the model to improve the generalization ability and classification accuracy of the model. Experiments showed that the feature extraction ability of SAE was significantly better than that of traditional machine learning methods.

Some deep learning methods have been applied to smart grids. Ashrafuzzaman et al. [22] used a feed-forward artificial neural network to detect false data injection attacks against a power grid. Wilson et al. [23] used the SAE model to detect anomalies in a power system. Wang et al. [24] used the SAE model in their research and found that a deep learning model reduced the uncertainty of electric load forecasting and thereby indirectly improved the performance of anomaly detection. Existing research shows that deep learning also has potential in industrial system security.

Existing research has shown that the deep learning method has superior performance in the field of anomaly detection and attack classification. In this study, we used these studies to promote the use of raw industrial traffic data to perform anomaly detection and attack classification tasks. We attempted to convert industrial data streams into different images and use a CNN for learning and testing. To the best of our knowledge, this is the first attempt to apply the CNN method for industrial anomaly detection and classification.

3. Problem and Solution Statement

In this section, we analyze the characteristics of industrial networks in comparison to a traditional network. These statements will provide a theoretical basis for our proposed solution. The inadequacy of anomaly detection based on the traditional machine learning method is also taken into consideration.

3.1. Correlation between Features. Industrial network traffic variables have a stronger correlation than general-propose computer networks. Operating a particular variable in the industrial production process may cause a knock-on effect. Undoubtedly, the trend of changes in features caused by cyber-attacks is different from those of the normal behavior. This suggests that the rational application of the correlation between industrial characteristics will assist in the detection of anomalies.

3.2. High Cost of Error. The significance of national infrastructure determines the higher requirements for anomaly detection and classification of industrial network traffic compared with general-propose computer networks. Misclassification of industrial traffic may lead to catastrophic consequences. Anomaly detectors should strive for higher detection rates and response speeds to reduce associated costs. The precise classification of various attacks should be guaranteed to achieve timely analysis and identify the location of anomalies.

3.3. Imbalanced Instance Distribution. An ICS is in a stable and normal state for the majority of its operating time; therefore, it is relatively easy for researchers to obtain normal traffic samples. In reality, the probability of anomalous events occurring is lower than in normal events, which makes it difficult to collect attack traffic data. Imbalanced distribution may negatively impact the performance in machine learning-based anomaly detectors and classifiers. One of the traditional solutions is to synthesize minority class samples by exploiting the similarity of feature spaces, but this method's disadvantage is overgeneralized [25]. Another option is to reduce the size of majority samples using a random deletion method; however, some important information in majority of the samples will be lost. As a trade-off, feature engineering improves the model ability and solves the imbalance problem by selecting and extracting significant features that represent the characteristics of a sample. However, the current pattern of manually selecting features is very complicated and requires researchers to have heuristic expertise.

Based on the above analysis, we propose an anomaly detection and attack classification model based on a CNN. The CNN is a popular deep learning technique and has been confirmed as the best choice in the field of image classification. The solution we propose has the following advantages.

3.4. Feature Correlation. The method we propose considers the actual situation of the industrial production process and the relationship between traffic features. We used a feature mapping method based on the Mahalanobis distance before the data input to implement the correlation measure between the dimensions of traffic instances.

3.5. High Performance. The sparse connectivity and shared weights of a CNN greatly reduce the related parameters and improve the training speed. The feature extraction layer eliminates similarities and maintains differences between various class instances. In other words, removing redundant information will be more conducive for correct detection results.

3.6. Feature Selection Automation. Deep neural networks can perfectly solve the problem of sample imbalance and reduce the workload of manual design features because they have the ability to automatically learn from the training data and obtain new effective feature representations.

4. Methodology

Based on the discussion in Section 3, this section provides a detailed description of the relevant methods. Considering the input requirements and the full utilization of the feature correlation, we first encoded the captured industrial process traffic and mapped it into a matrix. Next, we used a CNN to learn the data features and extract deeper representations that were more conducive to recognition. Finally, anomaly detection and classification were performed by a supervised machine learning algorithm.

4.1. Feature Mapping. We propose a feature mapping method based on the Mahalanobis distance that transforms one-dimensional data into a two-dimensional matrix that can be used as CNN input. The Mahalanobis distance takes into account the relationship between features, and there is a certain correlation between the features in an ICS. We present the steps to convert an industrial process data stream into a Mahalanobis distance matrix.

First, we defined n industrial data traffic flow as $X = \{x_1, x_2, x_3, \ldots, x_n\}$. Based on the characteristics of network traffic for an industrial environment, x_i can be expressed as $x_i = [f_1^i, f_2^i, f_3^i, \ldots, f_m^i]$, where m is the number of features included in each data flow. In this study, the value of m was 26 and f_l^i represents the value of the l-th feature in the i-th data stream.

To make full use of the correlation between different features in the i-th network stream feature vector, x_i was converted into a matrix of m rows and m columns. The specific conversion method is

$$X_i = x_i^{\mathrm{T}} I_m = \left[f_1^i, f_2^i, f_3^i, \ldots, f_m^i\right] \cdot \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} f_1^i & 0 & \cdots & 0 \\ 0 & f_2^i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_m^i \end{bmatrix}_{m \times m}, \tag{1}$$

where I_m is an m-order identity matrix. Obviously, the diagonal elements of the matrix X_i are the values of the m-dimensional features.

We can represent each column of matrix X_i with an m-dimensional vector F_j^i:

$$F_j^i = \begin{bmatrix} \eta_{j,1}^i \\ \eta_{j,2}^i \\ \cdots \\ \eta_{j,m}^i \end{bmatrix}, \tag{2}$$

here,

$$\eta_{j,p}^i = \begin{cases} 0, & j \neq p, \\ f_j^i, & j = p. \end{cases} \tag{3}$$

Therefore, X_i can be expressed in m m-dimensional vectors F_j^i as

$$X_i = \left[F_1^i, F_2^i, F_3^i, \ldots, F_m^i\right]. \tag{4}$$

Next, we calculated the covariance matrix of the matrix X_i to find its inverse matrix:

$$\sum^{-1} = \operatorname{cov}(X_i)^{-1} = \begin{bmatrix} \operatorname{cov}(F_1^i, F_1^i) & \operatorname{cov}(F_1^i, F_2^i) & \cdots & \operatorname{cov}(F_1^i, F_m^i) \\ \operatorname{cov}(F_2^i, F_1^i) & \operatorname{cov}(F_2^i, F_2^i) & \cdots & \operatorname{cov}(F_2^i, F_m^i) \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{cov}(F_m^i, F_1^i) & \operatorname{cov}(F_m^i, F_2^i) & \cdots & \operatorname{cov}(F_m^i, F_m^i) \end{bmatrix}^{-1}. \tag{5}$$

The correlation between different features of the traffic flow feature vector is defined by the Mahalanobis distance as

$$\mathrm{MD}_{j,k}^i = \begin{cases} \sqrt{\left(F_j^i - F_k^i\right)^{\mathrm{T}} \sum_{j,k}^{-1} \left(F_j^i - F_k^i\right)}, & j \neq k, \\ 0, & j = k. \end{cases} \tag{6}$$

Ultimately, the i-th flow record x_i can be represented as a symmetric matrix MHD_{x_i} with m rows and m columns diagonally all zeros as

$$\mathrm{MHD}_{x_i} = \begin{bmatrix} \mathrm{MD}^i_{1,1} & \mathrm{MD}^i_{1,2} & \cdots & \mathrm{MD}^i_{1,m} \\ \mathrm{MD}^i_{2,1} & \mathrm{MD}^i_{2,2} & \cdots & \mathrm{MD}^i_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{MD}^i_{m,1} & \mathrm{MD}^i_{m,2} & \cdots & \mathrm{MD}^i_{m,m} \end{bmatrix}. \quad (7)$$

4.2. Feature Matrix Visualization. We can treat each element in the MHD matrix after feature mapping as a pixel point; therefore, each data set instance can be transformed into a grayscale map. Before generating the grayscale map, we used the following method to normalize the MHD matrix:

$$\mathrm{MD}^{i'}_{j,k} = \frac{\mathrm{MD}^i_{j,k} - \mathrm{MHD}_{x_i\mathrm{MinValue}}}{\mathrm{MHD}_{x_i\mathrm{MaxValue}} - \mathrm{MHD}_{x_i\mathrm{MaxValue}}}, \quad (8)$$

where $\mathrm{MHD}_{x_i\mathrm{MinValue}}$ and $\mathrm{MHD}_{x_i\mathrm{MaxValue}}$ represent the minimum and maximum values in the matrix MHD_{x_i}, respectively. After the linear transformation shown in equation (8), the original elements were mapped in [0, 1].

The size of the grayscale maps generated by the data set used in this study was 676 (26×26) bytes. The visualized images of each category are shown in Figure 1, and the descriptions of the categories are provided in Table 1.

Observable differences between the various types of SCADA traffic prove that the anomaly detection method was reliable.

4.3. Convolutional Neural Network. A CNN is a multilayer neural network developed from a traditional neural network. It essentially learns a deep nonlinear network structure, realizes complex function approximation, and represents the input-output mapping relationship [26]. At the same time, it learns the basic characteristics of a data set from a small sample set. The CNN consists of convolutional layers, subsampling layers, and fully connected layers. The common CNN architecture is primarily a combination of convolutional layers and subsampling layers. The fully connected layers are the upper layers, the input is the features extracted by the lower layers, and the output is the classification result.

Figure 2 depicts the CNN architecture used in this study. Compared with other classical architectures such as AlexNet, VGGNet, and GoogleNet, the model architecture and input size of LeNet-5 were more suitable for our requirement. We adjusted the CNN architecture of LeNet-5 appropriately in two aspects. First, the input layer of the network was designed to be 26×26 pixels according to the grayscale image size obtained by the feature mapping. Second, the number of nodes in the output layer was changed.

CNN takes the original image of size $26 \times 26 \times 1$ as input, and the convolution layer C1 performs convolution operations with six 3×3 kernels to obtain six 24×24 feature maps. The S2 layer takes the output of the C1 layer as an input and uses 2×2 windows to pool six images and then obtains six 12×12 feature maps. The kernel size of C3 was the same as that of C1, but we used the pretrained 16 channels for convolution operations; therefore, the result was 16 10×10 feature maps. The subsampling S4 layer pooled 16 inputs using 2×2 receptive fields, and the result was 16 5×5 feature maps. The last two layers were fully connected layers. The number of nodes set was referenced to LeNet-5, and there were eight final output nodes. The CNN architecture used the Softmax function to implement multiclassification and dropout to mitigate overfitting of the model and the nonlinear activation function ReLU to introduce sparsity into the neural network.

5. Evaluation

5.1. Data Set. The most well-known intrusion detection data set is the KDD data set [27], which is a transformation of the DARPA data set and contains nine weeks of network connectivity data collected on a simulated US Air Force LAN. NSL-KDD is an improved version of the KDD data set that enhances the performance of the classification method [28]. However, KDD and NSL-KDD are intrusion detection data sets on traditional networks. The data set used in this study consisted of real data collected from the SCADA system of a natural gas pipeline test platform designed and developed by the Mississippi State University [29]. The platform included a master terminal unit, a remote terminal unit, and a human-machine interface. A proportion integrals differential (PID) controller was used to maintain the stability of the air pressure in the pipeline. The natural gas pipeline data set had a total of 27 features grouped into two classes: network traffic features and payload content features.

The network traffic features were used to indicate the communication status of the SCADA system and include the device address of the request/response packet and the location and byte size of the request/response memory in the packet. The time feature was used to record the time interval between the request and response in the SCADA communication. There was not much difference between normal data instances. Similarly, when the SCADA system was in a normal communication state, the value of the command/response CRC error rate feature was small, and the value of this feature may increase when the system is attacked.

The payload content characteristics changed in different SCADA systems because of the variety of measurement variables and control methods. In the gas pipeline data set used in this study, the payload features included response/command function codes, current measured values/initial values of gas pressure, and system control modes. In addition to these parameters, PID-related attributes were also considered, and adjustments to related parameters could change the behavior of the PID controller.

Finally, the label feature was used to determine whether the data instance was normal behavior or a certain type of attack. The specific categories of natural gas pipeline data set are shown in Table 1. The data set contained eight types of labels. Table 2 shows the distribution of these data instances.

5.2. Experiment Description. We used TensorFlow developed by Google to implement our CNN model. The goal

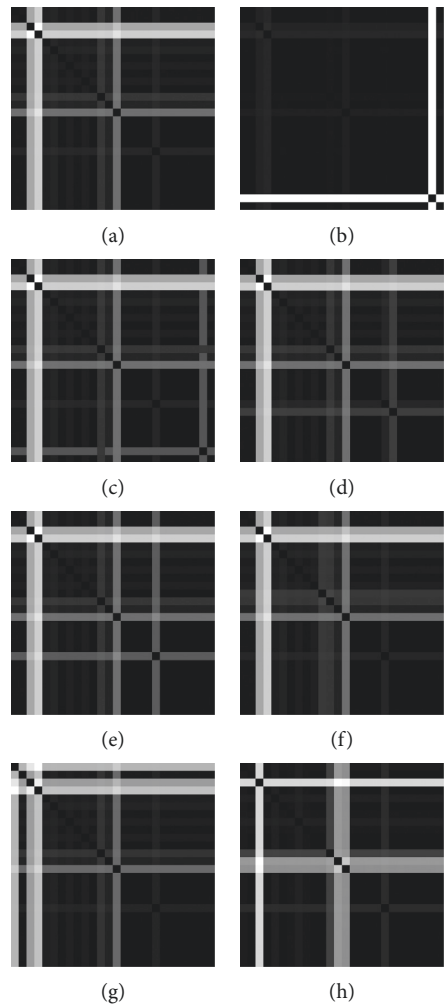


Figure 1: Visualization of all classes of data instances. (a) Normal. (b) NMRI. (c) CMRI. (d) MSCI. (e) MPCI. (f) MFCI. (g) DOS. (h) Recon.

Table 1: Attack classification in data set.

Abbreviation	Label	Label type
Normal	0	Normal behavior
NMRI	1	Naive malicious response injection
CMRI	2	Complex malicious response injection
MSCI	3	Malicious state command injection
MPCI	4	Malicious parameter command injection
MFCI	5	Malicious function code injection
DOS	6	Denial of service
Recon	7	Reconnaissance

of the proposed model was to achieve SCADA traffic classification. Eighty percent of the data set was used for training the model, and the other 20% was used for testing. The batch size was 50, and the learning rate was set as 0.001. The CNN was iteratively trained until its loss function converged. The time spent on training was 20 epochs. The detailed process of the experiment is shown in Figure 3.

The experimental process was executed as follows:

Step 1 (collecting raw data and processing it): the SCADA data set is divided into a training data set and a test data set, and feature mapping is performed.

Step 2 (initializing CNN): construct a CNN with reference to the architecture shown in Figure 2.

Step 3 (training the model): the training data are input into the model, and the weight coefficients of each layer of the CNN are determined through training.

Step 4 (testing the model): the test data are entered into a CNN to classify the SCADA traffic and determine if each metric is above a threshold. If it is larger than the lower limit, fine-tune the parameters and seek the optimal result; otherwise, directly adjust the parameters and repeat Step 3.

Experiments were conducted with an 8 GB RAM 3.2 GHz i5 CPU operating system. Each new instance spent approximately 0.253 s in the feature mapping process and 0.192 s in the detection process. The results show that our method had real-time significance in SCADA with a sampling rate of 1 Hz.

5.3. Evaluation Metrics. Two application scenarios of this method were taken into account. One case was binary classification, that is, a simple distinction between normal

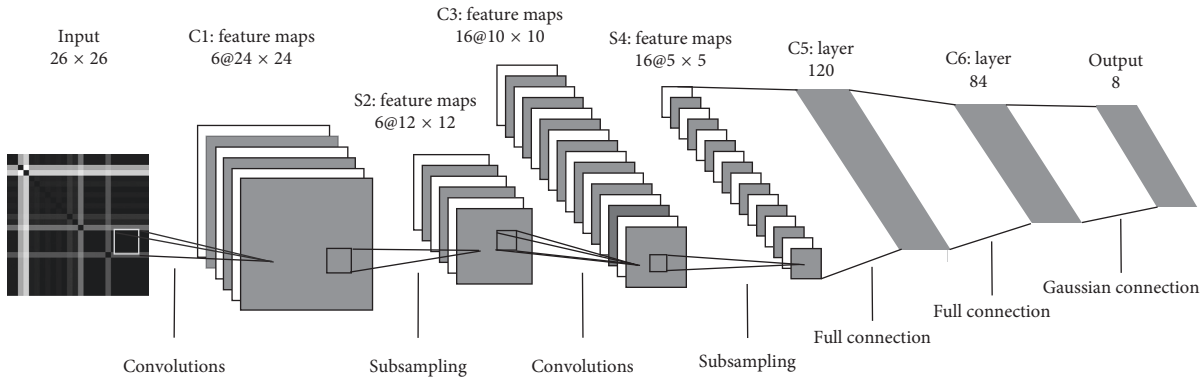


Figure 2: CNN architecture used in the study.

Table 2: Proportion of categories in the data set.

No.	Type	Count total	Proportion (%)
0	Normal	61,156	63.04
1	NMRI	2,763	2.85
2	CMRI	15,465	15.94
3	MSCI	782	0.81
4	MPCI	7,637	7.87
5	MFCI	573	0.59
6	DOS	1,837	1.89
7	Recon	6,805	7.01

traffic and abnormal traffic. The other case was multi-classification, in which the instance was directly tagged with a specific label. The following metrics were used to evaluate the performance of the classifier:

$$A = \frac{TP + TN}{TP + TN + FP + FN},$$
$$P = \frac{TP}{TP + FP},$$
$$R = \frac{TP}{TP + FN}, \quad (9)$$
$$F_1 = \frac{2PR}{P + R}.$$

Accuracy (A) represents the overall performance of the classifier. In addition, the results in Table 3 reflected a problem of unbalanced sample size; therefore, we calculated the precision (P) and recall (R) of each type of instance to ensure that the results were not distorted because of too many normal samples. The F_1 value acted to reconcile P and R.

5.4. CNN Design. We designed the model according to the basic hierarchical structure of LeNet-5 to analyze the influence of different architectures on the experimental results. First, considering the small input size, the sampling window of the subsampling layer was set as 2×2 to avoid information loss. Next, to ensure a fixed center point and sensitivity to edges in the convolution process, only convolution kernels with odd sizes were used. Finally, when the size of a convolution kernel was too large, it was easy to overfit, which is not conducive to subsequent training and testing. Therefore, the size of the convolution kernel in the designed network structure was no more than 5×5. The four architectures and their performance on data sets are shown in Table 3.

As shown in Table 3, the performance of the first structure was the best with accuracy and loss values of 99.32% and 0.025, respectively. The results of the second and third architectures were roughly the same, suggesting that the kernel order had no effect. Results of the fourth structure were the least ideal, suggesting that a large convolution kernel size is a poor choice.

5.5. Experiment Results. The metrics of the binary classification and the multiclassification were calculated. The accuracy of the two scenarios is shown in Table 4. The other three metrics of the binary classification and the multi-classification are shown in Tables 5 and 6, respectively.

The results in Table 4 show that our method satisfied the application requirements of binary classification and multiclassification (the overall accuracy of binary classification and multiclassification was 99.46% and 99.32%, respectively), and the average accuracy of the classifiers was 99.39%. Table 5 indicates that our method was good at solving the two-class problem; the lowest indicator also reached 98.76%. Our approach also performed well in the eight-class problem. Table 6 shows the accuracy, recall, and F_1 values of the multiclassifier where the reconnaissance attack achieved the optimal value (recall, precision, and F_1 values were 100%). The NMRI and MSCI indicators were slightly lower (93%–98%), and we will further explore in the future whether this is owing to certain characteristics of the two types of data.

5.6. Comparison. Before comparing with other machine learning methods, we conduct a lateral comparison experiment. The gas pipeline system reduced (10%) data set was used for model training and testing, and this data set also published by the Mississippi state university. The performance of the two data sets on different categories is shown in Table 7.

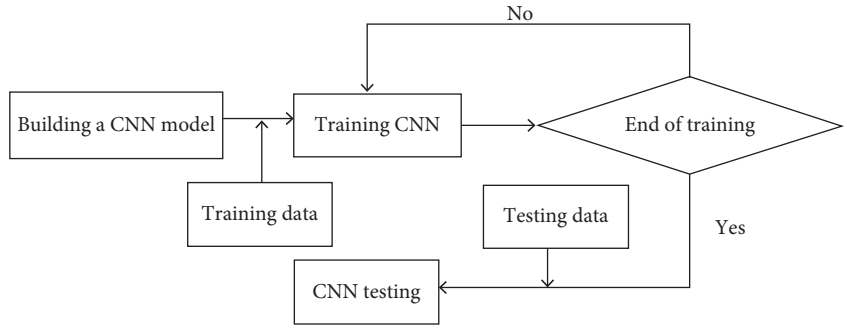


Figure 3: Anomaly detection process using CNNs.

Table 3: Four architectures in this research.

No.	C1		S1		C3		S4		Accuracy (%)	Test Loss
	Kernel	Output	Window	Output	Kernel	Output	Window	Output		
1	$6\times(3\times3)$	$6\times(24\times24)$	2×2	$6\times(12\times12)$	$16\times(3\times3)$	$16\times(10\times10)$	2×2	$16\times(5\times5)$	99.32	0.025
2	$6\times(3\times3)$	$6\times(24\times24)$	2×2	$6\times(12\times12)$	$16\times(5\times5)$	$16\times(8\times8)$	2×2	$16\times(4\times4)$	98.80	0.047
3	$6\times(5\times5)$	$6\times(22\times22)$	2×2	$6\times(11\times11)$	$16\times(3\times3)$	$16\times(9\times9)$	2×2	$16\times(4\times4)$	98.76	0.051
4	$6\times(5\times5)$	$6\times(22\times22)$	2×2	$6\times(11\times11)$	$16\times(5\times5)$	$16\times(7\times7)$	2×2	$16\times(3\times3)$	97.69	0.148

Table 4: Accuracy of two classifiers.

Metric	Binary classification	Multiclassification
Accuracy	99.46%	99.32%

Table 5: Recall, precision, and F_1 value of binary classifier.

Class type	Recall (%)	Precision (%)	F_1 value (%)
Normal	99.75	99.40	99.57
Anomaly	98.76	99.58	99.17

Table 6: Recall, precision, and F_1 value of multiclassifier.

Class type	Recall (%)	Precision (%)	F_1 value (%)
Normal	99.75	99.40	99.57
NMRI	93.81	98.52	96.11
CMRI	99.11	99.47	99.29
MSCI	94.88	94.88	94.88
MPCI	98.05	99.68	98.86
MFCI	95.81	100	97.86
DOS	96.03	100	97.97

We found that the performance of this research method on 10% data set is no better than the complete set, both on the recall and precision. Only three categories (Normal, CMRI and Recon) scored higher than 90%, and only Recon results are at the same level as the original performance. This seems to prove that data set size does affect the performance of deep learning methods. But in the long run, this is not a serious problem, because the era of industrial big data has arrived.

The performance comparison with other methods is divided into two parts, accuracy performance and time occupancy performance. Due to the different evaluation metrics used in different references, it is difficult to perform a full performance comparison for the same method. Considering that the time performance of similar methods should be similar, we compare the accuracy performance and time occupancy performance of different methods using the same evaluation metrics.

Table 7: Performance on two data sets (%).

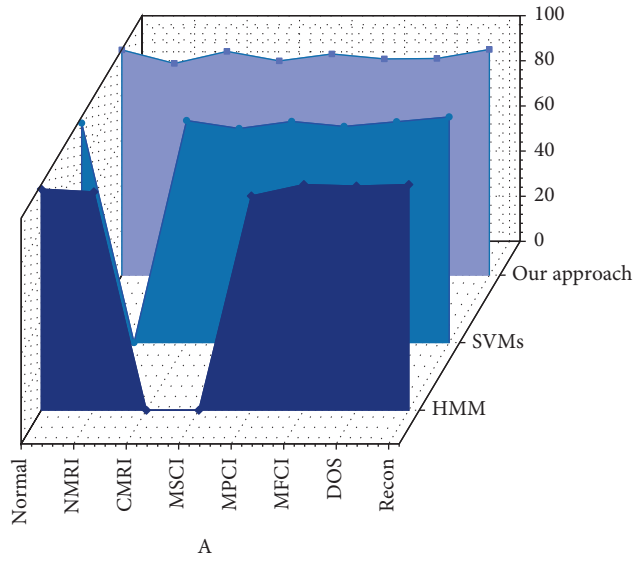
	Reduced data set		Complete data set	
Class type	Recall	Precision	Recall	Precision
Normal	92.25	91.26	99.75	99.40
NMRI	31.34	88.33	93.81	98.52
CMRI	98.19	97.20	99.11	99.47
MSCI	56.98	72.61	94.88	94.88
MPCI	86.93	70.24	98.05	99.68
MFCI	21.95	28.12	95.81	100
DOS	22.97	33.98	96.03	100
Recon	100	100	100	100

Table 8: Accuracy of different methods.

Detection method	Accuracy (%)
SVM ensemble method	94.5
HMM	93.4
Decision tree method	93.1
Our method	99.3

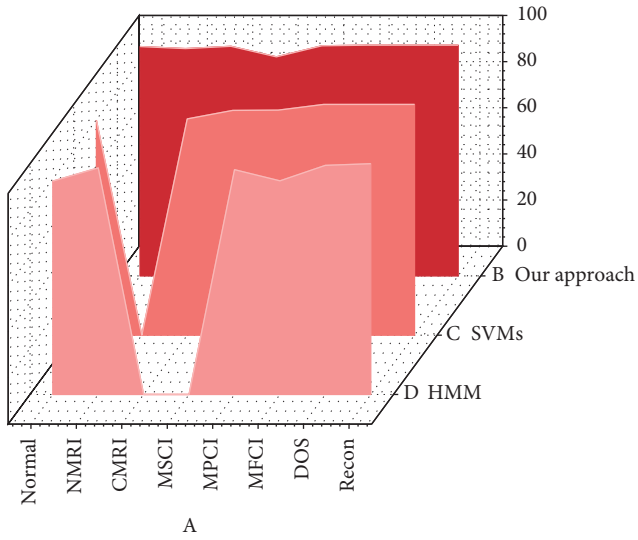
We compared the accuracy performance of our method with the following three methods: the SVM ensemble method derived from the study of Nader et al. [16] that used multiple support vector machine combinations to implement the construction of a multiclassifier, and the HMM [30] and decision tree method [31] which are classical machine learning algorithms.

Table 8 shows the overall accuracy of the four multiclassification methods. The performances of the traditional machine learning methods were similar. The accuracies of



	HMM	SVMs	Our approach
Normal	98	97.3	99.8
NMRI	96.7	0	93.8
CMRI	0	98.4	99.1
MSCI	0	94.9	94.9
MPCI	94.9	98	98.3
MFCI	100	95.8	95.8
DOS	99.3	97.9	96
Recon	100	100	100

Figure 4: Recall of different approaches (%).



	HMM	SVMs	Our approach
Normal	92.5	93	99.4
NMRI	98.2	0	98.5
CMRI	0	93.7	99.5
MSCI	0	97.4	94.9
MPCI	97.4	97.5	99.7
MFCI	92.6	100	100
DOS	99.3	100	100
Recon	100	100	100

Figure 5: Precision of different approaches (%).

the SVM ensemble method, the HMM method, and the decision tree method were 94.5%, 93.4%, and 93.1%, respectively. In contrast, our proposed anomaly classification method based on the CNN achieved the highest overall accuracy of 99.3%.

Figures 4 and 5 depict the performance of different methods in recall and precision, respectively. As can be seen from the recall analysis, our method obtained the highest score in five categories (Normal, CMRI, MSCI, MPCI, and Recon) where the instance of the normal state and the reconnaissance attack were nearly perfectly classified. In the case of the DoS attack, our method had a recall of 96%, which was slightly lower than that of the HMM and the SVM ensemble method. The precision of most types of traffic obtained through our method was mostly maintained at a high level above 98%, except for the MSCI attack with a detection precision of 94.9%. Note that this value was only 2.5% lower than that of the SVM ensemble method.

In addition, we found the metric values were zero in some situations. The HMM performed poorly at detecting both the CMRI attack and the MSCI attack, which the authors believe was owing to flaws in the training set [26]. Similar to the previous situation, the SVM ensemble method did not detect an NMRI attack; the metric value of both recall and precision was zero. However, in contrary to the above methods, our method successfully identified all kinds of attacks. In the detection of an NMRI attack, our method achieved 93.8% recall and 99.7% accuracy. Our method also had excellent performance in the CMRI attack and the MSCI attack which was not detected by the HMM. From the overall trend, the fluctuation range of our deep learning model's indicators was smaller than that of other methods,

Table 9: Time occupancy of different methods.

Detection method	Time occupancy per instance (s)
SAM-kNN	0.099
Pegasos	0.101
ARF	0.084
e-SREBOM	0.089
Our method (feature mapping)	0.253
Our method (detection)	0.192

and the recall rate and accuracy were concentrated in the ranges 93%–100% and 94%–100%, respectively. This phenomenon indicated that our proposed classifier had a more balanced performance.

We compared the time occupancy performance of our method with the following methods [32]: the self-adjusting memory (Sam) model for the k-nearest neighbor (k-NN) algorithm (SAM-kNN), the primal estimated subgradient solver for support vector machines (SVM) algorithm (Pegasos), the adaptive random forests (ARF) algorithm, and the evolving spiking restricted Boltzmann machines algorithm (e-SREBOM). The comparison results are shown in Table 9.

By comparison, it can be seen that the CNN operation consumes more time, but the difference is not large, and the Kappa statistic of the e-SREBOM is 74.31, which is the best method of accuracy in the comparison method. Considering that GPUs can greatly improve the computational efficiency of CNN and the continuous improvement of computing power in recent years, our method still has great advantages.

6. Conclusions and Future Work

ICSs are the lifeblood of countries, and it is necessary to implement anomaly detection to ensure their security. In this study, we proposed a method based on deep learning to achieve anomaly detection and attack classification for SCADA systems. Considering the characteristics of the relationship between the various features of ICSs, a feature mapping method based on the Mahalanobis distance was proposed. Our feature mapping method converted one-dimensional flow data into a two-dimensional matrix to be used as a CNN input.

The experimental results show that the proposed method achieved excellent performance on both the two-class problem and the multiclass problem, met the expected requirements of SCADA anomaly detection and attack classification, and provided assistance for the safety of an ICS.

At present, the method we proposed cannot detect new types of attacks, but it can theoretically detect the corresponding variant attacks by learning the basic knowledge and principles of existing attacks. In the future, we will design attacks according to the characteristics of SCADA scenarios and prove the effectiveness of our CNN method. At the same time, we will evaluate the possibility of using other deep learning methods in the SCADA anomaly detection and attack classification to achieve better industrial security defense.

Data Availability

The data used to support the findings of this study are included within the article. The original dataset we used in the experiments is industrial control system (ICS) cyber-attack dataset, which is published by Tommy Morris and Wei Gao from Mississippi State University. All data can be accessed from https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the Qinghai Province Natural Science Foundation (2017-ZJ-91), the National Natural Science Foundation of China (61872015), the Foundation of Science and Technology on Information Assurance Laboratory (No. 614211204031117), and the Beijing Polytechnic Research Fund (2017Z004-008-KXZ and 2018Z002-019-KXZ).

References

[1] O. Gonda, "Understanding the threat to SCADA networks," *Network Security*, vol. 2014, no. 9, pp. 17-18, 2014.

[2] N. Falliere, L. O. Murchu, and E. Chien, *W32.Stuxnet Dossier*, White Paper, 2011.

[3] Y. Yang, K. McLaughlin, S. Sezer et al., "Multiattribute SCADA-specific intrusion detection system for power networks," *IEEE Transactions on Power Delivery*, vol. 29, no. 3, pp. 1092–1102, 2014.

[4] S. Khandelwal, "Stuxnet-like "Havex" malware strikes European SCADA systems," June 2014, https://thehackernews.com/2014/06/stuxnet-like-havex-malware-strikes.html.

[5] Y. Mo, R. Chabukswar, and B. Sinopoli, "Detecting integrity attacks on SCADA systems," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1396–1407, 2014.

[6] A. Almalawi, X. Yu, Z. Tari, A. Fahad, and I. Khalil, "An unsupervised anomaly-based detection approach for integrity attacks on SCADA systems," *Computers & Security*, vol. 46, pp. 94–110, 2014.

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[8] A. Khalili and A. Sami, "SysDetect: a systematic approach to critical state determination for industrial intrusion detection systems using apriori algorithm," *Journal of Process Control*, vol. 32, no. 11, pp. 154–160, 2015.

[9] A. Carcano, I. N. Fovino, M. Masera, and A. Trombetta, "State-based network intrusion detection systems for SCADA protocols: a proof of concept," in *Proceedings of the International Conference on Critical Information Infrastructures Security*, Berlin, Germany, October 2009.

[10] N. Goldenberg and A. Wool, "Accurate modeling of Modbus/TCP for intrusion detection in SCADA systems," *International Journal of Critical Infrastructure Protection*, vol. 6, no. 2, pp. 63–75, 2013.

[11] C. Markman, A. Wool, and A. A. Cardenas, "A new burst-DFA model for SCADA anomaly detection," in *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy—CPS '17*, Oslo, Norway, September 2017.

[12] P. M. Nasr and A. Y. Varjani, "Alarm based anomaly detection of insider attacks in SCADA system," in *Proceedings of the Smart Grid Conference (SGC)*, Tehran, Iran, December 2015.

[13] V. L. Do, L. Fillatre, and I. Nikiforov, "A statistical method for detecting cyber/physical attacks on SCADA systems," in *Proceedings of the 2014 IEEE Conference on Control Applications (CCA)*, pp. 364–369, Juan Les Antibes, France, October 2014.

[14] W. Shang, L. Li, M. Wan, and P. Zeng, "Industrial communication intrusion detection algorithm based on improved one-class SVM," in *Proceedings of the 2015 World Congress on Industrial Control Systems Security (WCICSS)*, London, UK, December 2016.

[15] C. Zhou, S. Huang, N. Xiong et al., "Design and analysis of multimodel-based anomaly intrusion detection systems in industrial process automation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 10, pp. 1345–1360, 2015.

[16] P. Nader, P. Honeine, and P. Beauseroy, "Lp-norms in one-class classification for intrusion detection in SCADA systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2308–2317, 2014.

[17] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[18] T. Bodström and T. Hämäläinen, "State of the art literature review on network anomaly detection with deep learning," in *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, pp. 64–76, Springer International Publishing, Cham, Switzerland, 2018.

[19] A. Y. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the EAI International Conference on Bio-Inspired Information and Communications Technologies*, pp. 21–26, Hoboken, NJ, USA, 2016.

[20] V. L. L. Thing, "IEEE 802.11 network anomaly detection and attack classification: a deep learning approach," in *Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, USA, March 2017.

[21] B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," *IEEE Access*, vol. 6, pp. 41238–41248, 2018.

[22] M. Ashrafuzzaman, Y. Chakhchoukh, A. A. Jillepalli et al., "Detecting stealthy false data injection attacks in power grids using deep learning," in *Proceedings of the 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 219–225, Limassol, Cyprus, June 2018.

[23] D. Wilson, Y. Tang, J. Yan, and Z. Lu, "Deep learning-aided cyber-attack detection in power transmission systems," in *Proceeding of the 2018 IEEE Power & Energy Society General Meeting (PESGM)*, pp. 1–5, Portland, OR, USA, August 2018.

[24] H. Wang, J. Ruan, G. Wang et al., "Deep learning-based interval state estimation of AC smart grids against sparse cyber attacks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4766–4778, 2018.

[25] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[26] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[27] W. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of the IIEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 53–58, Ottawa, Canada, July 2009.

[28] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection," *International Journal of Engineering Research and Technology*, pp. 65–78, 2013.

[29] T. Morris and G. Wei, "Industrial control system traffic data sets for intrusion detection research," in *IFIP Advances in Information and Communication Technology*, Springer, Berlin, Germany, 2014.

[30] A. Almalawi, X. Yu, Z. Tari, A. Fahad, and I. Khalil, "An unsupervised anomaly-based detection approach for integrity attacks on SCADA systems," *Computers & Security*, vol. 46, pp. 94–110, 2014.

[31] W. Gao, "Cyberthreats, attacks and intrusion detection in supervisory control and data acquisition networks," Dissertations & Theses—Gradworks, Mississippi State University, Starkville, MS, USA, 2013.

[32] L. Xing, K. Demertzis, and J. Yang, "Identifying data streams anomalies by evolving spiking restricted Boltzmann machines," *Neural Computing and Applications*, pp. 1–15, 2019.