Computer Science and Information Systems 21(4):1725–1743 https://doi.org/10.2298/CSIS240314055S

Stacked Denoised Auto-encoding Network-based Kernel Principal Component Analysis for Cyber Physical Systems Intrusion Detection in Business Management

Zhihao Song

School of Business Administration, Zhengzhou University of Science and Technology
Zhengzhou 450064 China
zsshanln@163.com

Abstract. At present, the network data under the environment of industrial information physical system is larger and more complex. Traditionally, feature extraction by machine learning is cumbersome and computation-intensive, which is not conducive to anomaly detection of industrial network data. To solve the above problems, this paper proposes a stacked denoised auto-encoding network based on kernel principal component analysis for industrial cyber physical systems intrusion detection. Firstly, a novel kernel principal component analysis method is used to reduce the data feature dimension and obtain a new low-dimension feature data set. Then, a multi-stacked denoised auto-encoding network model is used to classify and identify the data after dimensionality reduction by voting. Experimental results show that the proposed method has better classification performance and detection efficiency by comparing the state-of-the-art intrusion detection methods.

Keywords: industrial cyber physical systems, intrusion detection, stacked denoised auto-encoding network, kernel principal component analysis.

1. Introduction

With the development of industrial control system information technology, more and more industrial control networks are connected with external public networks [1], forming an open network of information sharing, resulting in the increase of global industrial control network attack events. The risk root of industrial control system mainly includes the defects of system protocol technology, the loopholes of industrial software and the congenital deficiency of hardware design. And the industrial network traffic has the characteristics of stability, correlation, self-similarity and the difference between the network protocol technology and the general protocol, so that the traditional network anomaly detection scheme cannot be simply transplanted into the industrial network [2,3]. The methods of industrial intrusion detection are mainly misuse intrusion detection and abnormal intrusion detection. Misuse intrusion detection technology is a kind of model based on known intrusion behavior data, and then matches all the newly accessed data, and determines that the matched data has the intrusion behavior. Therefore, the misuse of intrusion detection technology can accurately describe the characteristics of attack behavior, reduce the false alarm rate of the system, but it can improve the missing alarm rate. The abnormal intrusion detection technology is a model based on the data of normal behavior, and then matches all the new access technology, and determines the data of matching as normal behavior [4]. This abnormal intrusion detection technology reduces the false alarm rate, but increases the false alarm rate at the same time.

In the application of industrial intrusion detection, attack types emerge one after another. At present, scholars have carried out many researches on invasion attack and security protection. The used algorithms include statistics-based [5], rule inference-based [6] and neural networks-based methods [7]. Shang et al. [8] selected Modbus/TCP function code as the research object, combined with SVM (Support Vector) Machine), he proposed a Modbus/TCP communication function code sequence anomaly detection method based on SVM, which realized the identification of attack behavior and anomaly behavior that firewall and intrusion detection system could not identify. But the study did not involve learning from new data later on. Therefore, the complexity of industrial network environment cannot be satisfied. Wan et al. [9] proposed a Modbus/TCP communication access control method based on function code deep detection based on deep packet resolution technology without changing Modbus/TCP protocol, which not only realized traditional function code and address range detection, but also realized range detection. However, the system used fixed protocol to match, and also had poor learning ability, which could not meet the complexity and expansibility of industrial network. The intrusion detection system designed by Ferrag et al [10] analyzed the telemetry data in the industrial control network and used the REPTree decision tree algorithm to identify the communication between different machines according to the hop number of the data propagating in the network, so as to distinguish the attacker and the engineer machine inside the network. This method could effectively identify the threats within the network. Ocaa et al. [11] converted Modbus RTU/ASCII data in the industrial control network to Modbus/TCP. Then it used the improved Snort software for threat analysis, it could effectively protect against denial of service, response injection, command injection and reconnaissance attack. Finally, the detected data was converted to the original Modbus RTU/ASCII format again. But this method could not effectively solve the attack of fake data injection in industrial network. Tong et al. [12] proposed two OCSVM intrusion detection algorithms to be applied in fault detection, and the Gaussian kernel parameter optimization models were DFN and DTL, respectively. In the application of fault diagnosis, it could effectively detect various types of faults and improve the accuracy. But the false detection rate had also increased significantly. This paper proposes a stacked denoised auto-encoding network based on kernel principal component analysis for industrial cyber physical systems intrusion detection. It shows a better detection efficiency compared with other methods through experiments.

This paper is organized as follows. In section 2, we give the related works. Section 3 detailed shows the proposed intrusion detection method. Experiments and analysis are displayed in section 4. There is a conclusion in section 5.

2. Related Works

In recent years, with the rapid development of automation control and communication, computer and other related technologies, the theory and application of cyber physical systems (CPS) have also been developed rapidly [13]. CPS integrates computer, communication network and control technology efficiently, so that the physical system is equipped with precision control, remote collaboration and autonomy. CPS is applied to industry, forming industrial cyber physical system systems (ICPS), which is widely used in key fields such as intelligent energy, intelligent mobility and intelligent manufacturing [14]. ICPS requires carrier-level quality of service for bearer networks, and puts forward higher requirements for network security, reliability, controllability and manageability [15].

The network security of communication bridge is particularly important. Intrusion Detection Systems (IDS) is another network security barrier of ICPS after network firewall. It mainly detects internal and external attacks through monitoring and analyzing activities on host or network [16]. Intrusion detection technology can be divided into two types. One is network-based IDS, which captures the communication data packets between the main control center and the site or on-site devices to evaluate whether the data poses threats to the system. According to different traffic characteristics, the detection model is established and the abnormal behavior of the system is detected. The other is application-based IDS. ICPS stores control and measurement data generated by sensors and actuators in a history server. Application-based IDS reads the application data of the history server to detect whether the system is attacked [17]. Since the information source of IDS based on ICPS application can be collected from different remote field devices such as programmable logic controllers and remote terminal units. Therefore, IDS based on ICPS applications can be deployed in a variety of ways, such as on history servers, standalone servers, and off-line servers.

At present, most of the intrusion detection systems of ICPS use the traditional machine learning methods to extract useful information from labeled data sets to train the detection model. However, the high-speed interconnection between devices in today's network environment leads to massive data, which brings new challenges to IDS design. How to realize stable and efficient intrusion detection in the network environment with massive data is the primary problem in the current research. Abbasi et al. [18] proposed a sensor cloud intrusion detection algorithm based on parallel discrete optimization feature extraction and machine learning method in fog computing mode to effectively improve the success rate and miss rate of anomaly detection in view of large-scale high-dimensional data and variable intrusion behavior of sensor cloud. Wook et al. [19] proposed a new method based on partial least squares (PLS) and core vector machine. PLS algorithm was used to extract the principal component of network data to construct feature set, and then CVM was used to complete abnormal intrusion detection and judgment. This method could deal with large-scale data quickly. Hadeel et al. [20] proposed a feature selection pigeon optimizer algorithm for intrusion detection systems. The algorithm performed well in true positive rate (TPR), false rejection rate (FRR), accuracy (Acc) and F-measure. Guo et al. [21] proposed a one-class SVM based on incremental single-class support vector machine. OCSVM algorithm was used to learn the normal behavior communication mode from Modbus/TCP data information, and the current training sample set was reduced by using adjacent classification interval and KKT condition. The reduced samples were trained again to improve the incremental learning speed of intrusion detection system while ensuring higher classification accuracy. Lv et al. [22] proposed a new intrusion detection model based on optimal mixing and functional extreme learning machine. This method used gravity search algorithm and differential evolution algorithm to optimize the parameters of hybrid kernel extreme learning machine, so as to improve the global and local optimization ability in the process of attack prediction, and introduced kernel

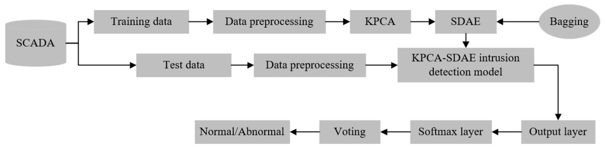

Fig. 1. Framework of ICPS intrusion detection model based on KPCA-SDAE

principal component analysis algorithm to reduce the dimension and feature extraction of intrusion detection data, with high detection accuracy and time saving.

The above methods can improve detection accuracy and other performance from different perspectives, but the following problems need further study. First, the network data in ICPS presents complex nonlinear relations, resulting in low correlation between feature attributes in the data set. Therefore, they are not suitable to use traditional linear dimensionality reduction strategies such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to screen out highly relevant features. Second, network data in industrial scenarios are faced with more severe noise interference than traditional IT networks, which may lead to data redundancy or even some missing attribute values. The above methods cannot effectively solve these problems. Therefore, this paper studies a deep learning hybrid strategy detection model to enhance the anti-jamming ability of the system. The improved KPCA is introduced as the data dimension reduction model for the system to enhance the robustness of the acquired data. Integrated learning method is used to integrate stacked denoised auto-encoding network with different structures, and softmax classifier is used to classify and identify the data after dimensionality reduction, so as to reduce the dependence of system performance on the parameter structure setting of a single deep belief network.

3. ICPS Intrusion Detection Model Based on KPCA-SDAE

KPCA structure and stacked denoised auto-encoding (SDAE) network model are used to construct ICPS intrusion detection framework, namely, KPCA-SDAE, as shown in Figure 1.

In the training stage, the original communication traffic data obtained from SCADA system is preprocessed, and then imported into KPCA model for feature extraction and data dimension reduction. Then, it adds labels to the data after dimensionality reduction as training samples of SDAE. Then Bagging technology is used to divide the training data. The KPCA-SDAE model is constructed by using the data of different groups to train multiple SDAE networks and fix the parameters of SDAE model. The parameters of KPCA-SDAE model are adjusted by supervised learning algorithm.

In the test phase, the preprocessed data is fed into the KPCA-SDAE model. After the output layer, the binary classification or multiple classification is realized by Softmax classification, and finally the Voting mechanism is adopted to determine the final predicted value. The performance of the intrusion detection algorithm is evaluated by ACC and FN. The main components of ICPS intrusion detection model based on KPCA-SDAE are KPCA data dimension reduction model and ensemble classifier construction,

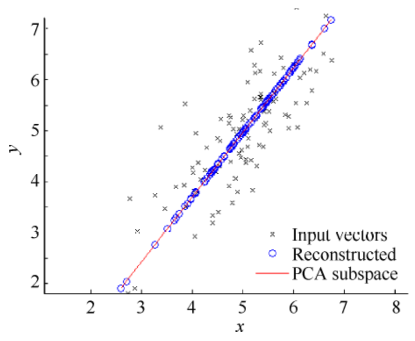

Fig. 2. PCA process

in which KPCA data dimension reduction model is mainly used to eliminate data redundancy and conduct data dimension reduction processing, and ensemble classifier construction is mainly used to classify and identify processed data.

3.1. Dimension Reduction Optimization Based on KPCA

Basic theory of principal component analysis (PCA).

Assuming that m samples have been selected and each sample contains d variables (d-dimensions) [23], the sample matrix can be expressed as:

$$x_{d\times m} = [x^1, \cdots, x^j, \cdots, x^m]. \tag{1}$$

The $j-th$ sample is $x^j = [x^j_1, x^j_2, \cdots, x^j_d]^T$. The basic principal component analysis is how to project sample points onto a hyper-plane, and make the projection of sample points as separate as possible on this hyper-plane, so as to find the dominant direction. Let the projection of our sample point x^j onto our new space be $U^T x^j$. U^T is the projection coefficient matrix. In order to keep the projection of the sample points as separate as possible, the variance of the sample points after projection should be maximized, as shown in Figure 2.

The variance of sample points after projection is $\sum_{j=1}^{m} U^T x^j (x^j)^T U$, so the projection matrix must satisfy:

$$\max_{U} tr(U^T X X^T U), s.t. U^T U = I. \tag{2}$$

Formula (2) can be obtained by Lagrange multiplier method.

$$X X^T u^i = \lambda_i u^i (i = 1, 2, \cdots, d). \tag{3}$$

The eigenvalue decomposition of Equation (3) is carried out, and the obtained eigenvalues are sorted from large to small $\lambda_1 > \lambda_2 > \cdots > \lambda_d$. Then it takes the eigenvectors corresponding to the first d' eigenvalues to form the reconstruction space. d' value can be selected according to the minimum dimension established by the following formula of reconstruction threshold t [12].

$$\frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq 1. \tag{4}$$

KPCA.

It is obvious that PCA method directly discards feature vectors after in the process of spatial reconstruction, ignoring a lot of information. And from high dimensional space to low dimensional space, it is a linear mapping, so it has certain limitations. Kernel principal component analysis [24] in nonlinear dimensionality reduction is one of the effective ways to solve this problem.

To introduce the kernel method, the following transformation is performed on Equation (3).

$$(\sum_{i=1}^{m} z_i z_i^T) u^i = \lambda_i u^i. \tag{5}$$

Where z_i is the image of sample point x^i in higher dimensional feature space. It can be known from Equation (5).

$$u^j = \frac{1}{\lambda_i} (\sum_{i=1}^{m} z_i z_i^T) u^j = \sum_{i=1}^{m} z_i \alpha_i^j. \tag{6}$$

Where $\alpha_i^j = \frac{z_i^T u^j}{\lambda_j}$. Suppose $z_i = \varphi(x^i)$, so equation (5) is transformed into:

$$(\sum_{i=1}^{m} \varphi(x^i) \varphi(x^i)^T) u^j = \lambda_j u^j. \tag{7}$$

Equation (6) is transformed into:

$$u^j = \sum_{i=1}^{m} \varphi(x^i) \alpha_i^j. \tag{8}$$

By introducing the kernel function,

$$\kappa(x^i, x^j) = \varphi(x^i)^T \varphi(x^j). \tag{9}$$

After substituting Equation (8) and Equation (9) into Equation (7), we can get,

$$K \alpha^j = \lambda_j \alpha^j. \tag{10}$$

Where $\alpha^j = [\alpha_1^j, \alpha_2^j, \cdots, \alpha_m^j]^2$. The kernel matrix K is:

$$K = \begin{bmatrix} \kappa(x^1, x^1) & \cdots & \kappa(x^1, x^j) & \cdots & \kappa(x^1, x^m) \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ \kappa(x^m, x^1) & \cdots & \kappa(x^m, x^j) & \cdots & \kappa(x^m, x^n) \end{bmatrix}$$

It is obvious that Equation (10) is the eigenvalue solution problem. After obtaining the eigenvalue and eigenvector of Equation (10), the following optimization design space can be reconstructed.

Reconstructing and optimizing the design space based on KPCA.

According to the above theory, for any sample x, after its projection, the j-dimension coordinate in the new design space is:

$$z_j = \sum_{i=1}^{m} \alpha_i^j \kappa(x^i, x). \tag{11}$$

The following kernel function is used in this paper.

$$\kappa(x^i, x^j) = (x^i)^T x^j. \tag{12}$$

Reconstruct the design space and substitute Equation (11) into Equation (10), we can get,

$$z_j = (\sum_{i=1}^{m} u_i^j (x^i)^T) x. \tag{13}$$

Therefore, the relationship between any sample x and the projected coordinate can be written in matrix form, i.e,

$$Z = Ax. \tag{14}$$

Where $Z = [z_1, z_2, \cdots, z_d]^T$. The coefficient matrix A is:

$$A = \begin{bmatrix} \sum_{i=1}^{m} \alpha_i^1 x_1^i & \cdots & \sum_{i=1}^{m} \alpha_i^1 x_d^i \\ \vdots & \cdots & \vdots \\ \sum_{i=1}^{m} \alpha_i^d x_1^i & \cdots & \sum_{i=1}^{m} \alpha_i^d x_d^i \end{bmatrix}$$

Multiply both sides of equation (14) by A^{-1}, we can get the following formula,

$$x = A^{-1} Z. \tag{15}$$

It is obvious that equation (14) establishes the mapping relationship between the original variable and the projected variable. Equation (15) gives the mapping relationship between projection variables and original variables, which is the key to optimize the design space after dimensionality reduction. The two relations can be used to reconstruct the optimization space and search for optimization in the reconstructed optimization space. Then the optimization results are re-mapped back to the original optimization space.

The dimension reduction optimization algorithm based on linear kernel principal component analysis is shown in **Algorithm 1**.

3.2. SDAE Network

Auto-encoding (AE) network [25] belongs to the non-recurrent neural network in feed-forward neural network, which is a 3-layer unsupervised learning network, including encoder and decoder, as shown in Figure 3.

The encoder transforms the input sample $x_i (i = 1, 2, \cdots, n)$, where n is the sample number, into a lower-dimensional space through the activation function $h(\cdot)$.

Algorithm 1 Linear kernel principal component analysis.

```
1: Input. Get sample matrix $X_{d\times m} = [x^1, \cdots, x^j, \cdots, x^m]$. And it is normalized and decentralized to form standardized data.
2: Output. Optimized solution.
3: Calculate the kernel matrix $K$ corresponding to the sample matrix using the above standardized data. The eigenvalues and eigenvectors of the matrix are obtained by eigenvalue decomposition.
4: Order the eigenvalues corresponding to the above determined eigenvectors from large to small, and determine the reconstruction threshold.
5: Use the sorted feature vectors in (3) to calculate the coefficient matrices $A$ and $A^{-1}$.
6: The original design space was reduced in dimension, and the design variables after dimension reduction were optimized.
7: End
```

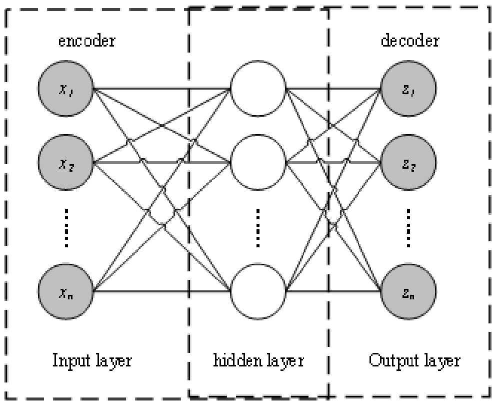

Fig. 3. AE network structure

$$y = h(xw + b). \tag{16}$$

Where y is the output matrix of the hidden layer. x is the input sample matrix, $x = [x_1, x_2, \cdots, x_n]$. w is the weight matrix of the encoder. b is the bias matrix of the encoder.

The decoder inverts the signal y from the low-dimensional space to the high-dimensional space through the activation function $f(\cdot)$, and then reconstructs the input sample.

$$z = f(x) = f(h(xw' + b')). \tag{17}$$

Where z is the reconstructed sample matrix, $z = [z_1, z_2, \cdots, z_n]$. z_i is the input sample of reconstruction. w' is the weight matrix of decoder. b' is the bias matrix of the decoder.

In practical application, the training samples are often mixed with noise, resulting in the deviation of features acquired through AE network learning. Denoised auto-encoding (DAE) network adds noise to AE network and extracts more robust features by learning noisy data [26]. DAE network structure is shown in Figure 4. Conditional distribution C_L is introduced, and the probability of damaged samples is given to obtain the input sample

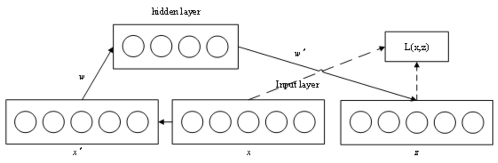

Fig. 4. DAE network

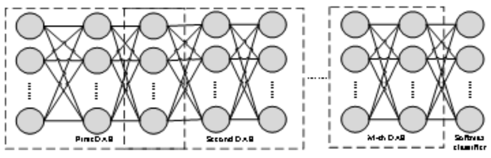

Fig. 5. DAE network

matrix x' after adding noise. The loss function $L(x, z)$ is used to calculate the reconstruction error, and the effective features of the input samples are extracted by minimizing the reconstruction error.

In order to extract deep features of input samples, DAE is stacked layer by layer with deep network structure to form SDAE network, as shown in Figure 5. The input of the hidden layer of the next DAE network is the output of the hidden layer of the previous DAE network, and the network uses unsupervised learning to complete the stacking process. SDAE network, as an unsupervised learning network, can extract deep features of input samples and carry out feature coding, but cannot classify samples. Therefore, the Softmax classifier [27] is added to the last layer of the SDAE network to make the entire network as a supervised learning network.

3.3. Classifier Construction Based on a Novel Bagging Algorithm

DBN (deep belief network) structure is introduced to classify and identify the KPCA model data after dimensionality reduction. Although DBN has strong ability of feature extraction and data mining, it requires artificial selection for the optimal structure (including DBN network layer number and node number) and the optimal parameters (including vector, momentum, etc.). Even the optimal structure of DBN is not fixed. Therefore, the optimization of DBN structure and parameters is a time-consuming process. Therefore, ensemble DBN (EDBN) uses parallel integration of multiple DBN structure learning models to avoid spending a lot of time on DBN structure selection and parameter adjustment, which can effectively reduce the dependence of detection model on the optimal parameters of a single DBN model.

DBN is composed of a series of Restricted Boltzmann machines (RBM) and reverse transmission stacked neural networks. The training steps of DBN model are as follows:

1) Low-level DBN training.

– Initialize model parameters.

Let the initial value of the weight in parameter $\theta_1 = (W_1, a, b_1)$ of the model be $W_1 \in (0,1)$. The initial value of both visible layer bias a and hidden layer bias b_1 is set to 0.

– DBN contrast divergence algorithm.
First, compute the hidden layer output forward. Let $V = (v_1^{c_1}, v_2^{c_1}, \cdots, v_k^{c_1})$ be the $c_1 - th$ updated input value in the visible layer. $w_{i,1j}$ is the connection weight between the $i - th$ neuron in visible layer V and the $j - th$ neuron in the forward hidden layer $H1$, where,

$$h_{1j} = \sum_{i=1}^{k} v_i^{c_1} w_{i,1j} + b_{1j}. \tag{18}$$

Neuron h_{1j} of the hidden layer is activated at the $c_1 - th$ update, that is, the output value of the hidden layer is,

$$P(h_{1j}^{c_1} = 1 | v_i^{c_1}) = S(h_{1j}^{c_1}). \tag{19}$$

Second, reverse reconstruct the visibility layer output. The output of the hidden layer reconstructs the input of the visible layer. At the $c_1 - th$ update, the input value of neuron $v_i^{c_1}$ reconstructed in the visible layer is:

$$v_{i'}^{c_1} = \sum_{j=1}^{l_1} S(h_{1j}^{c_1}) + a_i. \tag{20}$$

Similarly, the probability that neuron v_i of the reconstructed visible layer is activated at the $c_1 - th$ update is the output value of the reconstructed visible layer.

$$P(v_{i'}^{c_1} = 1 | h_{1j}^{c_1}) = S(v_{i'}^{c_1}). \tag{21}$$

Third, reconstruct the output to the front hidden layer. The hidden layer continues to be reconstructed from the output of the reconstructed visible layer. The probability that neuron h_{1j} of the hidden layer is activated again, namely, the output value of the forward hidden layer after reconstruction is,

$$P(h_{1j}^{c_1} = 1 | v_{i'}^{c_1}) = S(h_{1j}^{c_1}). \tag{22}$$

Where $h_{1j'}^{c_1} = \sum_{i=1}^{l_1} S(v_{i'}^{c_1}) w_{i,1j} + b_{1j}$ represents the input after the reconstruction of the forward hidden layer at the $c_1 - th$ update.
Fourth, update model parameters. After the visible layer and hidden layer are reconstructed, the updated model parameter θ_1 can be obtained by the following formulas:

$$w_{i,1j} \leftarrow w_{i,1j} + \lambda \times (S(h_{1j}^{c_1}) \times v_i^{c_1} - S(h_{1j'}^{c_1}) \times v_{i'}^{c_1}). \tag{23}$$

$$a_i \leftarrow a_i + \lambda \times (v_i^{c_1} - v_{i'}^{c_1}). \tag{24}$$

$$b_{1j} \leftarrow b_{1j} + \lambda \times (S(h_{1j}^{c_1}) - S(h_{1j'}^{c_1})). \tag{25}$$

Where λ represents the learning rate.
When $c_1 < T$, repeat step (2) with updated model parameters as initial values. If $c_1 = T$, perform Step (3).

- Stop DBN training. Output the underlying DBN model parameters, namely $\theta_1 = (W_1, a, b_1)$.

2) DBN training.

- DBN pre-training. Suppose that DBN is stacked with three RBMs, and the training process of each RBM is the same as step (1). The output of the RBM hidden layer of the upper layer serves as the input of the visible layer of the RBM of the next layer. When the model parameter $\theta_3 = (w_3, b_2, b_3)$ of the last layer RBM is output, the DBN pre-training is finished.
- (2)Fine-tuning of DBN model. DBN fine-tuning is a supervised training process. In the fine-tuning process, BP algorithm is used to reverse update the parameters of the model, and the cross entropy error function is used as the optimization objective function. The objective function can be calculated as follows:

$$J = \frac{1}{m}\sum_{k=1}^{m}(-\sum_{i=1}^{r_3}(o_i^k \ln \hat{o}_i^k)). \tag{26}$$

Where $\hat{o}_i^k$ and o_i^k respectively represent the output value (predicted value) and real label value of the $i-th$ neuron corresponding to the $k-th$ sample in the DBN output layer. By using the partial derivatives of the cross entropy error function to the weight terms and bias values of each network layer, the weight terms and bias terms of each network layer can be dynamically updated. When the J value converges or reaches the set number of generation selection, the weight items and offset values of each layer are fixed to complete the fine tuning of the system.
EDBN model adopts Bagging algorithm to integrate DBN structure. Assuming that the size of the training set is N, Bagging algorithm selects m sub-training sets with size N (including repeated samples) by self-sampling method. Then, m data sets are used to train m DBNSs with different hyperparameter settings. And m DBN base classifier models can be obtained. Finally, the predictive label value is obtained by majority voting mechanism. The pseudo-codes of Bagging algorithm are described in **Algorithm 2**.

4. Experiments and Results Analysis

In order to verify the feasibility of SDAE-KPCA intrusion detection algorithm, real data obtained from SCADA system is selected as the test data set [28]. Each instance data in the dataset contains 26 feature attributes and one tag attribute. There are eight types of labels: Normal, NMRI (naive malicious response injection attack), CMRI (malicious response injection attack), MSCI (malicious status command injection attack), MPCI (malicious parameter command injection attack), MFCI (malicious function code injection attack), DOS (denial of service attack), and Reco (reconnaissance attack).

The experiment is carried out under Inter Core i6 CPU, 32GB RAM, GTX1060 GPU environment and Windows10 operating system. Python3.6 is used for simulation experiments. The algorithm is implemented by deep learning framework Tensorflow. Accuracy (ACC) and false negative (FN) are used to measure the detection accuracy of IDS, while

Algorithm 2 Bagging algorithm.

```
    Input: Training set $S = (x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)$, where $x_k \in R^l$, number of classifiers $m$.
 2: Output: The final classifier $H_c$.
    for $t = 1, 2, \cdots, m$ do
 4: $S_t =$ sample with Bootstrap(S,N)
    $h_t = L(S_t)$
 6: end for
    sample with Bootstrap(S,N)
 8: $I = \phi$
    for $t = 1, 2, \cdots, N$ do
10: $r = Random - Iterger(1, N)$
    add $S[r]$ to $I$
12: return $I$
    end for
```

training time (TRT) and test time (TET) are used to measure the complexity of the algorithm. FN refers to the probability of abnormal traffic being misjudged as normal traffic in the sample. Considering that in the intrusion detection system, the threat of abnormal traffic being misjudged as normal traffic is greater than that of normal traffic being misjudged as abnormal traffic, FN is selected as one of the detection accuracy indexes.

The dataset contains 80000 data sets , and the distribution of sample data is shown in Table 1. There are only 780 samples from MSCI and 575 from MFSI. According to the ratio of 8:1:1, the sample is divided into three parts: training set, verification set and test set. The training set is used to train the IDS model and simulate the data samples. The validation set is used to adjust the hyperparameters of IDS model. The test set is used to test the accuracy of IDS model to identify network attacks and the generalization ability of detection model. Common parameters include weight values and bias values of each network layer. The hyperparameters include network depth (DAE and DBN layers), sparse index, and the setting of generation selection times, etc.

Table 1. Distribution of sample data

Normal	Anomaly	Anomaly	Anomaly	Anomaly	Anomaly	Anomaly	Anomaly
Type	CMRI	NMRI	MSCI	MPCI	MFCI	DOS	Reco
44407	15466	2763	780	7367	575	1837	6805

After pre-processing and min-max normalization of the original data, 64000 groups of data are transferred into the DAE model in batches after removing label values for pre-training. In order to make the acquired features better represent the original data features, the error between the output value of the hidden layer and the real value of the last layer of the regular de-noising autocoding network in DAE is fine-tuned. Then, the data set of DAE dimension reduction operation is collected by self-help sampling method to obtain the data of m group with the size of 80000. Then, m groups of DBN models are trained, and the optimal DBN models are obtained by changing the DBN network depth, generation times and other super parameters. Bagging ensemble method is used to integrate m DBN models, and the final prediction value is determined by voting mechanism.

Before the final testing of the SDAE-KPCA model, a process of parameter tuning is required. Since there is no automatic method for parameter tuning, empirical parameter tuning is used to adjust parameter settings to achieve the best performance of the model. The initial parameters of DAE model are as follows: network depth=2, learning rate=0.02, L1 regularization coefficient =0.01, sparse penalty factor=0.15 and sparse parameter=0.05, the number of pre-training rounds=100 and the number of fine-tuning rounds=50. The initial parameters of EDBN are set as follows: the number of integrated DBN=3, the network depth of each DBN=3, the learning rate=0.001, the number of training rounds=100, and the number of fine tuning rounds is 50.

In order to test the model's performance in identifying abnormal traffic, the model is tested in detecting binary samples, that is, Normal and Abnormal, and in the multi-classification samples,that is, NMRI, CMRI, MSCI, MPCI, and MFCI, DOS and Reco. The model is compared and analyzed, including the performance of different model parameters and network structure design, as well as the comparison with the current main-stream intrusion detection algorithms.

4.1. Classification Network Parameters Determination

We take the SDAE network depth=2, the learning rate=0.02, the L regularization coefficient =0.01, the sparsity penalty factor $\beta = 0.10$, the sparsity parameter $\rho = 0.05$, the number of pre-training rounds=100, and the number of fine-tuning rounds=50 as the standards. Only one parameter is changed in each group of experiments, and the experimental results are shown in Tables 2-7.

1. Network structure setting.
   Under the premise of fixed learning rate, L regularization coefficient and other parameters, the training error obtained by changing only the depth of network structure and the number of neurons in hidden layer is shown in Table 2 and Table 3. As shown in Table 2, when the network depth is 2 and the hidden layer neuron is set to 30-10, the mean square error (ERR) between the network output value and the real value is the smallest.

Table 2. Error values with different neuron structures

No.	Neuron structure	ERR
1	30-10	0.001657
2	30-20	0.002894
3	30-30	0.002944
4	40-10	0.005368
5	40-20	0.005813
6	40-30	0.005995

2. Parameter setting.

Table 3. Error values with different training periods

Training period	30-10	30-20	30-30	40-10	40-20	40-30
0	0.13	0.13	0.13	0.15	0.20	0.21
2	0.06	0.07	0.12	0.13	0.14	0.18
4	0.06	0.07	0.12	0.13	0.14	0.15
6	0.06	0.07	0.12	0.13	0.13	0.14
8	0.06	0.02	0.12	0.13	0.13	0.14
10	0.01	0.02	0.06	0.10	0.12	0.13
20	0.01	0.02	0.05	0.07	0.12	0.13
40	0.01	0.02	0.05	0.07	0.09	0.09
60	0.01	0.02	0.05	0.07	0.09	0.09
80	0.01	0.02	0.05	0.07	0.08	0.09
100	0.01	0.02	0.05	0.07	0.08	0.09

When SDAE network depth is fixed as 2, network structure is 30-10, learning rate and other parameters are preset values, the training error obtained by changing only the number of pre-training rounds (PTR) or fine-tuning rounds (FTR) is shown in Table 4. It can be seen from Table 3 that when the number of PTR and FTR values are set as 100 and 100, the ERR of the network is the smallest.

Table 4. Error values with different PTR and FTR

No.	PTR	FTR	ERR
1	100	50	0.002874
2	100	100	0.001963
3	100	25	0.003647
4	200	50	0.002971
5	50	50	0.003923
6	200	200	0.002685

Under the condition that the number of PTR and FTR of SDAE is fixed as 100 and 100, the network depth is 3 (50-50-20), the learning rate and other parameters are set as default values, table 5 shows the worth training error of only changing the coefficient penalty factor β or sparse parameter ρ. It can be seen from Table 5 that when β and ρ are set to 0.10 and 0.10, the error value is minimum.
Under the condition that the number of pre-training and fine-tuning rounds of SDAE is fixed as 100 and 100, $\beta = 0.10$ and $\rho = 0.10$, network depth=2, and learning rate is set as the default value, the training error obtained by only changing the L1 regularization coefficient is shown in Table 5. According to Table 6, when L1 regularization coefficient is 0.010, ERR reaches the local minimum value. When the regularization coefficient is too small, it is easy to cause under-fit phenomenon and the error will increase. When the regularization coefficient is too large, it is easy to cause over-fit phenomenon, and the data learning time will increase accordingly.
In SDAE, the number of pre-training and fine-tuning rounds are 100 and 100, $\beta = 0.10$, $\rho = 0.10$, L1 regularization coefficient=0.010, and network depth=2. The initial

Table 5. Error values with different β and ρ

No.	β	ρ	ERR
1	0.10	0.05	0.001943
2	0.20	0.05	0.003029
3	0.25	0.05	0.003141
4	0.10	0.10	0.001878
5	0.20	0.10	0.003104
6	0.25	0.10	0.003829

Table 6. Error values with different L1 regularization coefficients

No.	L1	ERR
1	0.005	0.002447
2	0.010	0.001877
3	0.015	0.003105
4	0.020	0.003218
5	0.025	0.004115
6	0.030	0.005934

learning rate is 0.01, and the results obtained by changing this initial value are shown in Table 7. As can be seen from the table, dynamic changing learning rate will reduce the influence of learning rate on data fitting accuracy. 0.01 is selected as the final SDAE learning rate.

Table 7. Error values with different learning rates

No.	L1	ERR
1	0.01	0.001775
2	0.02	0.001865
3	0.03	0.002467
4	0.04	0.002536
5	0.05	0.002917
6	0.06	0.003348

4.2. Classification Performance Analysis

The parameters obtained above are used in the classification experiments. 64000 training samples are obtained after dimensionality reduction. Self-sampling method is used to extract three groups of data sets, and the data extraction results are shown in Table 8.

Data sets D1, D2 and D2 are used to train SDAE with three different parameter settings respectively. The training results are shown in Table 9. Finally, the three groups of SDAE structure settings are shown in Table 10.

Table 8. Error values with different learning rates

Training set			
Data set	Normal	Anomaly	Total
D1	27000	23000	
D2	26000	24000	50000
D3	27000	23000	
Result			
Data set	Normal	Anomaly	Total
D1	7800	6200	
D2	7500	6500	14000
D3	7300	6700	

Table 9. Accuracy with different training periods

Method	0	10	20	30	40	50	60	70	80	90	100
SDAE1	0.905	0.925	0.926	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935
SDAE2	0.898	0.926	0.927	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935
SDAE3	0.888	0.926	0.927	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935

4.3. Comparison Experiments

The SDAE-KPCA algorithm is compared with other algorithms (including PCA-BP, DBN, MDCO [29], RFSS [30] and IDSGAN [31]) using the same data sets. The dichotomous comparison results of detection models are shown in Table 11. As can be seen from Table 11, compared with PCA-BP, DBN, MDCO, RFSS and IDSGAN model, the accuracy of proposed method increases by 5.12%, 3.90%, 2.32%, 1.85% and 1.26%, the FN decreases by 1.45%, 1.33%, 1.17%, 0.77% and 0.73%, respectively. The time complexity of the proposed algorithm is slightly higher than that of the comparison algorithms, and the TRT training time is higher than that of other three detection methods. However, the test time TET is similar to the comparison algorithm for the test set with a small amount of data. The comparison results of multiple classifications are shown in Table 12. It can be seen from Table 12 that SDAE-KPCA has significantly improved accuracy compared with other algorithms in the case of multiple classifications, and the error leakage rate is relatively low. However, due to the small number of samples of MSCI and MFCI attacks, the detection effect of these two types of attacks is relatively poor, but compared with other comparison algorithms, the detection performance is significantly improved.

Table 10. Three groups of SDAE structure parameter settings

Class	Neuron structure	Learning rate	Discard rate	ACC/%
SDAE1	100-100	0.001	0.2	94.79
SDAE2	100-50	0.002	0.1	94.82
SDAE3	50-100	0.001	0.1	94.83

Table 11. Results of dichotomies with different methods

Method	ACC/%	FN/%	TRT/s	TET/s
PCA-BP	92.67	2.61	37.4	8.8
DBN	93.89	2.49	40.9	9.1
MDCO	95.47	2.33	46.1	9.4
RFSS	96.21	1.93	47.8	10.8
IDSGAN	96.53	1.89	46.3	11.3
SDAE-KPCA	97.79	1.16	59.9	11.9

Table 12. Results of dichotomies with different methods

Type	MDCO	MDCO	RFSS	RFSS	IDSGAN	IDSGAN	SDAE-KPCA	SDAE-KPCA
Type	ACC	FN	ACC	FN	ACC	FN	ACC	FN
Normal	0.959	0.0224	0.953	0.0209	0.961	0.0197	0.983	0.0101
NMRI	0.927	0.00247	0.917	0.0174	0.934	0.0170	0.925	0.0091
CMRI	0.948	0.0204	0.939	0.0219	0.951	0.0207	0.975	0.0183
MSCI	0.776	0.0323	0.794	0.0375	0.903	0.0331	0.914	0.0181
MPCI	0.954	0.0247	0.955	0.0247	0.958	0.0228	0.968	0.0154
DOS	0.987	0.0178	0.981	0.0155	0.986	0.0129	0.999	0.0014
Reco	0.968	0.0135	0.982	0.0168	0.979	0.0154	0.999	0.0092

5. Conclusion

This paper studies an integrated intrusion detection model based on deep auto-encoding network. KPCA is used to reduce data redundancy. Integrating SDAE overcomes the difficulty of DAE structure selection and significantly reduces the dependence of model on DAE parameter setting. Compared with other methods, the proposed algorithm has a better performance in dealing with large-scale industrial network data, which significantly improves the robustness and anti-interference ability of intrusion detection system, while reducing the dimension of data.

References

1. A. Al-Abassi, H. Karimipour, A. Dehghantanha, et al., "An ensemble deep learning-based cyber-attack detection in industrial control system," *IEEE Access*, vol. 8, pp. 83965–83973, 2020.
2. X. Yan, Y. Xu, X. Xing, et al., "Trustworthy network anomaly detection based on an adaptive learning rate and momentum in IIoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6182–6192, 2020.
3. F. Souhe, A. Boum, P. Ele, et al., "Fault detection, classification and location in power distribution smart grid using smart meters data," *Journal of Applied Science and Engineering*, vol. 26, no. 1, pp. 23–34, 2022.
4. M. Novaes, L. Carvalho, J. Lloret, et al., "Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment," *IEEE Access*, vol. 8, pp. 83765–83781, 2020.
5. S. Anton, L. Ahrens, D. Fraunholz, et al., "Time is of the essence: Machine learning-based intrusion detection in industrial time series data," *2018 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE*, pp. 1–6, 2018.

6. J. Liu, W. Zhang, T. Ma, et al., "Toward security monitoring of industrial cyber-physical systems via hierarchically distributed intrusion detection," *Expert Systems with Applications*, vol. 158, 2020.
7. S. Latif, Z. Idrees, Z. Zou, et al., "DRaNN: A deep random neural network model for intrusion detection in industrial IoT," *2020 International Conference on UK-China Emerging Technologies (UCET). IEEE*, pp. 1–4, 2020.
8. W. Shang, P. Zeng, M. Wan, et al., "Intrusion detection algorithm based on OCSVM in industrial control system," *Security and Communication Networks*, vol. 9, no. 10, pp. 1040–1049, 2016.
9. M. Wan, J. Li, H. Luo, et al., "Dynamic Adjusting ABC-SVM Anomaly Detection Based on Weighted Function Code Correlation," *International Conference on Machine Learning for Cyber Security. Springer, Cham*, pp. 1–11, 2020.
10. M. Ferrag, L. Maglaras, A. Ahmim, et al., "Rdtids: Rules and decision tree-based intrusion detection system for internet-of-things networks," *Future internet*, vol. 12, no. 3, pp. 44, 2020.
11. W. Ocaña, C. Alex, G. Ricardo, et al., "Control and monitoring of electrical variables of a level process using Modbus RTU-TCP/IP industrial communication," *Indian Journal of Science and Technology*, vol. 11, no. 32, 2018.
12. W. Tong, B. Liu, Z. Li, et al., "Intrusion Detection Method of Industrial Control System Based on RIPCA-OCSVM," *2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE). IEEE*, pp. 1148–1154, 2019.
13. H. Habibzadeh, B. Nussbaum, F. Anjomshoa, et al., "A survey on cybersecurity, data privacy, and policy issues in cyber-physical system deployments in smart cities," *Sustainable Cities and Society*, vol. 50, 2019.
14. H. Li, S. Yin, J. Liu, et al., "Novel gaussian approximate filter method for stochastic non-linear system," *International Journal of Innovative Computing, Information and Control*, vol. 13, no. 1, pp. 201–218, 2017.
15. J. Lee, M. Azamfar, J. Singh, "A blockchain enabled Cyber-Physical System architecture for Industry 4.0 manufacturing systems," *Manufacturing letters*, vol. 20, pp. 34–39, 2019.
16. K. Rijswijk, L. Klerkx, M. Bacco, et al., "Digital transformation of agriculture and rural areas: A socio-cyber-physical system framework to support responsibilisation," *Journal of Rural Studies*, vol. 85, pp. 79–90, 2021.
17. J. Leng, H. Zhang, D. Yan, et al., "Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop," *Journal of ambient intelligence and humanized computing*, vol. 10, no. 3, pp. 1155–1166, 2019.
18. J. Abbasi, F. Bashir, K. Qureshi, et al., "Deep learning-based feature extraction and optimizing pattern matching for intrusion detection using finite state machine," *Computers & Electrical Engineering*, vol. 92, 2021.
19. M. Wook, N. Hasbullah, N. Zainudin, et al., "Exploring big data traits and data quality dimensions for big data analytics application using partial least squares structural equation modelling," *Journal of Big Data*, vol. 8, no. 1, pp. 1–15, 2021.
20. A. Hadeel, S. Ahmad, S. Eddin, "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," *Expert Systems with Applications*, 2020.
21. Y. Guo, Z. Zhang, F. Tang, "Feature selection with kernelized multi-class support vector machine," *Pattern Recognition*, vol. 117, 2021.
22. L. Lv, W. Wang, Z. Zhang, et al., "A novel intrusion detection system based on an optimal hybrid kernel extreme learning machine," *Knowledge-based systems*, vol. 195, 2020.
23. S. Kanwal, M. TAHIR, H. RAZZAQ, "Principal component analysis and assessment of Brassica napus l. accessions for salt tolerance using stress tolerance indices," *Pak. J. Bot*, vol. 53, no. 1, pp. 113–118, 2021.
24. W. Lee, G. Mendis, M. Triebe, et al., "Monitoring of a machining process using kernel principal component analysis and kernel density estimation," *Journal of Intelligent Manufacturing*, vol. 31, no. 5, pp. 1175–1189, 2020.

25. W. Ma, L. Peng, “Image Quality Transfer with Auto-Encoding Applied to dMRI Super-Resolution,” *2021 4th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE). IEEE*, pp. 828–831, 2021.
26. G. Liu, Y. Fan, J. Zhang, et al., Deep flight track clustering based on spatial-temporal distance and denoising auto-encoding, *Expert Systems with Applications*, vol. 198, 2022.
27. N. Udoh and M. Ekpenyong, “A Knowledge-Based Framework for Cost Implication Modeling of Mechanically Repairable Systems with Imperfect Preventive Maintenance and Replacement Schedule,” *Journal of Applied Science and Engineering*, vol. 26, no. 2, pp. 221–234, 2023. https://doi.org/10.6180/jase.202302_26(2).0008
28. J. Ling, Z. Zhu, Y. Luo, et al., “An intrusion detection method for industrial control systems based on bidirectional simple recurrent unit,” *Computers & Electrical Engineering*, vol. 91, 2021.
29. W. Liang, K. Li, J. Long, et al., “An industrial network intrusion detection algorithm based on multifeature data clustering optimization model,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2019.
30. X. Li, W. Chen, Q. Zhang, et al., “Building auto-encoder intrusion detection system based on random forest feature selection,” *Computers & Security*, vol. 95, 2020.
31. Z. Lin, Y. Shi, Z. Xue, “Idsgan: Generative adversarial networks for attack generation against intrusion detection,” *Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, Cham*, pp. 79–91, 2022.

Zhihao Song is with the School of Business Administration, Zhengzhou University of Science and Technology Zhengzhou 451460 China. Research direction: Business management, Image processing, data analysis, artificial intelligence.

Received: March 14, 2024; Accepted: June 29, 2024.