Intelligent Intrusion Detection of Measurement Automation System Based on Deep Learning

Tao Liu[1], Shaocheng Wu[1], Ziyv Guo[2(✉)], Jie Zhao[1], Wenlong Sun[1], and Xiaohong Cao[1]

[1] Shenzhen Power Supply Bureau Co. Ltd., Shenzhen 518001, Guangdong, China
[2] Beijing University of Posts and Telecommunications, Beijing 100876, China
zyguo@bupt.edu.cn

Abstract. In the smart grid with measurement automation system, the intrusion detection system judges the intrusion event by analyzing the transmission data in the grid. Aiming at the characteristics of traditional intrusion detection, such as loss of features, low detection efficiency and poor adaptability, an intrusion detection method based on stacked denoising convolutional autoencoders is proposed, which combine convolutional neural network and denoising autoencoder to strengthen feature recognition ability, using Dropout and regularization methods to prevent overfitting, and using Adam algorithm to obtain optimal parameters. Finally, the NSL-KDD data set is used to verify the proposed method. Experimental results show that the overall recognition rate of this method is 97.25%, which is 11.59%, 9.63% and 4.07% higher than the existing NN, SVM, and ICNN accuracy rates respectively.

Keywords: Measurement automation system · Intrusion detection · Convolutional neural network · Autoencoder

1 Introduction

In recent years, with the continuous expansion of the grid scale and the integration of different types of energy, the centralized control grid is no longer suitable for the needs of today's society. In order to solve the challenges faced by the existing power grid, a new concept of smart grid has emerged in the power industry [1]. Smart grid is considered as the infrastructure of modern power grid. Through automatic control, high-power converter, modern communication means, sensor metering technology, modern energy management and other technologies, based on the demand, the reliability of energy network is optimized, so as to improve the efficiency and reliability [2].

As one of the cores of smart grid, measurement automation system undertakes the important functions of data acquisition and analysis [3]. With the deepening of the application of measurement automation systems, measurement automation terminals have adopted a large number of wireless communication methods to connect to the smart grid [4]. The wireless communication channel carries important business information such as remote control, telemetry, and remote signaling. Wireless attacks can cause information

X. Sun et al. (Eds.): ICAIS 2021, CCIS 1422, pp. 70–83, 2021.
https://doi.org/10.1007/978-3-030-78615-1_7

tampering or leakage, which can lead to the loss of accuracy and credibility of grid control, cause cascading grid failures or equipment damage, and cause economic losses, and endanger personal and social safety [5]. The complex network environment and endless attack methods also make the metering automation system face many challenges, and related security defense issues have gradually become current research hotpots.

Intrusion Detection System (IDS) is a security management system used to detect network intrusions. It can monitor the network transmission data in real time and take intrusive actions without affecting the internal network [6]. It can take measures such as monitoring, analysis, and early warning to improve the network's ability to respond to external threats. IDS anomaly detection is a behavior-based detection technology that detects unknown attacks by checking whether the actual network behavior deviates from its normal behavior. Many researches on machine learning have developed intrusion detection technologies with machine intelligence, and achieved good results, such as support vector machines, artificial neural networks and genetic algorithms [7–9]. However, because the measurement automation system needs to consider the complexity of wireless communication combined with power service detection, the use of traditional intrusion detection models often has the following problems: (1) The collected network traffic data is high-dimensional, and manual selection of features is not effective enough and has little basis. Important features may be lost and redundant features may be retained. (2) Poor self-adaptation ability. As the network operating environment and structure changes, it is necessary to continuously update the model to detect new and unknown attacks. (3) The model fitting ability is poor. The traditional machine learning model has a simple structure and limited feature extraction and learning capabilities. When faced with a large-scale data set, it cannot form an effective non-linear mapping of the data distribution.

Therefore, in response to the problems of traditional methods, this paper designs and implements a learning method that can automatically extract intrusion features and analysis, and proposes an intrusion detection model based on stacked denoising convolutional autoencoding. The traffic data is processed as two-dimensional gray image as input. Combine the denoising autoencoder with the convolutional neural network, use the convolutional characteristics of the Convolutional Neural Network (CNN), fully learn the network features, solve the model by reconstructing the error, and use the adaptive algorithm Adam to optimize the network, so that it can learn the intrusion characteristics as much as possible and get a better recognition effect.

2 Related Work

Many machine learning algorithms have been applied to intrusion detection, using different machine learning algorithms to reduce the false alarm rate and detect abnormal network behaviors, currently. Shon et al. [10] proposed an intrusion detection classifier that combines genetic algorithm and support vector machine (SVM), which has a wide range of adaptability to real environment data sets; Zhao [11] proposed a Least Squares Support Vector Machine (LSSVM) model for network intrusion detection; Hussain et al. [12] proposed a two-stage hybrid classification method. In the first stage, SVM is used for anomaly detection, and in the second stage, artificial neural network is used for misuse

detection. Traditional machine learning methods are very effective in intrusion detection and have a certain accuracy rate, but they cannot adjust the parameters of preprocessing and feature extraction independently. They need the manual participation of experts to complete the learning and classification goals, and the model performance depends on the quality of parameter tuning and selection features.

In order to solve the above problems, researchers have introduced deep learning technology. In recent years, deep learning has been widely used in speech recognition, image recognition and natural language processing. Since deep learning can automatically extract features from raw data and can effectively process large, complex and multi-dimensional network data, many scholars have also begun to combine deep learning methods with the field of intrusion detection, and have achieved good results. Erfani et al. [13] proposed a hybrid model that combines a deep belief network (DBN) with a single-class vector machine, and first uses restricted Boltzmann machine (RBM) to eliminate the negative effects of noise and abnormal data on the network and then use a single-class vector machine to achieve the classification task. Staudemeyer [14] pointed out for the first time that an LSTM recurrent neural network can be used for intrusion detection. LSTM can learn to look back in time and find some associations from a time perspective. Javaid et al. [15] combined the coding layer of the Sparse Autoencoder (used for feature extraction) and the soft-max function (used for class probability estimation), and designed a "self-learning" classification mechanism for NSL-KDD classification. Khan et al. [16] used CNN-based residual network (ResNet) and GoogleNet model for malware detection. Although the method based on deep learning has improved sample recognition ability and performance, it is prone to over fitting problems in the network training process, and there are many parameters, the training time is long, and the detection accuracy and efficiency need to be further improved.

3 Intrusion Detection Model of Measurement Automation System

The model framework of this paper is an intrusion detection model based on stacked denoising convolutional autoencoding neural network. The overall framework of the model is shown in Fig. 1.

3.1 The Overall Architecture of the Intrusion Detection Model

It can be seen from Fig. 1 that the model mainly includes the following steps for the identification of intrusion detection in the measurement automation system:

Data Acquisition. The measurement automation system environment is built to obtain real-time network traffic data by monitoring and recording network traffic, including source address, target address, connection attribute and other related information.

Data Preprocessing. Process the data into a constructed and processable format. First, the character attributes are mapped to numerical attributes through one-hot encoding, and then the data is normalized to the [0,1] interval to eliminate the influence of the large dimension of different features in the network connection on the training of the intrusion detection model, and finally map the data to a two-dimensional grayscale image.

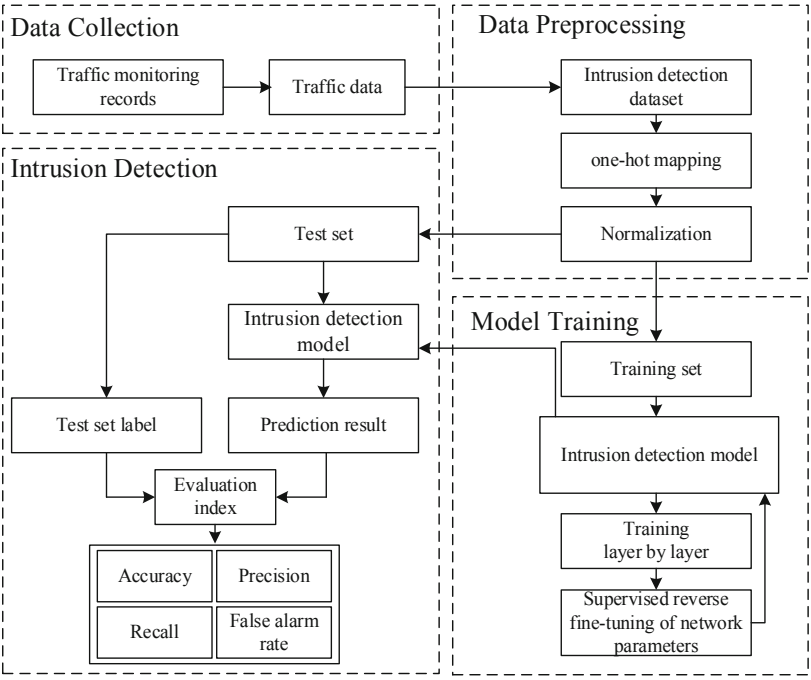

Fig. 1. Intrusion detection model of measurement automation system

Model Building and Training. The stacked denoising convolutional autoencoder model was built to extract and analyze features. It pre-trains and adjusts the parameters of the standard data set to achieve the optimal extraction of standard data features. The input layer of the model is a two-dimensional gray image processing format, and the hidden layer is composed of the encoding and decoding of the convolution layer, the pooling layer and the full connection layer, the activation function of the convolutional layer adopts ReLU to learn feature information independently, the fully connected layer introduces the Dropout method to prevent over-fitting, and the output layer uses the soft-max classifier to output classification decisions.

3.2 Stacked Denoising Convolutional Autoencoding Network

The convolutional denoising autoencoder combines the convolution and pooling operations of the convolutional neural network on the basis of the autoencoder, so as to realize feature extraction and better solve the redundancy and distortion of various data information in the measurement automation system, effectively improve the detection rate. The stacked denoising convolutional autoencoder network designed in this paper first constructs an improved convolutional denoising autoencoder which let the data go through two convolution operations and then perform a pooling operation to strengthen the feature learning ability of the network. Then stack two convolutional denoising autoencoders, the first of which retains the complete structure, and the second retains only the encoding part. The output of the second convolutional denoising autoencoder

is used as the input of two fully connected layers, and finally the recognition result is obtained through the Softmax output layer. The overall network structure is shown in Fig. 2.

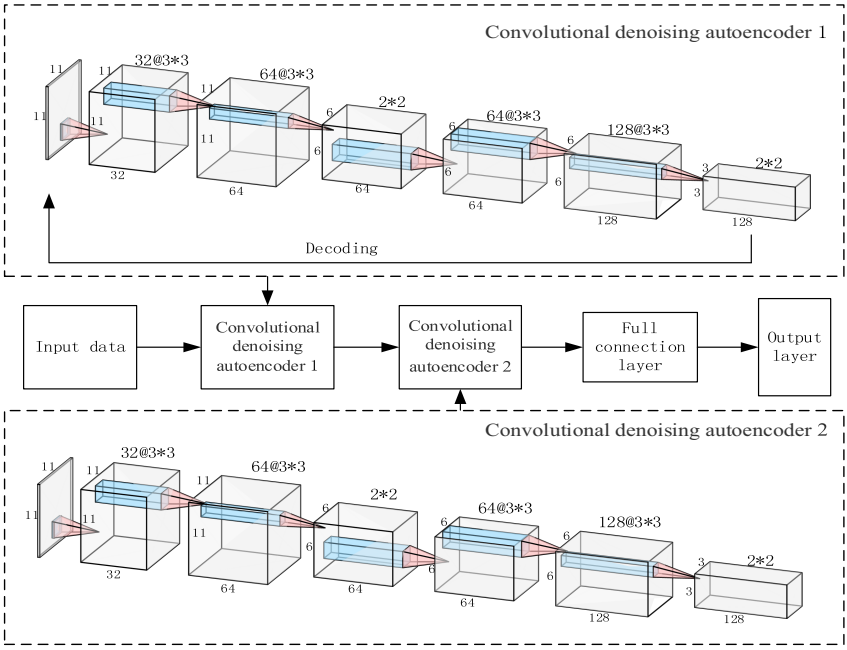

Fig. 2. Overall architecture of stacked denoising convolutional autoencoding network

Convolutional Autoencoding Network. The detailed encoding and decoding process is derived as follows:

Encoding Process:

The output of the convolutional layer can be expressed as:

$$h_1 = f\left(x \otimes W'_{11} + b'_{11}\right) \tag{1}$$

Where x represents the input feature vector, $\otimes$ is a convolution operation, W'_{11} represents the first layer weight, and b'_{11} represents the first layer bias, f is a nonlinear activation function, such as Sigmoid, Tanh, and ReLU. Compared with other activation functions, ReLU can make the network converge faster and reduce the training time. Therefore, this paper adopts the ReLU activation function, which is:

$$f(x)_{\mathrm{ReLU}} = \begin{cases} 0 \ (x \leq 0) \\ x \ (x > 0) \end{cases} \tag{2}$$

The output of the pooling layer can be expressed as:

$$h_2 = pool(h_1) = f\left(down(x) + b'_{11}\right) \tag{3}$$

Where *pool* represents the pooling operation. This paper uses maximum pooling to reduce the information redundancy caused by the convolution operation and $down(\bullet)$ represents downsampling.

Decoding Process:

$$h_2' = f\left(h_2 \otimes W_{22}' + b_{22}'\right) \tag{4}$$

$$h_1' = upsample\left(h_2'\right) = f\left(upsample(x) + b_{22}'\right) \tag{5}$$

$$x' = f\left(h_1' \otimes W_{21}' + b_{21}'\right) \tag{6}$$

Where x' is the reconstructed x, W_{22}' and b_{22}' are the weight and offset of the first layer of convolution during decoding, h_2' is decoded convolution output, *upsample* is upsampling, h_1' is decoded pooled output, f is decoding activation function.

Training of Convolutional Autoencoding Network. The training process of the stacked denoising convolutional autoencoding network is as follow.

Forward Propagation:

① Randomly select batch input from the standard data set, and the data parameter dimension is (batch_size, h, w, c).
② Enter the convolutional denoising autoencoding neural network shown in Fig. 2, and the convolution operation and pooling operation are as shown in Eqs. (7) and (8) respectively.

$$h_j^l = f\left(\sum_{i \in M_J} h_j^{l-1} \otimes W_{ij}' + b_j'\right) \tag{7}$$

$$Z_j^l = \beta\left(W_j^l down\left(Z_j^{l-1}\right) + b_j^l\right) \tag{8}$$

③ Using the conv2d_transpose function and upsample function in Tensorflow to perform deconvolution pooling decoding, and input to the fully connected layer to output the result. The fully connected layer (FC) is calculated as follows:

$$y_j^l = f\left(\sum\nolimits_{i \in M_J} y_j^{l-1} \otimes W_{ij}^l + b_j^l\right) \tag{9}$$

④ Solve the reconstruction error and use Softmax for data classification.

Back Propagation:

① Calculate the overall loss function $J(\omega, b)$ according to the classification results of the training set samples.

② Back propagate the weights and biases of the training network until convergence. In the model training process, in order to speed up the convergence time and improve the convergence accuracy, this paper uses Adam [17] to update the network model parameters. This method solves the problem of slow convergence and easy local optimization, and saves computer resources.

The loss function of this model is:

$$J(\omega, b) = J\left(\omega, b; x^i, y^i\right) + \frac{\lambda}{2} \sum_{l=1}^{n_l-1} \sum_{i=1}^{s_i} \sum_{j=1}^{s_j+1} (\omega_{ji}^l)^2 \tag{10}$$

$$J\left(\omega, b; x^i, y^i\right) = \frac{1}{N} \sum_{i=1}^{N} y \ln a + (1-y) \ln(1-a) \tag{11}$$

Where $J\left(\omega, b; x^i, y^i\right)$ is the cross-entropy loss function, which can reduce the difficulty of different classification problems in unbalanced data. $\frac{\lambda}{2} \sum_{l=1}^{n_l-1} \sum_{i=1}^{s_i} \sum_{j=1}^{s_j+1} (\omega_{ji}^l)^2$ is a regularization term, that is, weight attenuation. Its purpose is to reduce the weight range and prevent training from overfitting. $a = \sigma(h)$, $h = \omega * x + b$. The activation function Sigmoid and its derivatives are shown in Eqs. (12) and (13):

$$\sigma(h) = \frac{1}{1+e^{-h}} \tag{12}$$

$$\sigma'(h) = \frac{e^{-h}}{(1+e^{-h})^2} = \sigma(h)\left(1-\sigma'(h)\right) \tag{13}$$

The cross-entropy derivation of weights and biases is as follows:

$$\frac{\partial J\left(\omega, b; x^i, y^i\right)}{\partial \omega_j} = \frac{1}{N} \sum_{i=1}^{N} \frac{\sigma'_{\omega_j}(h) x_j}{\sigma(h)(1-\sigma(h))} (\sigma(h) - y) \tag{14}$$

$$\frac{\partial J\left(\omega, b; x^i, y^i\right)}{\partial b_j} = \frac{1}{N} \sum_{i=1}^{N} \frac{\sigma'_{b_j}(h)}{\sigma(h)(1-\sigma(h))} (\sigma(h) - y) \tag{15}$$

By substituting into Eq. (13), it can be obtained that:

$$\frac{\partial J\left(\omega, b; x^i, y^i\right)}{\partial \omega_j} = \frac{1}{N} \sum_{i=1}^{N} x_j (\sigma(h) - y) \tag{16}$$

$$\frac{\partial J\left(\omega, b; x^i, y^i\right)}{\partial b_j} = \frac{1}{N} \sum_{i=1}^{N} (\sigma(h) - y) \tag{17}$$

In this paper, Adam optimization algorithm is used to update the weight and bias. The algorithm is shown in Algorithm 1.

Algorithm 1. Adam algorithm

Require:α:Step size
Require:β_1，$\beta_2 \in [0,1)$:Exponential decay rates for the moment estimates
Require:$f(\theta)$ Stochastic objective function with parameters θ
Require:θ_0:Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1st moment vector)
$v_0 \leftarrow 0$ (Initialize 2nd moment vector)
$t \leftarrow 0$ (Initialize time step)
while θ_t not converged do
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at time step t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t/(1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
$\hat{v}_t \leftarrow v_t/(1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t/(\sqrt{\hat{v}_t} + \varepsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

In general, $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$.

4 Experimental Data and Evaluation Index

4.1 Experimental Data Set

In this paper, the NSL-KDD [18] data set is selected as the experimental benchmark data. It is an optimized version of KDDCup99, which solves the problem of redundant data in the KDDCup99 data set. Its original training set KDDTrain contains 125,973 pieces of data, and the original test set KDDTest contains 22,544 pieces of data. This article uses 25,192 pieces of data of KDDTrain +20% as experimental data. Each row of data in the data set has 41 characteristic attributes and 1 label attribute, which mainly include 4 types of attacks: Dos (Denial of Service Attack), Probe (Port Vulnerability Scanning Attack), R2L (Remote Illegal Access Attack), U2R (unauthorized access attack).

4.2 Data Preprocessing

The NSL-KDD data set contains 41 feature attributes, including symbolic features (tcp, udp, icmp, …) and numerical features. The data needs to be standardized before it can be applied to the detection algorithm.

Character Data Mapping Numeric Data
"0, udp, ftp_data, SF, 491, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 1, 0, 0, 150, 25, 0.17, 0.03, 0.17, 0, 0, 0, 0.05, 0, Normal" is a piece of data in the data set. The analysis shows that the 2, 3 and 4 dimensional values are character types and need to be converted to numeric types, for example, the second dimension has three types (tcp, udp, icmp), and the third dimension has ('auth', 'bgp', 'courier', etc.) 70 types, the 4th

dimension has 11 types ('OTH', 'REJ', 'RSTO', etc.), which are processed according to one-hot encoding, and finally 41-dimension is converted into 122-dimension attribute, one-hot encoding is shown in the Fig. 3 shown.

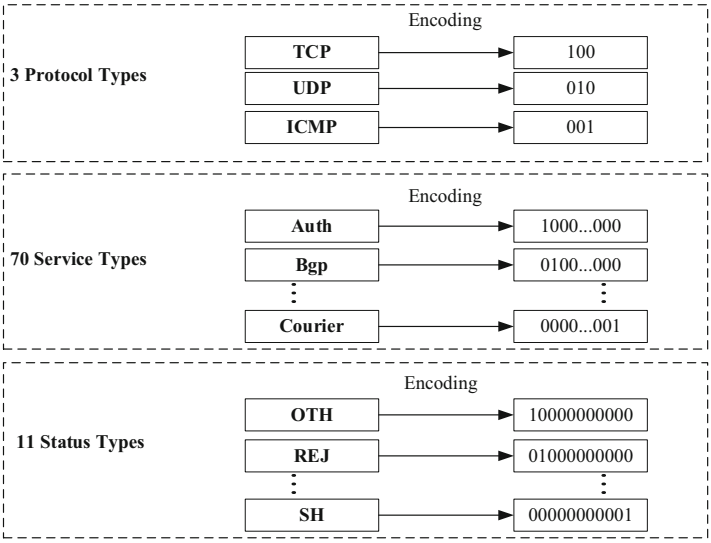

Fig. 3. One-hot encoding

Numerical Normalization. In feature vectors, different features generally have different dimensions and magnitudes. This situation will cause large differences in the size of the different feature values of each sample, which will affect the performance of the model. Features with a large order of magnitude will greatly affect the results of the model classification. Therefore, it is necessary to adopt Min-Max standardized processing to eliminate the influence of different orders of magnitude on the experimental results.

$$X_{normal} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{18}$$

Where x represents the original value of the sample feature, x_{min}, x_{max} represents the minimum and maximum value of the data, and X_{normal} represents the new characteristic value after normalization of each data.

4.3 Evaluation Index

This paper uses a confusion matrix to measure the experimental results, as shown in Table 1.

The evaluation indexes are as follows:

Accuracy (ACC): The ratio of the number of correctly classified samples to the total number of samples.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{19}$$

Table 1. Intrusion detection confusion matrix

Actual	Variable value	
	Normal	Attack
Normal	TN	FP
Attack	FN	TP

Precision (P): The data correctly judged as intrusion/normal accounts for the total number of data predicted as intrusion/normal.

$$P = \frac{TP}{TP + FP} \tag{20}$$

Detection rate/Recall rate (Recall, R): The data correctly judged as intrusion/normal accounted for the total number of intrusion/normal data.

$$R = \frac{TP}{TP + FN} \tag{21}$$

False Alarm Rate (FAR): The number of normal data predicted incorrectly accounts for the total number of normal data.

$$FAR = \frac{FP}{FP + TN} \tag{22}$$

F1-Score: This indicator is the harmonic average of Precision and Recall.

$$F1 - Score = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{23}$$

5 Experiment and Result Analysis

In order to verify the advantages of the intrusion detection method proposed in this paper under the measurement automation system, the intrusion detection model of this paper is simulated, and evaluation indicators are designed to test the performance.

5.1 Experimental Environment and Parameter Selection

This experiment uses Tensorflow to carry out the experiment simulation and chooses the Python programming language. The computer hardware configuration is Inter(R) Core(TM)i7-6700CPU@2.60 GHz processor, 16 GB memory, and the operating system is 64-bit Windows10. The main parameter variables in the model include convolutional autoencoding network structure parameters, learning rate, connection probability, and training times. The specific values of the parameters are shown in Table 2.

Table 2. Experimental variable parameters

Variable name	Variable value
Input layer	Input = (11,11,1)
Convolutional layer 1	Conv1 = (3,3,32), ReLU
Convolutional layer 2	Conv2 = (3,3,64), ReLU
Pool layer 1	Pool1 = Max pooling
Convolutional layer 3	Conv3 = (3,3,64), ReLU
Convolutional layer 4	Conv4 = (3,3,128), ReLU
Pool layer 2	Pool2 = Max pooling
Upsampling layer	Upsample = (6,6)
Convolutional layer 5	Conv5 = (3,3,32), ReLU
Convolutional layer 6	Conv6 = (3,3,64), ReLU
Pool layer 3	Pool3 = Max pooling
Convolutional layer 7	Conv7 = (3,3,64), ReLU
Convolutional layer 8	Conv8 = (3,3,128), ReLU
Pool layer 4	Pool2 = Max pooling
Full connection layer	Number of nodes 512, ReLU
Output layer	Number of nodes 5, Softmax
Dropout	0.6
Learning rate	0.001

5.2 Result Analysis

In order to evaluate the performance of the model in this paper, the model in this paper is compared with the classic network model NN [19], SVM [19], and the improved convolutional neural network model in literature [20]. The results are shown in Table 3 respectively. Figure 4 is a comparison chart of the accuracy of each type of attack in different models.

Table 3. Results of comparison with other models

Model	Accuracy	Recall	FAR	F1-score
This paper	97.25	96.77	0.97	96.90
NN [14]	85.66	84.18	2.31	84.52
SVM [14]	87.62	86.33	1.96	86.75
ICNN [15]	93.18	92.89	1.16	92.91

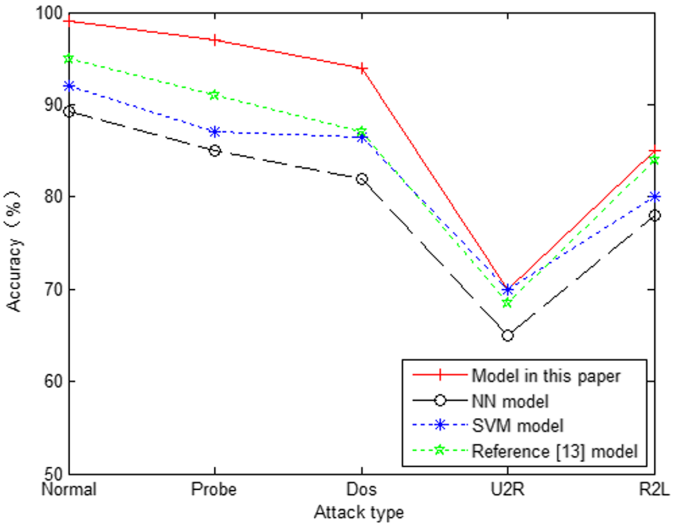

Fig. 4. Comparison of accuracy of various attacks

It can be seen from Table 3 that the accuracy of the model proposed in this paper is 11.59% higher than the NN model, 9.63% higher than the SVM model, 4.07% higher than the improved convolutional neural network model in [13], and the detection rate is respectively improved by 12.59%, 10.44% and 3.88%, it is also better than other models in false alarm rate and F1-Score performance. It can be seen from Fig. 4 that the model proposed in this paper is significantly better than the NN model, SVM model and the model in the literature [13] in the recognition of Normal, Probe, and Dos attacks, and there has also been a slight improvement in the identification of U2R and R2L attacks. In summary, it can be concluded that the model proposed in this paper combining the characteristics of convolutional neural network and autoencoding network can be well applied to the intrusion detection system of the measurement automation system, and has good classification and detection performance.

6 Conclusion

At present, in view of the problems of low recognition efficiency, severe feature loss, and poor adaptive ability in the intrusion detection technology of measurement automation systems, this paper proposes an intrusion detection model based on stacked convolutional denoising autoencoding network. Compared with traditional models, this model can learn the internal characteristics of data more fully. The model in this paper uses Dropout and regularization to avoid the occurrence of over-fitting, and uses Adam to optimize the reconstruction error, speed up the convergence speed, and avoid local optimization. Compared with other models, the accuracy and detection rate of the model proposed in this paper are significantly improved, reaching 97.25% and 96.77% respectively, and the false alarm rate is also slightly reduced.

Although the method in this paper has improved the intrusion detection of the measurement automation system, there are still problems that need to be further solved, mainly in the following three aspects: (1) How to save the node storage space in the measurement automation system and ensure the efficiency of intrusion detection; (2) Considering further optimization of the algorithm for the possible problems of gradient disappearance and local optimization during model training; (3) To further strengthen the generalization ability of the model, try to verify the model with multiple intrusion detection data sets.

Funding Statement. This work was supported in part by National Key R&D Program of China under Grant 2016YFF0204001.

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

References

1. Farhangi, H.: The path of the smart grid. IEEE Power Energy Mag. **8**(1), 18–28 (2010)
2. Gungor, V.C., Sahin, D., Kocak, T., et al.: Smart grid technologies: communication technologies and standards. IEEE Trans. Indust. Inf. **7**(4), 529–539 (2011)
3. Wang, Y., Yang, J.F., Shen, Y.H.: Design and application of metering automation system in large power supply enterprise. Electr. Measure. Instrum. **48**(011), 63–66 (2011)
4. Wu, S.C., Zhou, S.L.: Design and development of smart diagnosis system for station faults in metrology automation. GUANGDONG Electric Power **24**(8), 59–62 (2011)
5. Tang, Y.Q., Li, W., Zhang, L.J.: Endogenous secure communication based on characteristics of wireless channel. Radio Commun. Technol. **46**(274)(02), 31–39 (2020)
6. Allen, J.H., Christie, A., Fithen, W., Willke, B.: State of the Practice of Intrusion Detection Technologies. Carnegie Mellon Software Engineering Institute (2000)
7. Zhang, J.S., Liang, S.J., Zuo, J.L.: Research on Internet of Things intrusion detectionby optimizing SVM using Grey Wolf Optimization algorithm. Inf. Technol. Netw. Secur. **39**(10), 44–48 (2020)
8. Liu, K.K., Xie, F., Guo, X.X.: Cloud computing environment intrusion detection technology based on bp neural network. Comput. Dig. Eng. **12**, 141–145 (2014)
9. Liu, F.F.: Network intrusion detection based on ant colony optimization algorithm selecting parameters of neural network. Mod. Electr. Tech. **21**, 80–83 (2017)
10. Shon, T., Moon, J.: A hybrid machine learning approach to network anomaly detection. Inf. Sci. **177**(18), 3799–3821 (2007)
11. Zhao, F.: Detection method of LSSVM network intrusion based on hybrid kernel function. Mod. Elect. Tech. **38**(21), 96–99 (2015)
12. Hussain, J., Lalmuanawma, S., Chhakchhuak, L.: A two-stage hybrid classification technique for network intrusion detection system. Int. J. Comput. Intell. Syst. **9**(5), 863–875 (2016)
13. Erfani, S.M., Rajasegarar, S., Karunasekera, S., Leckie, C.: High-dimensional and large-scale anomaly detection using a linear one. Pattern Recogn. **58**(C), 121–134 (2016)
14. Staudemeyer, R.C.: Applying long short-term memory re-current neural networks to intrusion detection. South Afr. Comput. J. **56**(1), 136–154 (2015)
15. Javaid, A., Niyaz, Q., Sun, W., et al.: A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies, pp. 21–26. Springer, Cham (2016)

16. Khan, R.U., Zhang, X.S., Kumar, R.: Analysis of ResNet and GoogleNet models for malware detection. J. Comput. Virol. Hack. Tech. **15**, 29–37 (2018)
17. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. Computerence (2014)
18. Tavallaee, M., Bagheri, E., Lu, W., et al.: A detailed analysis of the KDD CUP 99 data set. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pp. 53–58. Ottawa, ON, Canada (2009)
19. Gao, N., Gao, L., Gao, Q., et al.: An intrusion detection model based on deep belief networks. In: 2014 Second International Conference on Proceedings of Advanced Cloud and Big Data (CBD), pp. 247–252, [S. l.]. IEEE (2014)
20. Hang, M.X., Chen, W., Zhang, R.J.: Abnormal flow detection based on improved one-dimensional convolutional neural network. J. Comput. Appl.