Hindawi
Mobile Information Systems
Volume 2022, Article ID 8124831, 8 pages
https://doi.org/10.1155/2022/8124831

Research Article

Network Intrusion Detection Method Based on Multi-Scale CNN in Internet of Things

Xiuye Yin[1] **and Liyong Chen**[2]

[1]*School of Computer Science and Technology, Zhoukou Normal University, Zhoukou 466001, China*
[2]*School of Network Engineering, Zhoukou Normal University, Zhoukou 466001, China*

Correspondence should be addressed to Xiuye Yin; 20111036@zknu.edu.cn

Received 25 July 2022; Revised 8 September 2022; Accepted 19 September 2022; Published 6 October 2022

Academic Editor: Le Sun

Network intrusion detection is a powerful means to identify and analyze the state of the Internet of things. For the reliability requirements of the Internet of things, an intrusion detection analysis method of the Internet of things based on a deep network model is proposed. First, based on the Inception network architecture as the backbone network, this method constructs a multi-scale convolutional neural network (M-CNN) intrusion detection analysis network model. The long-term and short-term memory network models are introduced into M-CNN to enhance the local feature extraction ability of the model. At the same time, batch normalization and global average pooling layers are introduced to make the data distribution of each layer uniform, reduce the model training time, reduce the model calculation gradient, and further improve the efficiency of the network. The simulation experiment takes the KDDcup99 data set as an example, and the results show that the M-CNN intrusion detection model can achieve better results. The precision *Pre* and recall *Re* of the detection model are 93.90% and 93.59%, respectively.

1. Introduction

The industrial Internet of things, regarded by scholars as a new generation of interconnection modes [1, 2], is an ubiquitous network based on the Internet. It connects a large number of microdevices (smartphones, electricity meters, etc.) with the Internet according to the agreed protocol and realizes intelligent communication between people, people, and things, as well as things and things [3–5]. At a time when the interconnected Internet of things is evolving towards science, at the same time, the external situation of the Internet of things is also suffering from massive network attacks. Moreover, the methods of attacks are also emerging day by day, and the inexplicable offenses that need to be resisted are becoming more and more complex [6, 7]. Therefore, an effective and accurate network state analysis method is of great significance to support the efficient information interaction function of the Internet of things. Intrusion detection technology can detect and analyze the network data by collecting all kinds of information in the network traffic so as to judge whether there is abnormal behavior in the network traffic and provide reliable protection support for users or terminal equipment [8, 9].

The emergence of deep learning provides a new solution for the intrusion detection research of the Internet of things. Taking the status of the Internet of things as a network sample data set, it is imported into the deep learning model for continuous training and learning, and the corresponding label set is constructed to realize real-time network status analysis and attack detection [10, 11]. However, the current deep network learning has too many network layers, which is easy to cause problems such as overfitting and gradient explosion.

In order to solve the problems of the overfitting and gradient explosion of depth network, this paper proposes an intrusion detection method of the Internet of things based on a multiscale convolutional neural network (M-CNN). Adopt the inception V3 network structure as the backbone network to build the M-CNN network analysis model. At the

same time, use the batch normalization and global average pooling layers to further improve the efficiency of the network. On the basis of ensuring a certain network structure, improving the computing power of the model, and solving the problems of overfitting and gradient explosion.

The remaining chapters of this paper are arranged as follows: Chapter 2 introduces the relevant research in this field. In Chapter 3, the proposed network intrusion detection model based on multiscale CNN is introduced in detail; the fourth chapter takes the KDDCUP99 data set as an example, and the experimental results show that the proposed model can achieve better results; and the fifth chapter is the conclusion.

2. Related Works

Intrusion detection technology is an active security defense strategy, which can actively detect and identify abnormal behaviors in network access data. It is an important security defense measure deployed behind the firewall and is the second firewall for system security [12–14]. The Internet is only a small part of the huge industrial Internet of things system [15]. According to the data of the Weekly Report on Network Information Security and Dynamics in December 2020 released by the China Internet Emergency Center, the number of hosts infected with network viruses in China has reached 609 thousand. It can be seen that improving the performance of intrusion detection to effectively identify malicious traffic has become an inevitable requirement of the development of network security technology.

Traditional intrusion detection methods are mainly composed of anomaly detection models for normal behavior characteristics and misusing detection models for known attack behavior characteristics. They are mainly based on feature detection and rule-based to identify attack behavior, which has many shortcomings [16]. First, traditional intrusion detection methods often lack accuracy and the ability to automatically update relevant features. Second, when identifying attack behaviors, detection methods of anomaly detection models include threshold detection and statistical methods, etc. Reference [17] used statistical analysis methods and identified DOS and DDOS based on Kullback-Leibler distance. However, it is too dependent on the features of normal network behavior, and there is a problem with a high false positive rate [18]. Detection methods of misusing detection models include pattern matching, expert systems, etc. Reference [19] developed a malware pattern-matching engine. However, it is only effective for the currently known attacks, and the missing rate of unknown network attacks is high. Network traffic data has a trend of high-dimensional and massive changes, and it is difficult to automatically distinguish between attack behavior and normal behavior by manual analysis alone [20]. Due to its strong representational learning ability, the deep learning model can effectively solve the problem of difficult high-dimensional data, which is suitable for application to the task of detection and classification [21].

The main idea of deep learning is to recognize and classify new network traffic data in real-time by using a trained classification model. Reference [22] proposed an intrusion detection system based on deep learning for the multi-cloud Internet of things environment, and the feasibility of the method was verified based on the NSL-KDD data set; Reference [23] determined the network security level, constructed a security intrusion detection system based on real-time sequence and extreme learning machine model, and analyzed the state of the Internet of things network. Based on the security interoperability requirements of the Internet of things, reference [24] built a distributed intrusion detection model based on the bidirectional long and short-term memory model to realize the network protection of the Internet of things for smart contracts; Reference [25] adopted the stacking depth learning method to analyze and determine the network state of the SCADA system in the power network. Reference [26] proposed a new unsupervised dimensionality reduction method to detect network attacks. It uses t-SNE combined with a hierarchical neural network to map the high-dimensional network data space to the low-dimensional potential space. Reference [27] proposed a hybrid method of multiobjective genetic algorithms and neural networks to establish an integrated solution for effectively detecting network intrusion, and this method has achieved good results. However, when dealing with network state analysis, the current deep learning model mainly relies on the stacking of network structures to improve the recognition accuracy, which makes the network model prone to the problem of over-fitting and introduces the gradient explosion phenomenon in the calculation of the network model. It is difficult to ensure the stability of the intrusion detection model.

Therefore, this paper proposes an intrusion detection method based on M-CNN. The proposed method improves the performance of network analysis, simplifies the structure of the network model, reduces computational memory, and improves the efficiency of model analysis.

3. Network Intrusion Detection Based on Multiscale CNN

3.1. Overall Framework of the Proposed Method. The overall training framework of the M-CNN model proposed in this paper is shown in Figure 1. The idea is based on the Inception structure of GoogLeNet, which includes three parts:

3.1.1. Data Preprocessing and Data Transformation. The symbolic feature attributes in the KDD data set are digitized and normalized to obtain a standardized data set. Then each network data is transformed into two-dimensional data so that it can conform to the CNN model input format and the data is read into the model through the data reading tool Pandas.

3.1.2. Specific Structure of CNN. The optimal features obtained by CNN training on the transformed data set are used to identify five attack states in the data set: DoS (denial of service), Probe (watch and other probe activities), R2L (illegal access of ordinary users to local superuser privileges), U2R (illegal access from remote machines), and Normal (normal records).

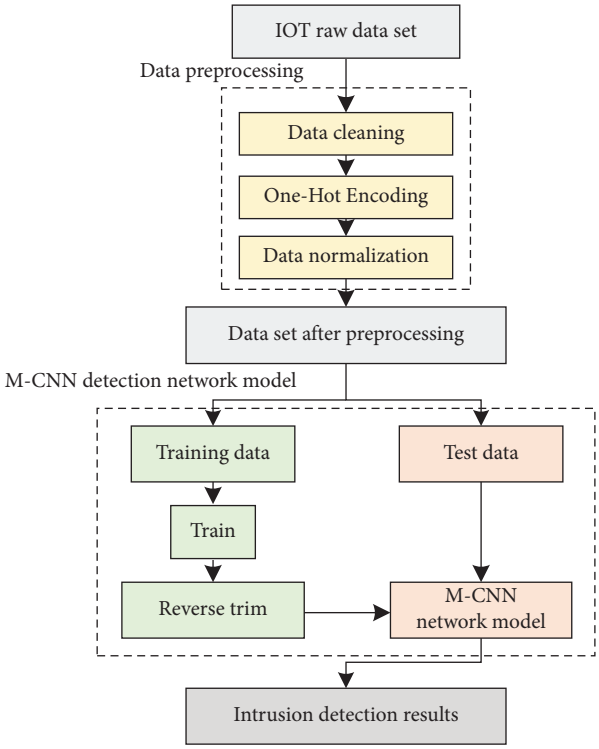

Figure 1: M-CNN learning process.

3.1.3. Model Training and Reverse Fine-Tuning Improve the Performance of the Model. In the CNN model, the BP algorithm plays the role of fine-tuning the model parameters in the whole network.

3.2. Data Preprocessing. Before analyzing the network model, the original data set should be processed accordingly. Data processing can be divided into three parts: data cleaning, one-hot coding, and data normalization.

3.2.1. Data Cleaning. Data cleaning is to correct or clear the wrong data in the data file, including the processing of invalid values and missing values in the data and the rationality detection of the data, that is, to replace, modify and delete the dirty data.

As shown in Figure 2, the main process of data cleaning is as follows:

Step 1: Find the missing data. Some faults or human errors will occur occasionally during data collection, resulting in the value of one or more items in the data being in a blank state. In this experiment, the missing value is found by the Excel tool, and the positioning can be realized by setting the positioning condition to a null value.

Step 2: Because the sample data set used in this paper has few missing values, that is, the information contained in the corresponding part is limited, so the method of directly deleting this row of data is adopted to deal with the missing values.

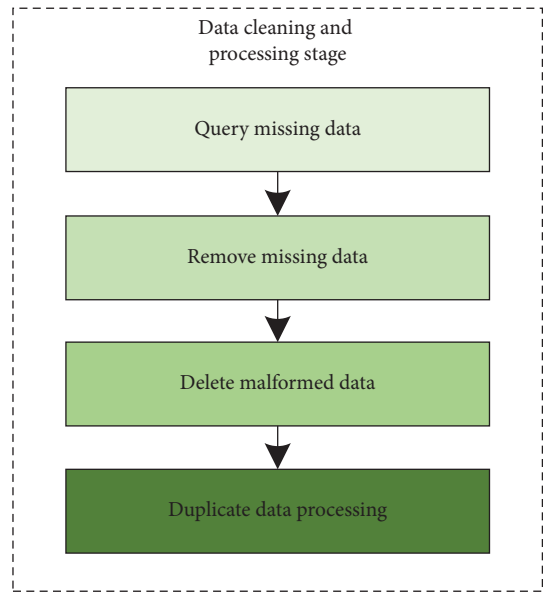

Figure 2: Internet of things data cleaning process.

Step 3: After excluding missing values, there will also be some obvious inaccuracies in the data. There are many data with content errors in this experimental data set, especially more than 1000 data with "Infinite" and "NaN" in the two columns of Flow Bytes/s and Flow Packets/s. However, the data in these two columns should be speed in numerical form, which is an obvious content error, so the whole row of data involved in the error will be deleted.

In addition to finding missing values and content errors, data cleaning also includes operations such as removing duplicate values. Whether it is logical errors or excessive data repetition, it will affect the performance of the model. After searching, no logical error data is found, and the data repetition is less, so no treatment measures are taken.

3.2.2. One-Hot Encoding. The data collected by the Internet of things has a large number of discrete data. Its data format is a string, so it is considered to digitize the labels in string format. If labels are converted into numbers, the above data cannot be directly used in the model.

One-Hot Encoding can solve this problem. The method is to encode n states. One digit represents one state, and n digits are needed to represent n states. When the result is in a certain state, the corresponding digit of the state is 1, and the other digits are 0. It can also be understood that for each feature, if it has m possible values, it will become m binary features after One-Hot Encoding. Moreover, these features are mutually exclusive, and only one feature is activated at a time. Therefore, the data will become sparse. One-Hot Encoding not only solves the problem of data attributes but also expands multiple labels in the experimental data set into sparse label variables with corresponding dimensions.

3.2.3. Data Normalization. Data normalization is to scale the data in proportion so that all values are within the required range. There are two main advantages to normalizing

data. One is to improve the convergence speed of the model, and the other is to improve the accuracy of the model.

According to statistics, the size of the data used in the experiment in this paper is unstable. In order to more truly simulate the data situation of the actual intrusion scene, the standard deviation is not used. At the same time, when new data is added, the maximum value will change and need to be redefined. Therefore, this experiment adopts the method of normalization of standard deviation to realize the normalization of the standard deviation of data.

The calculation formula of normalization of standard deviation is shown in formula (1), where x_{norm} is the normalized data set. After normalization of the standard deviation, the data present a Gaussian distribution, with a mean of 0 and a variance of 1.

$$x_{norm} = \frac{x - \mu}{\sigma}, \tag{1}$$

where x is the original sample data set of the Internet of things, μ is the mean value, and σ is the standard deviation. When using standard deviation normalization, the data distribution is generally close to the Gaussian distribution, otherwise, the normalization effect may be poor.

3.3. M-CNN Model Structure. According to the inception architecture, this paper creates a multiscale convolution module (M-Module), and adds long and short-term memory (LSTM) layers and a convolution pooling layer to complete the complementary operation of local feature extraction. It improves the efficiency of the network by using batch normalization and global average pooling (GAP) layer and finally completes the classification by using the Softmax layer. The M-Module based on Inception is shown in Figure 3 below.

In the M-Module used in this paper, the convolution kernel with a size of $5 * 5$ is decomposed into two $3 * 3$ convolutions, and the $3 * 3$ convolutions in the two branches are further decomposed into $1 * 3$ and $3 * 1$ convolution to minimize the number of parameters. At the same time, the step size of 1 and the same padding method are used in the module to maintain the edge features and maintain the size of the feature map. In each branch, $1 * 1$ convolution is used to unify the number of channels, so as to facilitate the fusion of feature maps.

The components and parameters of the model M-CNN are shown in Table 1.

In M-CNN, three M-Modules are used. The depth of the three module filters gradually increases, and an average pooling layer is added after the last M-Module to reduce the size of the feature map. Then, the feature map is mapped into a tensor with a size of $1 * 1 * D$ through the GAP layer, converted into a format consistent with the input of the LSTM layer through the Reshape layer, and then input into the LSTM layer to extract sequence features. Finally, the classification results are output through the Softmax layer.

3.4. Selection of a Pooling Layer. The original intention of the pooling layer was to reduce parameters and speed up network training, but it is a process of feature loss. In order to improve the classification accuracy of the model, this paper uses only one average pooling layer at the end. The reason for choosing average pooling is that the features in each network data are different in size and vary greatly. Even if they are normalized and mapped to the interval $(0, 1)$, some elements in the interval tend to 0, and some tend to 1. There is still a great difference. If the max pooling method is adopted, the feature tending to 0 will never be selected, resulting in errors in classification.

The pooling layer is divided into overlapping pooling and nonoverlapping pooling according to the size of the convolution kernel and sliding step size. Overlapping pooling can converge faster and have higher accuracy. This paper selects the overlapping pooling method to make the M-CNN model optimal.

3.5. Batch Normalization. In the process of deep learning network model training, the parameters of the network model are constantly updated and changed, so the probability distribution of the input data at each layer is also constantly changing, and the internal covariate shift phenomenon appears. It needs to use a smaller learning rate and better parameters in the model training to avoid this phenomenon. However, this method will prolong the training time of the model and make it difficult to train the model when using saturating nonlinearities activation functions

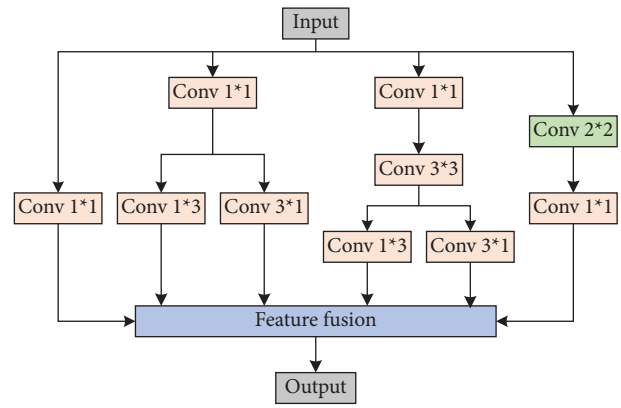

Figure 3: Multi-scale convolution module.

Table 1: Composition and parameters of M-CNN.

No	Network layer	Size (depth)	Step (stride)	Filling method
L1	Input layer	$12 * 12 * 2$	—	—
M2	M-module	128	$S = 1$	Same
C3	Conv layer	$3 * 3 * 128$	$S = 1$	Same
M4	M-module	256	$S = 2$	Same
C5	Conv layer	$3 * 3 * 256$	$S = 2$	Same
M6	M-module	512	$S = 2$	Same
C7	Conv layer	$3 * 3$	$S = 4$	Same
F8	Gap layer	—	—	—
R9	Reshape layer	—	—	—
D10	LSTM layer	128	—	—
F11	Full connection layer	15	—	—

(such as sigmoid positive and negative oversaturation). Because of the existence of the covariate shift phenomenon, according to the chain rule, when the number of layers of the deep learning network model continues to increase, the existing problems will also continue to be expanded. For the generalization of the deep learning network model, this is a serious problem. A neural network is representational learning. If the input of data always changes, the neural network must relearn the new distribution. The algorithm normalizes all layers of the model, which makes the input data irrelevant. By normalizing the input of each layer to readjust the data distribution, the data distribution of each layer is uniform, so as to reduce the impact of Internal covariate shift. The batch normalization (BN) algorithm can reduce the impact of internal covariate shift and has other advantages. At the same time, the BN algorithm is a very popular optimization algorithm in the field of deep learning. The BN algorithm is shown in Algorithm 1. In this paper, the BN algorithm is applied to the proposed model to improve the performance of the network model.

3.6. Weight Update. The weight update uses the BP algorithm and a gradient descent algorithm to update the weight and bias. The gradient descent algorithm is realized by calculating the derivative of the loss function for the weight and bias.

$$\frac{\partial}{\partial w_i^{(k)}}(W,b) = \left[\frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial w_i^{(k)}}J(W,b;x_i,g) + zw_i^{(k)}\right],$$

$$\frac{\partial}{\partial w_i^{(k)}}(W,b) = \frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial b_i^{(k)}}J(W,b;x_i,g), \tag{2}$$

where $W_i^{(k)}$ is the weight of node i in layer k, $b_i^{(k)}$ is the bias of node i in layer k, and z is the normalized parameter. The feedback error is:

$$\kappa_i^{(k)} = \left(\sum_{i=1}^{n} w_i^{(k)}\kappa_i^{(k+1)}\right)\cdot v_i^{(k)},$$
$$\kappa_i^{(\tau k)} = -\left(g - a_i^{(\tau k)}\right)\cdot v_i^{(k)}, \tag{3}$$

where τk is the final output layer, $\kappa_i^{(\tau k)}$ is the residual of node i in layer k, $v_i^{(k)}$ is the activation value input to node i in layer k, and $v_i^{(k)}$ is the activation value of node i in layer τk.

4. Experiments

The M-CNN model was built based on the deep learning framework Keras 2.2.0, and its programming language is Python 3.5.2 to verify the feasibility of the proposed detection method. The specific experimental parameter configuration is shown in Table 2.

The data set selected for the experiment comes from the KDDcup99 data set collected by the Lincoln Laboratory of the Massachusetts Institute of Technology in the United States. The data set has a total of 5 million records, in which 10% of the test set and training set are provided. The category, number, and proportion of training data and test data used in this experiment are shown in Tables 3 and 4.

4.1. Model Performance Analysis. The weights and biases of all layers are initialized by the 0-means Gaussian function, and the loss rate is 0.02. At the same time, due to the uneven distribution of data in the dataset, a normalization layer is used before convolution to evenly distribute the data and speed up the learning time of the network structure. In order to prevent the network from falling into overfitting and improve the generalization ability of the network structure, the dropout method is used to connect the map features in the fully connected layer. The layer-to-layer connection probability is experimentally explored, as shown in Figure 4.

It can be seen from Figure 4 that when the dropout probability is 0.8, the model accuracy of M-CNN is the highest, reaching 92.89%. Therefore, the dropout probability parameter is fixed at 0.8. At the same time, this paper uses the mini-batch gradient descent method for training. Each training only needs a small number of samples, which not only ensures that the convergence speed is not too slow but also ensures that the optimal solution is not too local.

The batch normalization layer can speed up the network learning speed. This paper also performs an experimental analysis of the M-CNN model with or without the BN layer, and the results are shown in Figure 5.

It can be seen from Figure 5 that the use of the BN layer can significantly accelerate the convergence speed and improve the learning efficiency of the model. After 30 rounds of iterative calculation, the accuracy of the model reaches 92.47%, which can support relatively accurate Internet of Things network intrusion judgment. At the same time, during training, because the BN layer updates the weights according to each mini-batch, set "use-global-stats" to "false." During the test phase and new sample prediction, set "use-global-stats" to "true" to prevent nonconvergence.

4.2. Analysis of Detection Results. The experiment uses the precision *Pre*, recall *Re*, and F_1 value as measures to analyze the optimality of the detection method.

$$Pre = \frac{TP}{TP + FP},$$
$$Re = \frac{TP}{TP + FN}, \tag{4}$$
$$F_1 = \frac{2Pre * Re}{Pre + Re},$$

where TP is the accuracy of model identification of network intrusion and FP is the proportion of correct model identification. The higher the precision and recall, the better the detection effect.

The test data set adopts part of the data set in the KDDcup99 data set. The test results of the test data set are shown in Table 5 and Figure 6.

Input: set the minimum batch by x: $\theta = \{x_1, x_2, x_3, \ldots, x_n\}$
Parameters to learn: r, θ
Output: $\{g = BN_{r,\theta}(x_i)\}$
(1) $\mu\theta \leftarrow (1/n)\sum_{i=1}^{n} x_i$ //mini-batch mean
(2) $\sigma_\theta^2 \leftarrow (1/n)\sum_{i=1}^{n} (x_i - \mu_\theta)^2$ //mini-batch variance
(3) $\widehat{x}_i \leftarrow (x_i - \mu_\theta / \sqrt{\sigma_\theta^2 + \gamma})$ //normalize
(4) $g \leftarrow r\widehat{x}_i + \theta = BN_{r,\theta}(x_i)$ //scale and shift

Algorithm 1: BN algorithm in network intrusion detection method.

Table 2: Experimental parameter configuration of the M-CNN detection method.

Item	Details
Operating system	Linux 6.3.1
CPU	Intel core i7-6700
Memory	32 GB
Programing language	Python 3.5.2
Deep learning framework	Keras 2.2.0

Table 3: Training set of the M-CNN detection method.

Category	Quantity	Proportion (%)
Normal	97 278	19.69
DoS	391 458	79.23
Probe	4 107	0.83
R2L	1 126	0.22
U2R	52	0.01

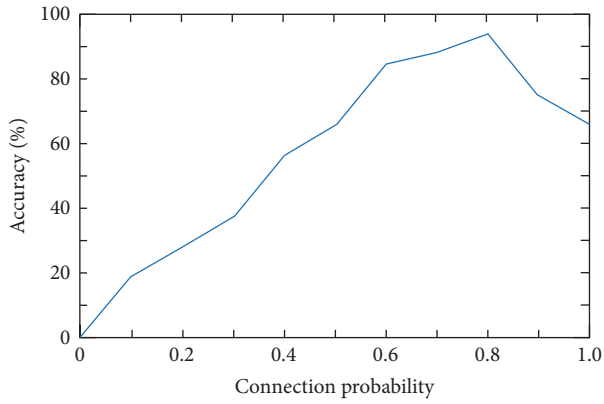

Figure 4: Influence of dropout probability parameters on the model.

As shown in Table 4, for various network states in the KDDcup99 data set, the M-CNN method can achieve more accurate classification and identification, and the identification accuracy of various situations can reach 91%. The detection accuracy is from high to low: *Dos* > *Normal* > *Probe* > *R2L* > *U2R*. Except that the detection accuracy of U2R with less training data is low, the detection accuracy of other types of data is greater than 92%. Figure 6 shows the evaluation of the model based on different indices.

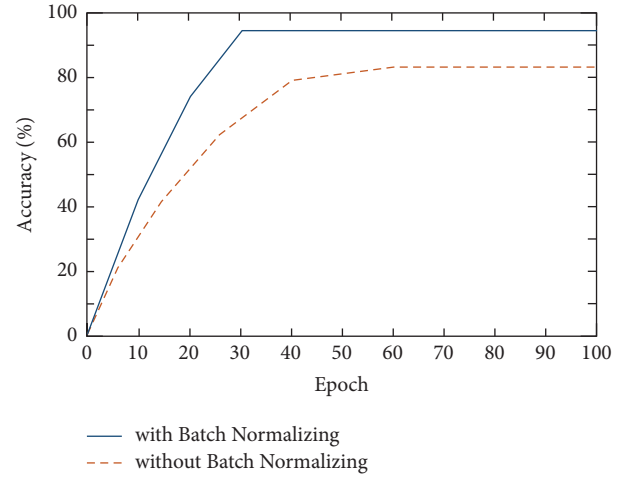

Figure 5: Experimental comparison with and without batch normalization layer.

The identification results of precision *Pre*, recall *Re*, and F_1 value in Figure 6 are consistent for different network states. Combined with the results in Table 4, it shows that the model has insufficient detection power for U2R. This is because the sample numbers of the U2R data set is insufficient, and the model fails to fully learn its data features. Therefore, the ability to distinguish this type of data is greatly reduced, and it cannot achieve high-precision detection, which also leads to a higher false positive rate of U2R data than the other four types of data. Therefore, according to the number of data and the detection results, it is determined that the data detection rate is in direct proportion to the number, while the false positive rate is in inverse proportion to the number. The larger the data scale, the better the detection results. Using the inception network architecture as the backbone network, a multiscale CNN network analysis model is built. The LSTM layer and convolution pooling are added to the model to complete the operation of local feature extraction, so as to improve the model's identification and classification performance of network intrusion.

In order to test the overall detection performance of intrusion data, this paper takes the KDD99 data set as the benchmark method, uses reference [22, 25] as the comparison method, and runs in the same experimental environment as the M-CNN detection method to verify the optimal performance of the proposed method. The comparison results of different detection methods are shown in Table 6. The M-CNN detection model has good performance, and the accuracy is higher than other models.

Table 4: Test set of the M-CNN detection method.

Category	Quantity	Proportion (%)
Normal	60 593	19.48
DoS	229 853	73.90
Probe	4 166	1.33
R2L	16 189	5.20
U2R	228	0.07

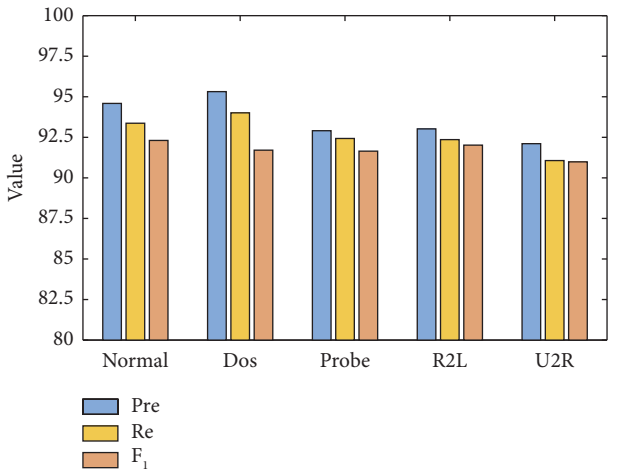

Figure 6: Analysis of test results under different indices.

Table 5: Test results for test data set.

Item	Accuracy (%)
Normal	93.37
DoS	94.01
Probe	92.43
R2L	92.36
U2R	91.07

Figure 7 shows the analysis of detection methods based on different indices.

It can be seen from Table 5 and Figure 7 that the M-CNN intrusion detection model designed in this paper can achieve better results when using the same dataset, with precision, recall, and value of 93.90%, 93.59%, and 93.31%, respectively. The results show that the detection rate of this model is higher than other models, the detection results are more accurate, and the overall performance of this model is more optimized. Therefore, this model is feasible. The reason is that the Inception V3 network structure used in the M-CNN model still maintains the expression ability of the model while simplifying the network model. Using the LSTM layer, more key information features in the data set can be extracted, which effectively removes the redundant features. At the same time, using the BN layer greatly alleviates the overfitting problem of the model. Comparison methods ignore the increase in parameters and calculations caused by the superposition of network layers. The method of directly increasing the number of network layers can improve the amount of information obtained, but it is also prone to problems such as overfitting and gradient explosion to a certain extent.

Table 6: Test results under different methods.

Method	Accuracy (%)
The proposed method	93.87
Reference [22]	92.18
Reference [25]	91.09

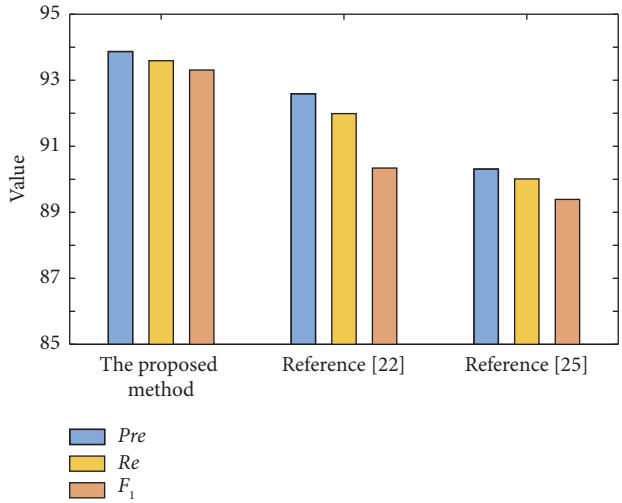

Figure 7: Comparative analysis of detection methods under different indices.

5. Conclusion

Reliable and accurate network intrusion detection analysis is an important guarantee for the stable and secure networking of users or terminals in the Internet of things. This paper proposes a network intrusion detection method based on the multiscale CNN network model. In the M-CNN network intrusion detection model, batch normalization and global average pooling layers are introduced to adaptively adjust the distribution of input data. It makes the data distribution of each layer uniform, reduces the model training time, reduces the model calculation gradient, and further improves the efficiency of the model. The simulation results confirm that the precision of the proposed M-CNN intrusion detection method for identifying the network state of complex data sets is 93.90%. The proposed method can accurately judge and analyze the heterogeneous network state of the Internet of things and effectively support the reliable and stable interconnection state of the Internet of things.

Although the data used in the simulation experiment in this paper is the standard detection data commonly used in the field of intrusion detection, its generation method is different from the data generated in the actual network environment. Therefore, the next stage of research will further study the intrusion data modeling and network state analysis generated in the actual network environment.

Data Availability

The data used to support the findings of this study are included in this article.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61402350, U1404620, U1404622), the Key Scientific and Technological Project of Henan Province (182102310034, 212102210400, 212102210098), the Key Research Projects of Henan Provincial Department of Education (22A520051).

References

[1] X. Xu, J. Li, Y. Yang, and F. Shen, "Toward effective intrusion detection using log-cosh conditional variational autoencoder," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6187–6196, 2021.

[2] B. B. Li, J. R. Song, Q. Y. Du, and J. J. He, "DRL-IDS:Deep reinforcement learning based intrusion detection system for industrial Internet of things," *Computer Science*, vol. 48, no. 7, pp. 47–54, 2021.

[3] L. Nie, Y. Wu, X. Wang et al., "Intrusion detection for secure social Internet of things based on collaborative edge computing: a generative adversarial network-based approach," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 1, pp. 134–145, 2022.

[4] Y. Zhang, P. Li, and X. Wang, "Intrusion detection for IoT based on improved genetic algorithm and deep belief network," *IEEE Access*, vol. 7, no. 1, pp. 31711–31722, 2019.

[5] H. M. Lu, T. Wang, X. Xu, and T. Wang, "Cognitive memory-guided AutoEncoder for effective intrusion detection in Internet of things," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3358–3366, 2022.

[6] N. Mazhar, R. Salleh, M. Asif, and M. Zeeshan, "SDN based intrusion detection and prevention systems using manufacturer usage description: a survey," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 12, pp. 717–737, 2020.

[7] M. Abdel-Basset, H. Hawash, R. K. Chakrabortty, and M. J. Ryan, "Semi-supervised spatiotemporal deep learning for intrusions detection in IoT networks," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12251–12265, 2021.

[8] I. Ullah and Q. H. Mahmoud, "Design and development of a deep learning-based model for anomaly detection in IoT networks," *IEEE Access*, vol. 9, no. 1, pp. 103906–103926, 2021.

[9] M. Emec and M. H. Ozcanhan, "A hybrid deep learning approach for intrusion detection in IoT networks," *Advances in Electrical and Computer Engineering*, vol. 22, no. 1, pp. 3–12, 2022.

[10] J. Lansky, S. Ali, M. Mohammadi et al., "Deep learning-based intrusion detection systems: a systematic review," *IEEE Access*, vol. 9, no. 1, pp. 101574–101599, 2021.

[11] G. Kornaros, "Hardware-assisted machine learning in resource-constrained IoT environments for security: review and future prospective," *IEEE Access*, vol. 10, no. 1, pp. 58603–58622, 2022.

[12] N. Mishra and S. Pandya, "Internet of things applications, security challenges, attacks, intrusion detection, and future visions: a systematic review," *IEEE Access*, vol. 9, no. 1, pp. 59353–59377, 2021.

[13] M. Boopathi, "Henry MaxNet: tversky index based feature selection and competitive swarm henry gas solubility optimization integrated Deep Maxout network for intrusion detection in IoT," *International Journal of Intelligent Robotics and Applications*, vol. 6, no. 2, pp. 365–383, 2022.

[14] R. Marzouk, F. Alrowais, N. Negm et al., "Hybrid deep learning enabled intrusion detection in clustered IIoT environment," *Computers, Materials & Continua*, vol. 72, no. 2, pp. 3763–3775, 2022.

[15] Y. Otoum and A. Nayak, "Anomaly and signature based IDS for the Internet of things," *Journal of Network and Systems Management*, vol. 29, no. 3, pp. 1–26, 2021.

[16] A. C. Li and S. J. Yi, "Intelligent intrusion detection method of industrial Internet of things based on CNN-BiLSTM," *Security and Communication Networks*, vol. 2022, no. 1, pp. 1–8, 2022.

[17] B. Bouyeddou, F. Harrou, B. Kadri, and Y. Sun, "Detecting network cyber-attacks using an integrated statistical approach," *Cluster Computing*, vol. 24, no. 2, pp. 1435–1453, 2021.

[18] L. Chen, Z. Wang, J. Wu, Y. Guo, F. Li, and Z. Li, "Dynamic threshold strategy optimization for security protection in Internet of Things: an adversarial deep learning-based game-theoretical approach," *Concurrency and computation: practice and experience*, vol. 34, no. 1, pp. 1-2, 2022.

[19] D. Oh, D. Kim, and W. Ro, "A malicious pattern detection engine for embedded security systems in the Internet of things," *Sensors*, vol. 14, no. 12, pp. 24188–24211, 2014.

[20] D. Li, L. Deng, M. Lee, and H. Wang, "IoT data feature extraction and intrusion detection system for smart cities based on deep migration learning," *International Journal of Information Management*, vol. 49, no. 1, pp. 533–545, 2019.

[21] H. N. Bhor and M. Kalla, "TRUST-based features for detecting the intruders in the Internet of Things network using deep learning," *Computational Intelligence*, vol. 38, no. 2, pp. 438–462, 2021.

[22] D. Selvapandian and R. Santhosh, "Deep learning approach for intrusion detection in IoT-multi cloud environment," *Automated Software Engineering*, vol. 28, no. 2, pp. 19–17, 2021.

[23] A. Haider, M. Adnan Khan, A. Rehman, M. Rahman, and H. Seok Kim, "A real-time sequential deep extreme learning machine cybersecurity intrusion detection system," *Computers, Materials & Continua*, vol. 66, no. 2, pp. 1785–1798, 2021.

[24] O. Alkadi, N. Moustafa, B. Turnbull, and K. K. R. Choo, "A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9463–9472, 2021.

[25] W. Wang, F. Harrou, B. Bouyeddou, S. M. Senouci, and Y. Sun, "A stacked deep learning approach to cyber-attacks detection in industrial systems: application to power system and gas pipeline systems," *Cluster Computing*, vol. 25, no. 1, pp. 561–578, 2021.

[26] H. Yao, C. Li, and P. Sun, "Using parametric t-distributed stochastic neighbor embedding combined with hierarchical neural network for network intrusion detection," *International Journal on Network Security*, vol. 22, no. 2, pp. 265–274, 2020.

[27] G. Kumar, "An improved ensemble approach for effective intrusion detection," *The Journal of Supercomputing*, vol. 76, no. 1, pp. 275–291, 2020.