Nonlinear Science and Intelligent Applications
https://www.acadlore.com/journals/NSIA

Identification of Computer Network Node Intrusion Behavior Based on Improved Recursive Residual Network

Hongzhang Han*

School of Computer Engineering, Jiangsu University of Technology, 213001 Changzhou, China

* Correspondence: Hongzhang Han (hhz@jsut.edu.cn)

Received: 07-02-2025 **Revised:** 09-03-2025 **Accepted:** 09-22-2025

Citation: H. Z. Han, "Identification of computer network node intrusion behavior based on improved recursive residual network," *Nonlinear Sci. Intell. Appl.*, vol. 1, no. 1, pp. 1–17, 2025. https://doi.org/10.56578/nsia010101.

Abstract: Accurate identification of node intrusion behavior in computer networks remains challenging due to the highly dynamic and complex nature of modern network environments, where benign activities are frequently misclassified as malicious events. To address the degradation in detection reliability resulting from such misjudgments, an intrusion identification framework based on an enhanced Recursive Residual Network (RRN) was developed. A statistical classification paradigm was incorporated to process the heterogeneous characteristics of network node intrusion data, enabling a more robust separation of normal and anomalous activity patterns. Features associated with abnormal nodes were extracted, and the range of intrusion-related behavioral deviations was optimized iteratively through an error-minimization function, allowing the model to adapt effectively to fluctuations in network states. A recursive structure derived from Recurrent Neural Network (RNN) principles was subsequently embedded within the residual regression architecture, through which node credit values were continuously iterated and updated to refine the distinction between legitimate behavior and genuine intrusion attempts, enhancing the stability of intrusion identification. In the final stage, a potential loss metric was computed to quantify the expected impact of detected anomalous behaviors on network assets, thereby enabling abnormal behaviors to be classified rigorously as intrusion events when their estimated loss exceeds a critical threshold. Experimental results demonstrate that the proposed method achieves high sensitivity, maintains a stable attack-type identification rate throughout the evaluation period, and reduces the trust value of compromised nodes below 0.15 within 75 seconds, indicating strong effectiveness in distinguishing authentic intrusion behaviors from normal variations. The overall findings suggest that the enhanced RRN offers a resilient and adaptive mechanism for intrusion behavior identification under conditions of complex network dynamics.

Keywords: Improved RRN; Computer network; Node intrusion; Behavior identification; Principal Component Analysis (PCA)

1 Introduction

With the popularization of computer technology and the Internet, network node intrusion occurs frequently, posing a major threat to personal, corporate and national security, easily leading to data leakage, system collapse, etc., thereby interfering with network operation and affecting social stability. Therefore, it is particularly important to identify network node intrusion behaviors and take corresponding prevention and countermeasures [1, 2]. However, the existing intrusion detection systems are faced with many challenges. On the one hand, because of the diversity and complexity of network node intrusion behaviors, it is often difficult for traditional detection methods to accurately identify all intrusion behaviors. On the other hand, with the continuous increase of network traffic, the existing detection system is prone to performance bottlenecks when dealing with large-scale data, resulting in the decline of detection efficiency and accuracy. In response to these problems, researchers have begun to explore intrusion detection methods based on deep learning.

Cao and Huang [3] explored host intrusion detection based on user file access logs. A graph was used to model file access behavior in a highly abstract way, deriving a set of behavioral features from the graph model that can be used by a machine learning algorithm to identify intruders. However, this method fails to fully consider the complexity of the network environment, resulting in poor performance in obtaining behavior features and affecting the identification effect of intrusion behavior. Reka et al. [4] developed a Coati Optimization Algorithm (COA) that

incorporates network centrality measures to perform multi-attack intrusion identification, using a clustering-gradient mechanism to improve cluster formation. In this study, a cluster-head selection algorithm for Mobile Ad Hoc Networks (MANETs) based on dual network centrality was developed to solve the problem of node mobility and energy. The COA was used to obtain compact clusters. Finally, a Multi-Scale Attention Graph Convolutional Neural Network (MSA-GCNN) was used to identify multiple attacks. However, this method fails to consider the dynamic changes of the network, resulting in a poor clustering effect and a poor identification rate of the MSA-GCNN model.

In addition, Xie et al. [5] combined the distributed computing resources of Federated Learning and the decentralized characteristics of blockchain and proposed a blockchain-based Federated Learning framework for intrusion detection in Internet of Vehicles (IoV) networks. However, this method fails to take into account the influence of multiple factors on node behavior, which leads to the easy misjudgment of normal behavior as intrusion behavior, affecting the intrusion identification performance. Sathiya and Yuvaraj [6] developed an intrusion detection approach that integrates binary group optimization with differential evolution and moment-probability extremum learning to model node behavior. An intrusion detection system was constructed using moment-probability extremum learning in combination with a feature-selection mechanism based on a Goodman-activated binary group optimization differential evolution algorithm. However, this method tends to overlook subtle feature variations, resulting in incomplete or suboptimal feature selection and, consequently, reduced intrusion detection accuracy.

Therefore, to further improve the identification effect of intrusion behavior, a method to identify intrusion behavior in network nodes based on an improved RRN was proposed to more effectively guarantee network security. Compared with existing research, the main innovation of this study lies in the organic combination of an improved RRN, Depthwise Separable Convolution (DWConv), reputation fusion mechanisms, and Stress Majorization for Coordinate Finding (SMACOF) range localization methods to construct an end-to-end node intrusion behavior recognition framework, as shown in Table 1.

Table 1. Summary of main innovations and contributions of this study

Innovation Dimension	Specific Content	Achieved Effects and Quantitative Metrics
Model structure innovation	Integration of improved recursive residual structure and DWConv	• Number of parameters reduced by approximately 28% from 39 million to 28.1 million • Training efficiency improved while maintaining high accuracy (98.2%)
Detection process innovation	Construction of an end-to-end framework:"classification preprocessing → Principal Component Analysis (PCA) → SMACOF localization → deep network → reputation rusion"	• Ablation experiments showed that removing the classification preprocessing module caused a 7% drop in accuracy • Systematically enhanced input data quality and detection targeting
Decision mechanism innovation	Introduction of a dynamic direct + indirect comprehensive reputation value as a key criterion	• High sensitivity to malicious nodes, with the reputation value dropping below 0.15 after 75 seconds • Effectively resisted reputation camouflage attacks and reduced misjudgment rate
Comprehensive performance improvement	Comprehensive evaluation using the Canadian Institute for Cybersecurity Intrusion Detection System 2022 (CIC-IDS-2022) dataset	• F1-score reached 0.92, representing an average improvement of over 20% compared to baseline methods • Identification rate remained stable around 95.3% with minimal fluctuation • Particularly significant improvement in identifying complex attack types like User to Root (U2R) and Remote to Local (R2L)

2 Feature Extraction and Localization of Computer Network Nodes

Network intrusion has different data characteristic patterns. Classification distinguishes normal data from potential intrusion data according to these patterns to provide a more accurate basis for subsequent processing.

2.1 Classified Processing of Intrusion Data

As a basic data analysis method in statistics, classification of the network node data aims to distinguish the normal data from the potential intrusion data. According to the characteristics of the intrusion data, a statistical classification paradigm was adopted to realize the detection of intrusion data. Let the data of each node in the computer network constitute a set z_j, with the corresponding weight coefficient represented by b_j. These data need to meet the conditions of Eq. (1).

$$\sum_{j=1}^{p} z_j b_j = 1, b_j > 0 \tag{1}$$

Under the above conditions, the maximum value of the intrusion data feature in the computer network node can be found using Eq. (2).

$$\gamma = \sum_{j=1}^{p} b_j - \frac{1}{2} \sum_{j,k=1}^{p} b_j b_k z_j z_k l\left(y, y_i\right) \tag{2}$$

where, $l(y, y_i)$ is the kernel function; z_k is the normal data set [7], b_k is the weight coefficient, and p is the data volume of computer network nodes.

Using Eq. (3), the intrusion data in computer network nodes are processed by quadratic programming.

$$z(y) = sign\left(\sum_{j=1}^{p} \beta_j z_j l\left(y, y_i\right) \gamma + c\right) \tag{3}$$

where, β_j is the classification coefficient [8], c is a normal number, and γ is the weight coefficient for intrusion behavior. The dual conversion of Eq. (3) is expressed as follows:

$$\begin{cases} \max \sum\limits_{k=1}^{m} \beta_j - \frac{1}{2} \sum\limits_{j=1}^{m} \sum\limits_{k=1}^{m} z_j z_k \beta_j \beta_j l\left(y, y_i\right) \\ \text{s.t.} \sum\limits_{k=1}^{m} z_k \beta_j = 0 \\ 0 \leq \beta_j \leq y_k \quad k = 1, 2, \cdots, m \end{cases} \tag{4}$$

After derivation and analysis, the intrusion detection problem is reformulated as the dual form of a quadratic programming problem, which is subsequently solved to obtain the optimal classification coefficients. The resulting solution is expressed as follows:

$$\beta^* = \left(\beta_1^*, \beta_2^*, \cdots, \beta_m^*\right)^T \tag{5}$$

Therefore, according to the optimal classification coefficient solution, the optimal classification function is obtained as follows:

$$g(y) = \operatorname{sgn}\left\{\beta^* l\left(y, y_i\right) + c\right\} \tag{6}$$

The computer network node data y can be substituted into the above Eq. (6) to complete the classification processing and obtain the intrusion data. To realize effective identification of intrusion behavior, it is necessary to not only identify the intrusion data but also further extract the abnormal characteristics of these data for the subsequent research of intrusion behavior identification.

2.2 Abnormal Node Feature Extraction

On the basis of data classification, to extract the features of abnormal nodes more effectively, the PCA method can be used to optimize the feature space. PCA is able to retain the information of the original data to the greatest extent possible, while reducing the data complexity. The following are the specific steps for feature extraction of abnormal nodes in the computer network:

Step 1: In the computer network environment, in the face of various eigenvalue variables presented by any abnormal node, PCA is used to implement the dimensionality reduction strategy and refine it into a comprehensive variable with positive characteristics:

$$Y = \begin{bmatrix} Y_1 = E_{11}W_1 + \cdots E_{1n}W_n \\ Y_2 = E_{21}W_1 + \cdots E_{2n}W_n \\ \vdots \\ Y_n = E_{n1}W_1 + \cdots E_{nn}W_n \end{bmatrix} = EW \tag{7}$$

where, Y is the comprehensive variable, E is the original abnormal node data variable, W is the characteristic weight variable [9], and n is the number of variables.

Step 2: In computer network analysis, the comprehensive variable with the least variance is used as the main component, and the mean and standard deviation of abnormal nodes are obtained after the standardization of the covariance matrix.

$$\begin{cases} \bar{l} = \frac{\sum_{i=1}^{n} Y_i}{n} \\ s = \sqrt{\frac{\sum_{i=1}^{n} \left(Y_i - \bar{l}\right)^2}{n}} \end{cases} \tag{8}$$

where, $\bar{l}$ is the mean value, s is standard deviation [10], and Y_i represents the synthetic variable of node i.

Step 3: Based on the statistical characteristics of the abnormal nodes, the standard observation matrix and the associated matrix of the sample are constructed.

$$\begin{cases} W' = \frac{W_j - \bar{l}_j}{s_j} & j = 1, \cdots n \\ T = \frac{W' W^U}{a} \end{cases} \tag{9}$$

where, W' is the observation matrix, T is the correlation matrix, W_j represents the feature weight variable of node J, a is the main component number, and U is the correlation coefficient [11].

Step 4: Each element in the correlation matrix of the computer network is calculated. The feature decomposition formula of element T_{ij} is shown in Eq. (10) where k is the number of elements.

$$T_{ij} = \frac{\sum_{k=1}^{m} T\left(W' - \overline{l_i}\right)\left(W' - \overline{l_j}\right)}{\sqrt{\sum_{k=1}^{m} T\left(W' - \overline{l_i}\right)^2 \sum_{k=1}^{m} \left(W' - \overline{l_j}\right)^2}} \tag{10}$$

Step 5: Elements with eigenvalues greater than 1 are selected, and the cumulative contribution of these elements should exceed 90%. The principal component load matrix is constructed, which is the key feature of the abnormal nodes in the computer network. The cumulative contribution rate is expressed as:

$$Q = \frac{\sum_{i=1}^{n} \sum_{j=1}^{q} T_{ij}\mu_j}{\sum_{i=1}^{n} \mu_i} \tag{11}$$

where, μ is the characteristic value [12], and q is the number.

2.3 Determining the Network Intrusion Node Range

After feature extraction is complete, the next step is to determine the range of network intrusion nodes. This step is critical because it can narrow the detection range, improve efficiency, and re-determine the target node range based on the new network state to ensure the effectiveness of intrusion detection. By using the SMACOF theory to correct the distance between target and target node, the error function is iteratively optimized, and the target node range of network intrusion is accurately solved. The specific steps are described below. For a system containing a

specific number of N network nodes, a function S is defined to measure the error between the estimated distance of the network node to the target node and the actual distance.

$$S\left(x_1, x_2, \cdots, x_N\right) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[\hat{\delta}_{i,j} - d_{i,j}\left(x_i, x_j\right)\right]^2 \tag{12}$$

where, $i \neq j, \hat{\delta}_{i,j}$ is the estimated distance between nodes [13], and $d_{i,j}\left(x_i, x_j\right)$ is the Euclidean distance. The above formula can be expanded to derive the following expression:

$$S = \frac{1}{2}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\delta}_{i,j}^2 + \sum_{i=1}^{N}\sum_{j=1}^{N}d_{i,j}^2\left(x_i, x_j\right) - 2\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\delta}_{i,j}^2 d_{i,j}\left(x_i, x_j\right)\right] \tag{13}$$

Eq. (13) is transformed into a spatial matrix to represent the determination of the target node range of network node intrusion.

$$\begin{aligned} S(X) &= \frac{1}{2}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\delta}_{i,j}^2 + \sum_{i=1}^{N}\sum_{j=1}^{N}d_{i,j}^2(X) - \sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\delta}_{i,j}^2 d_{i,j}(X)\right. \\ &= \frac{1}{2}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\delta}_{ij}^2 - \frac{N}{2}trace\left\{XHX^T\right\} + trace\left\{X\psi(X)X^T\right\}\right. \end{aligned} \tag{14}$$

where, $trace\{\cdot\}$ is the trace operation, X is the matrix composed of all nodes $[x_1, x_2, \cdots, x_N]$, X^T is the transpose of X, H is the center matrix of node positioning space [14], with $H = I - ee^T/N, I$, I is the identity matrix, and $\psi(X)$ is the center matrix about X.

$$[\psi(X)] = \begin{cases} -\frac{\hat{\delta}_{i,j}}{d_{i,j}(x_i, x_j)}, i \neq j.d_{i,j} \neq 0 \\ -\sum_{p=1, p\neq i}^{N} w_{i,p}\left(x_i, x_p\right), i = j \end{cases} \tag{15}$$

where, $w_{i,p}\left(x_i, x_p\right)$ represents the spatial weight between node X_i and node X_p.

Based on the SMACOF theory, an auxiliary function is constructed to determine the target node coordinate $\varphi(X, Z)$ of network node intrusion:

$$\varphi(X, Z) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\delta}_{i,j}^2 - \frac{1}{2}Ntrance\left\{XHX^T\right\} + \text{trace}\left\{Z\psi(Z)X^T\right\} \tag{16}$$

where, Z represents the coordinate matrix of the target node in the low dimensional space, and $\psi(Z)$ represents the center matrix about Z.

For each iteration k, $Z = X_{k-1}$ is utilized, since the gradient function of $\varphi(X, Z)$ satisfies the following condition:

$$\nabla_{x\varphi(X,Z)} = \frac{1}{2}NXH - Z\psi(Z) \tag{17}$$

Setting the gradient to 0 derives the following expression:

$$X_k = \frac{2}{N}X_{k-1}\psi\left(X_{k-1}\right)H^{-1} \tag{18}$$

where, X_k is the space matrix composed of all nodes.

According to the characteristics of the node positioning space center matrix H, the network intrusion node range X' is derived as follows:

$$X' = X_k - \frac{2}{N}\varphi\left(X, X_{k-1}\right) \tag{19}$$

3 Identification of Intrusion Behavior of Computer Network Nodes

In computer network node intrusion detection, improving the RRN structure can deeply dig the characteristics of network node data, which is very key to accurately identifying the intrusion behavior, because many complex intrusion behaviors are often hidden in these unobtrusive characteristics. Residual connections allow the network to more easily learn small changes in the input data, thereby improving the detection of intrusion behavior. To improve the identification accuracy of intrusion behavior in subsequent computer network nodes, the improved RRN results were constructed, and the intrusion detection was carried out in the range of intrusion nodes preliminarily determined above.

3.1 Improvement of the RRN Structure

Based on the previous data processing and feature extraction results, the RRN architecture was constructed and optimized. The model can capture the features of network node intrusion behavior more effectively to enhance the detection accuracy and efficiency, laying a solid foundation for the subsequent optimization detection and intrusion determination. The iterative structure of RNN was introduced and the regression residual model was constructed by using the concept of jump connection in assisted residual network. The jump connection and recursive structure in the RRN help to speed up the training of the model, reduce the risk of overfitting, and make the model have higher detection efficiency and stability in practical applications. At the same time, using deep separable convolution technology to simplify model parameters can greatly reduce the number of model parameters, computational complexity and memory consumption while maintaining model performance. In this improved architecture, depth-separable convolution plays a central role, as shown in Figure 1.

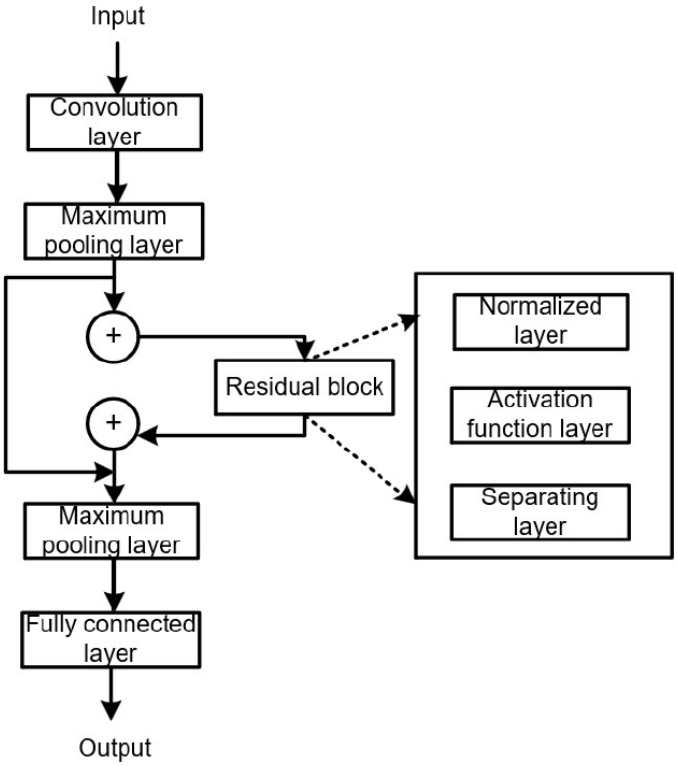

Figure 1. Improved recursive residual network structure

Based on the results of previous data processing and feature extraction, the RRN architecture was constructed. The network consists of multiple recursive blocks, each of which is composed of multiple residual units. The residual unit uses a pre-activated architecture to reduce training difficulty and improve performance. Designs that apply activation functions before the weight layer are named "pre-activation" architectures. The "pre-activated" architecture is less difficult to train and shows better performance. The residual unit in this architecture can be expressed as:

$$H^u = F\left(H^{u-1}, W^u\right) + H^{u-1} \tag{20}$$

where, $u = 1, 2, \cdots, U$, with U being the number of residual units [15], H^{u-1} and H^u are the input and output of the units, respectively, and F is the residual function.

In this study, Eq. (20) was adjusted, and the adjusted residual unit is defined as follows:

$$H^u = G\left(H^{u-1}\right) = F\left(H^{u-1}, W\right) + H^0 \tag{21}$$

where, G represents the adjusted residual function, and H^0 is the output of the first convolution layer.

Let B represent the number of recursive blocks in the network, x_{b-1} and $x_b (b = 1, 2, \cdots, B)$ represent the input and output of the b-th recursive block, and $H_b^0 = f_b\left(x_{b-1}\right)$ represent the result of x_{b-1} passing through the convolutional layer in the b-th recursive block [16]. Then, according to Eq. (21), the u-th residual element is:

$$H_b^u = G\left(H^{u-1}\right) = F\left(H_b^{u-1}, W_b\right) + H_b^0 \tag{22}$$

Then the output x_b of the b-th recursive block is:

$$x_b = H_b^U = G^{(U)}\left(f_b\left(x_{b-1}\right)\right) = G\left(G\left(\cdots\left(G\left(f_b\left(x_{b-1}\right)\right)\right)\cdots\right)\right) \tag{23}$$

In the RRN, the DWConv technique is used to reduce the model parameters. By dividing the standard convolution into deep convolution and point-by-point convolution, the calculation and parameter number of the model can be greatly reduced. The model was trained with a marked network node intrusion dataset. At the same time, the weight coefficient transfer and sharing mechanism in the recursive structure was used to effectively control the scale of network parameters. After training and tuning, the improved RRN was deployed to the actual computer network intrusion detection system. In practical applications, the model can monitor the behavior of network nodes in real time and identify the potential intrusion behavior accurately.

3.2 Calculation of the Node Reputation Value

In a complex network environment, the behavior of nodes may be affected by many factors and fluctuate. For example, a node may experience a short period of increased network traffic due to a temporary business need, but this does not necessarily mean that it has been compromised. By considering many factors comprehensively to calculate the credit value, the normal behavior fluctuation can be accurately distinguished from the real intrusion behavior to improve the accuracy of intrusion detection. Therefore, multiple factors were considered in this study to calculate the comprehensive reputation value of a node, aiming to evaluate the node security more comprehensively, effectively reduce misjudgment, and improve the reliability of detection. The recursive structure of the RRN enables the model to deal with more complex network node behavior patterns and realize recursive transfer and sharing of weight coefficients among residual modules. Based on this recursive structure, the credit value of nodes can be iterated and updated constantly so that it can reflect the actual node behavior more accurately and realize intrusion detection. The change of node reputation value is based on the dynamic status of network nodes, and is not a simple numerical accumulation. The specific steps are as follows:

Step 1: Direct credit value calculation. The Bayesian method is used to estimate the success rate of node communication. The subjective probability of incomplete or unknown information is evaluated first, and then the Bayesian formula is used for correction. Finally, the revised credit value is used for evaluation. The specific application of the Bayes formula is as follows:

$$P\left(D_j \mid E\right) = \frac{P\left(D_j\right) P\left(E \mid D_j\right)}{\sum_{i=1}^{n} P\left(D_j\right) P\left(E \mid D_j\right)} (j = 1, 2, \ldots, n) \tag{24}$$

where, D_j is a subset of the standard set [17], E is the element in the annotation set, $P\left(D_j\right)$ is the communication success rate of standard set subset nodes, and $P\left(E \mid D_j\right)$ indicates the communication success rate of element nodes in the standard set subset. The $Beta$ distribution function is used to implement the fitting analysis of intrusion behavior identification. Assuming that the credit value of node X represents the number of successful data transfers of node X, then the following equation can be derived:

$$Beta(\alpha, \beta) = \frac{\tau(\alpha + \beta) P\left(D_j \mid E\right)}{\tau(\alpha)\tau(\beta)} \sigma^{\alpha-1}(1 - \sigma)^{\beta-1} \tag{25}$$

where, α and β are parameters greater than 0. Probability transformation of the $Beta$ distribution function is performed, and the final result of credit value is obtained as follows:

$$T_{ij}^{d} = E\left(R_{ij}\right) = E(Beta(\alpha, \beta)) = \frac{r+1}{r+s+1} \tag{26}$$

where, s is the number of information transmission failures [18].

Step 2: Indirect credit value calculation. The data of neighboring nodes are used to calculate the credit value of the current node. By analyzing and calculating the data of other nodes, the reputation behavior of the current node can be evaluated, and more accurate results can be obtained.

Let the underlying nodes x, y, and z be the same. For node x, the indirect reputation value can be transferred between nodes, and its calculation formula is as follows:

$$T_{kj}^{i} = \frac{\sum_{i=1}^{n} \sqrt{R_{ik} + R_{kj}}}{n} \tag{27}$$

where, R_{ik} and R_{kj} represent the credit value of data flowing between y nodes and z nodes [19]. There is a risk in the calculation of the indirect credit value because the internal intrusion node may adopt a camouflage strategy to improve its own credit value or reduce the credit value of normal nodes. To deal with this problem, another credit calculation method needs to be adopted to identify and judge the intrusion node more accurately.

Step 3: Comprehensive credit value calculation. In view of the limitations of the above two credit value quantification methods, a comprehensive credit value evaluation method is adopted. Based on the comprehensive credit value of each node in the wireless network at any time, the intrusion optimization detection model of data nodes transmitted in the computer network is constructed.

$$X_m = c * T_{ij}^{d} + (1-c) * T_{kj}^{i} \tag{28}$$

where, X_m is the comprehensive credit value, and the value range of parameter c is [0, 1]. Therefore, the results of Eq. (28) are substituted into the above Eq. (23) as the input of the improved RRN, and the detection results of computer network node intrusion are the output.

3.3 Determining the Intrusion Behavior

On the basis of optimized detection, the potential intrusion behavior was further judged, involving calculating potential loss values or other relevant metrics and analyzing the detected abnormal behavior in depth to determine whether it constitutes a true breach. Once a network anomaly is detected, the relevant node can quickly send the abnormal information to the base station, which, as the center of the network, has sufficient energy and powerful computing power and is responsible for in-depth identification of the abnormal information. The goal of identification is to accurately determine whether there is network node intrusion to isolate malicious nodes in time and ensure the normal operation of the network and the security of data transmission.

This study evaluates the potential loss value L of the abnormal behavior to determine whether it should be recognized as an intrusion behavior. If the value exceeds the threshold, it is an intrusion behavior, and measures are taken immediately to isolate malicious nodes and maintain stable network operation. The relevant calculation formula is shown in Eq. (29).

$$a_i = \sum_{s \in M_i} \sum_{k=1}^{K} x_b w_s * \tau_k \tag{29}$$

where, a_i is the threat indicator of node i affected by abnormal behavior, W_s is the number of times that node s monitoring reports the intrusion action to the base station, K indicates the number of detected intrusions, τ_k is the weight of detecting intrusion behavior k [20], and M_i is a set of monitoring sets for node i. Thus, L, the potential loss of node I, can be obtained, as shown in Eq. (30).

$$L = a_i * c_i \tag{30}$$

where, c_i is the importance of node i, and the potential loss value L of each node is calculated by considering the potential harm of abnormal behavior comprehensively. Once the value exceeds the preset threshold, the base station is able to determine that the node is a malicious node, confirm the existence of intrusion, and immediately perform isolation procedures to remove it from the network to prevent further damage. The threshold is not arbitrarily

set, but is based on in-depth analysis of a large amount of historical network intrusion data and normal network behavior data. By conducting a detailed study of past intrusion events, the actual losses caused by different types of intrusion behaviors to the network were calculated, including data leakage, service interruption duration, and degree of network performance degradation. Meanwhile, considering the potential loss fluctuation range under normal network behavior, a threshold that can distinguish intrusion behavior from normal abnormal fluctuations was comprehensively determined. In addition, the expected utility-based method was used to calibrate the threshold, completing the identification of intrusion behavior in computer network nodes.

4 Experimental Results and Analysis

4.1 Preparation for Experiment

To verify the effectiveness of the research on intrusion behavior identification in computer network nodes based on the proposed improved RRN, the designed method was compared with some baseline methods [3–6].

In terms of experimental setup, the mini-batch size was 128, the training period (epochs) was 10 rounds, and the learning rate was set to 0.0001. The test dataset used is CIC-IDS-2022, which contains 85 features and 3119346 communication session instance samples. In addition, the cycle number of 50,000 and the batch size of 128 were set. The Adam optimizer was simultaneously used by employing its adaptive learning rate characteristics to accelerate convergence and improve training stability. To prevent overfitting, multiple regularization strategies were implemented. A dropout layer with a dropout rate of 0.5 was introduced after the convolutional layer of the RRN to reduce the complex co-adaptation relationships between neurons. At the same time, a weight decay coefficient of 1e-5 was set in the optimizer to constrain the complexity of the model. To address the issue of class imbalance in the CIC-IDS-2022 dataset, the focal loss based on the cross entropy loss function was introduced, which enhanced the model's ability to recognize minority class intrusion behaviors by adjusting the weights of difficult and easy samples. In addition, a cosine annealing scheduling strategy was adopted for the learning rate, which gradually decreased with training epochs, thus achieving more precise parameter adjustment in the later stage of training.

To build a stable and controllable measurement environment, the TensorFlow framework was adopted, and the execution program was written in Python. To ensure the specificity and anti-interference of the execution program, Java was additionally employed to invoke the Python program. In experimental evaluation, the training objective function is usually aimed at minimizing the difference between the predicted results and the true labels to improve the model's accurate identification ability of intrusion behavior. The cross entropy loss function was adopted as the training objective function for multi-classification problems. By minimizing the loss function, the model can continuously adjust its own parameters to make the prediction results closer to the real situation, thereby improving the accuracy of intrusion behavior identification. To effectively evaluate the identification effectiveness of the proposed method, the identification accuracy, the F1 value, the intrusion behavior identification rate, and the changes in malicious node trust values were used as indicators. The identification accuracy is a commonly used indicator to measure the overall performance of the model, which represents the proportion of correctly predicted samples to the total number of samples. The calculation formula is expressed as:

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{31}$$

where, TP represents the true example, which is the number of samples that are actually intrusion behaviors and correctly predicted by the model; TN represents true negative cases, which refers to the number of samples that are actually normal behavior and correctly predicted by the model; FP stands for false positive examples, which refers to the number of samples that are actually normal behavior but incorrectly predicted by the model as invasive behavior; FN stands for false negative examples, which refers to the number of samples that are actually invasive behavior but incorrectly predicted by the model as normal behavior.

The F1 value is the harmonic mean of precision and recall, taking into account both the accuracy and recall capability of the model. Accuracy represents the proportion of samples predicted by the model as positive cases that are actually positive cases, while recall represents the proportion of samples predicted correctly by the model as positive cases. The calculation formula is as follows:

$$F1 = 2\frac{P \times R}{P + R} \tag{32}$$

where, P represents precision and R represents recall.

In the identification of intrusion behavior in computer network nodes, the early stop strategy aims to prevent overfitting of the model during training by monitoring the performance indicators of the model on the validation set (such as validation loss or validation accuracy) to stop training. The specific steps are as follows:

Step 1: Setting of the patience value. The patience value is a pre-set integer that represents the number of epochs that the model is allowed to continue training without further improvement in the validation set's performance. For example, setting the patience value to 5 means that if there is no improvement in performance on the validation set for five consecutive epochs, training can be stopped.

Step 2: Monitoring of the performance of the validation set. After each epoch of training, the model is applied to the validation set and the performance metrics such as validation loss or validation accuracy are calculated.

Step 3: Determining whether to stop early. The performance of the validation set for the current epoch is compared with the previous best performance. If the current performance does not improve, a counter is added. If there is an improvement in current performance, the counter is reset to 0. When the counter reaches the patience value, the training is stopped and the previously best-performing model parameters are used as the final model parameters.

Through the above training objective function, the indicator calculation method, and the early stopping strategy on the validation set, it is possible to effectively train and optimize an intrusion behavior identification model for computer network nodes based on the improved RRN and to improve its accurate identification ability of intrusion behavior. A medium-sized enterprise network testing environment was established, in which the Open Shortest Path First (OSPF) routing protocol was employed, with a topology structure including core switches, routers, database servers, web servers, and multiple client subnets. A total of 150 logical nodes were deployed in the network to simulate the distribution and communication relationships of devices in a real enterprise network, with a simulation time set at 150 seconds.

In the configured system test environment, programming tools were used to convert the design part into program code and import it into the main test computer. Then, the network was adjusted to the running state and the corresponding network running data were generated. Then, the computer network was connected to the system software running program through real-time data acquisition, protocol analysis, feature extraction and other steps, finally outputting the intrusion behavior identification results for computer network nodes. The experimental environment diagram is shown in Figure 2.

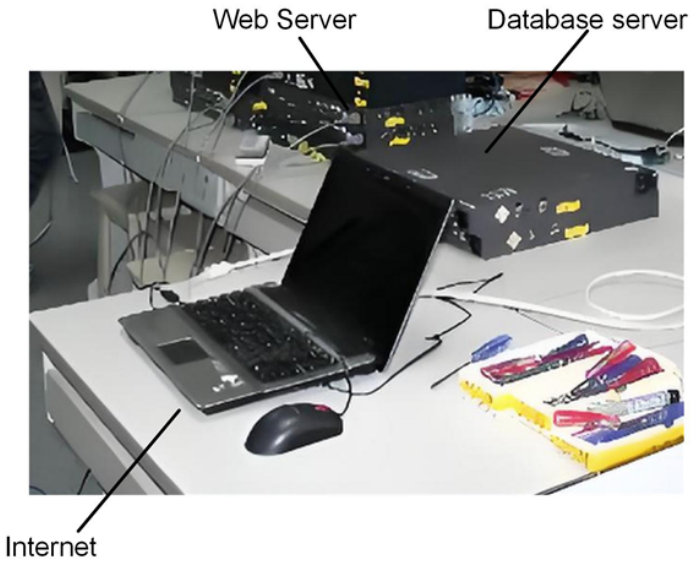

Figure 2. Experimental environment diagram

A total of 16200 communication session samples were randomly selected from the CIC-IDS-2022 dataset for in-depth analysis. After the network system was started, network traffic data were collected through the mirror port, and mixed analysis was performed based on 16200 session instances in the CIC-IDS-2022 dataset. The selected sample data cover a wide range of normal communication patterns and intrusion activities, including denial-of-service (DoS) attacks, R2L attacks, port scanning and monitoring activities (Probe), and unauthorized attempts to gain local superuser privileges (U2R), as shown in Table 2.

All configurations were consistent on each node. In the same network environment, the maximum working bandwidth was set to 300 Mbps.The sparse coding technology was adopted as the coding method of nodes. Keras was employed as the learning library of this experiment. To facilitate the reproducibility and verification of this research, the training scripts, data preprocessing pipeline, model implementation, and the complete hardware/software environment are described in detail below.

Table 2. Number of node intrusion detection samples

Network State	Training Samples	Test Samples
Normal	10000	2000
DoS attacks	1000	200
R2L	1000	200
Port scanning and monitoring activities (Probe)	1000	200
U2R	500	100

(a) Training script and model implementation

The core architecture of the improved RRN was implemented using the TensorFlow and Keras frameworks. The model leverages DWConv and a pre-activation residual structure within recursive blocks. The following pseudo-code illustrates the key components of the model-building process:

```python
import tensorflow as tf
from tensorflow.keras.layers import *
from tensorflow.keras.models import Model

def pre_activation_residual_unit(input_tensor, filters, kernel_size=3):
    """
    Defines a pre-activation residual unit.
    """
    x = BatchNormalization()(input_tensor)
    x = Activation('relu')(x)
    x = DepthwiseConv2D(kernel_size, padding='same')(x)
    x = Conv2D(filters, (1, 1), padding='same')(x)
    return x

def build_recursive_block(input_tensor, num_units, filters):
    """
    Builds a recursive block containing multiple residual units.
    """
    x = input_tensor
    for _ in range(num_units):
    residual = pre_activation_residual_unit(x, filters)
    x = Add()([x, residual])
    return x

def build_improved_rrn(input_shape, num_classes, U=5, N=8, base_filters=64):
    """
    Constructs the complete Improved RRN model.
    Args:
    U: Number of recursive blocks.
    N: Number of residual units per block.
    base_filters: Base number of filters for convolutional layers.
    """
    inputs = Input(shape=input_shape)
    x = inputs

    # Initial Convolution
    x = Conv2D(base_filters, (3, 3), padding='same')(x)

    # Stack of Recursive Blocks
    for _ in range(U):
```

```
    x = build_recursive_block(x, N, base_filters)

    # Final Classification Head
    x = BatchNormalization()(x)
    x = Activation('relu')(x)
    x = GlobalAveragePooling2D()(x)
    outputs = Dense(num_classes, activation='softmax')(x)

    model = Model(inputs, outputs)
    return model

# Example: Model Instantiation
model = build_improved_rrn(input_shape=(None, None, 1), num_classes=5, U=5, N=8)
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

(b) Data preprocessing pipeline

The CIC-IDS-2022 dataset was processed through the following pipeline to prepare it for model input: data cleaning-feature scaling-sequence construction-label encoding. Rows with missing values and duplicate communication sessions were first removed. Numerical features were standardized using Z-score normalization to have a mean of 0 and a standard deviation of 1. Then continuous communication sessions were segmented into fixed-length time-series windows to capture temporal behavior. Multi-class labels (normal, DoS, R2L, Probe, and U2R) were converted into a one-hot encoded format.

(c) Hardware and software environment

The experiments were conducted in the following environment to ensure consistency:

• Hardware configuration: Intel Xeon Gold 6226R CPU (2.90 GHz); two NVIDIA Tesla V100 GPUs (32 GB VRAM each); and 128 GB DDR4 memory.

• Software configuration: Ubuntu 20.04.6 LTS operating system; Python 3.8.10; and key libraries including TensorFlow 2.10.0, Keras 2.10.0, NumPy 1.21.5, Pandas 1.3.5, and Scikit-learn 1.0.2.

4.2 Experimental Results

The proposed intrusion behavior identification method for computer network nodes integrates multiple modules such as classification preprocessing, PCA feature extraction, SMACOF range localization, improved RRN, and reputation value fusion. To further explore the contribution of each module to the overall method performance, a step-by-step ablation experiment was conducted. By sequentially removing specific modules and comparing the performance of methods under different conditions, the key roles of each module in intrusion behavior identification can be clarified. The identification accuracy is used for measurement, and the results of the ablation experiment are shown in Table 3.

Table 3. Results of the ablation experiment

Plan	Identification Accuracy
Design method	0.97
Removal of the classification preprocessing module	0.90
Removal of the PCA feature extraction module	0.92
Removal of the SMACOF range positioning module	0.93
Removal of the improved RRN module	0.87

According to the results in Table 3, it can be seen that after removing the classification preprocessing module, the recognition accuracy significantly decreases, indicating that classification preprocessing can effectively distinguish normal data from potential intrusion data, providing a more accurate basis for subsequent processing and reducing the impact of interference data on detection results. Removing the PCA feature extraction module, the SMACOF range localization module, and the reputation value fusion module results in a decrease in performance. Without PCA's dimensionality reduction and feature optimization, the model may face difficulties in processing high-dimensional data, which affects the accurate recognition of intrusion behavior. Without SMACOF, it is difficult to determine

the range of network intrusion target nodes, which is affected by the interference of irrelevant node data and affects the accuracy of detection. Without reputation value fusion, relying solely on a single detection result for judgment can easily lead to misjudgment or omission of intrusion behavior. The removal of the improved RRN module has a significant impact on performance. This module can deeply explore the characteristics of network node data, and residual connections and recursive structures help accelerate model training speed and reduce the risk of overfitting.

Traditional neural network structures have limited ability to handle complex network node behavior patterns, resulting in a significant decrease in detection performance. Through gradual ablation experiments, the important role of each module in identifying intrusion behavior of computer network nodes was clarified. Classification preprocessing, the PCA feature extraction, the SMACOF range localization, the improved RRN, and the reputation value fusion module all contribute to improving the performance of the method. Among them, the improvement of the RRN module has the most significant impact, followed by the classification preprocessing module. In practical applications, the advantages of each module should be fully utilized to ensure the accuracy and reliability of identifying intrusion behaviors of computer network nodes.

Sensitivity analysis on parameters was conducted, such as the number of recursive blocks (U), the number of residual units (N), and the DWConv configuration in the improved RRN, to determine the optimal parameter configuration. Among them, the number of recursive blocks controls the number of repetitions of the recursive structure, and the testing range is from U = 2 to U = 8. The range of residual unit number testing is from N = 4 to N = 16. DWConv was evaluated in two configurations: standard convolution and depthwise convolution. When DWConv was enabled, the expansion factor α was tested over the range of 1.0 (no channel expansion) to 2.0 (doubling the number of channels). The results are shown in Table 4 below.

Table 4. Sensitivity analysis results

Parameter Configuration	Identification Accuracy (%)	Parameter Quantity (M)	Training Time (h)
$U = 5, N = 8$, standard convolution	94.7	39.0	12.5
$U = 5, N = 8$, DWConv ($\alpha = 1.5$)	95.3	29.5	11.8
$U = 8, N = 12$, standard convolution	93.8	68.3	18.7
$U = 3, N = 4$, DWConv ($\alpha = 1.0$)	93.6	12.4	7.2
$U = 5, N = 8$, DWConv ($\alpha = 1.0$)	98.2	28.1	11.2

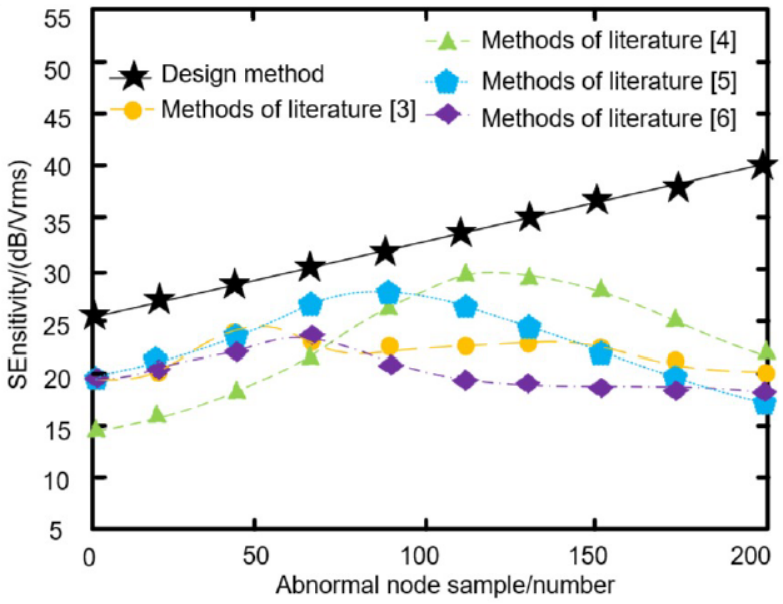

Figure 3. Results of F1 value identification for abnormal data using different methods

According to the results in Table 4, it can be seen that when the number of recursive blocks (U) is 5, the number of residual units (N) is 8, and the DWConv configuration is α = 1.0, the network performance reaches its optimal level, which can provide reliable support for identifying intrusion behavior of computer network nodes. Using literature review methods and the proposed method, intrusion behavior identification was carried out on various computer

network nodes, with the F1 value as the evaluation index. A total of 200 samples was randomly selected from the test sample for testing, and the F1 values of each party are shown in Figure 3.

As shown in Figure 3, there is a significant difference in the F1 value between the design method and the other four methods. Specifically, the F1 value of the proposed method consistently remains above 0.9, while the F1 values of the other four literature methods are all below 0.8. By comparing the F1 values of five methods for intrusion behavior identification of various computer network nodes, it can be concluded that for abnormal data, the intrusion behavior identification method of the designed method shows good identification accuracy, significantly better than other methods. To evaluate the performance of the methods more scientifically, the 95% confidence intervals of the F1 values for each method were calculated, and the results are shown in Table 5.

Table 5. 95% confidence interval results of F1 values for each method

Method	Mean F1 Value	95% Confidence Interval
Design method	0.92	[0.90, 0.94]
Method in the study [3]	0.58	[0.52, 0.60]
Method in the study [4]	0.76	[0.50, 0.80]
Method in the study [5]	0.65	[0.55, 0.72]
Method in the study [6]	0.58	[0.55, 0.63]

According to the results in Table 5, the lower limit of the F1 confidence interval for the design method is 0.90, which consistently remains above 0.90. The upper limit of the F1 confidence interval for the other four literature methods is 0.80 or below. The design method demonstrates excellent identification accuracy. Its high F1 mean value (0.92) and narrow confidence interval ([0.90, 0.94]) indicate that this method has not only excellent average performance but also strong performance stability. This means that the design method can reliably achieve accurate identification of intrusion behavior in different sample situations, reducing the significant performance changes caused by sample fluctuations. The design method has significant advantages in identifying intrusion behaviors of computer network nodes and can more accurately and stably identify abnormal data, providing more reliable technical support for network security protection. The change of intrusion behavior identification rate of the proposed algorithm and other methods over time is shown in Table 6.

Table 6. Changes of the identification rate of intrusion behavior (%)

Time (s)	Design Method	Method in the study [3]	Method in the study [4]	Method in the study [5]	Method in the study [6]
10	88.2	80.3	85.4	85.5	85.6
20	93.5	80.5	85.7	85.9	86.0
30	94.4	87.5	87.8	87.9	87.7
40	95.0	86.4	86.9	86.8	86.7
50	95.1	80.7	85.1	85.6	85.5
60	95.2	80.5	82.3	82.6	82.4
70	95.3	89.8	80.9	80.7	80.6

As can be seen from Table 6, at each measured time points (10 seconds to 70 seconds), the identification rate of intrusion behavior of the design method remained between 88.2% and 95.3%, showing high stability and accuracy. The identification rate of intrusion behavior increased from 88.2% of 10 seconds to 95.3% of 70 seconds, an increase of 7.1 percentage points, indicating that the performance of the design method is constantly improving over time. The identification rate of intrusion behavior of the method in the study [3] fluctuated greatly, from 80.3% in 10 seconds to 80.5% in 20 seconds, and then to 80.5% and 80.7% in 60 seconds and 70 seconds, with both increases and decreases during the period. Using the method in the study [4], the identification rate of intrusion behavior increased from 85.4% in 10 seconds to 87.8% in 30 seconds, but then decreased slightly to 80.9% in 70 seconds. The identification rate of intrusion behavior of the method in the study [5] showed an overall upward trend, from 85.5% in 10 seconds to 85.6% in 50 seconds, but then fluctuated slightly. Using the method in the study [6], the identification rate of intrusion behavior increased from 85.6% in 10 seconds to 87.7% in 30 seconds, but then gradually decreased to 82.4% in 70 seconds. In summary, the identification rate of intrusion behavior of the design method maintains a high level at each time point, and the fluctuation is small, which indicates that it has good stability.

To show the advantages of the proposed method more clearly, the results of the comparison of different methods on the trust value of malicious nodes are shown in Figure 4.

It can be clearly seen from the presentation of Figure 4 that the node trust value of the design method dropped below 0.15 after 75 seconds, while the node trust value of the literature method was above 0.2. Therefore, the proposed method is highly sensitive to malicious nodes, and can effectively identify and respond to malicious nodes to ensure that malicious nodes cannot interfere with normal trust relationships, which has a positive impact on network security. Therefore, normal nodes in the network can trust each other more, improving the overall security and stability of the network. The changes of network nodes under different methods are shown in Figure 5.

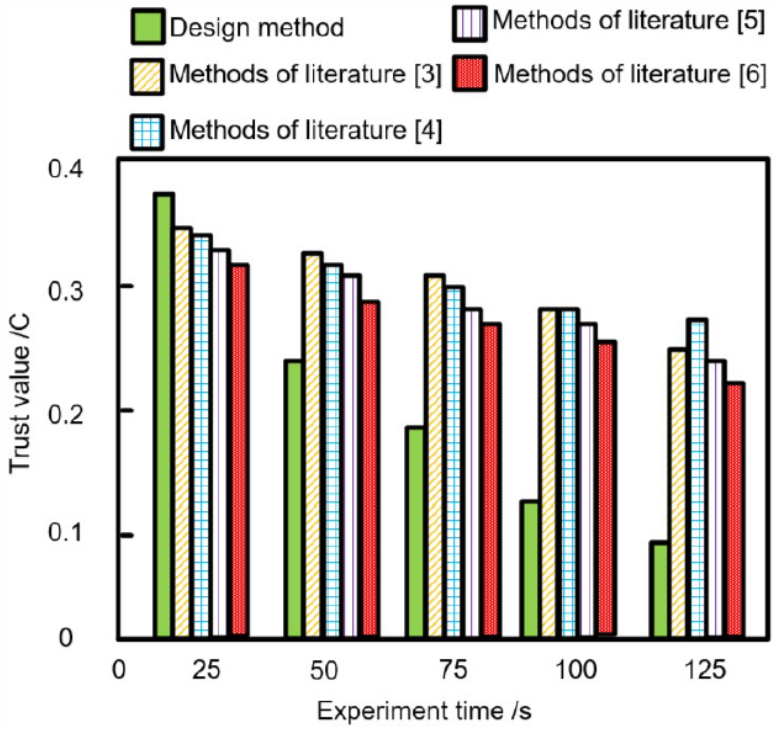

Figure 4. Influence of different methods on the trust value of malicious nodes

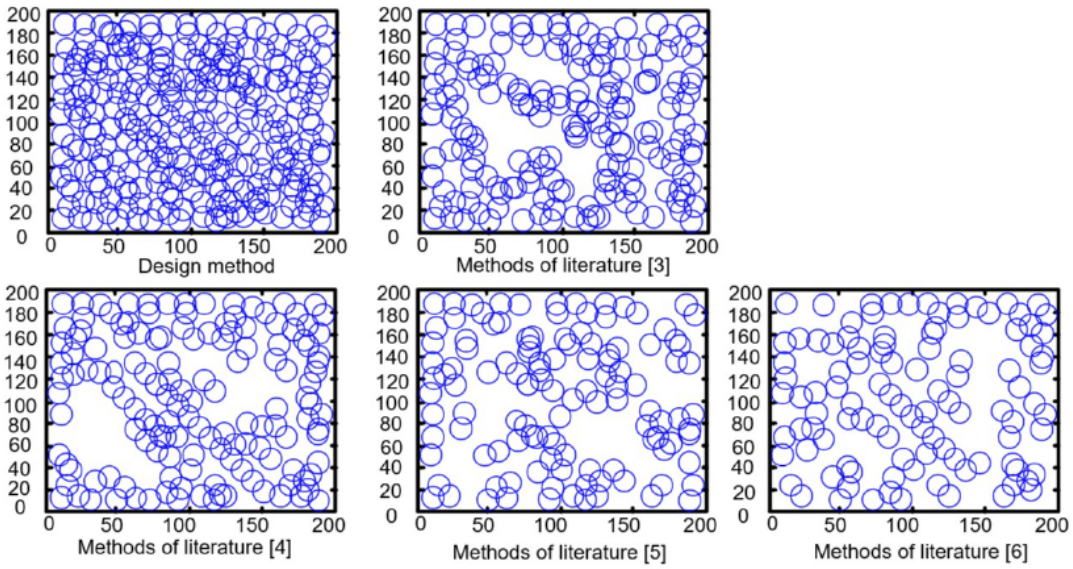

Figure 5. Changes of network nodes under different methods

As can be seen from the results in Figure 5, in actual network applications, the design method has a large coverage area of network nodes, which can prevent intruders from touching the network core and control nodes to achieve self-protection. In the face of an intrusion situation, when the other four methods were adopted, the area of network

nodes was not completely covered, and some nodes were eroded by intrusion nodes, reducing the coverage area. Therefore, the proposed design method is more practical in evaluating the results.

On the basis of the above analysis, to further verify the real-time performance of the proposed method, the performance differences between the proposed method and the other four methods [3–6] in identifying intrusion behavior of computer network nodes were compared. Delay refers to the time interval from the occurrence of an anomaly to the output of the model's identification results, measured in milliseconds. The above methods were used to identify five groups of samples, each containing 1000 samples. The identification delay of each method was calculated, and the results are shown in Table 7.

Table 7. Identification delay of various methods

Group Number	Identification Delay (ms)				
	Proposed method	Method in the study [3]	Method in the study [4]	Method in the study [5]	Method in the study [6]
1	12.3	45.7	38.9	31.4	27.8
2	11.6	44.5	37.3	30.2	26.5
3	11.7	44.8	37.6	30.8	26.9
4	12.0	45.2	38.4	31.1	27.4
5	12.2	45.4	38.8	31.3	27.7

According to the results in Table 7, the highest average detection delay of the proposed method (12.3 ms) is significantly lower than that of the literature method (27.8–45.7 m s). This indicates that the proposed method has significant advantages in intrusion behavior identification latency, with lower latency verifying the effectiveness of the improved RRN in real-time performance.

5 Conclusions

In this design, a recursive structure was introduced into the improved RRN-based node intrusion behavior identification framework. This structure enables the use of recursive weight coefficients within the residual module, allowing the scale of network parameters to be effectively controlled and the overall network architecture to be simplified. According to the comprehensive credit value of the node, the intrusion detection model of the data node in computer network transmission was established. Based on the detection results, the loss value was used to identify the intrusion behavior of network nodes. When the potential loss value exceeds the specified threshold, the base station is able to determine that the node is a malicious node with intrusion behavior and immediately start the isolation program to ensure the stable operation of the network and the safe transmission of data. Although the improved RRN performs well in detection, its generalization capabilities still need to be further improved. Especially in the face of new or unknown intrusion behavior, the model may not be able to accurately identify it. By integrating various regularization techniques and integrated learning strategies, the generalization performance of the model can be enhanced. In addition, exploring transfer learning and other means to apply cross-domain knowledge to intrusion detection can improve the adaptability of the model. At the same time, the research and development of distributed intrusion detection technology can be strengthened to improve the detection ability of distributed network attacks.

Data Availability

The data used to support the research findings are available from the corresponding author upon request.

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] S. Wang, W. Xu, and Y. Liu, "Res-TranBiLSTM: An intelligent approach for intrusion detection in the Internet of Things," *Comput. Netw.*, vol. 235, p. 109982, 2023. https://doi.org/10.1016/j.comnet.2023.109982

[2] S. Rajabi, S. Asgari, S. Jamali, and R. Fotohi, "An intrusion detection system using the artificial neural network-based approach and firefly algorithm," *Wireless Pers. Commun.*, vol. 137, no. 4, pp. 2409–2440, 2024. https://doi.org/10.1007/s11277-024-11505-5

[3] Z. Cao and S. H. S. Huang, "A behavioral graph model for host-based intrusion detection," *J. Inf. Assur. Secur.*, vol. 18, no. 2, p. 48, 2023.

[4] R. Reka, R. Karthick, R. S. Ram, and G. Singh, "Multi head self-attention gated graph convolutional network based multi attack intrusion detection in MANET," *Comput. Secur.*, vol. 136, p. 103526, 2024. https://doi.org/10.1016/j.cose.2023.103526

[5] N. Xie, C. Zhang, Q. Yuan, J. Kong, and X. Di, "IoV-BCFL: An intrusion detection method for IoV based on blockchain and federated learning," *Ad Hoc Netw.*, vol. 163, p. 103590, 2024. https://doi.org/10.1016/j.adhoc.2024.103590

[6] R. Sathiya and N. Yuvaraj, "Swarm optimized differential evolution and probabilistic extreme learning based intrusion detection in MANET," *Comput. Secur.*, vol. 144, p. 103970, 2024. https://doi.org/10.1016/j.cose.2024.103970

[7] F. Ullah, S. Ullah, G. Srivastava, and J. C. W. Lin, "IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic," *Digit. Commun. Netw.*, vol. 10, no. 1, pp. 190–204, 2024. https://doi.org/10.1016/j.dcan.2023.03.008

[8] L. Xing, K. Wang, H. Wu, H. Ma, and X. Zhang, "FL-MAAE: An intrusion detection method for the internet of vehicles based on federated learning and memory-augmented autoencoder," *Electronics*, vol. 12, no. 10, p. 2284, 2023. https://doi.org/10.3390/electronics12102284

[9] B. Xu, "Design of intrusion detection system for intelligent mobile network teaching," *Comput. Electr. Eng.*, vol. 112, p. 109013, 2023. https://doi.org/10.1016/j.compeleceng.2023.109013

[10] X. Wang and X. Liao, "Mathematical modeling and simulation of network multi intrusion behavior identification," *Comput. Simul.*, vol. 39, no. 2, pp. 452–456, 2022.

[11] Y. Song, "Public cloud network intrusion and internet legal supervision based on abnormal feature detection," *Comput. Electr. Eng.*, vol. 112, p. 109015, 2023. https://doi.org/10.1016/j.compeleceng.2023.109015

[12] Y. Zhang, "The WSN intrusion detection method based on deep data mining," *J. Cyber Secur. Technol.*, vol. 7, no. 3, pp. 115–133, 2023. https://doi.org/10.1080/23742917.2022.2162195

[13] B. E. Benaissa, C. Bessenouci, O. M. Ikumapayi, A. Q. Al-Dujaili, A. I. Abdulkareem, A. J. Humaidi, G. Lorenzini, and Y. Menni, "Optimizing the average hop-count and node distance using an adjusted DV-Hop algorithm with a distance error rate," *Instrum. Mes. Métrol.*, vol. 22, no. 1, pp. 1–9, 2023. https://doi.org/10.18280/i2m.220101

[14] L. Xia and X. Xia, "Network security intrusion detection methods combining optimization algorithms and neural networks," *Procedia Comput. Sci.*, vol. 228, pp. 582–592, 2023. https://doi.org/10.1016/j.procs.2023.11.067

[15] H. Qu, L. Di, J. Liang, and H. Liu, "U-SMR: U-SwinT & multi-residual network for fabric defect detection," *Eng. Appl. Artif. Intell.*, vol. 126, p. 107094, 2023. https://doi.org/10.1016/j.engappai.2023.107094

[16] A. B. Mohammed, L. C. Fourati, and A. M. Fakhrudeen, "Comprehensive systematic review of intelligent approaches in UAV-based intrusion detection, blockchain, and network security," *Comput. Netw.*, vol. 239, p. 110140, 2024. https://doi.org/10.1016/j.comnet.2023.110140

[17] J. Liang and J. Liu, "Maritime transportation risk assessment: A multilevel node relationship-based fuzzy bayesian network," *Ocean Eng.*, vol. 312, p. 119204, 2024. https://doi.org/10.1016/j.oceaneng.2024.119204

[18] R. Ghanbarzadeh, A. Hosseinalipour, and A. Ghaffari, "A novel network intrusion detection method based on metaheuristic optimisation algorithms," *J. Ambient Intell. Humaniz. Comput.*, vol. 14, no. 6, pp. 7575–7592, 2023. https://doi.org/10.1007/s12652-023-04571-3

[19] N. Omer, A. H. Samak, A. I. Taloba, and R. M. Abd El-Aziz, "A novel optimized probabilistic neural network approach for intrusion detection and categorization," *Alexandria Eng. J.*, vol. 72, pp. 351–361, 2023. https://doi.org/10.1016/j.aej.2023.03.093

[20] S. Devaraju, D. Soni, S. Jawahar, J. P. Maurya, and V. Tiwari, "Performance exploration of Network Intrusion Detection System with Neural Network classifier on the KDD dataset," *Int. J. Saf. Secur. Eng.*, vol. 14, no. 5, pp. 1431–1437, 2024. https://doi.org/10.18280/ijsse.140510