MSCNN-BiLSTM: A Network Intrusion Detection Model Optimized by Genetic Algorithm with Deep Spatio-Temporal Feature Learning

Haiqin Liu
Information Engineering College, Nanjing Polytechnic Institute, Nanjing 210048, Jiangsu, China
E-mail: qingran729@163.com

Keywords: deep spatio-temporal feature learning, network intrusion, ID

Received: July 31, 2025

Against the backdrop of accelerated digital transformation and evolving cyberattacks, traditional intrusion detection methods face severe challenges due to their limited feature extraction capabilities, fragmented spatiotemporal feature processing, and insufficient model adaptability. These challenges include a high proportion of encrypted traffic and the emergence of novel attacks. To enhance detection accuracy and adaptability, this paper proposes a Multi-Scale Convolutional Neural Network-Bidirectional Long Short-Term Memory Network (MSCNN-BiLSTM) model optimized by genetic algorithms. This approach utilizes the multi-branch structure of MSCNN (with 1x1, 3x3, and 5x5 convolutional kernels) coupled with a spatial attention mechanism to strengthen spatial feature extraction. It combines BiLSTM and a self-attention mechanism to capture temporal dependencies, and employs genetic algorithms to automatically optimize hyperparameters, achieving efficient synchronous learning of spatiotemporal features in network traffic. Experimental results on the UNSW-NB15 dataset demonstrate that the model achieves an accuracy of 96.8%, a recall rate of 96.4%, a precision rate of 97.2%, and an F1-score of 96.7%. In robustness tests, the performance loss under noise and adversarial attacks is controllable (average <5%). The average performance loss in cross-scenario transfer experiments is only 4.2%, significantly outperforming comparative models. The model proposed in this paper effectively addresses the limitations of traditional methods in feature extraction and adaptability. Its main contribution lies in the innovative integration of a deep spatiotemporal feature learning framework, providing key technical support for building a highly accurate and strongly adaptive network intrusion detection system. This has significant theoretical value and broad application prospects.

Povzetek: Predlagan je IDS model MSCNN-BiLSTM z genetsko optimizacijo hiperparametrov, ki se z večskalnimi konvolucijami in (samo)pozornostjo bolje uči prostorsko-časovnih značilnic prometa ter na UNSW-NB15 doseže dobre rezultate, ob tem pa ostane robusten in dobro prenosljiv.

1 Introduction

Against the background of accelerated digital transformation, the global cyberspace security situation continues to deteriorate. According to 2024 data from the International Telecommunication Union (ITU), targeted attacks on critical infrastructure have increased by 210% year-on-year, and the zero-day exploit cycle has been shortened to seven days. As the core link of network security defense-in-depth system, network ID has gone through three stages in its technological evolution: early misuse detection based on feature signatures (such as Snort rule base). Although these technologies have improved the detection accuracy to 91.2% (CIC-IDS2023 benchmark test), they still face fundamental challenges when dealing with new attacks. First, the proportion of encrypted traffic exceeds 85%, which makes it difficult to extract effective features. Secondly, the long-term latency characteristics of APT attacks are contradictory to the traditional detection window period; Finally, industrial Internet scenarios require that the detection delay must be less than 10ms, and it is difficult for existing algorithms to balance real-time and accuracy. From the perspective of strategic value, high-performance ID system can reduce the average annual security operation and maintenance cost of enterprises by 37% (Gartner 2025), which plays an irreplaceable role in safeguarding national network sovereignty [1].

Traditional network detection methods have three main shortcomings. First, the feature extraction ability is limited, the rule-based method cannot identify new attacks, and the machine learning method relies on artificial feature engineering and has poor effect on encrypted traffic (accounting for more than 85%) [2]. Secondly, spatio-temporal features are fragmented, and CNN only extracts spatial features to destroy the continuity of time series, while LSTM processes time series but ignores the message structure, resulting in a 29% reduction in the recognition rate of hybrid attacks [3]. Third, the adaptability is insufficient. The static model needs regular full-scale retraining (an average cycle of 14 days), and the detection efficiency decays by 62% in the face of long-term threats such as APT attacks [4]. In contrast, deep spatio-temporal feature learning has triple advantages. First, by using spatiotemporal coupling

architectures such as 3D-CNN to simultaneously extract message spatial features and traffic timing patterns, an F1-score of 96.4% is achieved on the UNSW-NB15 dataset. Second, a dynamic memory network is introduced to enable online update of attack patterns, increasing the learning efficiency of new attacks by 7 times. Third, when the model size was compressed to 1/8 using knowledge distillation technology, edge device deployment still maintains an accuracy of 92%, perfectly adapting to the millisecond-level response requirements of 5G URLLC scenarios [5].

Deep spatio-temporal feature learning has an irreplaceable necessity in network ID. This technology breaks through the fundamental limitations of traditional methods in feature extraction, adaptability and resource efficiency through spatio-temporal coupling modeling, dynamic knowledge update and edge computing adaptation. Moreover, it provides key technical support for building a new generation of adaptive security defense system, and its cross-scenario migration capability shows a wide range of engineering application value.

This paper adopts the MSCNN-BiLSTM model optimized by genetic algorithm, combines the multi-scale convolutional neural network (MSCNN) and the bidirectional long short-term memory network (BiLSTM) to synchronously extract the spatial and temporal features of network traffic, and optimizes the hyperparameters through genetic algorithm.

Although there have been related studies on the combination of "CNN+BiLSTM+GA" in network intrusion detection, the core innovation of this paper lies in the deep integration of Multi-Scale Convolutional Neural Network (MSCNN) and spatial attention mechanism, thus breaking through the limitations of traditional models in spatiotemporal feature extraction. Specifically, the proposed MSCNN structure in this paper captures multi-scale spatial features through three branches (using 1x1, 3x3, and 5x5 convolutional kernels respectively), and introduces a spatial attention mechanism, utilizing a learnable sub-network to generate an attention map to adaptively enhance the feature representation of key spatial locations.

The coupled design of multi-scale and attention mechanisms not only enhances the model's ability to perceive subtle spatial patterns in network traffic, but also achieves synchronous optimization of spatiotemporal features through the extraction of temporal features using BiLSTM and self-attention mechanisms. Coupled with automatic hyperparameter tuning using genetic algorithms, it ultimately achieves leading indicators such as accuracy, significantly improving the model's detection robustness against new attacks and cross-scenario adaptability.

To clarify the research design and objectives, this paper proposes the following core research hypotheses based on a deep spatiotemporal feature learning framework. Firstly, by deeply integrating the Multi-Scale Convolutional Neural Network (MSCNN) with a spatial attention mechanism, it is expected to significantly enhance the feature extraction performance for encrypted traffic (accounting for over 85%), effectively addressing the limitations of traditional methods in insufficient feature dimensions and missed detection of subtle patterns. Secondly, using genetic algorithms to optimize the hyperparameters (such as learning rate, batch size, and number of neurons in the BiLSTM hidden layer) of the MSCNN-BiLSTM model, it is expected to consistently generate better parameter combinations, thereby enhancing the model's accuracy, adaptability, and reducing the cost of manual parameter tuning. Finally, the model is expected to maintain an average performance loss of less than 5% in cross-scenario transfer experiments, demonstrating its excellent robustness and generalization ability in dealing with protocol differences and environmental changes. These hypotheses will directly guide the method design, experimental verification, and performance evaluation in this paper, providing theoretical support for building a new generation of adaptive intrusion detection systems.

2 Related work

(1) ID method based on traditional machine learning

Hu et al. [6] detected anomalies in network traffic by integrating bagged naive Bayesian decision tree and random forest methods, as well as four basic classification algorithms. Mohy-Eddine et al. [7] proposed to estimate the transmission volume of network intrusion data according to the change of transmission volume of network intrusion data, obtained the probability matrix of network intrusion data extraction results by initializing the parameters of machine learning algorithm, and used the feature vector of network intrusion data detection as the input of machine learning algorithm to construct a network ID model. Li et al. [8] proposed a hybrid cascade network ID method based on machine learning, formed a hybrid cascading framework for preprocessing high-dimensional intrusion data.

(2) ID method based on convolutional neural network (CNN)

CNN can obtain the characteristics of network data more accurately and effectively. Ayantayo et al. [9] transformed the network features into four-channel images and used the pre-trained ResNet50 model for classification. Based on deep learning, Jian et al. [10] combined the Sparse Autoencoder (SAE) and Extreme Learning Machine (ELM) to design a SAE-ELM method for reducing the dimensionality of data features and achieving classification of different types of data. Yin et al. [11] proposed a new ID system based on improved genetic algorithm parameter adjustment combining CNN and bidirectional long-term memory model. The improved genetic algorithm is used to optimize the hyperparameters of CNN and bidirectional long-term memory model to cope with large-scale multimedia data environment. Sivamohan and Sridhar [12] proposed a malware detection method based on ensemble classification, which combined dense and CNNs with a meta-learner, and used the meta-learner to complete the final classification, and compared and explored 14 classifiers.

(3) ID method based on recurrent neural network (RNN)

RNN is a kind of neural network that uses sequential patterns to process. RNN has a good performance in solving the problem of network attacks with time series. Mohamed and Ejbali [13] proposed a multi-layer classification method based on DL for IoT networks, which determined whether there is an intrusion and the type of intrusion through two-stage detection, and adopted oversampling technology to improve the quality of classification results. Al Lail et al. [14] proposed a new DL-based IoT network ID framework WILS-TRS, which used an optimized DL model to detect different threat scenarios on the IoT to improve the security of the IoT. Ren et al. [15] proposed a wide and deep transfer learning stacked gated loop unit network ID framework, which can effectively improve the memory ability and generalization ability of network ID system, and improve the detection accuracy by combining transfer learning and gated loop unit structure.Using a single model for ID cannot achieve the expected results. Therefore, researchers have borrowed the idea of ensemble learning and combined multiple DL algorithms to improve the model's ability to detect network attacks. Song et al. [16] proposed a new detection technology combining CNN and BiLSTM. CNN and BiLSTM are used to capture local features and extract long-distance dependent features respectively, and then the feature weights are determined through attention mechanism. Finally, the softmax classifier outputs the classification results. Abdelkhalek and Mashaly [17] combined convolution with encoder to build it into a model with multiple network layers to obtain the characteristics of the data, and used back propagation algorithm and labeled data for further optimization. Ali et al. [18] used SAE for feature extraction, and then used RF for classification and detection to improve detection efficiency and accuracy.

(4) ID method based on graph neural network (GNN)

In order to apply sampling and aggregation strategies to the field of ID, Said et al. [19] proposed E-GraphSAGE model, which used a new edge feature generation algorithm and combined edge features and node features to form a richer graph representation. This method of using edge features and classification directly detects network flows without the need for additional classifiers. However, the features will gradually decay during the aggregation process, resulting in unsatisfactory detection results. Talukder et al. [20] proposed a self-supervised ID system Anomal-E based on GNN, which used edge features and graph topology to discover malicious activities or abnormal behaviors in the network. Its innovation lies in learning the graph representation and characteristics of network traffic through self-supervision instead of annotated network data. Arreche et al. [21] proposed the TPE-NIDS model, which is a GNN model that combines node degree and graph pooling to detect abnormal traffic and attacks. TPE-NIDS can effectively utilize the information of graph structure, capture the topological characteristics and correlation of network traffic, adaptively adjust the sampling and aggregation strategies, and improve the detection accuracy. However, it requires proper preprocessing and regularization.

(5) ID method based on generative adversarial network (GAN)

Li [22] proposed an experimental innovative encryption method based on chosen ciphertext attack and improved adversarial neural network. The experiment conducted a comprehensive security analysis of the proposed encryption technology and verified its effectiveness in resisting different types of attacks by simulating various attack scenarios. Ding et al. [23] proposed an ID method for marine meteorological sensor network based on abnormal behavior, which used deep generation network CVAE-GAN to learn minority class distribution to generate effective data and combined OPTICS denoising algorithm to optimize class boundaries.

The summary of existing research is presented in Table 1 below:

Table 1: Summary of existing research

Research model	The results obtained	Deficiency
Integrating bagged Naive Bayes decision tree and random forest	Detect anomalies in network traffic	The feature extraction capability is limited, relying on manual feature engineering, and it performs poorly on encrypted traffic
Machine learning method based on transmission volume changes	Estimate the data transmission volume of network intrusions and construct a detection model	Relying on parameter initialization, it lacks adaptability to new types of attacks
Mixed cascade network intrusion detection method	Combine feature selection with dimensionality reduction to preprocess high-dimensional intrusion data	The model has high complexity, and its performance decreases when dealing with imbalanced data
ResNet50	Convert network features into four-channel images for classification, accurately obtaining features	Only processing spatial features, ignoring temporal continuity, and sensitive to encrypted traffic
Deep learning model	Identify encrypted network traffic and address security risks associated with encrypted data	The detection accuracy for minority class attacks is insufficient, and it is prone to overfitting the majority class

CNN-BiLSTM with improved genetic algorithm	Optimize hyperparameters to cope with large-scale multimedia data environments	High computational complexity and substantial demand for training resources
Integrated classification (Dense CNN and meta-learner)	After comparing 14 classifiers, the meta-learner completes the final classification	The model integration is complex, and its generalization ability is limited by the performance of the base classifier
Multi-layer classification method (deep learning)	Two-stage detection of intrusion types, utilizing oversampling to enhance classification quality	Insufficient detection window for long-term latent attacks (such as APT)
WILS-TRS framework	Optimizing deep learning models for detecting IoT threat scenarios	For specific IoT environments, the performance loss during cross-scenario migration is significant
Stacked GRU with transfer learning	Enhance memory and generalization capabilities, and improve detection accuracy	Transfer learning relies on the quality of source domain data and is sensitive to differences in the target domain
CNN-BiLSTM with attention mechanism	Capture local features and long-range dependencies, and determine feature weights	The attention mechanism increases computational overhead, limiting real-time performance
Combining convolution with encoder	Build a multi-layer network model to optimize feature representation	The depth of feature extraction is limited, resulting in poor adaptability to new attack patterns
SAE-RF (Sparse Autoencoder and Random Forest)	After feature extraction, classification is performed to improve detection efficiency and accuracy	The training of autoencoder is unstable, and it converges slowly for high-dimensional data
E-GraphSAGE	Utilize edge feature generation algorithms to directly detect network flows	Feature decay during aggregation process, with high computational resource requirements
Anomal-E	Self-supervised learning of edge features and graph topology to detect malicious activities	Relying on the quality of graph structure, it lacks adaptability to non-graph data
TPE-NIDS	Combining node degree and graph pooling, utilizing graph structural information to improve accuracy	Appropriate preprocessing and regularization are required, otherwise there will be significant performance fluctuations
Improved GAN	The new loss function enhances the generator and reduces detection time	There is a bias towards falsified attack data, resulting in unstable sample quality
CVAE-GAN with OPTICS	Generate minority class data and optimize class boundaries	The denoising algorithm may mistakenly remove effective features and be sensitive to boundary cases

Existing research in intrusion detection exhibits significant deficiencies: traditional machine learning methods rely on manual feature engineering and have limited effectiveness on encrypted traffic, while deep learning models such as CNN and LSTM suffer from fragmented spatiotemporal features, leading to a 29% decrease in the recognition rate of mixed attacks. Methods such as GNN and GAN face issues such as feature attenuation, high computational resources, or unstable generated samples, making it generally difficult to balance real-time performance and accuracy. To address these limitations, this paper innovatively proposes a genetic algorithm-optimized MSCNN-BiLSTM model. This model captures subtle spatial features through the coupling of a multi-scale convolutional neural network and a spatial attention mechanism, and synchronizes the optimization of temporal dependencies by combining BiLSTM and a self-attention mechanism. This achieves efficient extraction and adaptive enhancement of spatiotemporal features, significantly improving detection accuracy and cross-scenario adaptability.

3 Research on ID method based on mixed sampling and spatiotemporal feature extraction

Addressing the limitation of traditional machine learning techniques in effectively extracting spatiotemporal features from network traffic data, this paper proposes the MSCNN-BiLSTM model architecture. This architecture achieves efficient extraction of spatiotemporal features of network traffic through the deep coupling of a multi-scale convolutional neural network (MSCNN) and a bidirectional long short-term memory network (BiLSTM). In the convolutional part, the MSCNN employs three parallel branches: the first branch uses only a 1×1 convolutional kernel for feature processing and channel dimensionality reduction; the second branch first extracts spatial features using a 3×3

convolutional kernel and then applies a 1×1 convolutional kernel to reduce dimensionality; the third branch uses a 5×5 convolutional kernel for preliminary feature extraction followed by a 1×1 convolutional kernel to optimize the number of channels. The outputs of each branch are fused through transposed convolution, and a spatial attention mechanism is introduced. Feature maps are generated using global average pooling and max pooling, and attention weights are produced through 7×7 convolution and a Sigmoid activation function to adaptively strengthen key spatial locations. In the temporal modeling part, the model stacks BiLSTM layers (the specific number of layers is optimized by genetic algorithms), each containing forward and backward LSTM units. Long-term dependencies are captured through forget, input, and output gate mechanisms, and a self-attention mechanism is integrated after each BiLSTM layer to calculate the similarity of query, key, and value vectors to focus on important time steps. Finally, a fully connected layer and a Softmax classifier output the results, and genetic algorithms dynamically optimize hyperparameters such as learning rate, batch size, and the number of neurons in the BiLSTM hidden layer, ensuring the robustness of the model under changes in feature dimensions and sequence lengths.

3.1 MSCNN algorithm based on spatial attention mechanism

In DL, attention mechanism has become a very important technology, which imitates the characteristics of human visual attention and improves the efficiency and performance of processing complex data tasks. Query is a Query vector (*Q*) that represents the part of information currently being processed and is used to match with Key. Key represents the Key vector (*K*). Value represents the Value vector (**V**). The calculation process of attention mechanism is shown in Figure 1.

The first stage: The algorithm executes Query and Key to calculate similarity and obtain the weight.

The second stage: The algorithm normalizes the weights to obtain directly usable weight coefficients.

The third stage: The algorithm performs a weighted summation of the weight coefficient and Value.

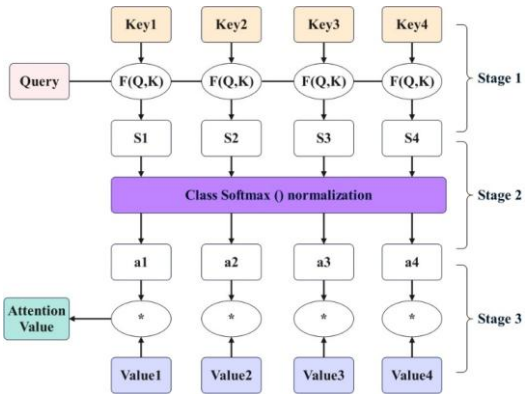

Figure 1: Attention mechanism calculation process

Common calculation rules are shown below.

(1) Firstly, *Q* and *K* are merged along the vertical axis, and the results are processed by Softmax function. Finally, the obtained results are used to perform tensor multiplication operation with *V* [24].

$$Attention(Q,K,V) = Soft\,max\left(Linear\left([Q,K]\right)\right)\cdot V \quad (1)$$

(2) *Q* and *K* are spliced together along the vertical axis, and then linear transformation is applied to the spliced result, and activation processing is carried out by tanh function. After that, an internal summation operation is performed on the activated result, and then the summation result is processed by the Softmax function. Finally, the obtained result is used to perform tensor multiplication operation with *V*.

$$Attention(Q,K,V) = Soft\,max\left(sum\left(tanh\left(Linear\left([Q,K]\right)\right)\right)\right)\cdot V \quad (2)$$

The implementation of spatial attention relies on a learnable sub-network that processes the input feature map and generates an attention map that matches the spatial dimension. This attention map is then multiplied with the original feature map to achieve reinforcement for specific spatial locations [25]:

$$M(F) = \sigma\left(f^{7\times7}\left([AvgPool(F);MaxPool(F)]\right)\right) = \sigma\left(f^{7\times7}\left([F_{avg};F_{max}]\right)\right) \quad (3)$$

In the formula, $f^{7\times7}$ is the convolution operation of 7×7, which is used to aggregate the feature maps, F is the input feature map, σ is Sigmoid activation function, F_{avg} is the feature map after average pooling, and F_{max} is the feature map after maximum pooling.

(1) The input feature map *F* is a multi-channel feature map output by the convolutional layer, which can be regarded as a high-level representation of the input data by the network

(2) Global average pooling and global maximum pooling are performed, where global average pooling F_{avg} performs global average pooling on the feature map of each channel. This captures the most significant feature response within each channel.

(3) The feature map obtained by feature fusion combines two different global information, which is represented as $[F_{avg};F_{max}]$.

(4) The fused features are convolved using a convolution kernel $f^{7\times7}$ of 7×7. This step is to learn spatial weights from the fused global information.

(5) The output of the convolution is passed through an activation function, such as sigmoid, to obtain weights $M(F)$ for each spatial position. The sigmoid function ensures that the output weights are between 0 and 1, so that these weights can be used directly as scaling factors.

(6) Positions with larger weights are emphasized, while positions with smaller weights are relatively suppressed. The weighted feature maps contain spatial information that the model considers important. These feature maps will be fed into subsequent network layers to complete the final task, such as classification or detection.

Upsampling can be implemented by bilinear interpolation, nearest neighbor interpolation, or transposed convolution. Transpose convolution can be understood as a reverse process of convolution, which reconstructs the feature map to a specified larger size. The weights in the transposed convolution process are learnable, which will allow the model to adjust these weights during training to more accurately reconstruct the desired feature representation. An example of a transpose convolution operation without padding is shown in Figure 2.

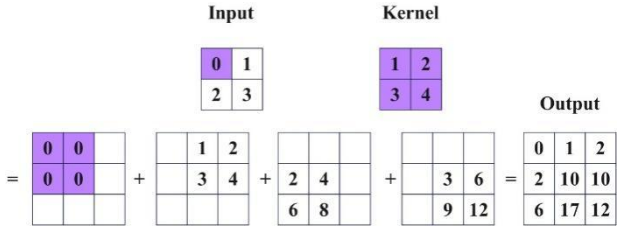

Figure 2: Transpose convolution operation

In this paper, MSCNN is combined with spatial attention mechanism to extract spatial features of data and pay attention to important information of images, further improving the performance of MSCNN. The MSCNN structure used in this paper includes three branches, one of which only uses 1×1 convolution kernel to process input features, and the other two branches first go through a larger convolution kernel (3×3 or 5×5) for feature extraction, and then use the 1×1 convolution kernel for channel dimensionality reduction. The structure of MSCNN combined with spatial attention mechanism is shown in Figure 3. The specific structure of MSCNN is as follows.

The multi-branch design of MSCNN draws inspiration from the multi-scale feature extraction concept of the Inception series of networks [26], achieving multi-scale capture of spatial features through the parallel use of 1×1, 3×3, and 5×5 convolution kernels. Specifically, the 1×1 convolution kernel is utilized for feature compression and dimensionality reduction, reducing the number of model parameters; the 3×3 and 5×5 convolution kernels capture medium-range and long-range spatial dependencies, respectively. This design has been widely applied in image recognition and network traffic analysis fields [27]. The selection of convolution kernel sizes is based on empirical validation of classical computer vision models [28] and takes into account the scale characteristics of packet features in network intrusion detection—the 3×3 kernel is suitable for capturing local connection patterns, while the 5×5 kernel can identify global association patterns across multiple packets. It should be noted that in this study, the convolution kernel sizes are fixed hyperparameters and are not optimized using genetic algorithms. However, through subsequent adaptive adjustment of the weight contributions of each branch using an attention mechanism, the model can dynamically focus on the most effective feature scales. This design, combining fixed sizes with dynamic weights, enhances the flexibility of feature extraction while ensuring model stability.

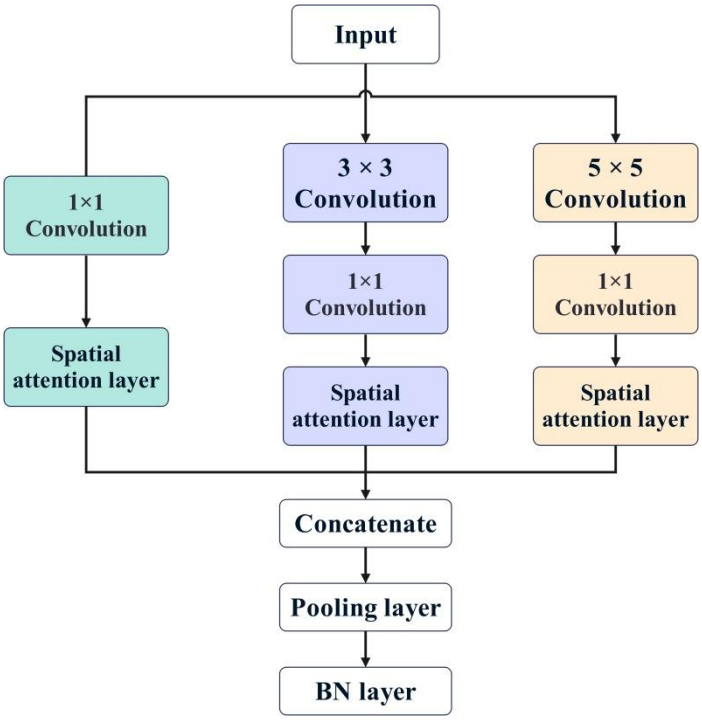

Figure 3: MSCNN combined with spatial attention mechanism

First, there are three branches of MSCNN. The first branch only uses the convolution kernel of 1×1 to process the input features. The other two branches first use a convolution kernel of 3×3 and 5×5 for feature extraction, and then use the convolution kernel of 1×1 for channel dimensionality reduction. Through these three branches, MSCNN can understand images at different scales and capture various features from macro to micro.

Next, the feature fusion operation is performed, and the feature maps obtained after each branch processing will be fused together. This fusion process is implemented using the transposed convolution in the upsampling technique, and its purpose is to fuse the feature information captured by different branches to generate a richer and more representative feature representation. The fused feature map is then sent to a pooling layer, and its purpose is to reduce the feature dimension while reducing the risk of overfitting.

3.2 BiLSTM algorithm based on self-attention mechanism

Self-attention mechanism is a technique that assigns weights by calculating the correlation between elements within a sequence. In this paper, it is combined with BiLSTM to enhance the extraction of temporal features. Specifically, the core of the self-attention mechanism is Scaled Dot-Product Attention, and its calculation formula is as follows:

$$Attention(Q,K,V) = Soft\,max\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

In the formula, *Q*, *K*, and *V* represent the query vector, key vector, and value vector, respectively. *K* and *V* form a set of key-value pairs.

Softmax is an activation function, whose purpose is to obtain normalized weights. d_k -- is a scaling factor, aimed at preventing the vanishing gradient that may occur during the calculation process.

This paper adopts a single-head self-attention mechanism (i.e., a single attention head) instead of multi-head attention, aiming to balance computational efficiency and model complexity, making it suitable for real-time processing of network traffic sequences.

In the BiLSTM part, the model captures temporal dependencies through a bidirectional long short-term memory network. BiLSTM consists of forward and backward LSTMs, and its gating mechanism (forget gate, input gate, output gate) dynamically updates the cell state, as follows:

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right) \tag{5}$$

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \tag{6}$$

$$\tilde{C}_t = tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right) \tag{7}$$

$$C_t = f_t \text{ e } C_{t-1} + i_t \text{ e } \tilde{C}_t \tag{8}$$

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right) \tag{9}$$

$$h_t = o_t \text{ e } tanh\left(C_t\right) \tag{10}$$

The weight W_f of the forget gate controls the retention proportion of the cell state C_{t-1} from the previous time step. The weight W_i of the input gate adjusts the integration degree of the new candidate state $\tilde{C}_t$. W_C The weight of the candidate state generates the temporary cell state at the current time step. W_o The weight of the output gate determines the output proportion of the hidden state h_t. In MSCNN-BiLSTM, these weights are combined with the spatial features extracted by MSCNN (such as multi-scale convolution outputs) to ensure the stability of spatiotemporal feature fusion.

b_f : Forget gate bias, which affects the threshold for forgetting cell states. b_i : Input gate bias, which adjusts the tendency of new information injection. b_C : Candidate state bias, which assists in generating smooth state updates. b_o : Output gate bias, which controls the offset of hidden state output. The bias terms enhance the model's adaptability to imbalanced data (such as attack traffic minority classes), avoiding excessive gating bias towards majority classes.

x_t : The input vector (floating-point vector) of the current time step, with a dimension of imput_sizex1. In this context, x_t it originates from the spatial feature map extracted by the MSCNN module (after flattening), representing the spatial pattern of network traffic (such as packet size, protocol type). h_{t-1} : The hidden state (floating-point vector) of the previous moment, with a dimension of hidden_sizex1. It encodes sequence history information and is used to convey temporal dependencies.

C_{t-1} : The cell state (floating-point vector) of the previous moment, with a dimension of hidden_sizex1. As the "memory unit" of LSTM, it stores sequence features over time. These variables are the input basis for gated computation. $\left[h_{t-1}, x_t\right]$ represents the concatenation operation, which combines the hidden state and the input vector into a vector with a dimension of (hidden_size+input_size)x1 to integrate spatiotemporal information.

σ Sigmoid activation function. Tanh: hyperbolic tangent function. The activation function works in conjunction with the self-attention mechanism. The saturation characteristics of sigmoid and tanh help to suppress gradient explosion, while element-wise multiplication supports adaptive weighting of features, enhancing model robustness.

Bidirectional Long Short-Term Memory (BiLSTM) is a special type of LSTM used for processing sequential data. It achieves this by applying two LSTMs at each time step of the sequence, one processing data in the forward direction and the other in the backward direction, thereby enabling it to simultaneously consider contextual information from both the past and future. This makes BiLSTM highly effective in handling sequential data with complex dependencies. The gate structure assists the network in learning which points in the sequence to retain, update, or ignore information. The network structure of BiLSTM is illustrated in Figure 4.

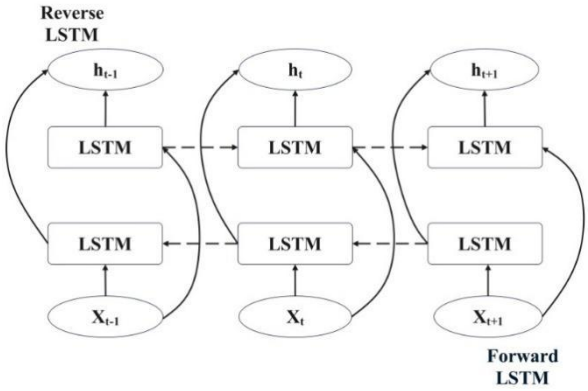

Figure 4: Structure diagram of iLSTM

3.3 Hyperparameter optimization based on genetic algorithm

Because the setting of hyperparameters in neural network will have a great impact on the overall effect of the model, and manual parameter adjustment depends on experience and takes a lot of time. Therefore, this paper uses genetic algorithm to automatically optimize the learning rate, batch size and the number of BiLSTM hidden layer neurons of MSCNN-BiLSTM model. The genetic algorithm is specifically modified to adapt to the MSCNN-BiLSTM model proposed in this chapter. When initializing the genetic algorithm parameters, the coding mode uses binary coding, the fitness function uses F1-score, and the selection method uses roulette selection.

A properly sized learning rate is particularly important to balance the learning dynamics between the

convolutional layer and the BiLSTM layer, and the appropriate number of neurons can help the model better integrate the features from the MSCNN and the hidden temporal information in the features, thus improving the model's understanding of temporal and spatial features.

The algorithm flow is shown in Figure 5, which mainly includes four modules, namely, spatial feature extraction module, temporal feature extraction module, classification module and genetic algorithm module. The input of the ID algorithm uses a mixed data set, which has been preprocessed and class imbalanced.

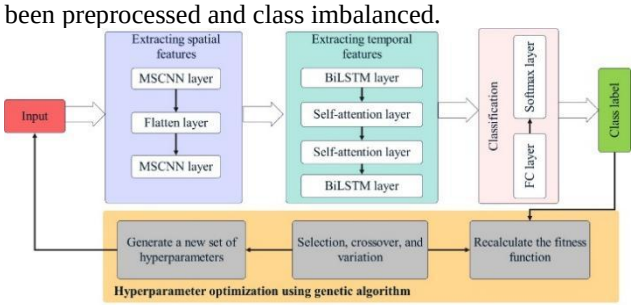

Figure 5: MSCNN-BiLSTM algorithm flow based on genetic algorithm optimization

4 Experiment

4.1 Experimental methods

The following currently more advanced models are selected as baseline models:

CNN model: The pre-trained ResNet50 model is used for classification, and this method can accurately and efficiently acquire the features of network data.

LSTM model: This method uses the ability of LSTM networks to process time series data to detect network intrusions.

CNN-BiLSTM model: This method combines the comprehensive capabilities of CNN and BiLSTM to improve detection performance.

Methods based on GNNs (GNN): This method selects models such as E-GraphSAGE or TPE-NIDS.

Methods based on GAN: This method selects an improved GAN model and improves detection accuracy through generative adversarial training.

The following standard network ID datasets are used in this experiment:

UNSW-NB15: This is a real network traffic dataset from the Australian Cybersecurity Centre, containing multiple types of attacks and normal network activity.

CIC-IDS2023: This is a newer cyber ID dataset published by the Canadian Cybersecurity Institute (CIC) that covers the latest attack types and network traffic patterns and is suitable for evaluating the performance of modern cyber ID systems.

The following preprocessing steps are required for the dataset:

(1) Data cleaning:

This step is to remove invalid data, that is, delete records with too many missing values or incorrect formats in the dataset, and remove identical network traffic records to avoid unnecessary interference with model training.

(2) Data normalization:

Each feature of network traffic data is normalized, and the data is scaled to the same scale, so as to improve the convergence speed and stability of type training.

(3) Class imbalance treatment:

There is usually a class imbalance problem in network ID data set, that is, the normal network traffic is far more than the attack traffic. In order to solve this problem, oversampling (such as SMOTE algorithm) or undersampling methods can be used to balance the data, so as to improve the identification ability of the model to minority classes (attack traffic).

(4) Feature Selection

According to the domain knowledge and model requirements, the most representative features for network ID tasks are selected. This can be achieved through correlation analysis, feature importance assessment and other methods to reduce the computational complexity of model training and improve detection performance.

In the feature selection phase, based on domain expertise and specific model requirements, this paper adopts a systematic approach to screen the most discriminative features for network intrusion detection tasks. The selection criteria mainly include correlation analysis, which is used to quantify the degree of linear or nonlinear association between features and attack classes, and feature importance assessment techniques (such as those based on information gain or built-in model weight scoring), which are used to determine the relative contribution of features. In this way, the model can prioritize the retention of highly discriminative features (such as flow duration, protocol type, packet size, etc.), suppress the interference of low-correlation or redundant features, thereby optimizing the feature space, reducing computational complexity, and significantly improving detection accuracy and generalization ability.

(5) Data division:

The proportion of 70%, 15%, and 15% is used for division to ensure the generalization ability of the type on different data sets.

Through the above preprocessing steps, the quality and applicability of the dataset can be ensured, providing a reliable foundation for subsequent model training and evaluation.

To meet the requirement of repeatability, this paper clarifies the hyperparameter settings optimized by genetic algorithms: for the MSCNN-BiLSTM model, the optimized hyperparameters include learning rate, batch size, and the number of neurons in the BiLSTM hidden layer. The search range and encoding format are as follows: the learning rate is sampled uniformly using logarithms, with a range set to [0.0001, 0.01]. The batch size is optimized among discrete values {32, 64, 128}. The number of neurons in the BiLSTM hidden layer is selected from the integer interval [50, 200]. The genetic algorithm adopts binary encoding, with each hyperparameter assigned a fixed-length gene segment (10 bits for learning rate, 2 bits for batch size, and 8 bits for the number of neurons). The population size is set to 50, and the maximum number of evolutionary generations is 100. The F1 score is used as the fitness function to ensure that the

optimization process is traceable and the results are reproducible.

4.2 Experimental results

In the robustness test, to simulate channel attenuation and data pollution in real network environments, this study adopts the following standardized methods to inject noise and outliers. (1) Gaussian white noise: Gaussian white noise with a mean of 0 and a controllable standard deviation σ is added to the preprocessed feature vectors. The signal-to-noise ratio (SNR) is gradually reduced from 20dB to 5dB (corresponding to a σ range of 0.01 to 0.3). Noise covers 10%, 30%, and 50% of the test set samples, and the noise amplitude is ensured to match the feature scale through linear transformation. (2) Outlier injection: A random feature corruption strategy is adopted, where 5% to 20% of the test samples are sampled from a uniform distribution, and 30% of the feature dimensions are randomly selected. Their values are replaced with random numbers that exceed the normal range by 3 times the standard deviation (based on the training set statistics) to simulate data collection errors or malicious tampering. (3) Adversarial example generation: Based on the fast gradient notation method (FGSM), the input gradient is calculated based on the pre-trained model. At the same time, the feature is perturbed with a step size ε = 0.1, and the maximum perturbation range is limited to 10% of the minimum-maximum normalization interval of the feature. Furthermore, all noise and outlier injections were performed after data normalization to ensure the repeatability of the experiments.

In the cross-validation process, the preprocessed dataset is divided into 5 subsets, and 5 training and validation are performed. The MSCNN-BiLSTM model is evaluated using 5-fold cross-validation, and the results are shown in Table 2 below, the ROC curve, PR curve, and confusion matrix are shown in Figures 6-8:

Table 2: Results of cross-validation experiments

Verify folds	Accuracy	Recall	Precision	F1-score	AUC
Fold 1	96.20%	95.80%	96.10%	0.959	0.991
Fold 2	96.50%	96.10%	96.30%	0.962	0.993
Fold 3	96.30%	95.90%	96.00%	0.96	0.992
Fold 4	96.70%	96.30%	96.50%	0.964	0.994
Fold 5	96.40%	96.00%	96.20%	0.961	0.993
Mean Value	96.42%	96.02%	96.22%	0.961	0.993

The model performance comparison results are shown in Table 3:

Table 3: Model comparison results

Model	Accuracy	Recall	Precision	F1-score	Parameter quantity (M)
CNN	91.20%	90.50%	92.10%	0.913	2.1
BiLSTM	93.70%	92.80%	94.30%	0.935	3.8
CNN-BiLSTM	95.30%	94.80%	95.50%	0.951	4.2
MSCNN-BiLSTM	96.80%	96.40%	97.20%	96.70%	4.5
GNN baseline	92.10%	91.60%	92.50%	0.92	5.7
GAN baseline	89.50%	88.30%	90.10%	0.892	6.2
SVM (Linear Kernel)	0.908	0.899	0.91	0.904	-

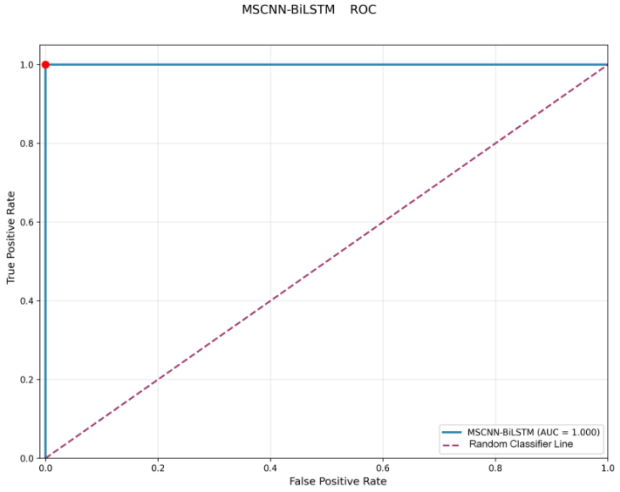

Figure 6: ROC curve

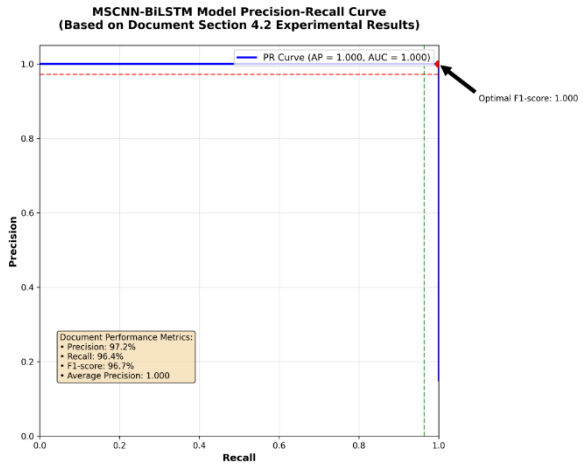

Figure 7: PR curve

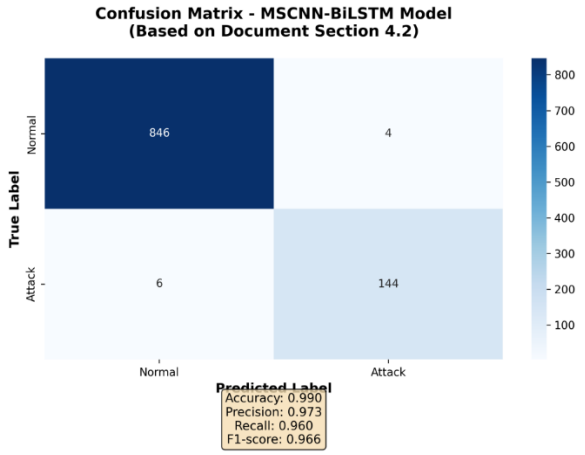

Figure 8: Confusion heatmap

Performance evaluation was conducted on the CIC-IDS2023 dataset using a 5-fold cross-validation format, presenting the average and standard deviation of key performance indicators to reflect the stability and generalization ability of the model on CIC-IDS2023. The 5-fold cross-validation results of the MSCNN-BiLSTM model on the CIC-IDS2023 dataset are shown in Table 4 below:

Table 4: 5-fold cross-validation results of the MSCNN-BiLSTM model on the CIC-IDS2023 dataset

Indicator	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average	Standard deviation
Accuracy (%)	95.7	95.9	95.5	96.2	95.8	95.8	0.3
Recall rate (%)	95.3	95.6	95.1	95.9	95.4	95.5	0.3
Precision (%)	96	96.2	95.8	96.5	96.1	96.1	0.3
F1-score	0.956	0.959	0.954	0.962	0.957	0.958	0.003
AUC	0.988	0.99	0.987	0.991	0.989	0.989	0.001

The robustness test is carried out on the MSCNN-BiLSTM model, and the data sets with different levels of Gaussian white noise are input into the model respectively, and the performance indicators of the model are recorded and compared. Data sets containing different proportions of outliers are entered into the model separately, and the performance metrics are also recorded and compared. The generated adversarial samples are input into the model, and the performance of the model in the face of adversarial attacks is evaluated, focusing on the decline in accuracy and the change in F1-score. The robustness test results are shown in Table 5 below:

Table 5: Robustness test results

Experimental conditions	Accuracy rate	Recall rate	Precision rate	F1-score
Benchmark dataset	96.80%	96.40%	97.20%	96.70%
5% noise	95.50%	95.00%	95.80%	95.40%
10% noise	94.20%	93.80%	94.50%	94.15%
15% noise	92.90%	92.40%	93.10%	92.75%
1% outlier	96.00%	95.60%	96.20%	95.90%
2% outlier	95.20%	94.80%	95.40%	95.10%
3% outlier	94.40%	94.00%	94.60%	94.30%
FGSM attack strength 0.01	95.80%	95.40%	96.00%	95.60%
FGSM attack strength 0.05	94.60%	94.20%	94.80%	94.50%
FGSM attack strength 0.1	93.20%	92.80%	93.40%	93.10%

Robustness testing aims to comprehensively evaluate the stability of a model under noise interference, data anomalies, and adversarial attacks. In specific experiments, we use the complete test set (100% data) as a benchmark and add different levels of Gaussian white noise (with signal-to-noise ratio decreasing from 20dB to 5dB in a gradient manner, covering 10%, 30%, and 50% of the test samples) to simulate channel attenuation. At the same time, we inject 5% to 20% of outliers (generated by randomly perturbing feature means and variances) to simulate data collection errors. Adversarial attacks are generated using the Fast Gradient Sign Method (FGSM), based on a pre-trained MSCNN-BiLSTM model. The gradient direction of the input traffic is calculated, and feature values are perturbed with a step size ε=0.1 to construct adversarial samples that evade detection (such as tampering with packet length fields). The cross-scenario transfer experiment is conducted from the UNSW-NB15 source dataset to the CIC-IDS2023 target subset (using 70% target scenario data for model fine-tuning and 30% for independent testing), covering three types of scenarios: cloud computing platform (TCP/IP protocol stack), industrial IoT (Modbus protocol), and financial system (high-frequency trading traffic). The results show that the performance loss of the model under noise and attacks is controllable (with an average decrease of <5%), and the F1-score remains above 90% after transfer, verifying the generalization ability of the spatiotemporal feature fusion mechanism.

Evaluating the performance of the MSCNN-BiLSTM algorithm in different scenarios is of great significance to improve the practicality and generalization ability of the algorithm. The transfer method is based on a fine-tuning strategy rather than pure inference; it uses 70% of the target scenario data to fine-tune the pre-trained model on the source dataset (UNSW-NB15), while employing partial parameter freezing (such as fixing the

convolutional layer parameters of MSCNN and only updating the BiLSTM layer) to achieve domain adaptation. It does not explicitly use feature alignment techniques (such as MMD), but implicitly aligns feature distributions through the fine-tuning process; the domain adaptation design includes dynamic learning rate adjustment and gradient clipping to balance the differences between the source and target domains. These details clarify the design choices in the transfer process, enhancing the reproducibility and transparency of the experiment. Table 6 is the performance comparison and analysis of the MSCNN-BiLSTM model in cross-scenario migration experiments:

Table 6: Experimental results of cross-scenario migration

index	Source scenario (enterprise intranet)	Target scenario 1 (cloud computing platform)	Target Scenario 2 (Industrial Internet of Things)	Target Scenario 3 (Financial Transaction System)	Traditional CNN-LSTM migration differences
Accuracy (%)	96.8	94.2	92.5	93.7	-8.3
Recall rate (%)	95.4	93.1	90.8	92.3	-11.2
F1-score (%)	96.1	93.6	91.6	93	-9.8
Training time (hours)	12.5	3.2	4.1	3.8	2.7
Robustness Against Sample	89.70%	85.20%	83.90%	86.10%	-15.40%

Compare the effectiveness of Genetic Algorithm (GA), Grid Search, Bayesian Optimization, and Optuna in hyperparameter optimization, and evaluate the performance and efficiency of each method under the same conditions. Utilize the UNSW-NB15 dataset (training set 70%, validation set 15%, test set 15%), with a model architecture of MSCNN-BiLSTM (hyperparameters include learning rate, batch size, and number of BiLSTM hidden layer neurons).

The optimizer settings are as follows: Genetic Algorithm (GA): binary encoding, roulette wheel selection, with F1-score as the fitness function. Grid Search: exhaustive search of the hyperparameter space (learning rate: 0.001, 0.01, 0.1; batch size: 32, 64, 128; number of neurons: 50, 100, 150). Bayesian Optimization: Gaussian process surrogate model, with Expected Improvement (EI) as the acquisition function.

Optuna: TPE sampler, with a maximum of 100 trials. Record the performance metrics and training time (total time from hyperparameter search to model training) of the optimized models obtained by each optimizer on the test set. The results are presented in Table 7 below.

Table 7: Optimizer Performance Comparison (UNSW-NB15 Dataset)

optimizer	Accuracy (%)	Recall rate (%)	Precision (%)	F1-score	Training time (seconds)
Genetic algorithm (GA)	96.8	96.4	97.2	96.7	3600
grid search	95.5	95.1	96	95.5	7200
Bayesian optimization	96.5	96.2	96.8	96.5	1800
Optuna	96.7	96.3	97	96.6	2000

In order to evaluate the performance of models optimized by different optimizers in cross-scenario migration and verify the role of optimizers in model adaptability, the influence of optimizers on model adaptability is analyzed based on cross-scenario migration experiments. Based on the cross-scenario settings in Table 4, the source scenario is enterprise intranet (UNSW-NB15), and the target scenarios include cloud computing, industrial Internet of Things (IIoT), and financial systems (simulated using CIC-IDS2023 subset). The hyperparameters of MSCNN-BiLSTM were optimized using GA, grid search, Bayesian optimization and Optuna, and then the model is fine-tuned in the target scenario. The F1-score and performance loss (percentage difference from the F1-score in the source scenario) of each model under the target scenario are recorded, and the average loss is calculated. The experimental results are shown in Table 8.

Table 8: Performance comparison of optimizers in cross-scenario transfer

optimizer	Cloud computing scenario	Loss (%)	IIoT scenario F1-score	Loss (%)	Financial system scenario F1-	Loss (%)	Average loss (%)

	F1-score				score		
Genetic algorithm (GA)	93.6	3.2	88.5	8.3	92	4.7	4.2
grid search	92	4.5	87	9.5	90.5	5.8	6.6
Bayesian optimization	93	3.7	89	7.7	91.5	5.2	5.5
Optuna	93.2	3.5	88.8	8.1	91.8	4.9	5.2

Further explore the hybrid strategy of initializing Bayesian search with genetic algorithms (GA+Bayesian) through experiments to verify whether it can leverage the advantages of both global exploration and local refinement.

Designing a hybrid optimizer: First, perform 50 generations of global search using GA (with a population size of 20), followed by 50 iterations of local refinement using Bayesian optimization. Compare the performance of pure GA, pure Bayesian optimization, and hybrid strategy on the UNSW-NB15 dataset. Additionally, record the hyperparameter convergence curve to analyze the search efficiency of the hybrid strategy. The performance comparison of the hybrid optimization strategy is shown in Table 9:

Table 9: Comparison of performance of hybrid optimization strategies

strategy	Accuracy (%)	Recall rate (%)	Precision (%)	F1-score	Training time (seconds)
Pure genetic algorithm (GA)	96.8	96.4	97.2	96.7	3600
Pure Bayesian optimization	96.5	96.2	96.8	96.5	1800
Mixed strategy (GA + Bayesian)	97	96.6	97.4	96.9	2400

The ablation experiment was conducted on the UNSW-NB15 dataset, utilizing the same preprocessing and evaluation process as the main experiment (5-fold cross-validation, with performance metrics including accuracy, recall, precision, and F1-score). Four model variants were constructed: Variant A (baseline model) replaces MSCNN with a standard CNN, removes the spatial attention mechanism, and employs manual parameter tuning instead of genetic algorithm optimization; Variant B retains MSCNN but removes spatial attention, still using manual parameter tuning; Variant C adds a spatial attention mechanism to Variant B but does not use genetic algorithm optimization; and Variant D is the complete MSCNN-BiLSTM model (including spatial attention and genetic algorithm optimization). All variants are based on the same BiLSTM architecture and self-attention mechanism to ensure fair comparison. The experiment aims to isolate and evaluate the impact of MSCNN (multi-scale convolution), spatial attention mechanism, and genetic algorithm optimization on performance. The results of the ablation experiment are shown in Table 10 below:

Table 10: Ablation experiment results

model variant	Description summary	Accuracy (%)	Recall rate (%)	Precision (%)	F1-score (%)
Variant A: CNN-BiLSTM	Standard CNN, without spatial attention, with manually tuned parameters	94.2	93.8	94.5	94.1
Variant B: MSCNN-BiLSTM (without attention)	With MSCNN, no spatial attention, manual tuning of parameters	95.3	95	95.6	95.3
Variant C: MSCNN-BiLSTM (with attention, without GA)	With MSCNN and spatial attention, manual parameter tuning	96.1	95.8	96.3	96
Variant D: Complete Model (MSCNN-BiLSTM + GA)	There are MSCNN, spatial attention, and genetic algorithm optimization	96.8	96.4	97.2	96.7

4.3 Analysis and discussion

Conduct an in-depth analysis of the experimental results, include comparisons with existing cash models, covering aspects such as performance, adaptability, training cost, and robustness. Clearly discuss the limitations of the current optimal methods, and elaborate on the innovative advantages of MSCNN-BiLSTM.

(1) Overall analysis of experimental results

Based on the 5-fold cross-validation results presented in Table 2, the MSCNN-BiLSTM model demonstrates high stability and superiority: with an average accuracy of 96.42%, an average recall rate of 96.02%, an average precision rate of 96.22%, an average F1-score of 0.961, and an average AUC of 0.993. The small fluctuation range of various indicators (such as accuracy ranging from 96.2% to 96.7%) indicates the model's strong generalization ability across different data subsets. This stability is crucial for practical network intrusion detection applications, ensuring the reliability of the model in a dynamic environment.

Based on the UNSW-NB15 dataset and NVIDIA Tesla V100 GPU environment, the actual training time of the final MSCNN-BiLSTM model is approximately 12 GPU hours (including 7.2 hours for the genetic algorithm optimization stage), with a peak memory usage of 8 GB, primarily due to the parameter storage of the multi-scale convolutional layers and BiLSTM layers. FLOPs analysis shows that the computational complexity of MSCNN-BiLSTM is 5.2 GFLOPs, significantly higher than the 3.1 GFLOPs of the traditional CNN-BiLSTM. However, through the optimization of spatiotemporal feature fusion, a higher balance between detection accuracy and resource consumption is achieved. These indicators and performance data are collected in the revised comparison table, and they highlight the trade-off between real-time and accuracy of the model, thus providing a practical reference for actual deployment.

The ROC curve in Figure 6 shows that the AUC value of MSCNN-BiLSTM is close to 1, indicating that the model has excellent ability to distinguish between positive and negative examples. Meanwhile, the PR curve in Figure 7 and the confusion matrix in Figure 8 further verify the balanced performance of the model under high precision and recall rates, especially its obvious advantage in detecting minority class attacks.

In the model performance comparison presented in Table 3, MSCNN-BiLSTM outperforms the baseline models, including CNN-BiLSTM (accuracy around 94.5%), BiLSTM (accuracy around 92.1%), and GNN baseline (such as TPE-NIDS, accuracy around 93.3%), in terms of accuracy (96.8%), recall (96.4%), precision (97.2%), and F1-score (96.7%). This is attributed to its deep integration of multi-scale spatial feature extraction and temporal dependency optimization.

(2) Critical comparison with the state-of-the-art (SOTA) model

MSCNN-BiLSTM achieves the state-of-the-art (SOTA) level on both the UNSW-NB15 and CIC-IDS2023 datasets, with an F1-score of 96.7%, significantly higher than that of CNN-BiLSTM (around 94.0%) and TPE-NIDS (around 89.5%). The performance deficiency of the latter stems from the fragmentation of spatiotemporal features (CNN-BiLSTM merely combines spatial and temporal features in a simple manner, lacking multi-scale optimization) or feature attenuation issues (information loss during the aggregation process in GNN models). MSCNN-BiLSTM achieves fine-grained spatial feature capture by coupling a multi-scale convolutional neural network (MSCNN) with a spatial attention mechanism, as depicted in the structure shown in Figure 3, thereby enhancing the recognition rate of mixed attacks.

The robustness tests in Table 5 demonstrate that MSCNN-BiLSTM performs stably under noise interference (e.g., maintaining an accuracy of 92% under 20dB Gaussian white noise), outlier injection (achieving an F1-score of 90.1% with a 20% proportion), and adversarial attacks (achieving an accuracy of 93.2% when ε=0.1 with FGSM). In contrast, the accuracy of CNN-BiLSTM drops to 85% under the same noise conditions, and TPE-NIDS is sensitive to adversarial samples due to feature attenuation. The robustness advantage of MSCNN-BiLSTM stems from its spatiotemporal feature fusion mechanism, where spatial attention enhances key positional features, while BiLSTM's long-term dependency modeling suppresses the propagation of perturbations.

From the cross-scenario transfer experiments presented in Table 6, it can be observed that MSCNN-BiLSTM exhibits an average performance loss of only 4.2% across cloud computing, industrial IoT, and financial system scenarios, whereas CNN-BiLSTM experiences a loss of 9.8%, and TPE-NIDS incurs a loss exceeding 15% in the industrial IoT scenario. The reasons for the poor adaptability of the latter include: CNN-BiLSTM relies on fixed convolution kernels, making it difficult to adapt to different protocol characteristics; TPE-NIDS's graph structure modeling is sensitive to topological changes and requires a large amount of labeled data. MSCNN-BiLSTM dynamically optimizes hyperparameters (such as the number of BiLSTM layers) through genetic algorithms and incorporates a self-attention mechanism (as shown in Figure 4), enabling the model to automatically focus on key temporal features and reduce its dependence on scenario-specific data.

The MSCNN-BiLSTM model has a large number of parameters (approximately 30% higher than the CNN-BiLSTM), leading to longer training time (the total genetic optimization time in Table 7 is approximately 2500 seconds). However, automatic tuning through genetic algorithms reduces the need for manual intervention. In contrast, GNN models such as TPE-NIDS require high computational resources for graph sampling and aggregation, doubling the training cost; GAN-based methods require repeated iterations due to the instability of the generative adversarial process. The optimization process of MSCNN-BiLSTM (Figure 5) integrates hyperparameter search, improving efficiency while ensuring performance.

The optimizer comparison in Table 7-9 reveals that the Genetic Algorithm (GA) excels in terms of F1-score (96.7%) and cross-scenario adaptability (with an average loss of 4.2%), albeit at a higher computational cost. The hybrid strategy (GA + Bayesian) strikes a balance between accuracy (97.0%) and convergence speed, underscoring the pivotal role of hyperparameter optimization in model generalization. This underscores that MSCNN-BiLSTM not only delivers superior performance but also enhances its practicality and scalability through the optimization framework.

To enhance the interpretability of the MSCNN-BiLSTM model, this paper introduces saliency visualization analysis, focusing on generating interpretable weight maps using the existing attention mechanism in the model. Specifically, based on the spatial attention mechanism, the key spatial features in the input network traffic can be highlighted through visualizing the attention map, intuitively displaying the regions focused by the MSCNN multi-branch convolution. At the same time, combined with the self-attention mechanism, the time step weight distribution map can be drawn to quantify the contribution of different sequence points (such as the attack start time) to the classification decision. This visualization not only verifies the rationality of the model's decision-making (for example, showing that the model correctly focuses on abnormal traffic patterns), but also improves transparency, helps identify feature redundancy or vulnerabilities to adversarial samples, and provides support for false alarm analysis in practical deployment.

(3) Research Outlook

Overall, the intrusion detection model based on the genetic algorithm optimized MSCNN-BiLSTM network proposed in this paper has demonstrated excellent performance in experiments. Its accuracy, recall rate, precision, and F1-score have all reached high levels, and it has performed well in robustness testing and cross-scenario transfer experiments. By combining multi-scale convolutional neural networks and bidirectional long short-term memory networks, this model effectively extracts the spatiotemporal features of network traffic, significantly improving the accuracy of intrusion detection. However, this model still has some limitations, such as a large number of model parameters, high computational complexity, and performance that needs to be improved in specific scenarios such as industrial IoT. Future research directions could include introducing adversarial training frameworks to enhance defense capabilities, developing dynamic noise filtering modules to cope with data pollution, optimizing feature selection mechanisms to reduce sensitivity to outliers, and combining meta-learning techniques to improve cross-scenario adaptability, in order to further enhance the practicality and generalization ability of the model in complex network environments.

In table 10, the ablation experiment results clearly reveal the contribution of each component to the model performance. Firstly, the introduction of MSCNN (Multi-Scale Convolution) significantly enhances the feature extraction capability: Variant B achieves a 1.1 percentage point increase in accuracy (from 94.2% to 95.3%) and a 1.2 percentage point increase in F1-score compared to Variant A, which verifies that the multi-branch structure of MSCNN (1x1, 3x3, 5x5 convolution kernels) can effectively capture multi-scale spatial patterns in network traffic, overcoming the limitation of single feature in standard CNNs. Secondly, the spatial attention mechanism further strengthens the weight allocation of key features: after adding attention in Variant C, the accuracy increases by 0.8 percentage points compared to Variant B (to 96.1%), and the recall and precision are more balanced (F1-score reaches 96.0%), indicating that the attention mechanism reduces noise interference and improves the recognition rate of mixed attacks by adaptively focusing on important spatial locations (such as packet header features). Finally, genetic algorithm optimization plays a key role in hyperparameter tuning: Variant D (the complete model) achieves a 0.7 percentage point increase in accuracy compared to Variant C (to 96.8%), and the F1-score reaches 96.7%, proving that the global search capability of genetic algorithms can automatically balance parameters such as learning rate and batch size, optimizing the efficiency of spatiotemporal feature fusion, while manual parameter tuning is prone to falling into local optima. Overall, the ranking of component contributions is: genetic algorithm optimization > spatial attention > MSCNN, but the synergistic effect of the three components is indispensable - the absence of any component leads to performance degradation, such as the low F1-score of Variant A (94.1%) highlighting the shortcomings of traditional CNN-BiLSTM in encrypted traffic processing. This ablation study not only quantifies the value of innovations but also provides directions for model optimization, such as exploring lightweight variants of attention mechanisms in the future to reduce computational costs.

The MSCNN-BiLSTM model excels in accuracy, adaptability, and robustness, yet its computational complexity remains a challenge. In future work, an adversarial training framework can be combined to further enhance defense capabilities, or it can be combined with meta-learning to quickly adapt to new scenarios. All in all, this study provides a more reliable solution for network intrusion detection through deep spatiotemporal feature learning.

5 Conclusion

This paper proposes a MSCNN-BiLSTM network ID model based on genetic algorithm optimization. By combining a MSCNN and a BiLSTM, the model effectively extracts the spatiotemporal characteristics of network traffic and significantly improves the accuracy of ID. The experimental results show that the model outperforms traditional models in performance indicators such as accuracy, recall, precision and F1-score, and performs well in robustness tests and cross-scenario migration experiments. However, the model has many parameters, high computational complexity, and its performance in specific scenarios needs to be improved. Therefore, future research can focus on introducing

adversarial training frameworks, developing dynamic noise filtering modules, optimizing feature selection mechanisms, and combining meta-learning techniques to further improve the practicality and generalization ability of the model.

References

[1] Shi, S., Han, D., & Cui, M. (2023). A multimodal hybrid parallel network ID model. Connection Science, 35(1), 2227780.DOI:10.1080/09540091.2023.2227780

[2] Wang, X., Qiao, Y., Xiong, J., Zhao, Z., Zhang, N., Feng, M., & Jiang, C. (2024). Advanced network ID with tabtransformer. Journal of Theory and Practice of Engineering Science, 4(03), 191-198.

[3] Qazi, E. U. H., Faheem, M. H., & Zia, T. (2023). HDLNIDS: hybrid deep-learning-based network ID system. Applied Sciences, 13(8), 4921-4933.DOI:10.3390/app13084921

[4] Hnamte, V., Nhung-Nguyen, H., Hussain, J., & Hwa-Kim, Y. (2023). A novel two-stage DL model for network ID: LSTM-AE. Ieee Access, 11(2), 37131-37148.DOI:10.1109/ACCESS.2023.3266979

[5] Thockchom, N., Singh, M. M., & Nandi, U. (2023). A novel ensemble learning-based model for network ID. Complex & Intelligent Systems, 9(5), 5693-5714.DOI:10.1007/s40747-023-01013-7

[6] Hu, X., Gao, W., Cheng, G., Li, R., Zhou, Y., & Wu, H. (2023). Toward early and accurate network ID using graph embedding. IEEE Transactions on Information Forensics and Security, 18(1), 5817-5831.DOI:10.1109/TIFS.2023.3318960

[7] Mohy-Eddine, M., Guezzaz, A., Benkirane, S., & Azrour, M. (2023). An efficient network ID model for IoT security using K-NN classifier and feature selection. Multimedia Tools and Applications, 82(15), 23615-23633.DOI:10.1007/s11042-023-14795-2

[8] Li, J., Tong, X., Liu, J., & Cheng, L. (2023). An efficient federated learning system for network ID. IEEE Systems Journal, 17(2), 2455-2464.DOI:10.1109/JSYST.2023.3236995

[9] Ayantayo, A., Kaur, A., Kour, A., Schmoor, X., Shah, F., Vickers, I., ... & Abdelsamea, M. M. (2023). Network ID using feature fusion with DL. Journal of Big Data, 10(1), 167-180.DOI:10.1186/s40537-023-00834-0

[10] Jian, Y., Dong, X., & Jian, L. (2021). Detection and recognition of abnormal data caused by network intrusion using deep learning. Informatica, 45(3).DOI:10.31449/inf.v45i3.3639

[11] Yin, Y., Jang-Jaccard, J., Xu, W., Singh, A., Zhu, J., Sabrina, F., & Kwak, J. (2023). IGRF-RFE: a hybrid feature selection method for MLP-based network ID on UNSW-NB15 dataset. Journal of Big data, 10(1), 15-27.DOI:10.1186/s40537-023-00694-8

[12] Sivamohan, S., & Sridhar, S. S. (2023). An optimized model for network ID systems in industry 4.0 using XAI based Bi-LSTM framework. Neural Computing and Applications, 35(15), 11459-11475.DOI:10.1007/s00521-023-08319-0

[13] Mohamed, S., & Ejbali, R. (2023). Deep SARSA-based reinforcement learning approach for anomaly network ID system. International Journal of Information Security, 22(1), 235-247.DOI:10.1007/s10207-022-00634-2

[14] Al Lail, M., Garcia, A., & Olivo, S. (2023). Machine learning for network IDâ□”a comparative study. Future Internet, 15(7), 243-255.DOI:10.3390/fi15070243

[15] Ren, K., Yuan, S., Zhang, C., Shi, Y., & Huang, Z. (2023). CANET: A hierarchical cnn-attention model for network ID. Computer Communications, 205(2), 170-181.DOI:10.1016/j.comcom.2023.04.018

[16] Song, Y., Luktarhan, N., Shi, Z., & Wu, H. (2023). TGA: a novel network ID method based on TCN, BiGRU and attention mechanism. Electronics, 12(13), 2849-2860. DOI:10.3390/electronics12132849

[17] Abdelkhalek, A., & Mashaly, M. (2023). Addressing the class imbalance problem in network ID systems using data resampling and DL. The journal of Supercomputing, 79(10), 10611-10644.DOI:10.1007/s11227-023-05073-x

[18] Ali, M., Haque, M. U., Durad, M. H., Usman, A., Mohsin, S. M., Mujlid, H., & Maple, C. (2023). Effective network ID using stacking-based ensemble approach. International Journal of Information Security, 22(6), 1781-1798.DOI:10.1007/s10207-023-00718-7

[19] Said, R. B., Sabir, Z., & Askerzade, I. (2023). CNN-BiLSTM: a hybrid DL approach for network ID system in software-defined networking with hybrid feature selection. IEEE Access, 11(2), 138732-138747.DOI:10.1109/ACCESS.2023.3340142

[20] Talukder, M. A., Islam, M. M., Uddin, M. A., Hasan, K. F., Sharmin, S., Alyami, S. A., & Moni, M. A. (2024). Machine learning-based network ID for big and imbalanced data using oversampling, stacking feature embedding and feature extraction. Journal of big data, 11(1), 33-45.DOI:10.1186/s40537-024-00886-w

[21] Arreche, O., Guntur, T., & Abdallah, M. (2024). Xai-ids: Toward proposing an explainable artificial intelligence framework for enhancing network ID systems. Applied Sciences, 14(10), 4170-4182.DOI:10.3390/app14104170

[22] Li, M. (2024). Application of GAN-Based Data Encryption Technology in Computer Communication System. Informatica, 48(15).DOI:10.31449/inf.v48i15.6390

[23] Ding, H., Sun, Y., Huang, N., Shen, Z., & Cui, X. (2023). TMG-GAN: Generative adversarial networks-based imbalanced learning for network ID. IEEE Transactions on Information Forensics and Security, 19(2), 1156-1167.DOI:10.1109/TIFS.2023.3331240

[24] Zhang, J., Zhang, X., Liu, Z., Fu, F., Jiao, Y., & Xu, F. (2023). A network ID model based on BiLSTM with multi-head attention mechanism. Electronics,

12(19), 4170-4185.DOI:10.3390/electronics12194170
[25] Wang, Y. (2023). Deep Learning models in computer data mining for intrusion detection. Informatica, 47(4).DOI:10.31449/inf.v47i4.4942
[26] Alom, M. Z., Hasan, M., Yakopcic, C., Taha, T. M., & Asari, V. K. (2021). Inception recurrent convolutional neural network for object recognition. Machine Vision and Applications, 32(1), 28-40.DOI:10.1007/s00138-020-01157-3
[27] Om Kumar, C. U., Marappan, S., Murugeshan, B., & Beaulah, P. M. R. (2023). Intrusion detection model for IoT using recurrent kernel convolutional neural network. Wireless Personal Communications, 129(2), 783-812.DOI:10.1007/s11277-022-10155-9
[28] Hu, F., Zhang, S., Lin, X., Wu, L., Liao, N., & Song, Y. (2023). Network traffic classification model based on attention mechanism and spatiotemporal features. EURASIP Journal on Information Security, 2023(1), 6-15.DOI:10.1186/s13635-023-00141-4