Journal of Cyber Security

Tech Science Press

Doi:10.32604/jcs.2025.066089

ARTICLE

Deep Learning Network Intrusion Detection Based on MI-XGBoost Feature Selection

Manzheng Yuan[1,2] **and Kai Yang**[2,*]

[1]School of Computer, Xijing University, Xi'an, 710123, China

[2]Xi'an Key Laboratory of Human-Machine Integration and Control Technology for Intelligent Rehabilitation, School of Computer Science, Xijing University, Xi'an, 710123, China

*Corresponding Author: Kai Yang. Email: 2308540402014@stu.xijing.edu.cn

Received: 29 March 2025; Accepted: 11 June 2025; Published: 07 July 2025

ABSTRACT: Currently, network intrusion detection systems (NIDS) face significant challenges in feature redundancy and high computational complexity, which hinder the improvement of detection performance and significantly reduce operational efficiency. To address these issues, this paper proposes an innovative weighted feature selection method combining mutual information and Extreme Gradient Boosting (XGBoost). This method aims to leverage their strengths to identify crucial feature subsets for intrusion detection accurately. Specifically, it first calculates the mutual information scores between features and target variables to evaluate individual discriminatory capabilities of features and uses XGBoost to obtain feature importance scores reflecting their comprehensive contributions in complex data relationships. Then, through adaptive weighted combination of the two types of scores to generate integrated feature weights, we can select the most valuable features for intrusion detection based on these weights. Furthermore, a deep learning detection model combining Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory Networks (BiLSTM) is developed based on the selected features. This model takes advantage of CNN's capability in extracting local feature associations and BiLSTM's strength in capturing long-term dependencies in sequential data, further enhancing the accuracy and efficiency of intrusion detection. Experiments have shown that on the NSL-KDD and UNSW-NB15 datasets, the accuracies reach 99.35% and 88.78%, respectively, demonstrating that this method significantly improves the accuracy and efficiency of network intrusion detection. This research provides new ideas for feature selection and model construction of network intrusion detection systems. It can improve the detection performance while reducing the computational overhead, and helps optimize the practical application effect of the IDS, having certain technical reference value for enhancing the intelligent level of network intrusion detection.

KEYWORDS: Network intrusion detection; feature selection; mutual information; extreme gradient boosting; deep learning

1 Introduction

With the rapid development of information technology and the widespread use of the internet, cybersecurity issues have become increasingly severe. Network attack methods are becoming more sophisticated and diverse. From traditional Denial of Service (DoS) attacks to Advanced Persistent Threats (APTs), the demand for a robust cybersecurity defense system has increased significantly. In this context, Network Intrusion Detection Systems (IDS) have emerged as a critical bulwark for cybersecurity, capable of real-time detection of anomalous network behaviors and early warning of potential threats, thereby becoming a vital technological means to ensure network security.

The high-dimensional features and complex temporal correlations in network traffic data lead to significant performance limitations for conventional machine learning models during training. Feature selection, a key data preprocessing step, has undergone several technological evolutions. Filter methods like mutual information (MI) use information-theoretic measures to assess feature relevance to target variables, with millisecond-level computational efficiency, yet they overlook feature interactions. Embedded methods typified by XGBoost evaluate feature importance via gradient-boosting mechanisms, capturing nonlinear feature relationships, but their computational complexity grows quadratically with the number of features. Despite differing in efficiency and accuracy, neither filter nor embedded methods alone can balance computational resource use and model generalization. The introduction of deep learning offers new ways to model complex patterns in network traffic data. CNNs excel at feature extraction with local receptive fields and weight sharing, making them suitable for locally correlated network traffic data. LSTMs and their variants like BiLSTMs effectively model temporal dependencies in time-series data through their gating mechanisms, which alleviate the vanishing gradient problem. However, existing studies mostly focus on single-model optimization or simple integration, without systematically addressing the challenge of joint spatiotemporal feature modeling in high-dimensional network traffic data.

In response to the aforementioned issues, this paper introduces a deep learning-based network intrusion detection method. It combines mutual information with XGBoost-weighted feature selection. The goal is to use this combination to identify highly discriminative feature subsets. Additionally, a deep learning model integrating CNN and BiLSTM is designed. This model efficiently captures the spatial and temporal features of network traffic data, enhancing the accuracy and robustness of intrusion detection systems. As a result, it offers a more reliable security solution for complex and dynamic network environments. The main contributions of this paper are as follows:

1. A feature selection method combining mutual information and XGBoost weighting is proposed. This approach evaluates the correlation between features and the target variable using mutual information and integrates the importance scores from XGBoost. A weighted strategy is employed to generate comprehensive weights, enabling the selection of the most representative feature subset. It effectively solves the traditional problem of balancing efficiency and accuracy in feature selection.
2. A deep-learning model integrating CNN and BiLSTM has been designed. CNN can capture local feature patterns and spatial correlations, while BiLSTM can model temporal dependencies in time-series data. By integrating the two, it effectively models complex network traffic data across spatial and temporal dimensions, leveraging their complementary strengths to address the gap in existing research on joint spatiotemporal feature modeling.
3. The method's performance was tested on public datasets like NSL-KDD and UNSW-NB15. It outperformed existing main methods in detection accuracy, recall, and F1-score, proving effective for network intrusion detection and offering a new way to boost cybersecurity protection.

The subsequent sections will provide an overview of the existing work, our proposed model, and the results. First, the current research status in the field is presented in Section 2. The theoretical foundation of the research is described in Section 3. Then, the datasets and experimental procedures of this paper are elaborated in Section 4, and the experimental conclusions, results analysis, and summary are presented in Section 5. Finally, the conclusions, limitations, and future research directions of this study are discussed in Section 6.

2 Current Research Status at Home and Abroad

As network environments evolve, traffic data becomes high-dimensional, complex, and dynamic. This raises higher requirements for feature processing and modeling in intrusion detection systems. Effective

feature selection can reduce redundancy, boost detection efficiency, and enhance model generalization. Meanwhile, deep learning, with its powerful automatic feature extraction and spatiotemporal modeling, offers new ideas for identifying covert attacks. Given network traffic data's characteristics and intrusion detection tasks' needs, feature selection and deep learning models are two current research focal points. Next, we'll separately review the research status of these two directions.

2.1 Current Research Status of Feature Selection Methods

Feature selection is a critical step in network intrusion detection, aiming to identify the most relevant features from high-dimensional network traffic data, reduce data redundancy, and enhance the efficiency and performance of detection models.

Feature selection methods based on filtering techniques evaluate features independently using statistical or information-theoretic metrics. These methods calculate the association between each feature and the target variable to identify a subset of features with strong relevance. Mutual Information (MI) is a classical filtering method that assesses the importance of features by measuring the degree of mutual dependence between the feature and the target variable. Recent studies have shown that MI performs exceptionally well in handling network traffic features with nonlinear relationships. For instance, Guilherme Nunes Nasseh Barbosa et al. [1] proposed a novel approach based on the Pareto dominance set, which optimized feature selection by minimizing both linear correlation and mutual information. This method significantly reduces the number of features while maintaining over 95% accuracy in network attack classification. However, the limitation of the mutual information method lies in its inability to capture interactions between features, potentially overlooking the importance of combined features.

Embedding-based feature selection methods integrate the feature selection process into model training, where feature importance is determined by learning model parameters. As an efficient gradient boosting method, XGBoost leverages gain and frequency to measure feature importance. In recent years, XGBoost has been widely applied in intrusion detection. For instance, Chen et al. [2] proposed a feature selection method combining XGBoost with Spearman's correlation coefficient. This combination, known as XGB-S, effectively identifies and selects the most influential features for classification tasks, filtering out redundant and irrelevant features to construct an optimal feature subset. This subset provides a more refined and effective input for subsequent model training and classification. Compared to filter methods, embedding methods can capture interactions between features but are computationally more expensive and more susceptible to noise in the data.

Thus, this paper proposes a mutual-information-based XGBoost feature selection method. MI-XGBoost uses mutual information to measure the dependency between features and target variables, enabling a more accurate selection of highly relevant features. Unlike linear correlation metrics, mutual information captures nonlinear relationships between features and targets, leading to more comprehensive and precise feature selection. This allows XGBoost to focus on the most informative features, improving model performance and efficiency. Additionally, XGBoost performs well on large-scale datasets. It can process data in parallel, resulting in faster training speeds. By first selecting features with mutual information, XGBoost can be trained on a smaller feature set, further enhancing training efficiency. However, compared with the proposed MI-XGBoost method, the traditional XGBoost feature selection method has its own limitations, which will be addressed by the combination of mutual information in our method.

2.2 Current Research Status of Deep Learning Models in Intrusion Detection

For the research on deep learning-based network intrusion detection techniques, Udurume et al. [3] compared and analyzed the application of the deep learning model CNN-BiLSTM with traditional machine

learning models in intrusion detection systems. Their study revealed the advantages of deep learning in terms of detection accuracy and emphasized the practical potential of lightweight machine learning models in resource-constrained environments. Additionally, they explored the trade-offs between model accuracy, computational complexity, and real-time performance. Zhang et al. [4] proposed a network intrusion detection model that integrates multi-head attention mechanisms and bidirectional long short-term memory networks (BiLSTM) to address the issue of relatively low practical detection accuracy in detection models. The model strengthens the relationship between features and attack types by assigning different attention weights to each vector in the feature vectors and leverages BiLSTM to capture long-distance dependencies to improve detection accuracy. Additionally, the dropout layers incorporated in the model help prevent overfitting, thereby further enhancing detection accuracy. Marc et al. employed four machine learning classifiers—logistic regression, support vector machines, multilayer perceptrons, and random forests—to train and evaluate synthetic network datasets generated by generative adversarial networks (CTGAN) and variational autoencoders (TVAE), as well as real network datasets. They assessed the statistical similarity of the synthetic data and its impact on classifier performance. Their findings revealed that at least 15% of real data is necessary to maintain classifier performance and avoid degradation in network intrusion detection tasks. Xu et al. [5] proposed a hybrid network intrusion detection method based on CNN-BLSTM-XGBoost, which comprehensively extracts spatial and temporal features of network intrusion data and utilizes XGBoost for efficient and accurate classification detection, significantly improving multi-class detection accuracy. However, the method has shortcomings in identifying small-sample attack types, handling data imbalance, and the depth of feature fusion. Particularly, the single XGBoost model in the classifier design is insufficient to fully explore the complex patterns of fused features.

This study focuses on optimizing the CNN-BiLSTM model and its feature selection method. Unlike previous research, this study proposes a feature selection method that combines mutual information with XGBoost weighting. It uses mutual information to evaluate feature relevance to target variables and integrates XGBoost's importance scores. A weighting strategy generates comprehensive weights to select the most representative feature subsets. In model design, we create a deep-learning model that combines CNN and BiLSTM. CNN extracts local feature patterns, while BiLSTM captures time-series dependencies, enabling efficient modeling of complex network traffic data. Furthermore, we validate our method on multiple public datasets (NSL-KDD, UNSW-NB15). Experimental results show that our approach outperforms existing mainstream methods in detection accuracy, recall, and F1 score. In summary, our study improves feature selection and model optimization, showing advantages in handling imbalanced data and recognizing small-sample attack types. This enhances the application value of the CNN-BiLSTM model in cybersecurity.

2.3 Frontier Exploration and Research Dilemmas of Feature Selection and Model Fusion

Recently, Bouke et al. [6] proposed a novel feature selection method based on central tendency (CTFS). This method significantly improves the execution efficiency and achieves a remarkable reduction in feature selection time. It takes approximately 0.1 s for the UNSWNB15 dataset and about 0.026 s for the NSLKDD dataset. However, the accuracy only reaches 95%. Moreover, since the CTFS method conducts feature selection based on central tendency, it may lack an intuitive interpretation of the feature selection results. There may be some selected features that are statistically significant, but their actual association with network attacks may not be clear, which may affect the understanding and application of the results by security experts. Logeswari et al. [7] proposed an improved collaborative dual-layer feature selection algorithm. They adopted the Synergistic Dual-Layer Feature Selection (SDFC) algorithm, combining statistical methods (such as mutual information and variance thresholding) with advanced machine learning model-based techniques, along with Recursive Feature Elimination (RFE) and Particle Swarm Optimization (PSO) to identify the

most relevant features. They used two-level classifiers, namely LightGBM and XGBoost, to efficiently classify network traffic as normal or malicious. However, this study only conducted experiments using the TON-IoT dataset [8]. Although it is a commonly used IoT security dataset, it may not fully cover all types of IoT network environments and attack scenarios. Other datasets may contain different attack types, device types, or network topologies, which may lead to the Intrusion Detection System (IDS) performing worse on other datasets than on the TON-IoT dataset. Moreover, this IDS uses two-level classifiers (LightGBM and XGBoost), but this design may be overly dependent on the output results of the first-level classifier. If the performance of the first-level classifier is not good, it may hurt the performance of the entire system. Sunil Kaushik et al. proposed a feature selection algorithm based on fundamental statistical methods. They applied basic statistical methods, namely the Chi-square test (Chi2) and the Pearson correlation coefficient, to extract the top 5, 10, and 20 features. Then, the vectors using these features were applied to traditional machine learning algorithms. This new feature selection technique helps improve the accuracy and other performance indicators of low-performance classifiers. However, this method is a traditional statistical approach and may not be able to fully explore the complex nonlinear relationships in the data. For some highly complex datasets, relying solely on these statistical methods may not be sufficient to extract the most representative features. Moreover, only traditional machine learning algorithms were used in the model selection, and deep learning methods were not attempted. The above-mentioned research has promoted the development of network intrusion detection technology from different perspectives, but it still faces three common challenges: firstly, there is insufficient integration between feature selection and deep learning models, and traditional methods are likely to lead to a mismatch between the feature subset and the model's requirements; secondly, the complexity of multi-model fusion is high, resulting in a large demand for computing resources, and the cost of hyperparameter tuning is high; thirdly, the cross-dataset generalization ability is weak, making it difficult to adapt to the dynamic and changeable network attack environment.

In response to the above problems, this paper proposes a CNN-BiLSTM network intrusion detection method that combines mutual information and XGBoost weighted feature selection. By constructing a dynamic weighted feature selection mechanism, it uses mutual information to quantify the correlation between features and attack labels and combines the feature importance evaluation of XGBoost to generate a feature subset that is highly suitable for the CNN-BiLSTM architecture, thus solving the problem of disconnection between feature selection and the model. At the same time, the CNN-BiLSTM network structure is optimized, and parameter sharing and inter-layer optimization strategies are applied to reduce the model complexity while retaining the ability to extract temporal and spatial features and reduce the cost of hyperparameter tuning. In addition, through cross-validation on various types of public datasets such as UNSW-NB15 and NSL-KDD, the adaptability of the model to different network environments and attack patterns is effectively improved. Experimental results show that this method significantly outperforms existing solutions on the NSL-KDD and UNSW-NB15 datasets, providing an efficient and robust solution for the practical application of network intrusion detection technology.

3 Relevant Theoretical Foundations

3.1 Fundamentals of Deep Learning

3.1.1 The Concept and Characteristics of Deep Learning

Deep Learning is an advanced machine learning method based on artificial neural networks, aimed at automatically extracting high-level features from large datasets through multi-layered network structures, thereby achieving complex pattern recognition and decision-making tasks [9]. Unlike traditional shallow learning models, the key advantage of deep learning lies in its powerful feature learning capability. By constructing deep neural networks that contain multiple hidden layers, the model can extract low-level

features of the data layer by layer, gradually combining these features into higher-order, more abstract representations. This hierarchical feature representation enables deep learning to achieve outstanding performance in complex tasks such as image classification, speech recognition, and natural language processing [10]. In recent years, the widespread application and success of deep learning can be attributed primarily to the development of big data, the enhancement of high-performance computing capabilities, and advancements in optimization algorithms.

The diverse and expressive features of deep learning make it a central area of research in current artificial intelligence studies. The following are the main characteristics of deep learning:

(1) Multi-layer Structure: Deep learning models consist of multiple layers, including an input layer, hidden layers, and an output layer. Each layer progressively extracts features from the data through complex nonlinear transformations. The greater the number of hidden layers, the deeper the model and the stronger its learning capability. This multi-layer structure enables the model to better fit complex functional relationships, thereby enhancing its expressiveness and generalization ability.

(2) Automatic Feature Extraction: Traditional machine learning algorithms rely on manually designed features, whereas deep learning can automatically extract data features through multi-layer networks. This not only reduces the dependence on domain knowledge but also enhances the model's applicability and flexibility, allowing it to be widely used across different fields and tasks.

(3) Powerful Representational Capability: Deep learning models can extract multi-level feature representations from data [11]. This powerful representational capability enables them to excel in fields such as image recognition, speech processing, and natural language processing [12]. Deep networks can abstract data features layer by layer, thereby achieving precise capture of complex patterns.

(4) Big Data Demand: The effectiveness of deep learning largely depends on the support of large-scale datasets. Big data not only aids in model training but also significantly enhances the model's generalization ability, making it more robust and accurate in practical applications.

(5) High Computational Demands: Training deep learning models typically requires substantial computational resources, particularly when dealing with high-dimensional data and deep networks. High-performance computing devices, such as GPUs and TPUs, can significantly enhance the speed and efficiency of model training, making them indispensable tools in deep learning.

(6) End-to-end Learning: Deep learning supports end-to-end learning, where the entire process from raw input to final output is directly optimized by the model. This integrated learning approach streamlines the system construction process and enhances the model's overall performance through global optimization.

(7) Model Maturity: Deep learning encompasses numerous mature model architectures, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Generative Adversarial Networks (GANs). Each of these models possesses unique characteristics, making them suitable for various types of tasks, including image processing, sequence data analysis, and data generation.

(8) Gradient Descent Optimization: The training of deep learning models typically relies on gradient descent and its variant optimization algorithms. These algorithms calculate the gradient of the loss function and update the model parameters, gradually approaching the optimal solution to ensure model convergence and performance improvement.

(9) Transfer Learning: Deep learning supports transfer learning by pre-training models on large-scale datasets and subsequently fine-tuning them for related tasks, significantly reducing the training data requirements and time for new tasks. This capability enables deep learning to quickly adapt to diverse tasks, which is a key factor in its widespread application in the industry [13,14].

(10) Wide Application: The powerful performance of deep learning has led to its extensive use across multiple fields, such as autonomous driving, intelligent healthcare, voice assistants, and image recognition, driving technological advancement and innovative development in various industries.
(11) Through these characteristics, deep learning not only excels in theoretical research but also demonstrates immense potential in practical applications, becoming the core driving force of current artificial intelligence technology.

3.1.2 Commonly Used Deep Learning Models

(1) CNN

CNN is a deep learning model adept at extracting local features. It captures local patterns in feature matrices through convolutional kernels and reduces data dimensionality and computational complexity using pooling operations [15]. In network intrusion detection tasks, CNN can extract spatial features from network traffic data, such as specific patterns in protocol fields or data distributions, providing strong support for detecting specific types of attacks [16]. Due to its parameter sharing and efficient computation capabilities, CNN performs exceptionally well in handling high-dimensional data [17,18].

(2) BiLSTM

Bidirectional Long Short-Term Memory Networks (BiLSTM) are a type of recurrent neural network capable of modeling bidirectional dependencies in sequential data. By employing forward and backward LSTM units, BiLSTM captures contextual information within sequence data, making it particularly suitable for dynamic network traffic data. In network intrusion detection, BiLSTM can extract pattern features that evolve from traffic data, demonstrating significant advantages in detecting complex, multi-step dependent attack behaviors.

The combination of CNN and BiLSTM can fully leverage the complementary advantages of both. The CNN is responsible for extracting local spatial features of network traffic, generating high-dimensional feature representations for the data. BiLSTM further models these features temporally, capturing the dynamic correlations of traffic over time [19]. The combined model, which integrates both local patterns and temporal dependencies, provides a more comprehensive and accurate feature representation for network intrusion detection, making it suitable for addressing complex and diverse attack patterns.

3.2 Weighted Feature Selection Using MI-XGBoost

3.2.1 Principles of the XGBoost Algorithm

XGBoost [20] is an efficient gradient-boosting algorithm that can enhance model performance by constructing multiple decision trees through ensemble learning. In feature selection, the embedded nature of XGBoost allows it to directly extract feature importance scores from the training process. Feature importance is typically represented by the following two metrics:

(1) Gain: Represents the contribution of a feature to the optimization of the objective function during each split;
(2) Split Count: Indicates the number of times a specific feature is selected as the splitting point.

By training an XGBoost model, a matrix of feature importance scores can be obtained.

$$S_{xgb} = [s_1, s_2, \ldots, s_n] \tag{1}$$

Among them, S_i represents the importance score of the i-th feature. Compared to mutual information, XGBoost is capable of capturing complex nonlinear relationships between features, but its computational

complexity is higher and it may be influenced by the size of the dataset and noise. Fig. 1 illustrates the feature selection process of XGBoost.

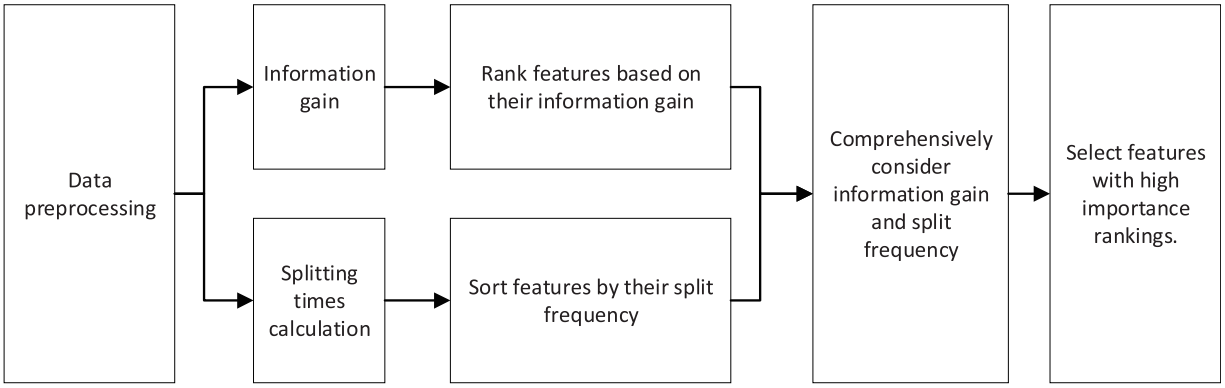

Figure 1: XGBoost feature selection flowchart

3.2.2 Principle of Mutual Information

Mutual Information (MI) is a feature selection method based on information theory, used to measure the dependency between two random variables [21]. Its core idea is to evaluate the importance of features by calculating the mutual dependence between the features and the target labels. The formula for calculating mutual information is as follows:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)} \tag{2}$$

Among them, X represents the feature variables, and Y represents the target labels; $p(x, y)$ denotes the joint probability distribution of the feature value X and the label value Y; $p(x)$ and $p(y)$ represent the marginal probability distributions of X and Y, respectively.

Higher mutual information scores indicate a stronger correlation between the feature and the target variable. By calculating the mutual information value for each feature, a ranking of feature importance can be obtained. However, the mutual information method only considers the relationship between individual features and the target variable, ignoring interactions between features, which may lead to certain limitations in the screening results.

3.2.3 MI-XGBoost Weighted Feature Selection Method

To address the shortcomings of these two methods, this paper combines mutual information scores with XGBoost scores through a weighting strategy. A tunable parameter is introduced during the weighting process to balance the influence of both methods, with its value determined based on performance metrics from the validation set [22]. When the parameter value favors mutual information, feature selection becomes more efficient; conversely, when it leans towards XGBoost, the nonlinear relationships among features are better represented. After weighting, the combined scores are ranked from highest to lowest, and the top-ranked features are selected as the final feature subset. During the feature selection process, the number of desired features can be further adjusted to balance computational efficiency and model performance. The advantage of this method lies in its ability to integrate the strengths of both feature selection mechanisms

while offering significant flexibility and adaptability, allowing for parameter adjustments to meet the needs of different datasets and detection tasks.

4 Deep Learning Network Intrusion Detection Based on MI-XGBoost Weighted Feature Selection

This paper presents a deep learning network intrusion detection method based on MI-XGBoost weighted feature selection, aimed at addressing the issues of feature redundancy and high computational complexity in network intrusion detection systems. By combining mutual information (MI) with XGBoost for weighted feature selection, the method effectively reduces redundant features while retaining the most discriminative key features. Building on this foundation, a deep learning model that integrates Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory networks (BiLSTM) is employed to fully exploit the spatial and temporal characteristics of the data, thereby enhancing the accuracy and efficiency of intrusion detection. Ultimately, this approach significantly improves the performance of network intrusion detection by optimizing feature selection and deep learning models, reducing computational complexity, and providing a more efficient and reliable solution. Fig. 2 illustrates the overall framework of this method.

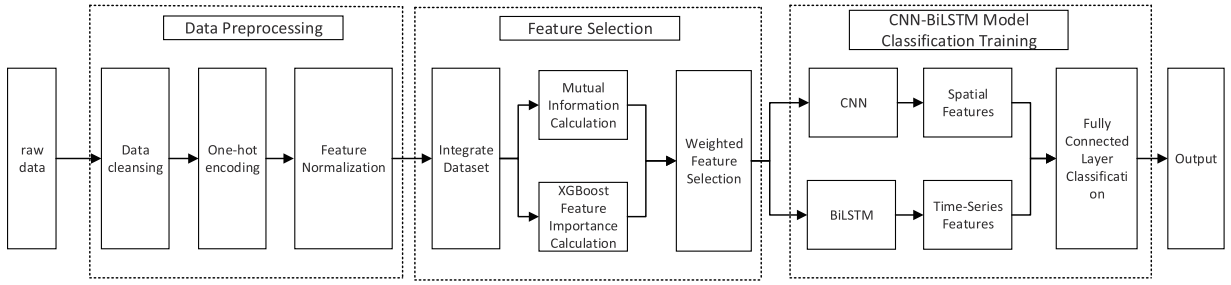

Figure 2: Deep learning network intrusion detection using MI-XGBoost and CNN-BiLSTM

4.1 Data Preprocessing

4.1.1 Data Cleaning

In the first step of feature processing, this study conducted data loading and cleaning on the NSL-KDD and UNSW-NB15 datasets to ensure data integrity and validity, providing high-quality input for subsequent modeling. The original NSL-KDD dataset is already divided into the training set KDDTrain+.txt and the testing set KDDTest+.txt, which contains 41 feature fields and 1 label field. After loading the data, explicit column names were assigned to each column to enhance readability, and the "difficulty_level" field, which was irrelevant to the task, was removed. Additionally, unnecessary invalid records were eliminated by checking data consistency. The UNSW-NB15 dataset comprises 2,540,000 network traffic records and 42 features, and it is divided into the training set UNSW_NB15_training-set.csv and the testing set UNSW_NB15_testing-set.csv. During the data cleaning process, the id column was removed as it served as a unique identifier and did not participate in model training. Finally, the training and testing datasets were merged into a complete dataset to facilitate unified feature processing in subsequent steps.

4.1.2 One-Hot Encoding

In data preprocessing, one-hot encoding is an important step for the numerical representation of categorical features in the dataset. This study applied one-hot encoding to the categorical features such as protocol_type, service, and flag in the NSL-KDD dataset. The variable "protocol_type" indicates the type of

network protocol (e.g., TCP, UDP, ICMP), "service" represents the type of target service (e.g., HTTP, FTP), and "flag" denotes the connection status (e.g., SF, REJ). Through one-hot encoding, each categorical value is transformed into a binary feature vector, where only one position is set to 1 and all other positions are set to 0. For instance, the values TCP, UDP, and ICMP in "protocol_type" are expanded into separate feature columns. After encoding, the feature dimensions of the dataset expanded from 41 to 121 dimensions, with the attributes protocol_type, service, and flag being expanded to 3, 70, and 11 feature columns, respectively. During the one-hot encoding process of the UNSW-NB15 dataset, three categorical features—proto, state, and service—were primarily addressed. First, the proto, state, and service columns were transformed into a new binary feature column. For the proto column, encoding resulted in the generation of three new feature columns, each representing different types of network protocols. The state column, which contains various connection states, was expanded into multiple feature columns after encoding. The service column, due to its greater variety of service types, was expanded into a large number of new feature columns. After one hot encoding, the three columns in the original data were replaced with multiple binary feature columns, significantly increasing the feature dimension of the data. To ensure that the feature dimensions of the training set and the test set are consistent, both sets were first merged and then one-hot encoded together. Subsequently, they were split back into the training and test sets, ensuring consistency in dimensionality. This approach not only transformed the categorical information but also provided numerical features suitable for processing by machine learning algorithms for subsequent modeling and training.

4.1.3 Feature Normalization

Feature normalization is a crucial step in data preprocessing, aimed at scaling numerical features to a uniform range. This process reduces the scale differences between various features, thereby enhancing the stability of model training and accelerating convergence speed.

In this study, the NSL-KDD and UNSW-NB15 datasets contain a large number of numerical features, such as src_bytes (source bytes) and dst_bytes (destination bytes). The range of values for these features varies significantly, which may lead to the model assigning excessive weight to features with larger value ranges during the training process, thereby affecting detection performance. To address this, Min-Max normalization is applied to all numerical features of the two datasets, using the following formula:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

In this context, X represents the original value of the feature, while $X_{\min}$ and $X_{\max}$ denote the minimum and maximum values of that feature, respectively. The normalized feature value is indicated, and after normalization, the range of all feature values is compressed to the interval $[0, 1]$, thereby reducing the impact of scale differences among features on model training. In the implementation process, the minimum and maximum values of each feature column are calculated individually, followed by normalization according to the aforementioned formula. This feature normalization not only enhances the model's ability to learn from different features but also effectively reduces the negative impact of inconsistent feature value ranges on model optimization, providing more balanced input data for subsequent feature extraction and classification.

In this study, the attack types in the NSL-KDD dataset were remapped into five major categories: Normal, DoS, Probe, U2R, and R2L, to facilitate multi-class classification tasks. The UNSW-NB15 dataset, on the other hand, contains nine types of attacks, with the original labels retained and annotated according to the attack categories. During the data preprocessing phase, the NSL-KDD and UNSW-NB15 datasets were divided into training and testing sets according to the official splitting standards. Subsequently, all categorical labels were converted into a numerical format using LabelEncoder to facilitate model processing.

Finally, the data was transformed into tensor format and loaded in batches using PyTorch's TensorDataset and DataLoader modules, ensuring the efficiency of the training process. These data processing steps provided a solid foundation for the subsequent construction and training of the model.

4.2 Feature Selection

This study employs a feature selection method that combines mutual information (MI) with importance weighting from XGBoost. First, MI assesses the correlation between each feature and the target variable by calculating the mutual information score. Subsequently, XGBoost computes the importance score for each feature through model training.

Standardization of mutual information (MI) scores and XGBoost feature importance scores is performed. The purpose of standardization is to convert two scores with different dimensions into a unified scale, facilitating subsequent weighted fusion. For each feature X_i, the standardization formulas for the mutual information score $MI(X_i, Y)$ and the XGBoost score $I_{XGBoost}(X_i)$ are as follows:

$$MI^*(X_i) = \frac{MI(X_i, Y) - \min(MI)}{\max(MI) - \min(MI)}$$

$$I^*_{XGBoost}(X_i) = \frac{I_{XGBoost}(X_i) - \min(I_{XGBoost})}{\max(I_{XGBoost}) - \min(I_{XGBoost})}$$

Among them, $\min(MI)$ and $\max(MI)$ represent the minimum and maximum values of the mutual information of all features, respectively; $\min(I_{XGBoost})$ and $\max(I_{XGBoost})$ denote the minimum and maximum values of the XGBoost feature importance scores. The standardized scores $MI^*(X_i)$ and $I^*_{XGBoost}(X_i)$ both range from 0 to 1, ensuring that they can be integrated on the same scale.

A weighted fusion approach is employed to combine the normalized mutual information scores and the XGBoost importance scores into a comprehensive score. The fusion method is as follows:

$$S(X_i) = \alpha \cdot MI^*(X_i) + (1 - \alpha) \cdot I^*_{XGBoost}(X_i)$$

In this context, $S(X_i)$ represents the comprehensive score of feature X_i, where α is the weight coefficient that controls the relative importance of the mutual information score and the XGBoost score during the fusion process. By adjusting α, the contributions of both can be flexibly modified for different tasks or datasets.

Finally, by calculating the comprehensive scores of all features, the features can be ranked according to their scores. Ultimately, the top K features with the highest scores are selected as the final subset of features.

$$\{X_{i1}, X_{i2}, \ldots, X_{ik}\} = \arg\max_{X_i}(S(X_1), S(X_2), \ldots, S(X_n))$$

This study selects the top 40 features as the final subset through comprehensive consideration. Firstly, these features rank highly in overall scores, indicating strong relevance to the target variable and high importance in model training, thus suggesting significant predictive power for intrusion detection. Secondly, retaining a moderate number of features balances model performance and computational complexity. While more features might enhance performance, they also increase complexity and resource use. The top 40 features capture the most critical information, maintaining high detection performance while reducing computations and boosting efficiency. Thirdly, fewer features simplify the model, reducing overfitting risk and enhancing generalization. Lastly, in practical intrusion detection, quick responses and low

resource consumption are crucial. The top 40 features meet these needs without compromising detection performance. This selection is a rational decision that weighs feature relevance, model performance, computational complexity, and practical application, enhancing model efficiency and usability while preserving detection effectiveness.

This method leverages the advantages of Mutual Information (MI) and XGBoost to fully consider the correlation and importance of features, allowing for the selection of the most discriminative features, optimizing the feature subset, and reducing the impact of redundant features. This feature selection method provided more precise inputs for subsequent deep learning model training, enhancing the model's efficiency and performance.

4.3 CNN-BiLSTM Network Model

The CNN-BiLSTM network combines the spatial feature extraction capabilities of Convolutional Neural Networks (CNN) with the temporal feature learning abilities of Bidirectional Long Short-Term Memory Networks (BiLSTM). This organic integration allows for efficient encoding of input data, making it suitable for complex tasks with spatiotemporal characteristics. The model first utilizes CNN to extract local spatial features, then employs BiLSTM to capture global temporal dependencies, and finally performs classification through a fully connected layer. The architecture of the CNN-BiLSTM network is illustrated in Fig. 3. The specific processes for spatial and temporal feature extraction are detailed as follows.

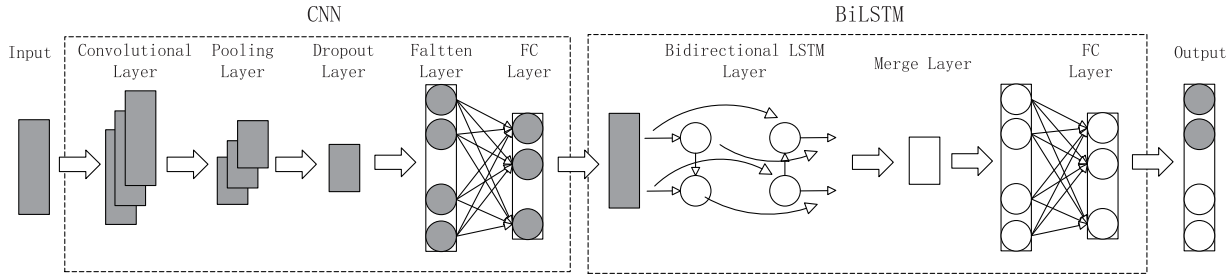

Figure 3: CNN-BiLSTM network model

4.3.1 Spatial Feature Extraction

The spatial feature extraction of the CNN-BiLSTM model is achieved through convolutional neural network layers, aiming to capture local spatial patterns within the input data. The input data is a processed feature matrix with dimensions $X \in R^{N*d}$, where N represents the number of samples and d denotes the feature dimension for each sample.

In this model, the input data first undergoes data input and format adjustment. The input feature matrix $X \in R^{N*d}$ is adjusted to a format suitable for 1D convolution operations. The data shape is transformed from $N*d$ to $N*1*d$, which provides the correct input format for the convolutional layer, ensuring that each sample can perform sliding window operations along the feature dimension as it passes through the convolutional layer. Then, the data goes through the first-layer 1D convolution processing. This layer uses convolution kernels to extract features. The specific calculation formula is as follows:

$$Z_{i,j} = \sum_{k=1}^{K} W_{j,k} \cdot X_{i,t+k-1} + b_j$$

Among them, $W_{j,k}$ is the j-th convolutional kernel with a size of K, and $X_{i,t+k-1}$ represents the feature of the i-th sample at position t, while b_j is the bias term. During this process, the ReLU activation function is used to introduce nonlinear expression capabilities. Subsequently, the output of the convolution is down-sampled through a max-pooling layer to reduce the spatial dimension of the data and retain the most significant features. The specific pooling calculation formula is as follows:

$$X_{pool,c,j} = \max\left(X_{c,j\cdot S:j\cdot S+K}\right)$$

Among them, X_{pool} refers to the pooled feature map, c denotes the channel index, j represents the pooling window position, K is the size of the pooling window, and S is the stride of the pooling operation. The dimensions of the pooled feature map are reduced by half. In the second convolutional layer, more filters are applied to extract advanced spatial features from the data. ReLU activation and batch normalization are again combined to boost nonlinear expression and speed up training. Then, the convolutional layer's output goes through max-pooling to further reduce the feature map dimension. Dropout regularization is also applied to mitigate overfitting and enhance the model's generalization.

$$Dropout\,(x) = mask \cdot x$$

Among them, the mask is a randomly generated binary mask with a probability of $p = 0.3$. Finally, the data processed by convolution, pooling, and Dropout is flattened and passed to the subsequent BiLSTM model for temporal feature extraction. These steps aim to effectively extract key features from the data while controlling model complexity to ensure good performance and stability in practical applications.

Through operations such as two layers of convolution, pooling, activation, and dropout, local spatial features are effectively extracted from the input data. The convolution operation captures local patterns at different scales, while pooling and dropout help to reduce overfitting and improve computational efficiency. After this series of processes, the data is passed to the BiLSTM layer for modeling temporal features.

4.3.2 Temporal Feature Extraction

Time series feature extraction is achieved through a Bidirectional Long Short-Term Memory (BiLSTM) network, which aims to mine temporal patterns in network intrusion data. Firstly, the results from spatial feature extraction are adjusted into a sequence shape suitable for LSTM processing, with tensor transformation applied to meet the input requirements of BiLSTM. Then, BiLSTM performs bidirectional sequence modeling on the data, computing hidden state sequences in both forward and backward directions to capture dependencies in the sequence. The specific implementation formulas are as follows:

$$h_t^{(f)} = H\left(W_{xh}^{(f)} x_t + W_{hh}^{(f)} h_{t-1}^{(f)} + b_h^{(f)}\right)$$

$$h_t^{(b)} = H\left(W_{xh}^{(b)} x_t + W_{hh}^{(b)} h_{t+1}^{(b)} + b_h^{(b)}\right)$$

In this context, $h_t^{(f)}$ and $h_t^{(b)}$ represent the forward and backward hidden states, respectively. W is the weight matrix, b is the bias, and H is the activation function. Then, the bidirectional hidden state sequences are concatenated to produce the final time-series feature output, ensuring comprehensive capture of temporal dependencies. Afterward, batch normalization and Dropout are applied to the output time features to ease the vanishing gradient problem and prevent overfitting. Subsequently, a fully connected layer generates the time-series feature vector. The spatial features extracted by CNN are used as inputs for BiLSTM, enabling it to model the temporal relationships between features in the sequence. The forward and backward

hidden states of BiLSTM jointly capture global dependencies in the time-series data, further refining the feature representation.

The spatial feature sequences extracted by the CNN serve as inputs to the BiLSTM, thereby modeling the temporal relationships among different features within the sequence. The forward and backward hidden states of the BiLSTM jointly capture the global dependencies of the time series, further enhancing the feature representation.

The CNN module was utilized to extract spatial features from the input data, primarily capturing local associations and patterns among features across different dimensions. Subsequently, the BiLSTM module was employed to extract temporal features, delving into the time-dependent relationships and dynamic characteristics of the data. The convolution operation, combined with pooling and batch normalization, effectively refined the spatial information of the input data while reducing redundant features. Meanwhile, the bidirectional LSTM aggregated information from both forward and backward flows, comprehensively considering the global characteristics of the time series. Ultimately, the deep features extracted from these two components were fused into a high-dimensional feature representation, laying a solid foundation for subsequent classification tasks.

5 Experimental Results and Analysis

In this paper, systematic quantitative experiments are conducted on the NSL-KDD and UNSW-NB15 datasets using the proposed feature selection method and model architecture to evaluate their performance in network intrusion detection. Also, ablation studies are designed to analyze and compare the contributions of each model component, further verifying the method's effectiveness.

5.1 Experimental Environment

The experiment was conducted using the Window11 operating system, an Intel(R) Core(TM) i7-14700HX CPU, 16 GB of RAM, and an NVIDIA GeForce RTX 4060 graphics card. The implementation was carried out using Python 3.9 and the Keras deep learning framework.

5.2 Dataset

5.2.1 NSL-KDD Dataset

This study employs the widely used NSL-KDD and UNSW-NB15 datasets to evaluate the proposed method. The NSL-KDD dataset is an improved version of the KDD Cup 1999 dataset, addressing issues of redundancy and class imbalance. It consists of a training set, KDDTrain+, and a test set, KDDTest+. The dataset includes normal traffic and four primary attack types: DoS, Probe, R2L, and U2R, which are further categorized into 39 specific attack classes. The dataset provides 122 initial features, covering dimensions such as basic, content, time, and traffic statistics.

5.2.2 UNSW-NB15 Dataset

The UNSW-NB15 dataset is a modern benchmark dataset for network intrusion detection, created by the University of New South Wales in Australia. It comprises approximately 1 million network connection records, each detailing 49 features spanning various categories such as basic content, timing, and statistical information. The dataset encompasses both normal traffic and nine primary attack types, including Denial of Service (DoS), Backdoor, Exploits, Fuzzers, Reconnaissance, and others, effectively simulating the complexities of contemporary network environments. Initially, the dataset includes 83 features, which are categorized into basic features, content features, timing features, and traffic statistical features.

5.3 Evaluation Metrics

To comprehensively evaluate model performance, accuracy, precision, recall, and F1 score were used as evaluation metrics. These metrics were calculated from the confusion matrix in Table 1, where: TP (True Positives) is the number of correctly classified attack records; TN (True Negatives) is the number of correctly classified normal records; FP (False Positives) is the number of normal records incorrectly classified as attacks. FN (False Negatives): The number of attack records incorrectly classified as normal.

Table 1: Confusion matrix

Confusion matrix		**Predicted value**	
		Normal	**Attack**
True value	Normal	TN	FP
	Attack	FN	TP

Using these metrics, we can calculate the model's accuracy, precision, recall, and F1 score, thereby comprehensively evaluating the model's classification performance.

Accuracy measures the proportion of correctly classified samples to the total number of samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Precision measures the proportion of samples predicted as positive by the model that are positive.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

Recall measures the proportion of all positive samples that are correctly predicted as positive by the model.

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

The F1-score is the harmonic mean of precision and recall, providing a balanced measure that considers both precision and recall.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6}$$

5.4 Experimental Procedure and Result Analysis

5.4.1 Parameter Settings

(1) Feature Selection Section

Feature selection is a method for identifying a subset of features relevant to the target task from the original feature set. By eliminating redundant or irrelevant features, feature selection not only reduces computational burden and dimensionality but also enhances model performance and generalization capability. This helps mitigate overfitting, improves model performance on new data, and accelerates both training and prediction processes.

This study employs three feature selection methods: Mutual Information (MI), XGBoost importance scoring, and the combined MI-XGB approach. The MI method calculates the mutual information scores between features and the target variable, selecting the top 40 features with the highest scores. The XGB method utilizes the XGBoost classification model to compute feature importance and selects the top 40 features based on importance. The MI-XGB combined method standardizes the mutual information scores and XGBoost importance, assigns respective weights to each, and calculates a comprehensive score to select the top 40 features. Each feature selection method outputs an optimized feature subset for model training. To validate the effectiveness and applicability of the proposed feature selection method, a comparative experiment was conducted under the same conditions, comparing the MI-XGB method with existing feature selection methods such as AE, PCA, XGBoost, and XGB-S. The specific results are presented in Table 2.

Table 2: Comparison of different feature selection methods (%)

Feature selection	NSL-KDD				UNSW-NB15			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
AE	95.87	95.65	95.85	96.08	83.66	84.35	83.85	83.98
PCA	97.79	97.83	97.82	97.91	85.75	95.96	85.90	86.12
XGBoost	98.08	98.23	97.95	98.25	86.58	86.75	86.89	87.05
XGB-S	98.08	98.23	97.95	98.25	88.71	88.69	88.67	87.95
MI-XGB	99.35	99.49	99.51	99.51	88.78	88.95	88.64	89.25

To investigate the impact of different weight configurations on model detection performance, the results are presented in Table 2. The table compares the classification performance of different feature selection methods on two datasets. Results show that the MI-XGB method outperforms others in all evaluation metrics. The superior performance of the MI-XGB method may be attributed to its ability to combine the advantages of mutual information and XGBoost, comprehensively considering feature correlations and importance in complex data relationships. To further verify the statistical significance of these performance differences, statistical tests were conducted to calculate the p-values between various feature selection methods. The results indicate that MI-XGB differs significantly from other methods in multiple metrics ($p < 0.05$). Notably, the differences between MI-XGB and the AE and XGB-S methods reach a highly significant level ($p < 0.01$). This confirms the superiority and stability of MI-XGB in feature selection tasks.

In addition to three distinct weight ratio settings, a more dynamic approach was also attempted, where weights were adaptively updated based on the training data during each training process. This method iteratively trains the model and gradually adjusts the weights of mutual information scores and XGBoost scores to achieve optimal detection performance. Ultimately, we compared the model performance under different weight configurations and evaluated the practical performance of each method. The comparison results revealed that the adaptive weight update approach significantly outperformed static weight ratio settings. This method can automatically optimize the feature selection process based on data characteristics, thereby enhancing the model's detection accuracy and stability, as detailed in Table 3.

Table 3: Comparison of different weights (%)

NSL-KDD		UNSW-NB15	
Weight (MI/XGB)	**Accuracy**	**Weight (MI/XGB)**	**Accuracy**
2/8	98.95	2/8	87.98
5/5	98.87	5/5	88.65
8/2	98.96	8/2	88.69
Adaptive	99.35	Adaptive	88.78

(2) Model Construction Section

The model employs a deep learning architecture combining "1D convolution with bidirectional LSTM." Initially, local features are extracted through a 1D convolutional layer with 64 filters and a kernel size of 3. Max pooling is then applied to reduce dimensionality, and batch normalization is used to accelerate training. Subsequently, a bidirectional LSTM is employed to capture temporal dependencies, with Dropout incorporated to prevent overfitting. Finally, a fully connected layer outputs classification results for five categories, and the softmax activation function is utilized to compute class probabilities. The Adam optimizer is adopted, with multi-class cross-entropy serving as the loss function.

(3) Training and Validation Section

During the training process, 6-fold stratified cross-validation was employed, with each fold trained for 10 epochs and a batch size of 32. The feature data were adjusted to fit the model input before training, while the label data were processed using One-Hot encoding. The model was trained on the training set for each fold and evaluated on the validation set for classification performance. The model's performance was comprehensively assessed using accuracy, weighted recall, and weighted F1 score.

5.4.2 Experimental Results

To evaluate the performance of the CNN-BiLSTM network intrusion detection model based on the MI-XGB feature selection proposed in this paper, experimental comparisons were conducted on the training and validation processes of the model using two classic network intrusion detection datasets: NSL-KDD and UNSW-NB15. Figs. 4 and 5 illustrate the changes in training accuracy and loss of the model concerning the number of iterations on the NSL-KDD and UNSW-NB15 datasets, respectively. It is evident from these figures that feature selection significantly impacts the model's training convergence and performance optimization.

On the NSL-KDD dataset, the training and validation accuracy of the CNN-BiLSTM model significantly improved after applying MI-XGB feature selection, stabilizing within a short period, with the final validation accuracy reaching 99.35%. The loss curves indicate that both training and validation losses decreased rapidly and gradually stabilized at lower levels. The trends in validation loss and training loss are consistent, demonstrating the model's strong generalization capability without evident overfitting. This confirms the effectiveness of MI-XGB feature selection in optimizing input features, as well as the advantages of CNN-BiLSTM in extracting spatial and temporal features.

On the UNSW-NB15 dataset, the model utilizing MI-XGB feature selection demonstrated commendable performance. The training and validation accuracy is boosted quickly, and the final validation accuracy hits 88.78%. The loss curves indicated that both training loss and validation loss decreased swiftly in the initial rounds, subsequently leveling off. Compared to the NSL-KDD dataset, although the accuracy on the UNSW-NB15 dataset was slightly lower, the model still exhibited rapid convergence and a consistent training-validation trend, further validating the effectiveness of the MI-XGB feature selection method.

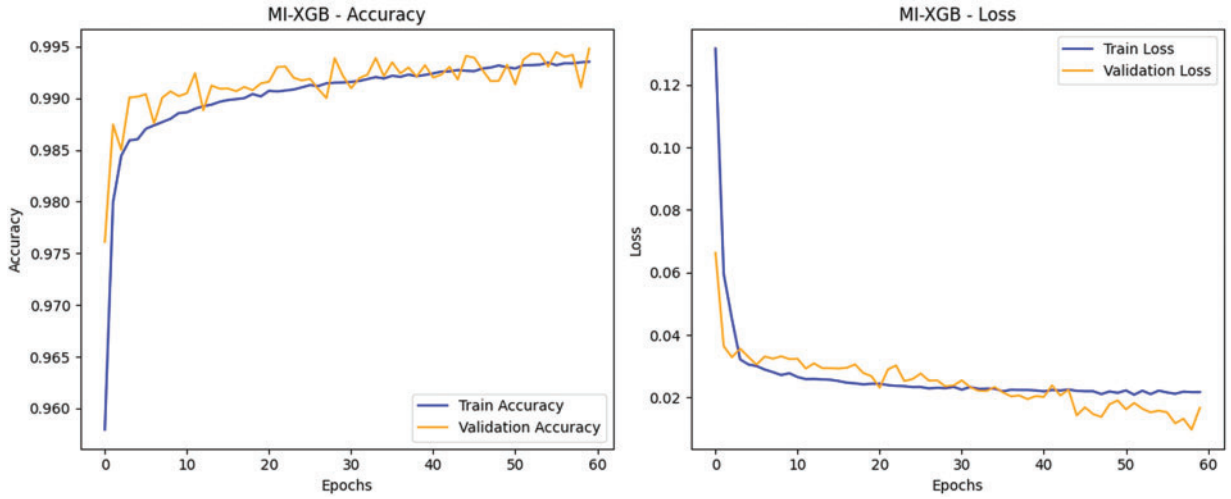

Figure 4: Changes in training accuracy and loss (NSL-KDD)

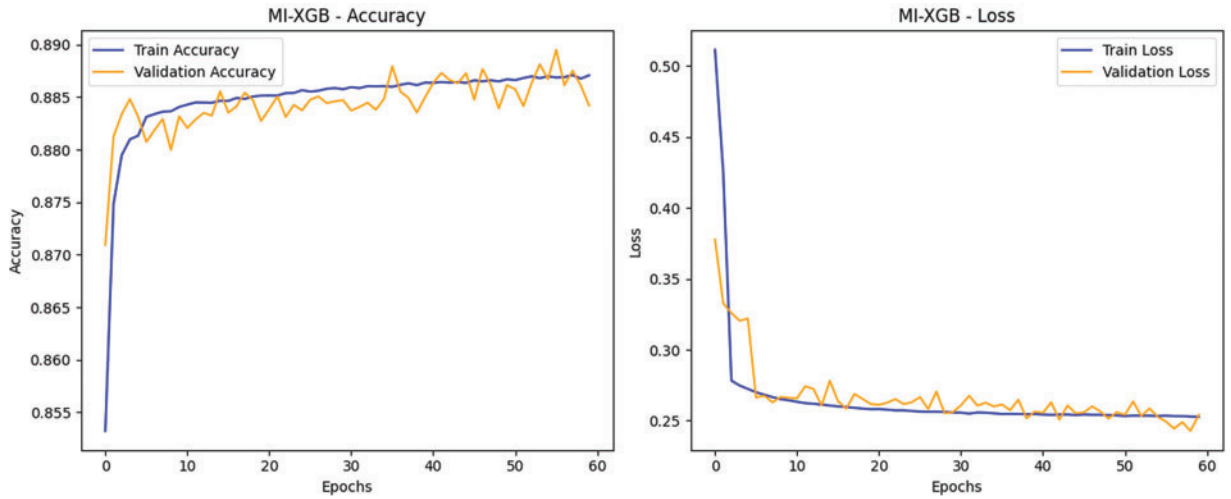

Figure 5: Changes in training accuracy and loss (UNSW-NB15)

To further verify the superiority of our method, we conducted comparative experiments on the NSL-KDD and UNSW-NB15 datasets against current mainstream intrusion detection methods. Table 4 shows the results.

Table 4: Comparison results (%)

Model	NSL-KDD				UNSW-NB15			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
TCN-BiLSTM [23]	80.70	83.20	82.80	82.90	–	–	–	–
AE-CNN-BiLSTM-AE [24]	85.70	84.70	87.50	85.10	–	–	–	–

(Continued)

Table 4 (continued)

Model	NSL-KDD				UNSW-NB15			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
OSS-SMOTE-CNN-BiLSTM [25]	–	–	–	–	72.38	69.94	87.42	77.71
PCC-RF-CNN-GRU [26]	–	–	–	–	84.72	85.62	86.18	83.05
The model proposed in this paper	99.35	99.49	99.51	99.51	88.78	88.95	88.64	89.25

Note: The "-" indicates that the model doesn't provide that metric in the experimental results.

The comparative results show that our method outperforms existing advanced models in key metrics like Accuracy, Precision, Recall, and F1-Score on the NSL-KDD and UNSW-NB15 datasets. On NSL-KDD, our method achieves 99.35% in accuracy and 99.51% in F1-Score, surpassing deep learning baselines like AE-CNN-BiLSTM. On UNSW-NB15, the F1-Score reaches 89.25%, a 6.2% improvement over PCC-RF-CNN-GRU.

These results indicate that the proposed mutual-information and XGBoost-weighted feature selection effectively identifies representative feature subsets, enhancing model generalization and detection accuracy. The CNN and BiLSTM hybrid structure also efficiently captures spatial and temporal features in network traffic, improving the recognition of complex attack patterns. Despite these achievements, the method has limitations. Model inference latency and resource consumption need optimization for large-scale, fast-changing real-time traffic. Also, the lack of complete metrics from some comparative methods on the same datasets limits the comprehensiveness of the comparison. Future work could involve lighter network structures or online learning mechanisms to enhance model real-time performance and adaptability.

5.4.3 Ablation Experiments

To validate the enhancement in model performance brought by the feature selection method, this paper conducts ablation experiments. The purpose of ablation experiments is to evaluate the impact of gradually removing certain key technologies or steps on the final model performance, thereby providing an in-depth analysis of the advantages of the MI-XGB feature selection method.

1. Experimental Setup

In the ablation experiments, we designed the following comparison strategies:

(1) Raw Full Feature Data: Utilizing all original features from the dataset for training, serving as the baseline model.
(2) Mutual Information Feature Selection: Feature selection is performed based solely on Mutual Information (MI), selecting features that exhibit a strong correlation with the target label.
(3) XGBoost Feature Selection: Utilizing only the feature importance scores from XGBoost to select features, based on the evaluation of relative importance through gain or frequency.
(4) MI-XGB Feature Selection: Combining MI and XGBoost feature importance for weighted ranking, the top k features with the highest weighted scores based on MI-XGB are selected.

All these methods are quantitatively compared and evaluated on the NSL-KDD and UNSW-NB15 datasets using the data preprocessing from Section 4.1. They are trained and tested with the same deep learning model (CNN + BiLSTM) for a comprehensive comparison.

2. Results of Ablation Experiments

By calculating accuracy, precision, recall, and F1 score, we conducted a quantitative analysis of the model's performance before and after feature selection. The experimental results are shown in Tables 5 and 6.

Table 5: Performance comparison of ablation experiments on the NSL-KDD dataset (%)

Method	Accuracy	Precision	Recall	F1-Score
Original feature set (122-Dimensional)	96.59	98.55	96.59	96.88
MI feature selection (Top 40 Dimensions)	99.18	99.39	99.43	99.42
XGBoost feature selection (Top 40 Dimensions)	99.26	99.24	99.26	99.24
MI-XGB feature selection (Top 40 Dimensions)	99.35	99.49	99.51	99.51

Table 6: Performance comparison of ablation experiments on the UNSW-NB15 dataset (%)

Method	Accuracy	Precision	Recall	F1-Score
Original feature set (122-Dimensional)	84.72	85.62	86.18	83.05
MI feature selection (Top 40 Dimensions)	87.69	88.61	88.39	87.77
XGBoost feature selection (Top 40 Dimensions)	87.81	88.63	88.21	87.33
MI-XGB feature selection (Top 40 Dimensions)	88.78	88.95	88.64	89.25

As shown in the table, the MI-XGB method outperforms other methods in all evaluation indicators, especially in the F1 score, indicating that the method effectively improves the performance of the model in the case of category imbalance.

3. Results Analysis

(1) Feature selection significantly enhances model performance.

The experimental results indicate that feature selection plays a significant role in enhancing model performance. Compared to directly using the raw data, models employing the MI (Mutual Information), XGB (XGBoost Feature Importance), and MI-XGB methods show notable improvements in key metrics such as accuracy, precision, recall, and F1 score. Among these, the MI-XGB method stands out, as it effectively combines mutual information and XGBoost feature importance through a weighted approach, thereby enhancing the model's ability to capture useful information and optimize overall performance. Specifically, the MI method filters the most discriminative features by measuring the correlation between input features and the target variable, while the XGB method selects the most contributive features based on the feature importance scores derived from tree models. The MI-XGB method organically integrates the advantages of both approaches, considering the complementarity of features while ensuring the predictive accuracy of the model, resulting in the best performance across all metrics.

(2) Feature selection aids in model convergence.

The comparative data from metrics such as accuracy indicate that feature selection has a positive impact on the convergence speed of the model. Utilizing feature selection methods can assist the model in achieving

optimal performance more rapidly, particularly in reducing redundant computations during the training process and mitigating overfitting. The MI-XGB method, by selecting the most representative features, enables the model to focus on key characteristics during training, thereby minimizing interference from irrelevant or noisy features. This optimization not only enhances the training efficiency of the model but also improves its generalization capability, resulting in more robust performance on test data. By reducing the involvement of ineffective features, the model is better positioned to learn the true patterns within the data, avoiding overfitting and demonstrating higher stability and accuracy in practical applications.

(3) Comprehensive advantages of the MI-XGB method.

In the comparison of all feature selection methods, the MI-XGB method demonstrated outstanding performance across multiple evaluation metrics, including accuracy, precision, recall, and F1 score. This method significantly enhances the overall detection capability of network intrusion detection models by effectively combining mutual information and XGBoost feature importance. Moreover, it successfully balances precision and recall when addressing class imbalance issues. Particularly on datasets with class imbalance, the MI-XGB method is more effective in detecting minority classes, thereby reducing the rate of missed detections. This is crucial in high-risk domains such as network intrusion detection, where a single missed intrusion can lead to severe consequences. Consequently, the application of the MI-XGB method not only achieves significant improvements in model performance but also provides an effective solution to critical issues encountered in practical applications.

In summary, the experimental results provide strong evidence for the positive impact of feature selection on model performance, particularly with the MI-XGB method. By organically integrating various feature selection strategies, this approach offers robust support for enhancing the performance and efficiency of network intrusion detection models. The outstanding performance across multiple metrics indicates a clear direction for further model optimization.

6 Conclusion

This paper proposes a network intrusion detection model that combines mutual information with an extreme gradient boosting feature selection method and integrates it with convolutional neural networks and bidirectional long short-term memory networks for training. Experimental results demonstrate that the MI-XGB feature selection method significantly enhances model performance on the NSL-KDD and UNSW-NB15 datasets. Comprehensive analysis of evaluation metrics such as accuracy, recall, and F1 score indicates that the MI-XGB method effectively identifies key features during the selection process, reducing model complexity while improving the accuracy and robustness of network intrusion detection. Feature selection not only accelerates model convergence but also effectively mitigates the negative impact of redundant features on the model, further enhancing its generalization capability.

However, despite the achievements of this study, there are still some limitations. Firstly, although the MI-XGB feature selection method has demonstrated excellent performance across multiple datasets, its effectiveness in handling certain specialized attack types remains to be improved. Secondly, the proposed method still relies on traditional convolutional neural networks and LSTM structures. In the future, exploring more advanced neural network architectures, such as graph neural networks (GNN) or Transformer models, could further enhance the model's ability to detect complex attack patterns. Also, to handle constantly updated datasets and emerging attack types, transfer learning strategies can be introduced. This enables the model to better adapt to new data distributions and attack patterns, enhancing its practicality and timeliness.

In future research, we plan to optimize the feature selection method by incorporating attention mechanisms and self-supervised learning to enhance the model's robustness and adaptability. Additionally,

we will combine other deep learning techniques, such as reinforcement learning, to further advance the intelligence of network intrusion detection systems. Overall, this study demonstrates the significant potential of combining feature selection with deep learning models to improve intrusion detection performance in the field of cybersecurity, with ample room for further research.

Acknowledgement: The authors would like to thank the Xi'an Key Laboratory of Human-Machine Integration and Control Technology for Intelligent Rehabilitation, School of Computer Science, Xijing University, for providing equipment and experimental environment support. Special thanks also go to Bosen Xu, Tao Jing, Jiaxiong Luo, Jiarui Wang and Xiaofang Dong for their participation in the research survey.

Funding Statement: This work was supported by the High-Level Talents Special Fund Project of Xijing University in 2022, "Research on Industrial Internet of Things Intrusion Detection Technology Based on Deep Learning" (XJ22B04).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization and project design, Manzheng Yuan and Kai Yang; investigation, supervision, project administration, and formal analysis, Kai Yang; methodology, software, validation, visualization, Manzheng Yuan; writing—original draft preparation, Manzheng Yuan; writing—review and editing, Manzheng Yuan. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data available on request from the authors. The data that support the findings of this study are available from the corresponding author, Kai Yang, upon reasonable request.

Ethics Approval: Not applicable.

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

References

1. Barbosa GNN, Andreoni M, Mattos DMF. Optimizing feature selection in intrusion detection systems: pareto dominance set approaches with mutual information and linear correlation. Ad Hoc Netw. 2024;159(1):103485. doi:10.1016/j.adhoc.2024.103485.
2. Chen H, You YZ, Jin HB, Wu C, Yang SW. Network intrusion detection method of improving feature selection and CNN-BiLSTM. Microelectron Comput. [cited 2025 Jun 2]. Available from: https://kns.cnki.net/kcms2/article/abstract?v=ywQ10qv17SH-BGVWj0Ii7r8q7ZutBx-BRO15KNOI32apJLo5RmxwX4jI4KDbw2leaACBRLTaC3d96rV4Az72IvMXWZ8A_yPMJaOsY40pxYTGw0ZZjcuMEyMHfMrWhTauFGEYkBLytSM2m4TOWBssjIe0Y-SYLPmAZ1XqXebKnu229YhjA0rLnnScnGDIdL0Ej7bix_g1Bpw=&uniplatform=NZKPT.
3. Udurume M, Shakhov V, Koo I. Comparative analysis of deep convolutional neural network—bidirectional long short-term memory and machine learning methods in intrusion detection systems. Appl Sci. 2024;14(16):6967. doi:10.3390/app14166967.
4. Zhang J, Zhang X, Liu Z, Fu F, Jiao Y, Xu F. A network intrusion detection model based on BiLSTM with multi-head attention mechanism. Electronics. 2023;12(19):4170. doi:10.3390/electronics12194170.
5. Xu DF, Xu HZ, Deng DJ. Intrusion detection based on CNN-BLSTM-XGB. Comput Eng Des. 2024;45(3):676–83. (In Chinese). doi:10.16208/j.issn1000-7024.2024.03.006.
6. Bouke MA, Abdullah A, Udzir NI, Samian N. Central tendency feature selection (CTFS): a novel approach for efficient and effective feature selection in intrusion detection systems. Multimed Tools Appl. 2025;2025:1–22. doi:10.1007/s11042-025-20837-8.
7. Logeswari G, Thangaramya K, Selvi M, Roselind JD. An improved synergistic dual-layer feature selection algorithm with two type classifier for efficient intrusion detection in IoT environment. Sci Rep. 2025;15(1):8050. doi:10.1038/s41598-025-91663-z.

8. Alrashdi I, Sallam KM, Alrowaily MA, Alruwaili O, Arain B. FIDWATCH: federated incremental distillation for continuous monitoring of IoT security threats. Ad Hoc Netw. 2024;165(11):103637. doi:10.1016/j.adhoc.2024.103637.
9. Du C, Guo Y, Zhang Y. A deep learning-based intrusion detection model integrating convolutional neural network and vision transformer for network traffic attack in the Internet of Things. Electronics. 2024;13(14):2685. doi:10.3390/electronics13142685.
10. Ullah S, Ahmad J, Khan MA, Alshehri MS, Boulila W, Koubaa A, et al. TNN-IDS: transformer neural network-based intrusion detection system for MQTT-enabled IoT Networks. Comput Netw. 2023;237(5):110072. doi:10.1016/j.comnet.2023.110072.
11. Xi C, Wang H, Wang X. A novel multi-scale network intrusion detection model with transformer. Sci Rep. 2024;14(1):23239. doi:10.1038/s41598-024-74214-w.
12. Wang P, Song Y, Wang X, Guo X, Xiang Q. ImagTIDS: an internet of things intrusion detection framework utilizing GADF imaging encoding and improved Transformer. Complex Intell Syst. 2024;11(1):93. doi:10.1007/s40747-024-01712-9.
13. Yan F, Zhang G, Zhang D, Sun X, Hou B, Yu N. TL-CNN-IDS: transfer learning-based intrusion detection system using convolutional neural network. J Supercomput. 2023;79(15):17562–84. doi:10.1007/s11227-023-05347-4.
14. Zhu F, Ye Z, Li P, Xu H. Transformer-based domain adaptation method for IoT traffic intrusion detection. Comput Sci. [cited 2025 Jun 2]. Available from: https://kns.cnki.net/kcms2/article/abstract?v=ywQ10qv17SGRIQLyPdKq1DQSAp9kJ6VCuGfhAGhcnOyVWpqQpIMTaifBKfQ1iMRmWHq9nozNMEtEDnWseHMrS5WCO8BMjsIxbyy5SjfN9lMW_XLx4VW8vrIp0zIrgqVWCPQG0NlaNvyJzh-ZF11EBVdATUR5obL9FMGmv1c8hOg6JLEMaJJ1JQ$==$&uniplatform=NZKPT&language=CHS.
15. Binbusayyis A. Hybrid VGG19 and 2D-CNN for intrusion detection in the FOG-cloud environment. Expert Syst Appl. 2024;238(19):121758. doi:10.1016/j.eswa.2023.121758.
16. Ren K, Yuan S, Zhang C, Shi Y, Huang Z. CANET: a hierarchical CNN-attention model for network intrusion detection. Comput Commun. 2023;205(16):170–81. doi:10.1016/j.comcom.2023.04.018.
17. Yang T, Chen J, Deng H, He B. A lightweight intrusion detection algorithm for IoT based on data purification and a separable convolution improved CNN. Knowl Based Syst. 2024;304(23):112473. doi:10.1016/j.knosys.2024.112473.
18. Momand A, Jan SU, Ramzan N. ABCNN-IDS: attention-based convolutional neural network for intrusion detection in IoT networks. Wirel Pers Commun. 2024;136(4):1981–2003. doi:10.1007/s11277-024-11260-7.
19. Li P, Wang H, Tian G, Fan Z. A cooperative intrusion detection system for the Internet of Things using convolutional neural networks and black hole optimization. Sensors. 2024;24(15):4766. doi:10.3390/s24154766.
20. Bhati BS, Chugh G, Al-Turjman F, Bhati NS. An improved ensemble based intrusion detection technique using XGBoost. Trans Emerg Telecommun Technol. 2021;32(6):e4076. doi:10.1002/ett.4076.
21. Alalhareth M, Hong SC. An improved mutual information feature selection technique for intrusion detection systems in the Internet of medical things. Sensors. 2023;23(10):4971. doi:10.3390/s23104971.
22. Fang Y, Yao Y, Lin X, Wang J, Zhai H. A feature selection based on genetic algorithm for intrusion detection of industrial control systems. Comput Secur. 2024;139(1):103675. doi:10.1016/j.cose.2023.103675.
23. Bai WR, Wei F, Zheng GY, Wang BH. Study on intrusion detection algorithm based on TCN-BiLSTM. Comput Sci. 2023;50(S2):941–8. (In Chinese).
24. Liang XY, Xing HY, Hou TH. CNN-BiLSTM network intrusion detection method based onself-supervised feature enhancement. J Electron Meas Instrum. 2022;36(10):65–73. (In Chinese). doi:10.13382/j.jemi.B2205445.
25. Wang Z, Liu Y, He D, Chan S. Intrusion detection methods based on integrated deep learning model. Comput Secur. 2021;103(6):102177. doi:10.1016/j.cose.2021.102177.
26. Wang XY, Zhao LH. Network intrusion detection model based on multi-stage feature selection and CNN-GRU. J N Univ China Nat Sci Ed. 2024;45(1):66–73. (In Chinese).