scientific reports

Check for updates

OPEN

An improved intrusion detection method for IIoT using attention mechanisms, BiGRU, and Inception-CNN

Kai Yang, JiaMing Wang ✉ & MinJing Li

In the field of Industrial Internet of Things (IIoT), existing intrusion detection models face challenges in three main areas: low accuracy in detecting attack traffic, feature redundancy when dealing with high-dimensional and complex attack traffic, making it difficult to capture critical information, and a tendency to favor learning common categories while neglecting rare categories when handling imbalanced data. To tackle these challenges, this study introduces an intrusion detection method that combines an attention mechanism, Bidirectional Gated Recurrent Units (BiGRU), and Inception Convolutional Neural Network (Inception-CNN) to enhance the model's detection rate. Simultaneously, the method employs a mixed sampling strategy for data resampling to address the bias learning issue caused by data imbalance. Additionally, the method employs a hybrid sampling strategy for data resampling to address the bias learning issue caused by data imbalance. It also incorporates denoising techniques to handle potential dataset noise introduced by hybrid sampling. Furthermore, a feature selection method combining Pearson correlation coefficient and Random Forest is applied to eliminate feature redundancy, enhancing the model's ability to capture crucial information from high-dimensional attack traffic. Experimental validation on internationally recognized datasets (Edge-IIoTset, CIC-IDS2017, and CIC IoT 2023) affirms the reliability of the proposed intrusion detection method. This approach underscores the significance of intrusion detection in the security of Industrial IoT and showcases its potential in addressing pertinent challenges in network security.

The rapid development of the IIoT has brought significant growth to the global economy. However, it has also brought security risks such as leakage of core industrial data and unauthorized manipulation of interconnected terminals. Industrial firewalls are a common means of protecting the Industrial Internet of Things. However, industrial firewall technology only provides passive defense mechanisms and may not effectively prevent files and programs containing threat codes.

To address these issues, intrusion detection technology has been deployed as an active network security measure. By continuously monitoring the network, it provides real-time protection against both internal and external intrusions. The characteristics of intrusion detection include proactiveness, real-timeliness, and dynamism, making it a promising research direction in the field of IIoT security.

Intrusion detection technology employs classifiers to distinguish between normal and abnormal data flows, generating alerts for potential attack behaviors. Researchers have developed diverse intrusion detection models tailored to specific network attacks, incorporating machine learning algorithms into these frameworks. These classifiers leverage various algorithms, including Bayesian, decision tree, neural network, and support vector machine. Notably, deep learning-based intrusion detection algorithms exhibit robust capabilities in feature extraction and classification, garnering significant attention in the field.

Deep learning techniques demonstrate more efficient and accurate performance compared to traditional intrusion detection methods when analyzing features of large volumes of network traffic data. Currently, widely adopted deep learning methods in the field of intrusion detection include Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Long Short-Term Memory networks (LSTM). Scholars both domestically and internationally have conducted extensive research in this field, advancing the application and development of deep learning techniques in intrusion detection.

School of Computer, Xijing University, Xi'an 710123, China. ✉email: wjm91447@gmail.com

Ma et al[1]. proposed a model that combines an improved bidirectional gated recurrent unit with residual connections (Res-BIGRU) and integrated dynamic extreme learning machine (IDELM) to address challenges in using RNN for anomaly network traffic detection. Their study focuses on enhancing RNN architecture and introducing the IDELM method suitable for imbalanced classification problems. Through a series of experiments, they obtained satisfactory results, demonstrating the significant effectiveness and superiority of this approach in handling anomaly traffic detection, particularly in dealing with imbalanced datasets and IoT environments, where its robustness is more pronounced.

Zhong et al[2]. proposed a Big Data-based Hierarchical Deep Learning System (BDHDLS) aimed at enhancing the performance of Intrusion Detection Systems (IDS) in machine learning. BDHDLS leverages behavioral and content features to understand the characteristics of network traffic and information stored in payloads, employing hierarchical deep learning models to handle different data distribution clusters. Compared to previous single-model approaches, the introduction of BDHDLS significantly improves the detection rate of intrusion attacks. Additionally, BDHDLS reduces model construction time significantly through parallel training strategies and big data technologies, further enhancing the practicality and efficiency of the system.

Ullah et al[3]. have developed a novel anomaly detection model tailored for Internet of Things (IoT) networks, employing CNN as the primary tool. By implementing CNN models in different dimensions such as 1D, 2D, and 3D, and utilizing transfer learning techniques for binary and multi-class classification tasks, they validated the effectiveness of their model across multiple datasets. The research findings demonstrate that their model significantly outperforms existing deep learning implementations in terms of accuracy, precision, recall, and F1 score, thus making a significant contribution to advancements in IoT intrusion detection.

Park et al[4]. proposed an Artificial Intelligence -based NIDS aimed at addressing data imbalance issues and improving the performance of existing systems. With the continuous evolution of communication technologies, various heterogeneous data transmitted within network systems have escalated network security concerns. Among various NIDS technologies, recent interest has surged in AI-based anomaly detection systems, with various models proposed to enhance NIDS performance. However, the challenge persists as AI models often fail to adequately learn malicious behaviors, resulting in inaccuracies in detecting network threats. To address this issue, the research team employed an advanced generative model capable of synthesizing plausible data for smaller attack traffic. Through comprehensive evaluations across various datasets, research findings demonstrate that the proposed system significantly outperforms previous AI-based NIDS in terms of performance.

Saheed YK et al.[5] proposed the IoT-Defender framework, which combines an improved Genetic Algorithm (MGA) and LSTM networks to detect attacks in IoT networks. By performing feature selection and parameter optimization in an edge computing (EC) environment, IoT-Defender successfully addresses the limitations of traditional cloud-based intrusion detection systems in terms of data confidentiality, network capacity, and response speed. Experimental results demonstrate that IoT-Defender exhibits outstanding performance on the BoT-IoT, UNSW-NB15, and N-BaIoT datasets, validating its effectiveness in IoT network security.

Additionally, Saheed YK et al.[6] proposed an ensemble learning technique based on the Grey Wolf Optimizer (GWO) for detecting intrusions in IoT networks. This model combines the probability averages of base learners through a voting technique and employs feature selection and extraction techniques to reduce dimensionality. Experimental results indicate that this model achieved high accuracy and low false positive rates on the BoT-IoT and UNSW-NB15 datasets, reaching accuracies of 99.98% and 100%, respectively.

Alzughaibi et al.[7] aimed to enhance the performance and efficiency of IDS in cloud environments to address the growing security concerns in cloud computing environments. They built two models based on Deep Neural Networks (DNN): one using MLP with backpropagation (BP), and the other utilizing MLP with Particle Swarm Optimization (PSO) for training. These models were employed for binary and multiclass classification on the CSE-CIC-IDS2018 dataset. The research aimed to improve the accuracy of IDS in detecting intrusion attacks in cloud environments and enhance other performance metrics. They meticulously documented various aspects of the experiments. Experimental results showed the best accuracy for binary classification as 98.97% and for multiclass classification as 98.41%.

Although significant progress has been made in the field of intrusion detection through previous research, there still exist some limitations. For instance, methods proposed by MA et al. may experience performance degradation when confronted with complex intrusion behaviors. The BDHDLS model introduced by Zhong et al. might encounter classification errors when dealing with small datasets or overly sparse feature vectors. Ullah et al.'s CNN model requires extensive time and computational resources for training, which may not be able to promptly adapt to the rapid evolution of network intrusion attacks. The generative model-based NIDS proposed by Park et al. could face high computational costs and may not comprehensively consider specific intrusion behaviors. Saheed YK et al.'s combination of the improved MGA and LSTM enhances detection effectiveness but also increases computational complexity and resource consumption, which may not be ideal for resource-constrained edge devices. Similarly, while the voting technique that integrates multiple machine learning methods improves detection performance, it also adds to the model's complexity and computational overhead, posing challenges for practical deployment.

Therefore, this paper proposes the NIDS-BAI model to address these issues. By combining a BiGRU network with an Inception-CNN network equipped with an attention mechanism, the performance degradation problem when facing complex intrusion behaviors is overcome. Meanwhile, the attention mechanism allows the model to focus on key features, thus reducing the required training time and computational resources. Additionally, the introduction of hybrid sampling strategies and feature selection algorithms resolves potential classification errors caused by handling small datasets and overly sparse feature vectors. The significant contributions of this article are summarized as follows:

1. This study proposes a hybrid sampling strategy that combines the ADASYN and RENN algorithms, aimed at addressing the issue of data imbalance and avoiding the model's bias towards learning features of the majority class. At the same time, the LOF algorithm is introduced to handle potential dataset noise introduced by hybrid sampling, thus enhancing the model's robustness.
2. This study proposes a feature selection method that combines random forests and the Pearson correlation coefficient to accurately assess the linear relationships between features. By utilizing the random forest algorithm to evaluate the importance of each feature and calculating the Pearson correlation coefficient between features, this method can effectively aid the model in capturing crucial information from high-dimensional attack traffic while eliminating redundant information.
3. This study combines a BiGRU network with an attention mechanism and an Inception-CNN network. The BiGRU network effectively captures long-term dependency relationships, while the attention mechanism allows the model to selectively focus on key parts. Meanwhile, the Inception-CNN network extracts multi-scale features, enhancing the understanding of crucial features. This fusion enhances the overall performance and detection accuracy of the model.

The article is divided into six sections. The first section serves as an introduction, providing background information and motivation for the research. The second section reviews the strengths and limitations of previous studies. The third section introduces the theoretical foundation, including concepts such as BiGRU, attention mechanism, Inception-CNN, data mixture sampling strategy, and feature selection. The fourth section details the process of model construction, including dataset introduction and preprocessing, data mixture sampling, feature selection, and the construction of the NIDS-BAI model. The fifth section presents the results of simulation experiments and conducts comparative analysis. Finally, the advantages of the proposed method in the context of industrial Internet of Things are summarized based on the experimental results.

Related work

In this section, we thoroughly review prior research in the field of network intrusion detection, focusing on the contributions and limitations of previous studies. Table 1 summarizes notable works, highlighting their key contributions and potential areas for improvement.

Methods

Bidirectional gated cycle unit architecture

During the feature extraction process, traditional GRU operates in a unidirectional manner, extracting features solely from past information. However, the limitations of unidirectional processing are addressed by the introduction of the BiGRU. Unlike the unidirectional GRU, BiGRU not only learns the influence of current information from past information but also considers the impact of future information on the present in reverse. The forward and backward networks are connected to the same output layer, providing a comprehensive contextual understanding for each moment in the input sequence.

The BiGRU layer comprises forward and backward GRU layers, facilitating the assimilation of long-term sequential data in both directions. As depicted in Fig. 1, the bidirectional GRU operates across consecutive time steps, enhancing the model's ability to capture subtle contextual information from both past and future inputs[17]. This effectively addresses the limitation of unidirectional recurrent neural networks that rely solely on the input data sequence to learn prior context.

The current hidden state of BiGRU is jointly determined by the current input x_t, the forward hidden state from the previous time step $\overrightarrow{h_{t-1}}$, and the backward hidden state from the subsequent time step $\overleftarrow{h_{t+1}}$. The final output of BiGRU is obtained by merging the forward and backward hidden states and then passing the combined representation to the output layer[18].

The forward derivation equations are as follows:

$$\overrightarrow{r_t} = \sigma(\overrightarrow{W_r}\overrightarrow{x_t} + \overrightarrow{U_r}\overrightarrow{h_{t-1}} + \overrightarrow{b_r}) \tag{1}$$

$$\overrightarrow{z_t} = \sigma(\overrightarrow{W_z}\overrightarrow{x_t} + \overrightarrow{U_z}\overrightarrow{h_{t-1}} + \overrightarrow{b_z}) \tag{2}$$

$$\tilde{h_t} = \tanh(\overrightarrow{W_h}\overrightarrow{x_t} + \overrightarrow{U_h}(\overrightarrow{r_t} \odot \overrightarrow{h_{t-1}} + \overrightarrow{b_h})) \tag{3}$$

$$\overrightarrow{h_t} = \overrightarrow{z_t} \odot \overrightarrow{h_{t-1}} + (1 - \overrightarrow{z_t}) \odot \tilde{h_t} \tag{4}$$

The reverse derivation equations are as follows:

$$\overleftarrow{r_t} = \sigma(\overleftarrow{W_r}\overleftarrow{x_t} + \overleftarrow{U_r}\overleftarrow{h_{t-1}} + \overleftarrow{b_r}) \tag{5}$$

$$\overleftarrow{z_t} = \sigma(\overleftarrow{W_z}\overleftarrow{x_t} + \overleftarrow{U_z}\overleftarrow{h_{t-1}} + \overleftarrow{b_z}) \tag{6}$$

$$\tilde{h_t} = \tanh(\overleftarrow{W_h}\overleftarrow{x_t} + \overleftarrow{U_h}(\overleftarrow{r_t} \odot \overleftarrow{h_{t-1}} + \overleftarrow{b_h})) \tag{7}$$

S/N	Authors	Contribution	Improvement space
[8]	Bahashwan AA, Anbar M, Manickam S, Al-Amiedy TA, Aladaileh MA, Hasbullah IH	This literature review provides a rigorous SDN architecture background, introduces a new research classification system, comprehensively analyzes machine learning and deep learning methods in the study of detecting DDoS attacks in SDN networks, and provides key tools, challenges, and future research directions, providing important references and guidance for the research community.	Different research methods can be compared in greater depth, more detailed analysis of the data sets can be provided, and future research directions can be discussed more specifically.
[9]	Fatani A, Dahou A, Abd Elaziz M, Al-Qaness MAA, Lu S, Alfadhli SA, et al.	This article introduces a novel Intrusion Detection System (IDS) model that combines deep learning and optimization techniques. Through exceptional feature extraction and selection methods, as well as the application of the MH (Metaheuristic) optimization algorithm, it has successfully enhanced the accuracy in detecting previously unknown attacks in cloud and IoT environments. This presents a highly promising new process for the field of network security.	The article does not discuss potential limitations and possible detailed shortcomings, nor does it consider applicability to other potential performance bottlenecks and environments.
[10]	Al-Imran M, Ripon SH.	The article introduces a multi-layered approach to IDS analysis, conducting experiments that compare machine learning and deep learning models. The results demonstrate notable performance enhancements, particularly with Ensemble Random Forest and deep neural network models, using the Kyoto Honeypot dataset. These findings underscore the method's suitability for IDS and employ metrics like MCC and LIME for comprehensive result assessment and explanation.	Introducing cross-validation on multiple datasets can verify the model's robustness across different environments and data distributions, ensuring the consistency and reliability of the research findings.
[11]	Halbouni A, Gunawan TS, Habaebi MH, Halbouni M, Kartiwi M, Ahmad R.	The article introduces a deep learning-based intrusion detection system using CNN and LSTM for spatial and temporal feature extraction from network traffic. It achieves high detection rates, accuracy, and a low False Alarm Rate (FAR) in binary and multi-class classification. The study evaluates the model on various datasets and compares it favorably to other deep learning and learning algorithms, emphasizing its performance advantages.	The deep learning model excels in intrusion detection but faces challenges with certain attack types like web attacks in CIC-IDS2017 and specific attacks in UNSW-NB15, leading to lower detection rates. Class imbalance in the dataset makes it difficult to achieve high detection rates and a low False Alarm Rate (FAR), requiring further improvements.
[12]	He J, Wang X, Song Y, Xiang Q, Chen C.	The article introduces a method named CWVAEGAN to tackle class imbalance in controversy detection. It involves transforming the dataset through data reconstruction and an enhanced VAEGAN with a restricted number of minority class samples. Subsequently, CWVAEGAN-1DCNN is devised using this approach and a 1DCNN to enhance intervention detection in class-imbalanced data. Results demonstrate that CWVAEGAN-1DCNN surpasses existing methods, providing an effective solution for intervention detection.	CWVAEGAN-1DCNN excels in mitigating class imbalance but slightly lags behind in comparison to advanced techniques on the UNSW-NB15 dataset, suggesting room for improvement. The article lacks information on the method's computational complexity, detection rates in real-world deployment, and performance across various network environments.
[13]	Hu R, Wu Z, Xu Y, Lai T.	This article introduces an intrusion detection method using Mosaic-encoded neural networks, effective in detecting various attack combinations with multi-classification capabilities. It transforms one-dimensional CAN ID data into a two-dimensional grid for efficient feature extraction with CNN while preserving temporal relationships. The method is tested on diverse attack datasets, allowing detection of multiple attack combinations. Autoencoders reduce dimensionality for limited computational resources in vehicular devices. Experimental results consistently demonstrate high performance in detecting all attack combinations.	Limitations of this approach include the restricted nature of the dataset, necessitating the availability of more real-time datasets encompassing a wider range of attack types for further method validation. Furthermore, for 29-bit CAN ID data, the current method requires sufficient padding of the remaining bits during the encoding process, which could be improved upon in the future. Lastly, while a CNN model is employed, future considerations might involve more intricate CNN model architectures for identifying more complex interactions.
[14]	Kim T, Pak W.	Paper introduces network intrusion detection method using deep learning and GAN.Enables real-time intrusion detection without waiting for a certain data packet count. GAN-based classifier accurately identifies unclassifiable packets, reducing hardware requirements. Although accuracy improvements are necessary, this approach provides a lightweight solution for real-time intrusion defense, enhancing network security and speed.	There is room for improvement in the accuracy of this method, and more efficient machine learning models may be needed, but this does not diminish the advantages of the approach. Future research can focus on enhancing precision to further improve network interaction detection performance.
[15]	Gautam, S. et al.	Sunil Gautam et al. proposed a hybrid deep learning model combining Bidirectional Long Short-Term Memory (BiLSTM) and Gated Recurrent Unit (GRU) for network intrusion detection systems. Through simulations on the CICIDS2017 public dataset, the model demonstrated high efficiency, successfully predicting most network attacks with a classification accuracy of 99.13%. The model outperformed the Naïve Bayes classifier in terms of prediction accuracy and false positive rate, and it maintained high performance using only 58% of the dataset attributes. Additionally, the study independently showcased the performance of LSTM and GRU with RNN.	Future research can validate the effectiveness of this model on different datasets to assess its performance in other environments. Optimizing the model can enhance its application in real-time intrusion detection, reducing detection latency. Investigating more advanced feature selection methods can improve the model's performance when reducing the number of features.

Continued

S/N	Authors	Contribution	Improvement space
[16]	Nam M, Park S, Kim D S.	This paper proposes an intrusion detection model that integrates bidirectional Generative Pretrained Transformer (GPT) networks to detect intrusions in the Controller Area Network (CAN) bus protocol. The model utilizes GPT to learn the patterns of normal CAN identifier sequences and effectively detect sequences containing a small number of attack identifiers. By minimizing the negative log-likelihood (NLL) value of the bidirectional GPT network for normal sequences, the model identifies sequences as intrusions when their NLL values exceed a predefined threshold. Compared to unidirectional GPT models and Long Short-Term Memory (LSTM)-based methods, the bidirectional GPT model demonstrates superior performance in estimating CAN identifier sequences.	Future research could assess the model's generalizability across different vehicle systems and variants of the CAN bus protocol. Optimizing the model's computational efficiency and resource consumption could enhance real-time performance. Additionally, improving detection accuracy for sequences with a small number of attack identifiers, simplifying the model structure to reduce computational complexity, and exploring the model's application to other network protocols or systems can further evaluate its cross-domain adaptability and effectiveness.

Table 1. Literature survey.

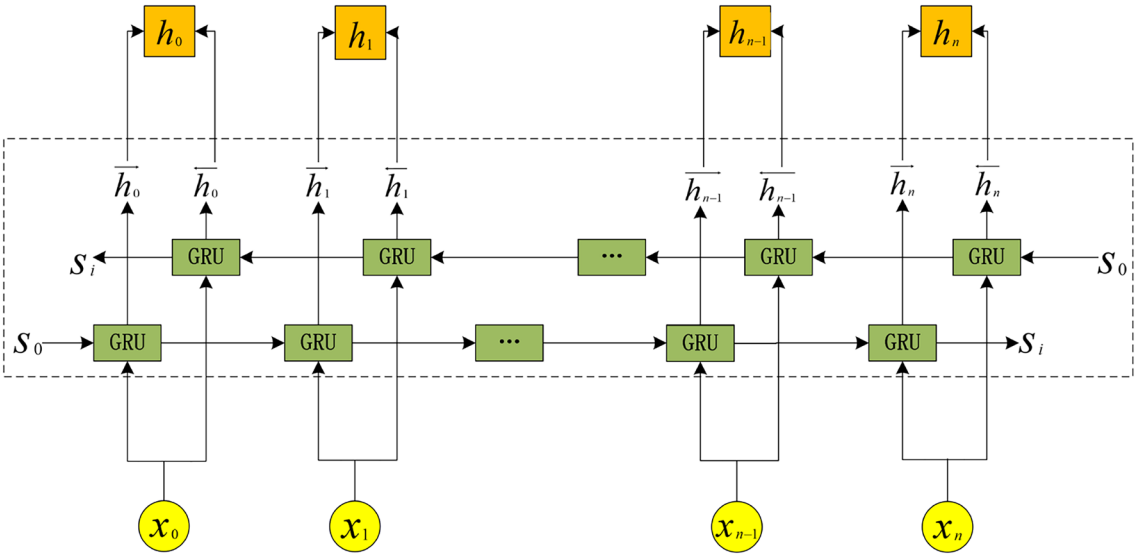

Figure 1. Cyclic unit structure of BiGRU network.

$$\overleftarrow{h_t} = \overleftarrow{z_t} \odot \overleftarrow{h_{t-1}} + (1 - \overleftarrow{z_t}) \odot \tilde{h_t} \tag{8}$$

The results from both forward and backward directions are linearly fused as:

$$h_t^{\text{combined}} = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{9}$$

Among them:

- $\overrightarrow{r_t}$ and $\overleftarrow{r_t}$ are reset gate vectors, determining the influence of the previous hidden state on the current candidate hidden state.
- $\overrightarrow{z_t}$ and $\overleftarrow{z_t}$ are update gate vectors, controlling how the previous hidden state is combined with the current candidate hidden state to update the hidden state.
- $\tilde{h_t}$ is the candidate hidden state, calculated based on the current input, the previous hidden state, and the values of reset gates.
- $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are the forward and backward hidden states at the current time step, determined by the update gate and the candidate hidden state.
- h_t^{combined} is the concatenation of the forward and backward hidden states. This concatenated vector contains information from both directions and can be used as input to the output layer for generating the final prediction.
- σ is the sigmoid activation function.
- $\odot$ denotes element-wise multiplication.

Attention mechanism

In the field of information processing, attention mechanism plays a crucial role in model design, akin to the workings of the human visual system. Its fundamental concept is to enable the model to selectively focus on specific information, akin to how humans concentrate attention on important areas when observing images. The

primary advantage of attention mechanism lies in consciously allocating computational resources to important areas, effectively capturing key information while suppressing irrelevant content.

In neural network architectures, attention mechanism selectively focuses on a portion of the overall information, distinguishing between important and irrelevant details, and directing the main attentional resources to critical areas to enhance network performance. By adjusting the weights of network parameters, attention mechanism significantly improves the neural network's ability to extract features, enabling it to better adapt to the requirements of complex tasks.

It is important to note that this mechanism requires determining the focus on the core task when processing input data. By judiciously allocating resources, it prioritizes processing key areas to meet task demands[19]. To illustrate its workings more clearly, please refer to Fig. 2.

Now, let $X = [x_1, \ldots, X_n] \in \mathbb{R}^{D \times N}$ represent N groups of input information, where each input is represented by a D-dimensional vector $x_n \in \mathbb{R}^D$, and $n \in [1, N]$. To efficiently utilize computational resources, it is not necessary to input all information into the neural network; relevant information can be selectively chosen from X. The computation of the attention mechanism can be divided into two steps: firstly, calculating the attention distribution over all input information; secondly, computing the weighted average of input information based on the attention distribution.

1. Calculating the attention distribution:Given a query vector q relevant to the task, an attention variable $z \in [1, N]$ is used to denote the index position of the selected information, where $z = n$ indicates the selection of the n-th input vector. Compute the probability a_n of selecting the n-th input vector given q and X:

$$a_n = p(z = n|X, q) = \text{softmax}(s(x_n, q)) = \frac{\exp(s(x_n, q))}{\sum_{j=1}^{N} \exp(s(x_j, q))} \tag{10}$$

Here, a_n is the attention distribution, and $s(x_n, q)$ is the attention scoring function.The attention scoring function can be computed using several models:

$$\text{Additive model: } s(x, q) = v^T \tanh(W_x + Uq) \tag{11}$$

$$\text{Dot product model: } s(x, q) = x^T q \tag{12}$$

$$\text{Scaling dot product model: } s(x, q) = \frac{x^T q}{\sqrt{D}} \tag{13}$$

$$\text{Bilinear model: } s(x, q) = x^T W q \tag{14}$$

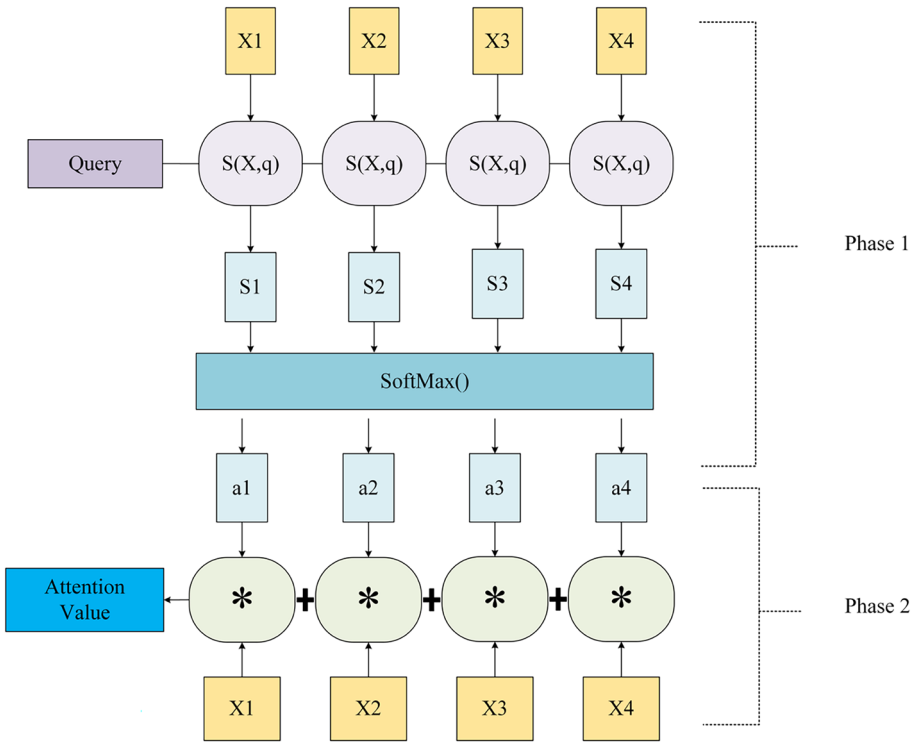

Figure 2. Structure diagram of attention mechanism.

Where W, U, and v are learnable parameters, and D is the dimension of the input vector. The Scaling Dot Product Model is adopted in this paper.

2. Computing the weighted average based on the attention distribution:Utilize the attention weights a_n to compute the weighted average of the input information x_n, resulting in the weighted average representation h:

$$h = \sum_{n=1}^{N} a_n x_n \tag{15}$$

This weighted average representation h incorporates the importance or relevance of the input information and can be used as input for subsequent tasks.

Inception-CNN network

The Inception-CNN is a powerful feature extraction network that enables convolutional neural networks to learn information from features at multiple different scales. Unlike relying on a single filter size, the Inception module simultaneously employs filters of different sizes, such as 1×1,3×1, and 5×1, to convolve the input data in parallel. Each filter generates a feature map, which are then concatenated together to form a new feature map containing features from multiple scales.

The expression of the Inception module is as follows:

$$O = \text{Concat}(Conv1D_{1\times1}(I), Conv1D_{3\times1}(I), Conv1D_{5\times1}(I), MaxPool_{3\times1}(I)) \tag{16}$$

Here, (I) represents the input data, $Conv1D_{1\times1}(I)$, $Conv1D_{3\times1}(I)$, and $Conv1D_{5\times1}(I)$ denote the 1×1, 3×1, and 5×1 convolution operations, respectively[20]. The Concat operation signifies the concatenation of the results, and O is the output of the Inception module.

In this article, the structure of the Inception module is depicted as shown in Fig. 3, utilizing 1×1 convolution, 3×1 convolution, and 5×1 convolution for feature extraction. To enhance computational efficiency and reduce the number of feature maps, 3×1 max pooling is applied after the 1×11, 3×1, and 5×1 convolutions. Additionally, a 1×1 convolution is employed to handle residual information between input feature maps.

Combining ADASYN, RENN, and LOF in a hybrid sampling method

This paper presents a hybrid sampling method named ARL (ADASYN-RENN-LOF), which combines the ADASYN, RENN, and LOF algorithms to address class imbalance and noise issues in data. The method mainly consists of the following steps: firstly, the original dataset is divided into majority and minority classes. Then, the ADASYN algorithm is applied to oversample the minority class samples, generating a new set of minority class samples. Simultaneously, the RENN algorithm is used to undersample the majority class samples, generating a new set of majority class samples. Next, the LOF algorithm is utilized to eliminate noise from the resampled dataset. Finally, the resampled and denoised majority and minority class samples are merged to form a balanced dataset, thus avoiding the model from biasing towards learning features from the more numerous class[21]. This contributes to training a more generalizable intrusion detection model. For detailed steps of the ARL algorithm, please refer to Algorithm 1.

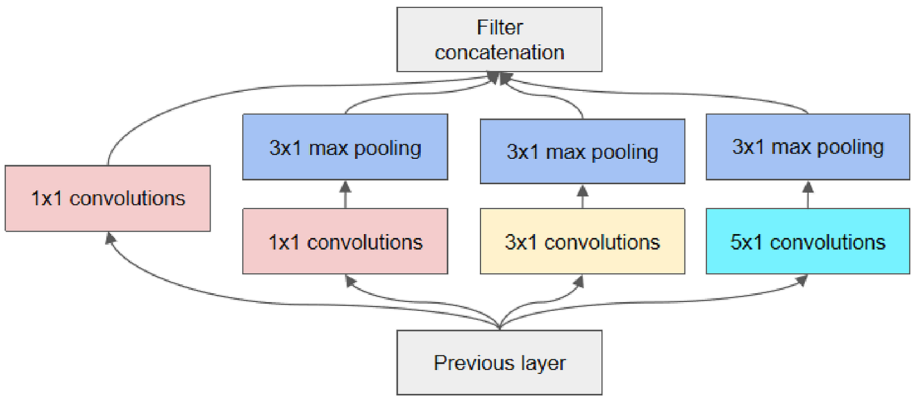

Figure 3. Structure of the Inception-CNN module.

Input: Minority class samples A, majority class samples R.
Output: Balanced minority class samples *newA*, balanced majority class samples *newR*.
Process:
Calculate the degree of imbalance: $d = \frac{|A|}{|R|}$.
if $d < d_{\text{th}}$ **then**
    Calculate the number of synthetic samples: $G = (|R| - |A|) \times b$.
    for i in $[1, |A|]$ **do**
        $neighborsA = \text{getNeighbors}(A, R, k1)$.
        $r_i = \frac{D_i}{k1}$, where D_i is the count of majority samples among $k1$ nearest neighbors of x_i.
        $\hat{r}_i = \frac{r_i}{\sum_{i=1}^{|A|} r_i}$.
        $g_i = \hat{r}_i \times G$.
    end for
    Initialize A as an empty set.
    for i in $[1, |A|]$ **do**
        for j in $[1, g_i]$ **do**
            $x_{zi} = \text{choose}(x_i, k1)$.
            $s_i = x_i + (x_{zi} - x_i) \times l$.
            $A = [A, s_i]$.
        end for
    end for
    $newA = A$.
    $C = |R| - |newA|$.
    $neighborsR = \text{getNeighbors}(R, newA, k2)$.
    for j in $[1, |R|]$ **do**
        $numReg(j) = \text{negRum(neighborsR(i))}$.
        if $numReg(i) > e$ **then**
            $R = \text{remove}(i, R)$.
        end if
    end for
    repeat
        steps (23) and (24).
    until Convergence
    $newR = R$.
    Apply LOF Algorithm to remove noise from *newA* and *newR*.
    Return *newA*, *newR*.
end if

Algorithm 1. ARL Balancing Algorithm

The algorithm accepts the original majority class sample set R and minority class sample set A as inputs. The resulting output includes the balanced majority class sample set labeled as *newR* and the minority class sample set labeled as *newA*.

1. Compute the dataset's imbalance degree d.
2. If $d < d_{\text{th}}$ (where d_{th} is a predefined maximum allowed degree of imbalance ratio), execute the following steps:
3. Calculate the number G of samples needed to be generated for the minority class.
4. For each sample in R, determine its k_1 nearest neighbors and compute the ratio r_i, where D_i represents the number of samples belonging to the majority class among the k nearest neighbors, and $|X|$ denotes the total number of samples. Normalize r_i to $\hat{r}_i$. Finally, calculate the number of samples that need to be synthesized for each minority class sample.
5. Generate g_i samples for each sample in R to form a new minority sample set.
6. Choose k_2 nearest neighbors from the set *newR* for each sample in A.
7. Identify the number of minority samples within the k_2 nearest neighbors of each majority sample and exclude the sample if the count exceeds e (where $e = 1$).
8. Repeat steps (6) and (7) until a stable majority class sample set is generated.
9. Eliminate noise in *newA* and *newR* to obtain the final *newR* and *newA*.

Feature selection algorithm based on random forest and pearson correlation coefficient

This study proposes a feature selection algorithm named RF-P, which integrates random forest algorithm and Pearson correlation coefficient to address the issue of redundant data features. The algorithm first evaluates the importance of sample features using the random forest algorithm and then combines the Pearson correlation coefficient to analyze the correlation between features.

Random Forest is a powerful machine learning algorithm that can effectively handle high-dimensional data and non-linear relationships. Meanwhile, the Pearson correlation coefficient is utilized to accurately measure the linear correlation between features. In the field of IIoT security, attack traffic data often exhibits highly complex patterns. Traditional feature selection techniques may not adequately capture this complexity[22]. By combining these two methods, RF-P can more accurately identify features relevant to attack behavior, thereby enhancing the robustness and performance of the model. For detailed algorithmic steps, please refer to Algorithm 2.

Input: Original dataset P
Output: Processed dataset $NewP$
Procedure:
Random Forest Importance:
for each base learner **do**
    Select corresponding out-of-bag data and calculate error ($error_a$).
    Introduce disruptions to the complete out-of-bag data set and calculate the error ($error_b$).
    Calculate feature importance.
end for
Ranking Features:
Rank the importance of the features.
Pearson Correlation Analysis:
Calculate Pearson correlation coefficients to measure feature correlations.
Feature Selection:
Utilize a combined approach of feature importance rankings and Pearson correlation analysis for optimal feature selection.
Output: $NewP$

Algorithm 2. RF-P Feature Selection Algorithm

Furthermore, the steps of the RF-P algorithm are explained as follows:

1. Select the corresponding out-of-bag data, i.e., the samples not used during the construction of the specific tree.
2. Calculate the error for this out-of-bag data, denoted as $error_a$.
3. Now, for the complete out-of-bag data sample:
4. Randomly introduce disturbances to the entire out-of-bag data sample.
5. Calculate the error for this perturbed out-of-bag data, denoted as $error_b$.
6. If the forest consists of M trees, compute the importance score I for a feature using the following formula: $I = \frac{error_b - error_a}{M}$
7. For two variables X and Y:
8. Calculate the covariance $cov(X, Y)$ between X and Y).
9. Calculate the standard deviations s_X and s_Y of X and Y.
10. Calculate the Pearson correlation coefficient $r_{X,Y}$ using the formula: $r_{X,Y} = \frac{cov(X,Y)}{s_X s_Y}$ The Pearson correlation coefficient $r_{X,Y}$ ranges from -1 to 1. If $r_{X,Y}$ is close to ±1, it indicates a high linear correlation between the two features. If it's close to 0,there is no linear correlation between the two features.

NIDS-BAI model building process

In the initial stage of modeling, the original dataset undergoes preprocessing. Subsequently, mixed sampling of the data is performed using the ARL algorithm, aiming to address the issue of data imbalance. This imbalance can cause the model to tend towards learning features from common categories while neglecting to learn from features of less represented categories[23]. Simultaneously, noise in the dataset after mixed sampling is eliminated to prevent its influence on model learning. Next, the RF-P algorithm is applied for feature selection to remove redundancy among features. Then, the NIDS-BAI model is constructed and trained. Finally, the model is evaluated. Please refer to Fig. 4 for the modeling process.

Data preprocessing

This paper utilized three datasets, namely Edge-IIoTset, CIC-IDS2017, and CIC IoT 2023, to evaluate the performance of the NIDS-BAI model in handling intrusion detection in the IIoT environment.

Firstly, the Edge-IIoTset dataset, specifically designed for IIoT applications, comprises 2,219,200 records with 60 features. This dataset covers 5 different types of attacks alongside normal traffic data. Additionally, the CIC-IDS2017 dataset contains 2,597,119 records and 79 features, encompassing 14 different types of attacks alongside normal traffic data. The CIC IoT Dataset 2023 includes 2,219,200 records and 46 features, covering 7 different

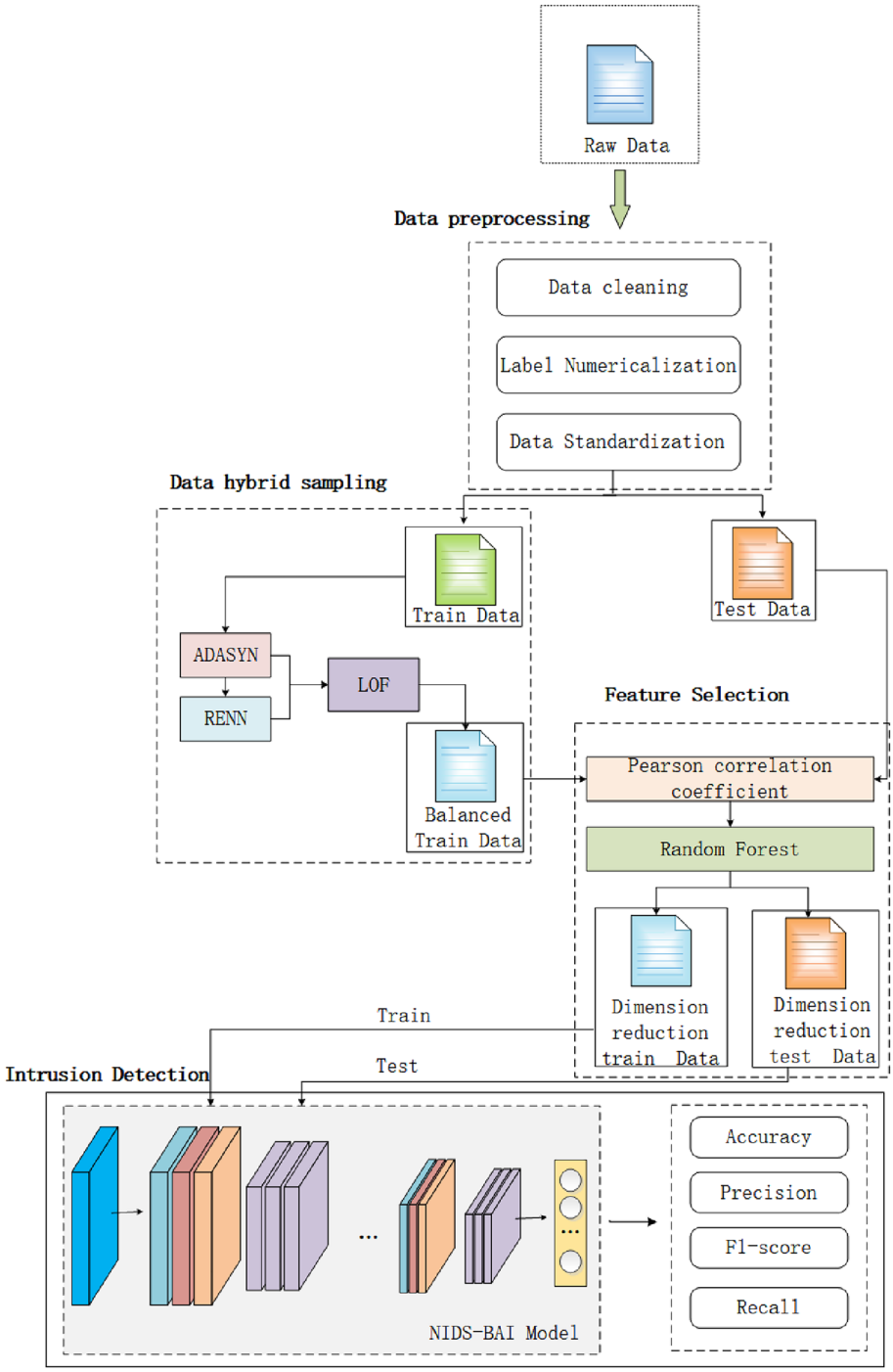

Figure 4. System model.

types of attacks alongside normal traffic data. These datasets' diversity contributes to evaluating the effectiveness of the proposed model in both traditional and contemporary network attack scenarios in IIoT environments.

The Edge-IIoTset dataset encompasses data from various IoT devices (over 10 types), including low-cost digital sensors used for sensing temperature and humidity, ultrasonic sensors, water level sensors, PH sensors, soil moisture sensors, heart rate sensors, flame sensors, etc. The CIC IoT Dataset 2023, on the other hand, comprises data obtained from 33 attacks executed within a topology composed of 105 IoT devices. Lastly, the CIC-IDS2017 dataset is derived from real network traffic and simulated attacks, utilized for evaluating the performance of IDS in detecting various network attacks, including DoS (Denial of Service), DDoS (Distributed Denial of Service), malware, and botnet attacks. The diversity of these datasets facilitates the assessment of the effectiveness of proposed models across different network attack scenarios.

In designing the NIDS-BAI model, this study specifically considers the specific IIoT protocols and device types reflected in datasets such as Edge-IIoTset, CIC-IDS2017, and CIC IoT 2023. These datasets encompass network traffic characteristics of various IIoT protocols.

Please refer to Table 2, which provides a detailed breakdown of the entire network traffic for each dataset.

One of the key steps in the preprocessing stage for the Edge-IIoTset, CIC-IDS2017, and CIC IoT 2023 datasets is data standardization. This step aims to transform the data into a standard normal distribution with a mean of 0 and a standard deviation of 1. The primary purpose of data standardization is to eliminate differences in scales among different features, thereby improving training stability and model accuracy. Standardized data exhibits higher consistency, mitigating the risk of certain features dominating the model learning process and facilitating the development of more robust and accurate models[24]. The standardization process is typically achieved through the formula shown in Equation 17.

$$z = \frac{x - \mu}{\sigma} \tag{17}$$

Data hybrid sampling

Taking the CIC IoT 2023 dataset as an example, significant optimization of category distribution was achieved through the application of the ARL algorithm. During this process, the ADASYN algorithm was utilized to expand minority class samples, including 'Benign', 'Spoofing', 'Recon', 'Web', and 'BruteForce', thereby enhancing the model's recognition ability for these categories. At the same time, the RENN algorithm effectively undersampled majority class samples, namely 'DoS', 'DDoS', and 'Mirai', reducing their sample count. The ARL algorithm mitigated the phenomenon of class imbalance in the dataset. Figure 5 illustrates the change in dataset distribution before and after the application of the ARL algorithm, demonstrating its effectiveness in addressing class imbalance issues.

The left plot in Fig. 5 illustrates the class proportions in the original data. It can be observed that the 'DDoS' class dominates, accounting for approximately 72.79%, while the proportions of 'Benign', 'Spoofing', 'Recon',

Classification		
	MultiCategorization	
Dataset	Traffic category	Label number
Edge-IIoTset	Normal	0
	DDoS	1
	Injection	2
	MITM	3
	Malware	4
	Scanning	5
CIC-IDS2017	Benign	0
	DDoS	1
	DoS Hulk	2
	DoS GoldenEye	3
	Slowloris	4
	DoS Slowhttptest	5
	FTP-Patator	6
	SSH-Patator	7
	Web Attack_Brute Force	8
	Web Attack_XSS	9
	Web Attack_Sql Injection	10
	PortScan	11
	Bot	12
	Infiltration	13
	Heartbleed	14
CIC IoT 2023	Benign	0
	DoS	1
	DDoS	2
	Mirai	3
	Spoofing	4
	Recon	5
	Web	6
	BruteForce	7

Table 2. Data set classification.

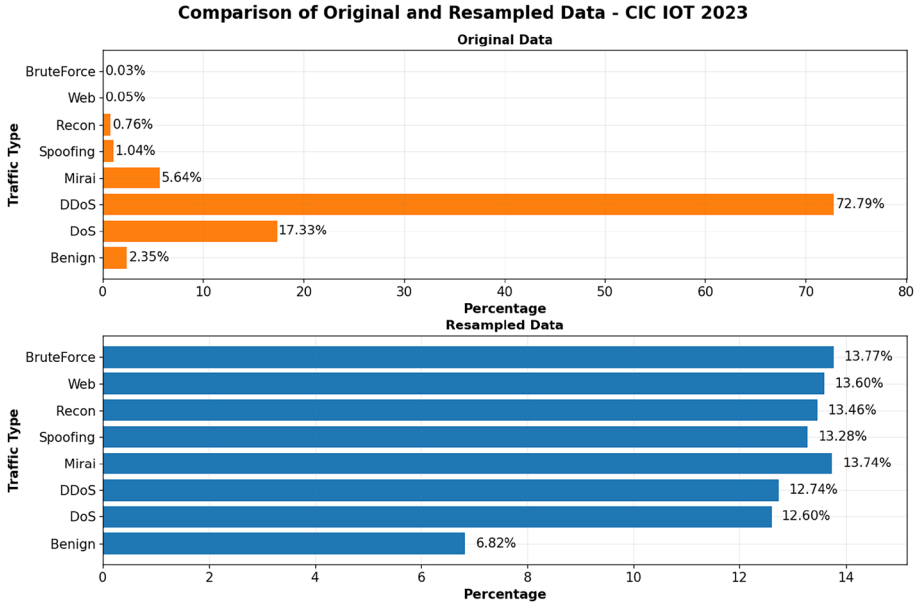

Figure 5. Comparison of original and resampled data.

'Web', and 'BruteForce' classes are relatively small, at 2.35%, 1.04%, 0.76%, 0.05%, and 0.03%, respectively. The data imbalance leads the model to tend towards learning the features of common classes, while neglecting the learning of less common ones.

The right plot in Fig. 5 demonstrates that after applying the ARL algorithm, the proportions of each class in the dataset become relatively balanced. The proportion of the 'DDoS' class decreases to 12.74%, while the proportions of 'Benign', 'Spoofing', 'Recon', 'Web', and 'BruteForce' classes increase to 6.82%, 13.28%, 13.46%, 13.60%, and 13.77%, respectively. The ARL algorithm avoids the over-learning of features from common categories by the model, thus enhancing the model's generalization capability and detection rate for all categories.

After hybrid sampling, the LOF algorithm is used to handle the introduced potential dataset noise. LOF is an algorithm for anomaly detection that measures the density of each sample point relative to its neighboring sample points. By utilizing LOF to identify and remove potential noise in the dataset, the model's learning process can be prevented from being influenced by samples deviating from the original data distribution. Figure 6 illustrates the distribution of outliers before and after LOF processing.The X-axis, Y-axis, and Z-axis respectively represent the position of data points in a three-dimensional space.

In Fig. 6, the left plot illustrates the distribution of data after hybrid sampling, with noise identified in red calculated by the LOF algorithm. The right plot represents the data distribution after removing the red noise points. It is evident from the figures that the LOF algorithm does not alter the overall distribution of the dataset.

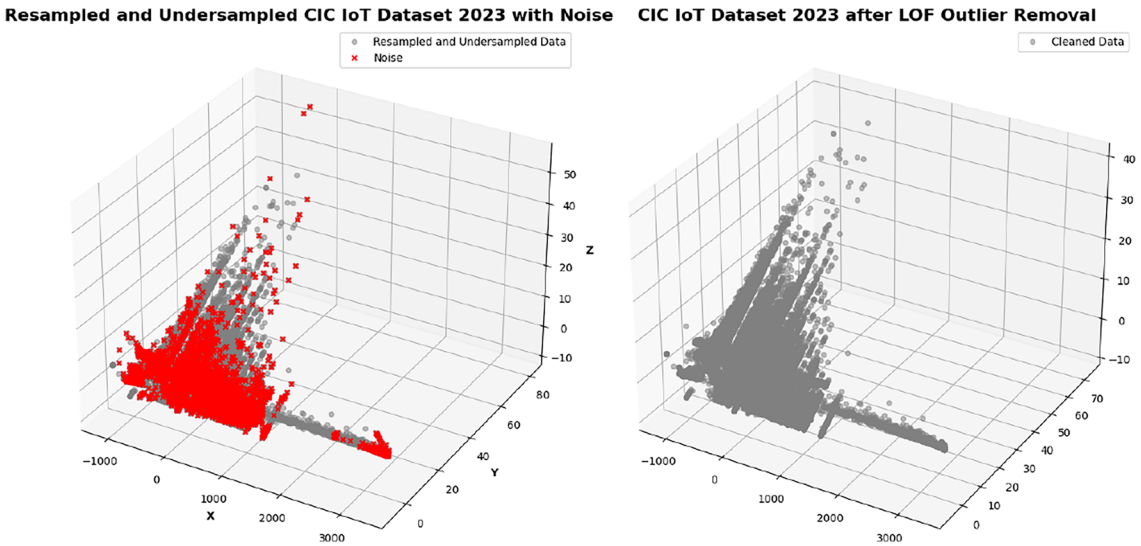

Figure 6. Comparison before and after LOF algorithm implementation.

This further ensures that the model can better capture meaningful features during training, aiding in strengthening the model's robustness and enhancing its generalization capability.

Feature selection

Using the CIC-IoT 2023 dataset as an example, let's illustrate the feature selection process of the RF-P algorithm. Firstly, the importance of each feature in the CIC-IoT 2023 dataset was computed using the random forest algorithm. Subsequently, the features were ranked based on their importance, as shown in Fig. 7.

According to Fig. 7, the importance of the IAT (Inter Arrival Time) feature is 0.5163, making it the most significant feature. IAT refers to the time interval between the arrivals of two consecutive packets. In network traffic analysis, monitoring changes in IAT can effectively identify anomalous traffic, especially in the detection of DoS and DDoS attacks. Furthermore, the importance measures of several other features are zero, indicating that they have minimal impact on the target variable. Therefore, these features can be excluded during the modeling process to reduce computational complexity and mitigate the risk of overfitting. However, relying solely on feature importance for feature selection is a single criterion, which may result in less convincing outcomes.

Hence, this study also utilized the results of Pearson correlation analysis and integrated them with the feature importance results for feature selection. Figure 8 presents the Pearson correlation results among these features.

To further explore the potential two-dimensional distribution correlation among different features, this study conducted a correlation analysis on each individual feature. The results revealed significant differences among these features. For example, the correlation score of the "Min" feature was 0.3087, while the "IRC" and "Telent" features were excluded from the scope of correlation analysis due to being unique values, making it impossible to calculate their correlation scores. The correlation metrics for all features ranged from -0.1249 to 0.3087.

The feature selection in this paper is based on the analysis results of Figs. 7 and 8. For features with strong linear correlations, important features were retained based on their importance metrics. For features with weak linear correlations, their correlation scores were analyzed, and if the score was below zero, the feature was removed. For features with correlation scores beyond the analyzed range, their importance based on Random Forest was considered, and if the importance metric exceeded 0.01, the feature was retained. Ultimately, the Edge-IIoTset dataset retained 39 features, the CIC-IDS2017 dataset retained 61 features, and the CIC-IoT 2023 dataset retained 35 features.

NIDS-BAI model

The NIDS-BAI model integrates two deep learning networks: a BiGRU network with attention mechanism and an Inception-CNN network. Firstly, the BiGRU network conducts bidirectional data analysis, effectively capturing long-term dependencies at both ends of the data, thus enhancing the model's ability to recognize complex attack traffic. Next, an attention mechanism is introduced to dynamically adjust the importance of different time steps in the output of the BiGRU network, improving the quality of its output. The attention mechanism enables the model to focus on analyzing key parts of the attack traffic, thereby enhancing detection accuracy and adaptability.

Following this, an Inception-CNN network is employed to optimize the output features of the attention mechanism, enhancing the model's representational capacity by extracting multiscale spatial information[25].

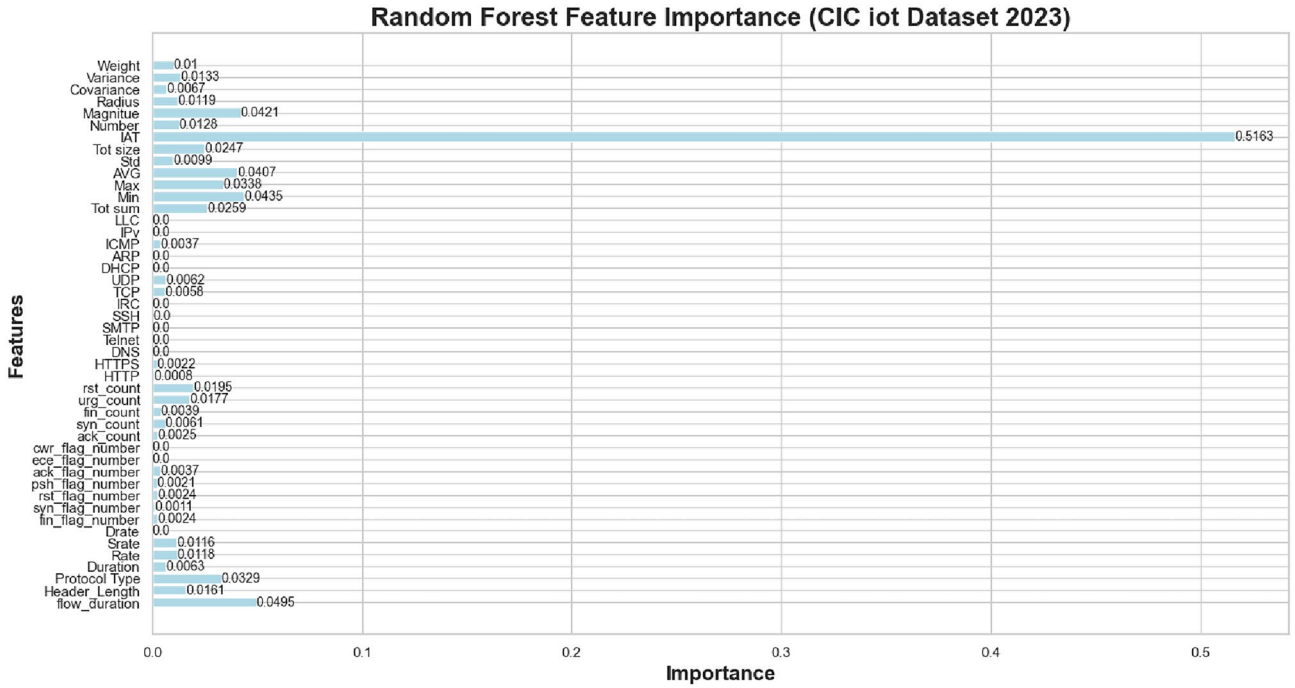

Figure 7. Results of feature importance analysis.

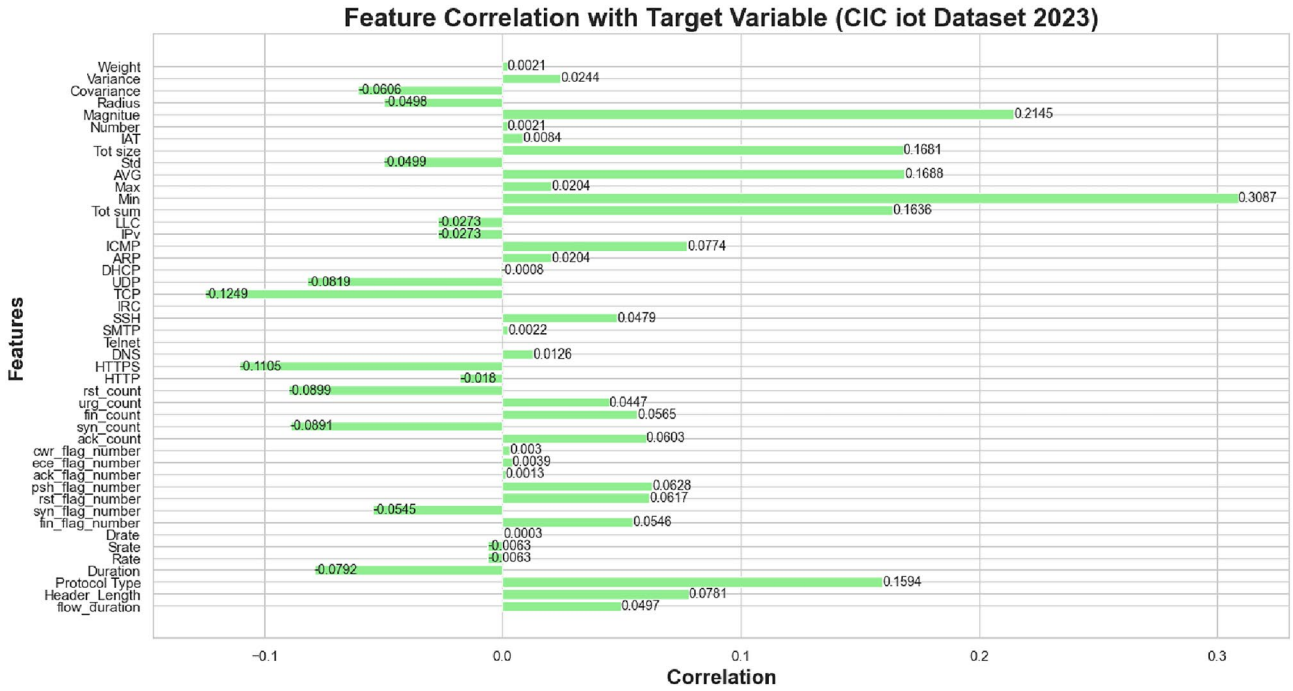

Figure 8. Results of linear correlation analysis.

The Inception-CNN utilizes convolutional kernels of different sizes to extract features and integrates them to enhance the model's understanding of features at different levels. Ultimately, classification of network traffic is performed through a decision layer.

The NIDS-BAI model fully leverages the capabilities of the BiGRU network with attention mechanism and the Inception-CNN network. While the BiGRU network effectively captures long-term dependencies, its utilization increases computational complexity. However, by introducing the attention mechanism, the model can selectively focus on key parts, thereby improving prediction accuracy. Furthermore, the Inception-CNN network extracts multiscale features, enhancing the understanding of crucial features, and consequently improving the model's classification accuracy. The schematic diagram of the NIDS-BAI model is depicted in Fig. 9.

Figure 9 demonstrates the NIDS-BAI model, which is comprised of two core components: a BiGRU network equipped with an attention mechanism and an Inception-CNN network. These two networks work collaboratively to jointly enhance the model's capability in recognizing attacks within network traffic. The modeling process of the NIDS-BAI model is as follows:

1. Initialization: The weights of the BiGRU layer are randomly initialized, and the query vector in the attention mechanism is assigned random weights.
2. Forward Pass: A forward GRU layer with 128 units is defined, and the input data X is passed through this layer to obtain the forward information representation $\overrightarrow{h_t}$.
3. Backward Pass: A backward GRU layer with 128 units is defined, and the input data X is passed through this layer to obtain the backward information representation $\overleftarrow{h_t}$.
4. Combine Outputs: The bidirectional information is merged by summing the outputs of the forward and backward GRU layers, resulting in the final output representation h_t.
5. Attention Mechanism: An attention mechanism is employed to calculate the attention weights for each time step of the input data, thereby enhancing the model's focus on critical information.
6. Inception Module: The Inception module comprises four branches. The first branch performs only a 1×1 convolution operation, while the subsequent three branches execute both convolution and pooling operations to extract features at different levels.
7. Final Layers: The model's output is transformed into a probability distribution over classes by applying a softmax layer.
8. Model Training: The model is trained on the input data X and target categories Y using specified settings, resulting in the final trained model M.

The specific pseudocode for the NIDS-BAI model is shown in Algorithm 3.

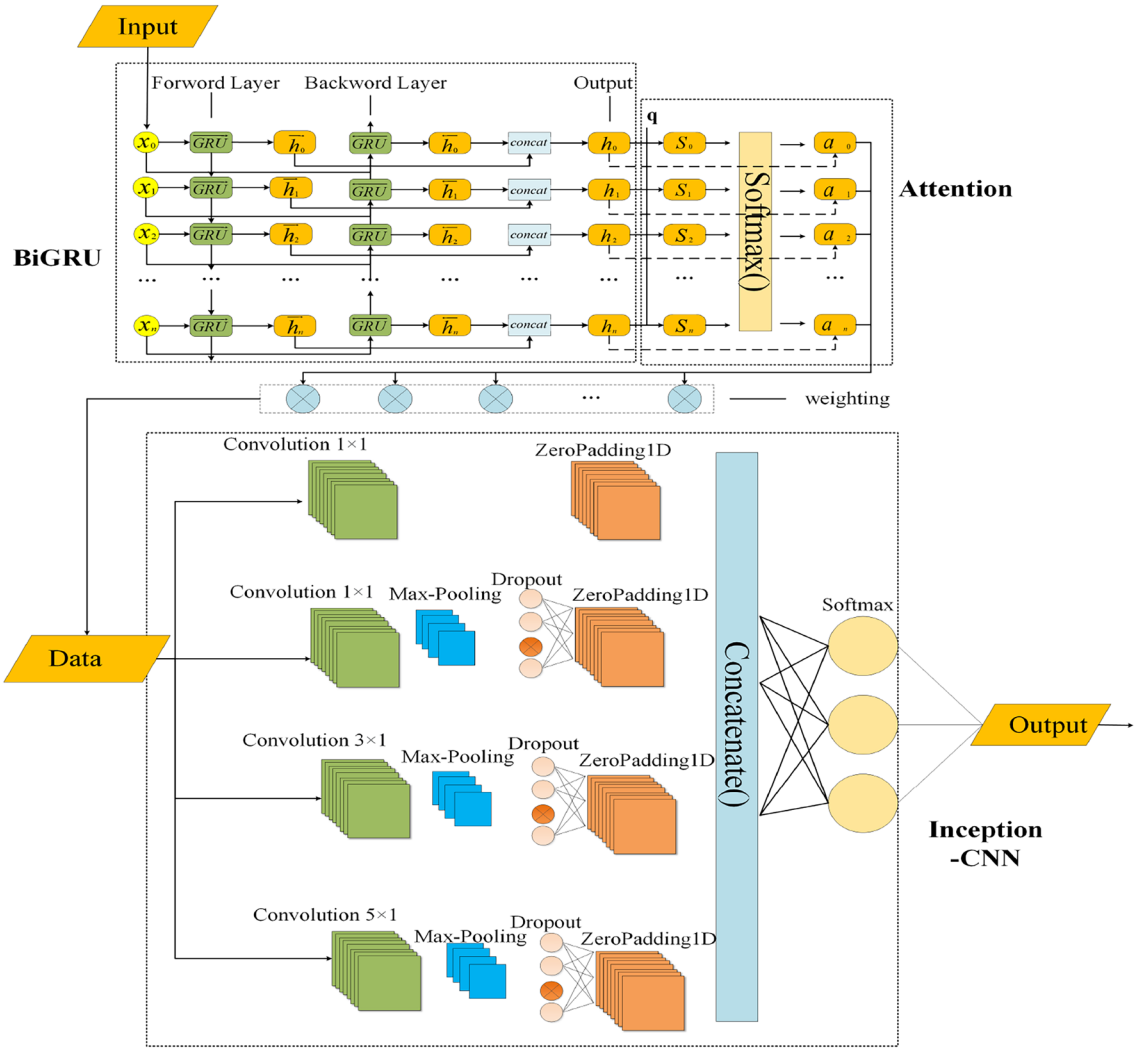

Figure 9. The internal structure of the NIDS-BAI model.

Require: Input data X of shape (None, 41, 1), target classes Y, optimizer (Adam), learning rate 0.001, batch size 1024, epochs 200

Ensure: Trained model M

Initialization:

Initialize Bidirectional GRU layers with random weights: $\overrightarrow{r_t}, \overrightarrow{z_t}, \tilde{h}_t, \overleftarrow{r_t}, \overleftarrow{z_t}, \tilde{h}_t$ ▷ Initialize with random weights or specific initialization strategy

Initialize query vector q for attention mechanism with random weights

Forward Pass:

Define forward GRU layer with 128 units, initialized with weights $\overrightarrow{r_t}, \overrightarrow{z_t}, \tilde{h}_t$

Forward output: $\overrightarrow{h_t} = \text{ForwardGRU}(X, \overrightarrow{r_t}, \overrightarrow{z_t}, \tilde{h}_t)$

Backward Pass:

Define backward GRU layer with 128 units, initialized with weights $\overleftarrow{r_t}, \overleftarrow{z_t}, \tilde{h}_t$

Backward output: $\overleftarrow{h_t} = \text{BackwardGRU}(X, \overleftarrow{r_t}, \overleftarrow{z_t}, \tilde{h}_t)$

Combine Outputs:

Combine forward and backward outputs: $h_t = \overrightarrow{h_t} + \overleftarrow{h_t}$ ▷ Combine outputs of BiGRU, Output Shape: (None, 41, 128)

Attention Mechanism:

Attention distribution: $a_n = \text{softmax}(h_t{}^T q)$ ▷ Compute Attention weights, Output Shape: (None, 41, 128)

Inception Module:

Inception module with three branches:

- Conv1D_1 × 1: Filters: 64; Output Shape: $(None, 41, 64)$
- Conv1D_1 × 1: Filters: 64; Output Shape: $(None, 41, 64)$; MaxPooling1D_3 × 1; Output Shape: $(None, 39, 64)$
- Conv1D_3 × 1: Filters: 64; Output Shape: $(None, 39, 64)$; MaxPooling1D_3 × 1; Output Shape: $(None, 37, 64)$
- Conv1D_5 × 1: Filters: 64; Output Shape: $(None, 37, 64)$; MaxPooling1D_3 × 1; Output Shape: $(None, 35, 64)$

Concatenate the outputs, Output Shape: $(None, 35, 256)$ ▷ Combine the outputs of the Inception Module

Final Layers:

Apply Softmax Layer, Output Shape: $(None, 35, Y)$ [Where Y is the number of classes]

Model Training:

Train the model on input data X and target classes Y using the specified optimizer, learning rate, batch size, and epochs.

return Trained model M

Algorithm 3. NIDS-BAI Model Training Algorithm

Result

Preparation for experiment

In Table 3 below, the experimental environment for this study is depicted.

Evaluation metrics

NIDS-BAI is a classification model for network traffic data, producing probabilities for each data type. The confusion matrix is constructed from the classification results, yielding True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) for each traffic type. Specifically, TP represents samples correctly classified as positive cases, FN indicates positive cases mistakenly classified as negative, FP denotes negative cases incorrectly classified as positive, and TN signifies samples correctly identified as negative cases. The confusion matrix is utilized to assess the model's performance, providing insights into its accuracy across

Module	Parameter
Processor	**Intel (R) Core (TM) i7-8750H**
Video Card	NVIDIA GTX1050Ti 4G
Main frequency	2.20GHz
RAM	40.00 GB
Operating system	Windows 10
Experimental tool	Python3.7 and TensorFlow2.11.0

Table 3. Experimental environment.

different categories and common misclassifications[26]. Examples of confusion matrices illustrating these concepts are detailed in Table 4.

Accuracy: It is calculated as shown in Formula 18 and represents the proportion of accurately predicted Normal and Attack samples to the total amount of data.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{18}$$

Precision: As shown in Formula 19, it represents the proportion of accurate predictions among the samples predicted as Attacks. Therefore, the precision is also referred to as the positive predictive value.

$$Pre = \frac{TP}{TP + FP} \tag{19}$$

F1-score: Formula 20 represents the harmonic average of the accuracy rate and recall rate. In scenarios where the categories in the dataset are imbalanced, the F1-score provides a useful metric for evaluating the model.

$$F1 = \frac{2 \cdot Pre \cdot Rec}{Pre + Rec} \tag{20}$$

The False Positive Rate (FPR) measures the ratio of incorrectly predicted normal samples as abnormal, quantifying the rate of false alarms in the model, as shown in Equation 21:

$$\mathrm{FPR} = \frac{FP}{FP + TN} \tag{21}$$

The True Positive Rate (TPR), also known as Recall(Rec) or Sensitivity, assesses the model's capability to accurately identify abnormal samples (true positives) among all actual abnormal samples, demonstrated by Equation 22:

$$TPR = \frac{TP}{TP + FN} \tag{22}$$

The influence of the ARL algorithm on training

To evaluate the impact of the ARL algorithm on model performance, this study utilized two CIC IoT 2023 datasets: one with the ARL algorithm applied and the other without. By analyzing the key performance indicators of these two datasets under NIDS-BAI model training, including training accuracy, testing accuracy, and the classification report heatmap, the effectiveness of the ARL algorithm in model training was assessed. The positive effect of the ARL algorithm on improving model generalization ability was emphasized. Experimental results are detailed in Fig. 10.

As shown in Fig. 10b, the dataset without the application of ARL algorithm exhibits poor generalization ability among different categories. Observing the accuracy curve, it can be seen that some categories dominate the training process, leading the model to prefer learning certain common network traffic. Therefore, after 50 training iterations, the training accuracy of this dataset is only 0.905, and the testing accuracy is 0.987. In contrast, as shown in Fig. 10a, after conducting experiments on the dataset with the application of ARL algorithm, it can be observed from the accuracy curve that the model's generalization ability among different categories has significantly improved. After 50 training iterations, the training accuracy of this model is 0.996, and the testing accuracy is 0.995.A comparative analysis of experimental results between datasets with and without the application of the ARL algorithm reveals that the model's generalization ability is significantly enhanced when the ARL algorithm is employed.

To provide a more persuasive assessment of the enhancement effect of the ARL algorithm on model performance, this paper presents two sets of data with classification reports after 50 iterations, including precision, recall, and F1 scores. The experimental results clearly demonstrate the positive impact of the ARL algorithm on the model's generalization ability. The specific classification report heatmap is shown in Fig. 11.

In Fig. 11a, it is evident that the application of the ARL algorithm to the dataset has enhanced the model's generalization ability, achieving higher precision, recall, and F1 scores in most categories. However, as shown in Fig. 11b, when the model is trained on the dataset without utilizing the ARL algorithm, the precision, recall, and F1 scores for the "Web" and "BruteForce" categories are all zero, indicating that the model failed to correctly identify these two categories.

In summary, when trained on a dataset without the application of the ARL algorithm, the model tends to learn categories with a larger quantity, leading to inferior recognition and classification performance for categories with

Confusion matrix	Predictive attack	Predictive normal
True attack	TP	FN
True normal	FP	TN

Table 4. Confusion matrix.

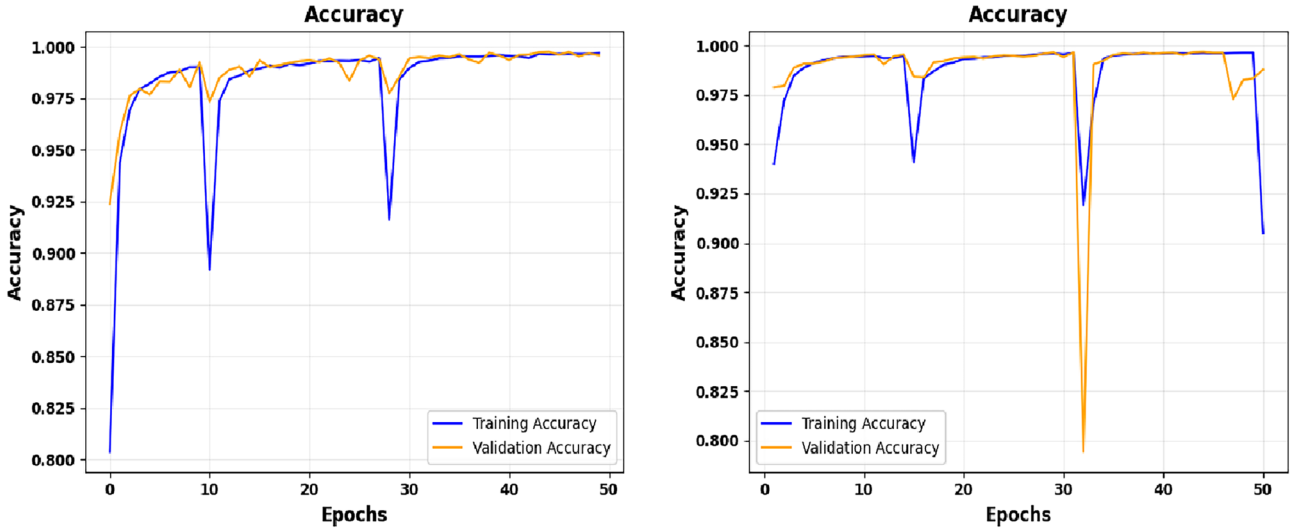

Figure 10. A comparison of model performance with and without the application of the ARL algorithm.

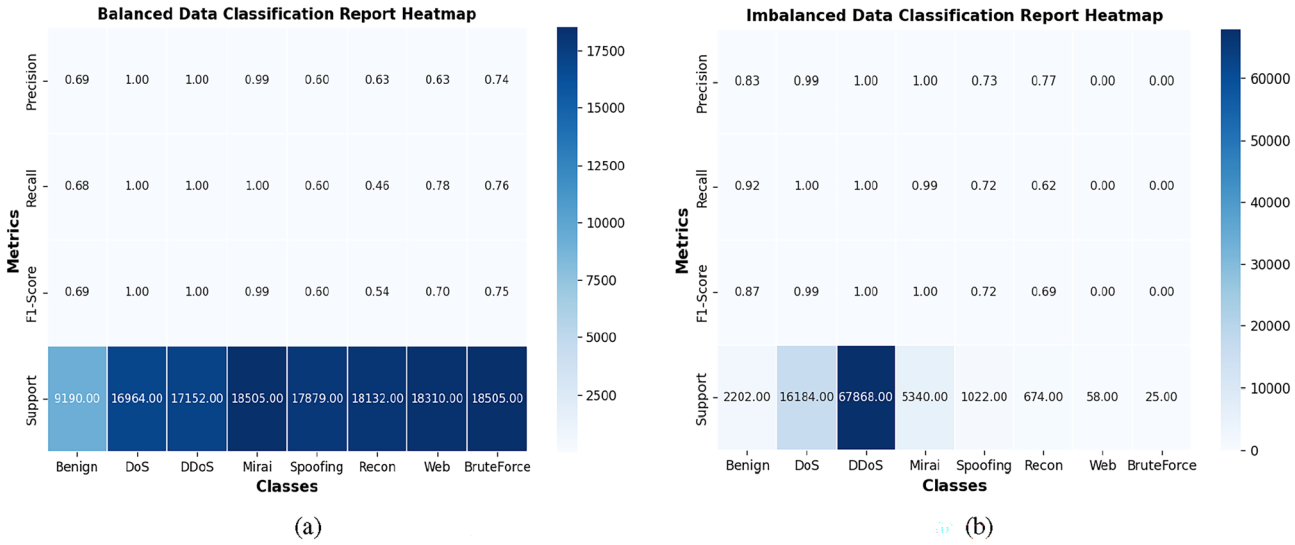

Figure 11. Classification reports for models with and without ARL algorithm application.

a smaller quantity. However, when trained on a dataset with the ARL algorithm, the model is able to learn each category more evenly, thereby improving its ability to recognize and classify categories with a smaller quantity. Therefore, the utilization of the ARL algorithm can enhance the robustness and generalization ability of the model, ensuring its excellent performance across all categories.

Comparative evaluation among distinct feature selection approaches

In this section, experiments were conducted on three datasets: Edge-IIoTset, CIC-IDS2017, and CIC IoT 2023. The objective of these experiments was to compare the proposed feature selection technique (RF-P) with Principal Component Analysis (PCA) and Information Gain (IG) methods in order to evaluate the improvement in model performance achieved by the proposed feature selection technique. The key metrics for comparison included accuracy, recall, precision, and F1 score. Detailed experimental results are presented in Table 5.

From the results in Table 5, it can be clearly observed that the use of the RF-P method for feature selection achieved superior performance. This is because the PCA method typically reduces the dimensionality by preserving features with larger variances in the data, but it sometimes overlooks those features with smaller variances that may contain important information, potentially affecting the overall performance of the model. On the other hand, the information gain method relies too heavily on the information gain between attributes for feature selection, which may result in incomplete feature selection and fail to capture all potentially useful information

Method	Edge-IIoTset				CIC-IDS2107				CIC IoT 2023			
	Acc	Rec	Pre	F1	Acc	Rec	Pre	F1	Acc	Rec	Pre	F1
PCA	0.923	0.923	0.916	0.919	0.896	0.896	0.930	0.906	0.935	0.935	0.933	0.922
IG	0.935	0.935	0.936	0.927	0.986	0.986	0.973	0.979	0.982	0.982	0.979	0.984
RF-P	0.947	0.947	0.948	0.946	0.993	0.993	0.993	0.993	0.995	0.995	0.995	0.995

Table 5. Evaluation metrics for feature selection methods.

for the model. In contrast, the RF-P algorithm used in this study comprehensively considers the correlation and importance of features, leading to a significant improvement in model performance and accuracy.

Comparison experiment between individual and hybrid models

The objective of this section is to verify whether the proposed NIDS-BAI model outperforms the individual use of its component modules in terms of performance. By conducting experiments on three datasets (Edge-IIoTset, CIC-IDS2017, and CIC IoT 2023), we independently evaluated two modules: BiGRU-Attention (abbreviated as BGA) and Inception-CNN (abbreviated as ICNN). We then compared their performance with that of the NIDS-BAI model to validate the performance enhancement of the NIDS-BAI model. The experimental results are presented in Table 6.

According to the evaluation results in Table 6, the proposed NIDS-BAI model significantly outperforms the BGA and Inception-CNN models in terms of performance. The remarkable improvement in the performance of the NIDS-BAI model can be attributed to its integration of the core advantages of the other two models. The BGA layer introduces an attention mechanism to process sequential data, enabling the model to focus on key parts in a targeted manner and thereby improving prediction accuracy. On the other hand, the ICNN layer employs parallel operations with multi-scale convolutional kernels, enhancing the understanding of key features and subsequently boosting the classification accuracy of the model. The NIDS-BAI model combines the characteristics of these two models, effectively leveraging their respective strengths, and achieves significant performance improvements.

Simulation experiment

In this section, a comprehensive evaluation of the NIDS-BAI model was conducted on the Edge-IIoTset, CIC-IDS2017, and CIC-IoT 2023 datasets. The evaluation primarily focused on key metrics, including training accuracy, training loss, precision, recall, F1 score, and ROC curve, to assess the performance of the NIDS-BAI model on these datasets.

For the Edge-IIoTset, CIC-IDS2017, and CIC-IoT 2023 datasets, the training accuracy of the NIDS-BAI model represents the proportion of correct predictions on the training set during the training process, while the training loss reflects the level of error during the training phase[27]. The validation accuracy and validation loss measure the predictive accuracy and error level of the model on the validation set. These metrics help to assess the model's learning ability and fit during the training process and serve as important references for further performance analysis. As shown in Fig. 12.

The training accuracy and loss curves for the Edge-IIoTset, CIC-IDS2017, and CIC-IoT 2023 datasets are shown in Fig. 12. From the curves, it can be observed that during the training process, the training accuracy of the NIDS-BAI model gradually increases while the training loss gradually decreases. Additionally, the validation accuracy and validation loss exhibit similar trends, indicating consistent performance of the model on both the training set and the validation set.

For instance, during the training process on the Edge-IIoTset dataset, the NIDS-BAI model achieved a training accuracy of 0.9467 and a training loss of 0.1262 at the 200th iteration. Simultaneously, the validation accuracy was 0.9476, and the validation loss was 0.1218. These results indicate that the model is capable of effectively learning the characteristics of this dataset and achieving a high accuracy on the test set.

During training on the CIC-IDS2017 dataset, the NIDS-BAI model demonstrated excellent training performance at the 200th iteration, achieving a training accuracy of 0.9915 and a training loss of 0.0369. Simultaneously, the validation accuracy was 0.9936, and the validation loss was 0.0314. These results indicate that the model is capable of classifying network traffic data in this dataset with high accuracy and low error rates.

On the CIC-IoT 2023 dataset, the NIDS-BAI model exhibited a training loss of 0.0099, a training accuracy of 0.9977, a validation loss of 0.0226, and a validation accuracy of 0.9953 during the final iteration of training. Although there were some fluctuations during the initial 0-45 iterations, significant improvements in accuracy

Model	Edge-IIoTset				CIC-IDS2107				CIC IoT 2023			
	Acc	Rec	Pre	F1	Acc	Rec	Pre	F1	Acc	Rec	Pre	F1
BGA	0.942	0.942	0.944	0.943	0.984	0.984	0.976	0.983	0.976	0.976	0.971	0.973
ICNN	0.918	0.918	0.942	0.916	0.937	0.937	0.931	0.927	0.882	0.882	0.883	0.879
NIDS-BAI	0.947	0.947	0.948	0.946	0.993	0.993	0.993	0.993	0.995	0.995	0.995	0.995

Table 6. Comparison between individual and hybrid models.

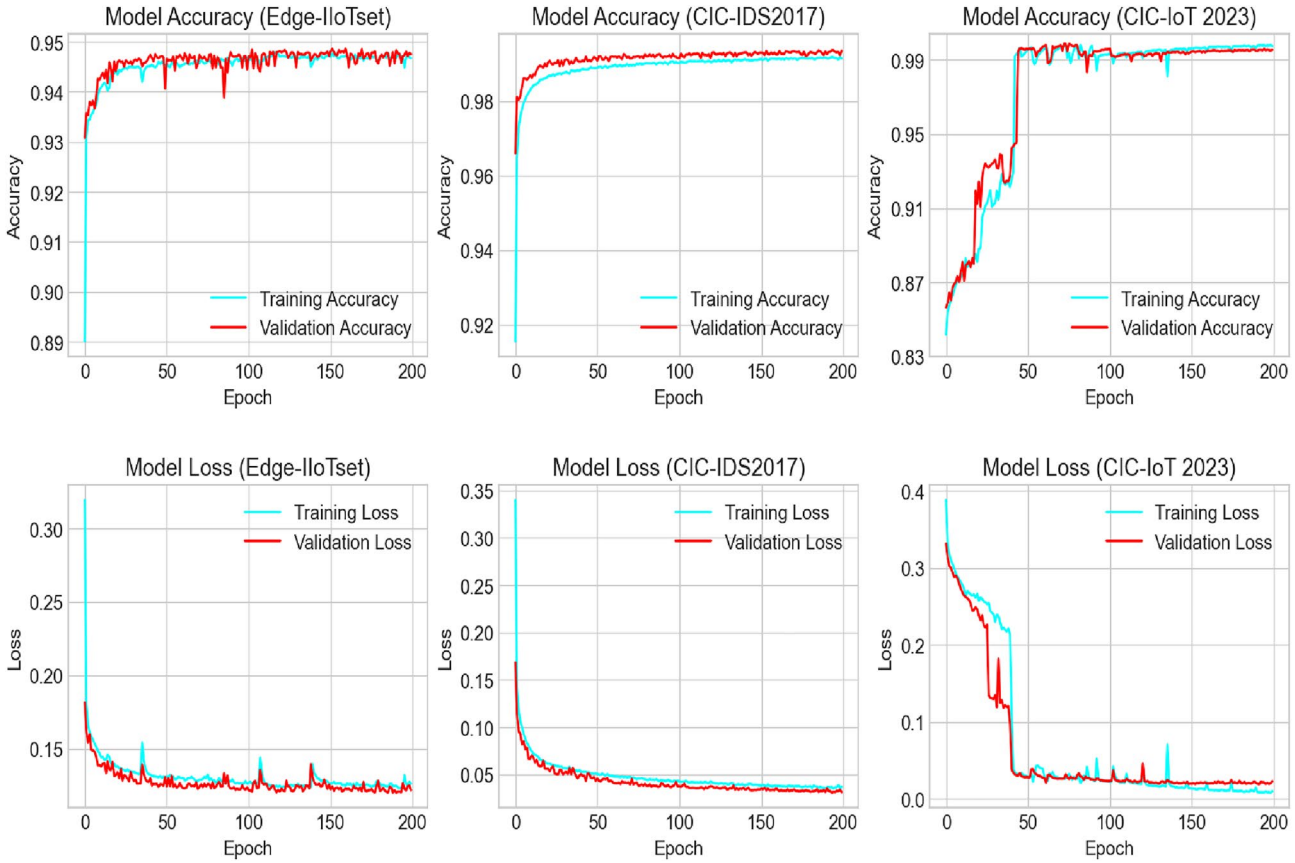

Figure 12. The training and validation loss and accuracy of the NIDS-BAI model on different datasets.

and loss were observed during the 45th to 47th iterations, indicating the model's ability to effectively learn and adapt to the unique characteristics of the CIC-IoT 2023 dataset.

In summary, the training accuracy and loss curves show that the proposed NIDS-BAI model can effectively learn the features of different network traffic datasets and achieve high accuracy in predicting malicious network traffic, making it a promising approach for intrusion detection in IoT networks.

To comprehensively evaluate the performance of the NIDS-BAI model on the Edge-IIoTset, CIC-IDS2017, and CIC-IoT 2023 datasets, this paper assesses key metrics including recall, precision, and F1 score for each type of network traffic in all three datasets. These metrics are crucial for a comprehensive assessment of the model's overall performance across different datasets[28]. For detailed evaluation results, please refer to Fig. 13.

According to Fig. 13, the NIDS-BAI model exhibits high precision, recall, and F1 scores for categories such as "Benign" and "MITM" in the Edge-IIoTset dataset, demonstrating its robust recognition capabilities. However, the model demonstrates lower precision, recall, and F1 scores for the "Injection" and "Scanning" categories, indicating that the NIDS-BAI model still faces challenges in accurately identifying certain attack traffic within this dataset.

In the CIC-IoT 2023 dataset, the NIDS-BAI model exhibits strong performance in categories such as "DoS," "DDoS," "Mirai," and "BruteForce." However, for the categories of "Spoofing" and "Recon," while the precision is relatively high, the recall and F1 scores are lower. The poorest performance is observed in the "Benign" category, primarily due to the high diversity of data within this category, which makes it challenging for the model to accurately identify this particular category.

In the CIC-IDS2017 dataset, the NIDS-BAI model demonstrates excellent performance in categories such as "DDoS," "DoS GoldenEye," "PortScan," "Bot," and "Heartbleed." However, the model exhibits poorer performance in categories like "Web Attack_XSS," "Web Attack_Sql Injection," and "Infiltration." These results indicate that the model requires improvement in these specific categories to enhance its overall performance.

To better assess the discriminative ability of the NIDS-BAI model, this paper introduces Receiver Operating Characteristic (ROC) curve analysis. The ROC curve illustrates the balance between TPR and FPR at different classification thresholds, providing deeper insights into the model's classification capability. By employing both macro-average and micro-average metrics, the evaluation in this paper becomes more comprehensive[29,30]. The ROC curve analysis offers a comprehensive perspective on the performance of the NIDS-BAI model, aiding in targeted improvements to meet the complex requirements of intrusion detection scenarios. Please refer to Fig. 14.

As shown in Fig. 14, the NIDS-BAI model demonstrates excellent performance on the Edge-IIoTset dataset, particularly in the " MITM" and "Benign" categories, with high AUC scores. Overall, the macro-average ROC curve area is 0.99277, and the micro-average ROC curve area is 0.99831.

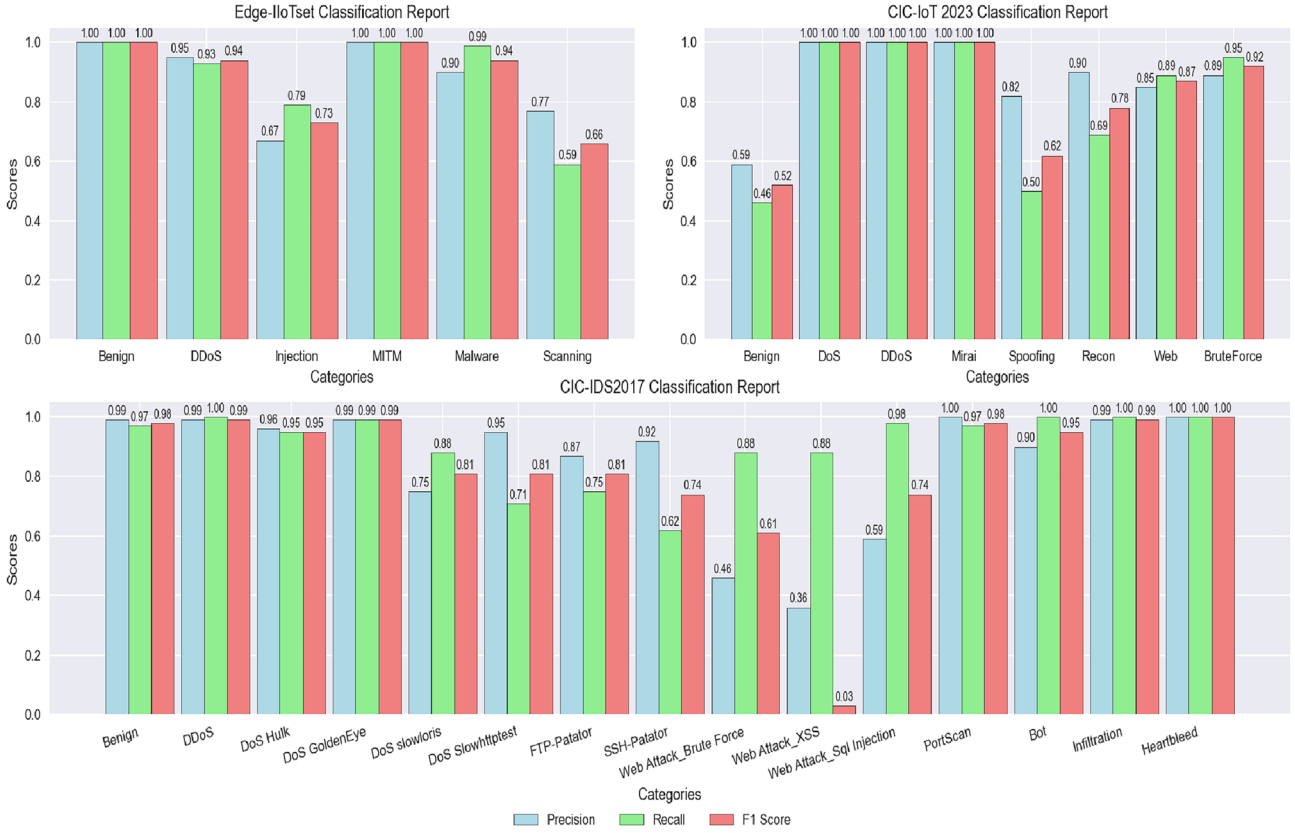

Figure 13. The performance of the NIDS-BAI model on each class across different datasets.

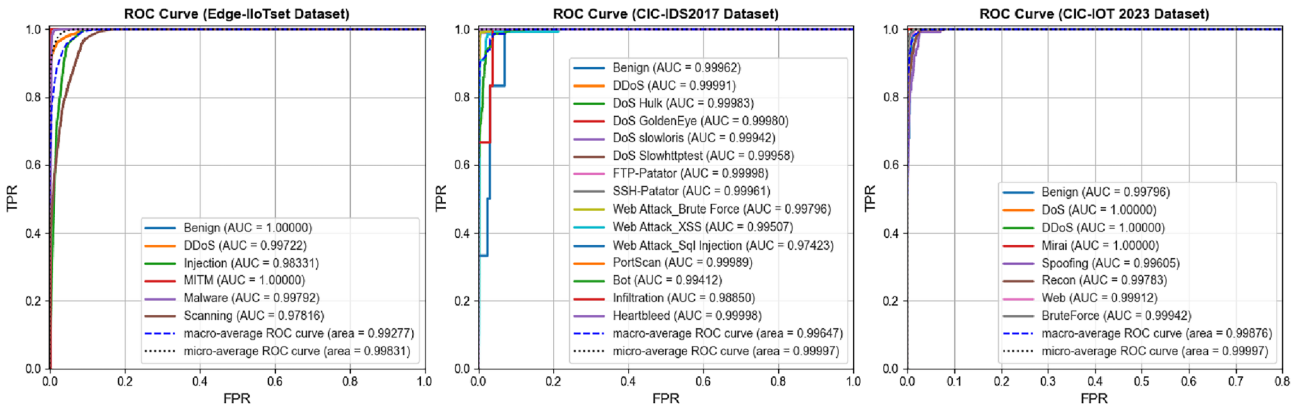

Figure 14. ROC analysis of the NIDS-BAI model.

For the CIC-IDS2017 dataset, most categories maintain high AUC scores, such as "FTP-Patator", "SSH-Patator", etc., but the AUC score for "Web Attack_Sql Injection" is relatively lower. Overall, the NIDS-BAI model achieves a macro-average ROC curve area of 0.99647 and a micro-average ROC curve area of 0.99997.

On the CIC-IoT 2023 dataset, the NIDS-BAI model performs excellently in identifying categories like "DoS", "DDoS", and "Mirai" (AUC=1.00000), but the AUC score for the "Spoofing" category is slightly lower. Overall, the macro-average ROC curve area is 0.99876, and the micro-average ROC curve area is 0.99997.

Based on the above, according to the display of training accuracy and loss curve, the NIDS-BAI model has demonstrated remarkable ability in effectively learning the characteristics of different network traffic datasets and achieved high accuracy in predicting malicious network traffic. This is attributed to the proposed RF-P algorithm, which can handle complex high-dimensional attack traffic and eliminate redundancy between features, enabling the model to accurately extract key information.

Analysis of key indicators such as recall, precision, and F1 score for each type of network traffic in the three datasets shows that the model performs well in identifying most categories[31]. This is because the ARL algorithm can learn each category more evenly, thereby improving the model's ability to recognize and classify categories with fewer instances. Although the recognition ability in certain categories still needs improvement, the use of

the ARL algorithm can enhance the model's robustness and generalization ability, ensuring its excellent performance across all categories.

Overall, the model proposed in this paper exhibits excellent macro-average and micro-average performance across the three datasets. These improvements in performance are attributed to the full utilization of the BiGRU network with attention mechanism and the Inception-CNN network, which jointly enhance the classification accuracy of the model.

Contrast experiment

To evaluate the feasibility of the NIDS-BAI model, we used the previously proposed methods on the Edge-IIoTset, CIC-IDS2107, and CIC IoT 2023 datasets to compare the performance of this model and other models previously proposed in terms of accuracy, recall, precision, and F1 score. The comparison results are shown in Table 7 below:

In the Edge-IIoTset dataset, the NIDS-BAI model achieved recall, precision, and F1 scores of 0.947, 0.948, and 0.946, respectively, with room for improvement in ACC. In the CIC-IDS2107 dataset, the overall performance of the DL-BiLSTM model was slightly higher. However, on the CIC IoT 2023 dataset, the NIDS-BAI model's overall performance significantly surpassed that of the DL-BiLSTM model.

The NIDS-BAI model demonstrates significantly superior robustness and generalization capabilities in complex network environments compared to other models, despite the fact that the DL-BiLSTM model performs slightly better in terms of overall metrics on the CIC-IDS2107 dataset, and the accuracy of the NIDS-BAI model is slightly lower than the DNN model on the Edge-IIoTset dataset[38]. This underscores the tremendous potential and practical value of the NIDS-BAI model in the field of IIOT intrusion detection.

Discussion

This article introduces a model named NIDS-BAI, which focuses on solving three core issues in IIoT network intrusion detection. Firstly, it solves the problem of low accuracy of the model in detecting attack traffic. Secondly, it addresses the issue of feature redundancy in high-dimensional and complex attack traffic. Finally, it corrects the model's tendency to learn categories with a larger number of instances, thereby enhancing its recognition and classification capabilities for categories with fewer instances.

Experimental results demonstrate that, in the context of IIoT intrusion detection, the feature selection method combining Pearson correlation coefficient with random forest significantly enhances performance. Random forest is effective in handling high-dimensional data and complex relationships. When combined with the Pearson correlation coefficient, it eliminates feature redundancy, further improving the accuracy of feature selection in the NIDS-BAI model and reducing the risk of overfitting.

Combining ADASYN, RENN, and LOF algorithms offers multiple advantages. These algorithms effectively address class imbalance and noise issues by applying ADASYN for oversampling and RENN for undersampling, thereby balancing the dataset and enhancing the model's capability to identify and classify minority classes. Additionally, the LOF algorithm efficiently identifies and eliminates noise, further improving the model's accuracy and generalization ability. Despite making some progress in handling data imbalance and noise, ARL methods still face challenges and limitations. Firstly, these techniques impact model training efficiency and computational requirements. Resampling and denoising processes necessitate additional computing resources, including storage space and computation time. Secondly, effectively tuning the parameters of ADASYN, RENN, and LOF algorithms to adapt to different datasets and scenarios, as well as dealing with extremely imbalanced data

Model	Edge-IIoTset				CIC-IDS2107				CIC IoT 2023			
	Acc	Rec	Pre	F1	Acc	Rec	Pre	F1	Acc	Rec	Pre	F1
SVM[32]	0.856	0.890	0.89	0.88	–	–	–	–	–	–	–	–
Knn[32]	0.839	0.845	0.875	0.851	–	–	–	–	–	–	–	–
DNN[32]	**0.960**	0.841	0.918	0.86	–	–	—-	–	–	–	–	–
LSTM	0.956	0.898	0.908	0.903	–	–	–	–	–	–	–	–
GRU	0.956	0.907	0.912	0.909	–	–	–	–	–	–	–	–
BiLSTM[33]	–	–	–	–	0.995	0.995	0.994	**0.995**	–	–	–	–
DL-BiLSTM[33]	–	–	–	–	**0.996**	**0.996**	**0.995**	**0.995**	–	–	–	–
SAE-RF-UDBB[34]	–	–	–	–	0.989	0.988	0.989	0.989	–	–	–	–
ANC[35]	–	–	–	–	0.982	0.987	0.991	–	–	–	–	–
GPC-FOS[36]	–	–	–	–	0.900	0.900	0.899	0.898	–	–	–	–
LR[37]	–	–	–	–	–	–	–	–	0.831	0.696	0.512	0.539
Adaboost[37]	–	–	–	–	–	–	–	–	0.351	0.487	0.464	0.368
DNN[37]	–	–	–	–	–	–	–	–	0.991	0.906	0.679	0.697
BiLSTM[33]	–	–	–	–	–	–	–	–	0.930	0.930	0.913	0.917
DL-BiLSTM[33]	–	–	–	–	–	–	–	–	0.931	0.931	0.918	0.919
NIDS-BAI	0.947	**0.947**	**0.948**	**0.946**	0.993	0.993	0.993	0.993	**0.995**	**0.995**	**0.995**	**0.995**

Table 7. Multi classification performance comparison. Significant values are in [bold].

distributions, remain unresolved issues in current research. In the future, optimizing algorithms and parameter settings can mitigate these additional costs while enhancing performance.

Finally, in IIoT intrusion detection, a combination of BiGRU network with attention mechanism and Inception-CNN network is employed. The BiGRU network is effective in capturing long-term dependencies in time-series data, making it suitable for handling temporal information in IIoT data. By introducing an attention mechanism, the BiGRU network focuses more on important features, effectively capturing complex temporal dependencies in the data and enhancing the model's accuracy in detecting intrusion behaviors. On the other hand, the Inception-CNN network extracts features of different scales in parallel, enhancing the model's understanding of features at different levels. Therefore, the NIDS-BAI model fully leverages the capabilities of the BiGRU network with attention mechanism and Inception-CNN network, collectively improving the model's ability to identify attacks in network traffic.

While this study did not extensively explore the robustness of the NIDS-BAI model against adversarial attacks, experimental results demonstrate its stable performance and high accuracy on training and validation sets. One of the important future directions is to investigate and evaluate the performance and robustness of the NIDS-BAI model against various forms of adversarial attacks. Adversarial attacks use carefully crafted input data to evade intrusion detection system detection, posing additional challenges to the security and practicality of the system.

The experiments and evaluations in this study primarily relied on specific datasets to validate the effectiveness and robustness of the NIDS-BAI model under specific protocols and attack scenarios. However, if the model overly depends on specific protocol features during design, its performance may be limited when applied to other protocols. For instance, different protocols may have varying data formats, transmission methods, or security requirements, necessitating customized models or parameter readjustments to adapt to new protocol environments.

Future research needs to delve deeper into the transferability and adaptability of the NIDS-BAI model across multiple IIoT protocols to ensure its practicality and reliability in widespread applications. This will involve comparative analysis of data characteristics among different protocols and research and experimental validation of model optimization and adaptation strategies for different protocol environments.

Data availability

The three datasets used in this study can be accessed through the following links: Edge-IIoTset: https://www.kaggle.com/datasets/mohamedamineferrag/edgeiiotset-cyber-security-dataset-of-iot-iiot. CIC-IDS2017: https://www.unb.ca/cic/datasets/ids-2017.html. CIC IoT 2023: https://www.unb.ca/cic/datasets/iotdataset-2023.html.

Received: 28 December 2023; Accepted: 13 August 2024
Published online: 20 August 2024

References

1. Ma, W., Zhang, Y., Guo, J. & Li, K. Unbalanced abnormal traffic detection based on improved res-bigru and integrated dynamic elm optimization. *Comput. Commun.* **179**, 112–130. https://doi.org/10.1016/j.comcom.2021.08.005 (2021).
2. Zhong, W., Yu, N. & Ai, C. Applying big data based deep learning system to intrusion detection. *Big Data Min. Anal.* **3**, 181–195. https://doi.org/10.26599/bdma.2020.9020003 (2020).
3. Ullah, I. & Mahmoud, Q. H. Design and development of a deep learning-based model for anomaly detection in IoT networks. *IEEE Access* **9**, 103906–103926. https://doi.org/10.1109/access.2021.3094024 (2021).
4. Park, C. *et al.* An enhanced AI-based network intrusion detection system using generative adversarial networks. *IEEE Internet Things J.* **10**, 2330–2345. https://doi.org/10.1109/jiot.2022.3211346 (2023).
5. Saheed, Y. K., Abdulganiyu, O. H. & Tchakoucht, T. A. Modified genetic algorithm and fine-tuned long short-term memory network for intrusion detection in the internet of things networks with edge capabilities. *Appl. Soft Comput.* **155**, 111434. https://doi.org/10.1016/j.asoc.2024.111434 (2024).
6. Saheed, Y. K. & Misra, S. A voting gray wolf optimizer-based ensemble learning models for intrusion detection in the internet of things. *Int. J. Inf. Secur.* **23**, 1557–1581. https://doi.org/10.1007/s10207-023-00803-x (2024).
7. Alzughaibi, S. & El Khediri, S. A cloud intrusion detection systems based on dnn using backpropagation and pso on the cse-cic-ids2018 dataset. *Appl. Sci.*https://doi.org/10.3390/app13042276 *(2023).*
8. Bahashwan, A. A. *et al.* A systematic literature review on machine learning and deep learning approaches for detecting ddos attacks in software-defined networking. *Sensors (Basel)*https://doi.org/10.3390/s23094441 *(2023).*
9. Fatani, A. *et al.* Enhancing intrusion detection systems for IoT and cloud environments using a growth optimizer algorithm and conventional neural networks. *Sensors (Basel).*https://doi.org/10.3390/s23094430 *(2023).*
10. Al-Imran, M. & Ripon, S. H. Network intrusion detection: An analytical assessment using deep learning and state-of-the-art machine learning models. *Int. J. Comput. Intell. Syst.*https://doi.org/10.1007/s44196-021-00047-4 *(2021).*
11. Halbouni, A. *et al.* CNN-LSTM: Hybrid deep neural network for network intrusion detection system. *IEEE Access* **10**, 99837–99849. https://doi.org/10.1109/access.2022.3206425 (2022).
12. He, J., Wang, X., Song, Y., Xiang, Q. & Chen, C. Network intrusion detection based on conditional wasserstein variational autoencoder with generative adversarial network and one-dimensional convolutional neural networks. *Appl. Intell.* **53**, 12416–12436. https://doi.org/10.1007/s10489-022-03995-2 (2022).
13. Hu, R., Wu, Z., Xu, Y. & Lai, T. Multi-attack and multi-classification intrusion detection for vehicle-mounted networks based on mosaic-coded convolutional neural network. *Sci. Rep.* **12**, 6295. https://doi.org/10.1038/s41598-022-10200-4 (2022).
14. Kim, T. & Pak, W. Early detection of network intrusions using a GAN-based one-class classifier. *IEEE Access* **10**, 119357–119367. https://doi.org/10.1109/access.2022.3221400 (2022).
15. Gautam, S. *et al.* A composite approach of intrusion detection systems: Hybrid RNN and correlation-based feature optimization. *Electronics*https://doi.org/10.3390/electronics11213529 *(2022).*
16. Nam, M., Park, S. & Kim, D. S. Intrusion detection method using bi-directional GPT for in-vehicle controller area networks. *IEEE Access* **9**, 124931–124944. https://doi.org/10.1109/access.2021.3110524 (2021).
17. Drewek-Ossowicka, A., Pietrołaj, M. & Rumiński, J. A survey of neural networks usage for intrusion detection systems. *J. Ambient. Intell. Humaniz. Comput.* **12**, 497–514. https://doi.org/10.1007/s12652-020-02014-x (2020).
18. Elsayed, R., Hamada, R., Hammoudeh, M., Abdalla, M. & Elsaid, S. A. A hierarchical deep learning-based intrusion detection architecture for clustered internet of things. *J. Sens. Actuator Netw.*https://doi.org/10.3390/jsan12010003 *(2022).*

19. Liao, P., Yan, J., Sellier, J. M. & Zhang, Y. Divergence-based transferability analysis for self-adaptive smart grid intrusion detection with transfer learning. *IEEE Access* **10**, 68807–68818. https://doi.org/10.1109/access.2022.3186328 (2022).
20. Liu, C., Yin, Y., Sun, Y. & Ersoy, O. K. Multi-scale resnet and bigru automatic sleep staging based on attention mechanism. *PLoS ONE* **17**, e0269500. https://doi.org/10.1371/journal.pone.0269500 (2022).
21. Liu, G. & Zhang, J. Cnid: Research of network intrusion detection based on convolutional neural network. *Discrete Dyn. Nat. Soc.* **1–11**, 2020. https://doi.org/10.1155/2020/4705982 (2020).
22. Kalpana, R. Recurrent nonsymmetric deep auto encoder approach for network intrusion detection system. *Meas. Sens.*https://doi.org/10.1016/j.measen.2022.100527 *(2022).*
23. Mananayaka, A. K. & Chung, S. S. Network intrusion detection with two-phased hybrid ensemble learning and automatic feature selection. *IEEE Access* **11**, 45154–45167. https://doi.org/10.1109/access.2023.3274474 (2023).
24. Mohammed, G. B., Shitharth, S. & Sucharitha, G. *A Novel Trust Evaluation and Reputation Data Management Based Security System Model for Mobile Edge Computing Network, 155–170* (Springer International Publishing, Cham, 2023).
25. Seo, S. *et al.* Hunt for unseen intrusion: Multi-head self-attention neural detector. *IEEE Access* **9**, 129635–129647. https://doi.org/10.1109/access.2021.3113124 (2021).
26. Thirimanne, S. P., Jayawardana, L., Yasakethu, L., Liyanaarachchi, P. & Hewage, C. Deep neural network based real-time intrusion detection system. *SN Comput. Sci.*https://doi.org/10.1007/s42979-022-01031-1 *(2022).*
27. Udas, P. B., Karim, M. E. & Roy, K. S. Spider: A shallow PCA based network intrusion detection system with enhanced recurrent neural networks. *J. King Saud Univ. Comput. Inf. Sci.* **34**, 10246–10272. https://doi.org/10.1016/j.jksuci.2022.10.019 (2022).
28. Wang, Z., Zeng, Y., Liu, Y. & Li, D. Deep belief network integrating improved kernel-based extreme learning machine for network intrusion detection. *IEEE Access* **9**, 16062–16091. https://doi.org/10.1109/access.2021.3051074 (2021).
29. Xing, L., Wang, K., Wu, H., Ma, H. & Zhang, X. Fl-maae: An intrusion detection method for the internet of vehicles based on federated learning and memory-augmented autoencoder. *Electronics*https://doi.org/10.3390/electronics12102284 *(2023).*
30. Wang, W. *et al.* Anomaly detection of industrial control systems based on transfer learning. *Tsinghua Sci. Technol.* 821–832 (2020).
31. Yu, H., Kang, C., Xiao, Y. & Ting, Y. Network intrusion detection method based on hybrid improved residual network blocks and bidirectional gated recurrent units. *IEEE Access*https://doi.org/10.1109/access.2023.3271866 *(2023).*
32. Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L. & Janicke, H. Edge-iiotset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access* **10**, 40281–40306. https://doi.org/10.1109/ACCESS.2022.3165809 (2022).
33. Wang, Z. *et al.* A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization. *Peer J. Comput. Sci.* **9**, e1569. https://doi.org/10.7717/peerj-cs.1569 (2023).
34. Le Jeune, L., Goedeme, T. & Mentens, N. Machine learning for misuse-based network intrusion detection: Overview, unified evaluation and feature choice comparison framework. *IEEE Access* **9**, 63995–64015. https://doi.org/10.1109/access.2021.3075066 (2021).
35. Pham-Quoc, C., Bao, T. H. Q. & Thinh, T. N. Fpga/ai-powered architecture for anomaly network intrusion detection systems. *Electronics*https://doi.org/10.3390/electronics12030668 *(2023).*
36. Shyaa, M. A. *et al.* Enhanced intrusion detection with data stream classification and concept drift guided by the incremental learning genetic programming combiner. *Sensors (Basel).*https://doi.org/10.3390/s23073736 *(2023).*
37. Neto, E. C. P. *et al.* A real-time dataset and benchmark for large-scale attacks in iot environment. *Sensors*https://doi.org/10.3390/s23135941 *(2023).*
38. Yao, R., Wang, N., Liu, Z., Chen, P. & Sheng, X. Intrusion detection system in the advanced metering infrastructure: A cross-layer feature-fusion cnn-lstm-based approach. *Sensors (Basel).*https://doi.org/10.3390/s21020626 *(2021).*

Acknowledgements

We sincerely appreciate the financial support provided by Xijing University's High-level Talents Special Fund Project in 2022 for the research on 'Industrial Internet of Things Intrusion Detection Technology Based on Deep Learning' (Project XJ22B04).

Author contributions

All authors made significant contributions to this work. K.Y. was involved in the conception and design of the study. J.M.W. conducted experiments and contributed to writing, while M.J.L. was responsible for interpreting experimental results. All authors participated in the review and approval of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to J.W.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.