International Journal on Semantic Web and Information Systems
Volume 20 • Issue 1

A Network Intrusion Detection Method for Various Information Systems Based on Federated and Deep Learning

Qi Zhou, School of Artificial Intelligence, Guangdong Open University, Guangzhou, China*
Chun Shi, School of Electronic and Information, Guangdong Polytechnic Normal University, Guangzhou, China

ABSTRACT

Under the premise of ensuring data privacy, traditional network intrusion detection (NID) methods cannot achieve high accuracy for different types of intrusions. A NID method combining transformer and federated learning (FedL) is proposed for this purpose. First, a multi-party collaborative learning framework was built based on FedL, which achieved data exchange and sharing. Then, by introducing the self-attention mechanism (AttM) to improve the traditional transformer, it could quickly converge. Finally, an NID model integrating transformer and FedL was constructed by combining DNN, GRU, and an encoder module composed of improved transformer, achieving accurate detection of network intrusion. The proposed NID method was compared with the other three methods. The results show that the proposed method has the highest NID accuracy and F1 score on the NSL-KDD and UNSW-NB15 dataset, with the highest accuracy reaching 99.65% and 89.25%, while the F1 score has the highest accuracy, reaching 99.45% and 88.13%, outperforming the other three comparative algorithms in terms of performance.

KEYWORDS

DNN, Federated Learning, GRU, Network Intrusion Detection, Transformer

INTRODUCTION

In recent years, the large-scale popularization and rapid development of the internet have brought great convenience to the development of enterprises and personal lives, followed by a series of network security issues and challenges (Hamad et al., 2020; Wang, et al., 2021; Fan et al., 2020). Bad actors often exploit network vulnerabilities, Trojan viruses, and other means to steal confidential information and valuable personal information. Network attacks can have wide coverage and endanger many areas of public production and security, causing huge burdens and losses. Research shows that sudden network attacks reveal that the existing basic network security protection technologies cannot flexibly adapt to resist complex network attacks. Therefore, there is an urgent need to propose network security technologies to address network security threats (Wang et al., 2020; Park et al., 2020; Wang et al., 2022).

DOI: 10.4018/IJSWIS.335495 *Corresponding Author

At present, the complexity of network threats is increasing, and the means of attack are becoming increasingly diverse, making the security protection of network systems particularly important. Firewalls, spam filters, and antivirus software are all tools used to protect network security. But currently the most widely used and powerful network security technology is NID systems. It is the most crucial link in the attack defense chain and can be used as the first or second defense mechanism for threats or attacks (Liu et al., 2020; Qin et al., 2020; Sharafaldin et al., 2021). The ultimate goal of the NID system is to quickly and accurately detect different types of attacks that may occur in the network, such as denial of service, port scanning, malware, distributed denial of service, or ransomware, by investigating network traffic (Meidan et al., 2022; Lin et al., 2021; Sattler et al., 2022).

The goal of NID is to detect abnormal behavior that damages the host as much as possible without interfering with the normal use of the network. The key to implementing NID is to find an effective detection algorithm to analyze network traffic (Yang et al., 2022; Cheng et al., 2020; Cheng et al., 2021).

Traditional machine learning (ML) technology has been proven to effectively identify important patterns in Internet of things (IoT) traffic, thus effectively targeting attacks. At present, the public is becoming increasingly sensitive to data privacy, and there is a risk of privacy leakage during data transmission (Zhu, et al., 2023; Cui, et al., 2023; de Caldas et al., 2023). Therefore, the data collected and transmitted from the device will be subject to legal regulatory limitations, which may lead to deviations in NID's task results. The delay generated by training based on global data and returning the results to edge nodes is relatively large, which is unacceptable for some delay sensitive applications (Ling & Hao 2022; Ling & Hao., 2022; Tembhurne, et al., 2022).

In 2016, Google proposed a distributed ML framework called FedL that can protect privacy, which is used to protect user privacy and information security during data exchange. FedL provides a collaborative and secure learning protocol that enables efficient learning among multiple participants while ensuring legal compliance (Srivastava, et al., 2022; Rahman, et al., 2020). Under this framework, each edge device can contribute to global model training while retaining the training data locally. In the FedL environment, edge devices typically collect sensing data from IoT nodes, typically time series data, and capture the behavior and operational status of IoT nodes through computational analysis (Mourad, et al., 2020; Abbas, et al., 2021).

This article solves the issues of sensitive information protection and incomplete data in training data by applying transformer and FedL to NID and improves the accuracy of NID. Compared with traditional methods, the proposed method provides the following innovations:

1. It utilizes sparse stacked autoencoders for feature dimensionality reduction and extracting deep level features of traffic using DNN. Using GRU to extract temporal features of traffic, the two feature maps are combined to ensure the comprehensiveness of the extracted features.
2. By introducing self AttM, the traditional transformer network has been improved to achieve fast convergence under massive computational data.
3. A multi-party collaborative learning framework was built based on FedL, and a NID model was constructed by combining DNN module, GRU module, and encoder module composed of improved transformer.

RELATED WORK

NID

The NID system is an important component of network security research. It detects intrusion behavior through proactive defense technology and takes emergency measures such as alerting and terminating the intrusion. Therefore, with the rapid development of learning technology, people have developed

various methods based on ML and deep learning. Researchers and scholars from foreign research institutions began to work on the application of deep learning in NID more than a decade ago.

By using convolution to collect local features of network traffic and extracting features using deep recurrent neural networks, Qazi and Zia (2023) constructed a hybrid NID system based on convolutional recurrent neural network architecture (Pan, et al., 2022). A white hat worm launcher suitable for defense systems in large-scale IoT networks has been proposed using the basic principles of machine learning. This launcher can effectively launch the white hat worm and deploy it to large-scale IoT networks using a divide and conquer algorithm to protect IoT systems from malicious attacks. However, due to the similarity between the communication mode of the white hat worm and normal devices when scanning and identifying malicious software, it may be misjudged as malicious software, causing unnecessary interference or damage to normal devices. In response to the attacks on vehicle networks, Alkhatib et al. (2022) compared and analyzed the real-time NID performance of different unsupervised deep learning and ML based NID algorithms on ethernet vehicle networks and provided their advantages and disadvantages. However, no new methods have been proposed that can effectively improve NID performance. By treating network traffic as vibrations, waves, or sounds, Aldarwbi et al. (2022) transformed network traffic features into waveforms, and based on this, implemented NID using deep learning technology based on audio/speech recognition, proposing a new NID system called "Voice of Intrusion." In response to the problem of signature recognition, Chellammal et al. (2023) proposed a deep learning based NID model by combining multiple hidden processing layers in the network to form a multi-layer deep learning network. However, this method only focuses on signature recognition of complex intrusion features involved in the NID process and does not fundamentally provide a path for NID (Takeda, 2022). In response to the increasing burden of intrusion attacks on servers, the reasons for the difficulty in implementing NID were analyzed, and a deep neural network for NID systems was proposed. In response to the complexity problem caused by network interactions when network traffic encounters known or unknown intrusions in the network, Farooq et al. (2023) proposed a NID scheme that integrates ML technology: IDS-FMLT, which achieves heterogeneous NID for different source network compositions. By combining reinforcement learning based on Q-learning with deep feedforward neural network methods, utilizing deep Q-learning to provide continuous automatic learning capabilities, a new NID method is proposed (Alavizadeh et al., 2022), which achieves intrusion detection for different types of networks through automatic trial and error. Although the above methods can improve the accuracy of NID to some extent compared to traditional ML based NID methods, they are all based on fixed or dynamic rules to identify attacks on the network. However, currently attackers use various techniques to disguise their attacks and disrupt the target's defense system. The intrusion detection system constructed using the above methods is no longer able to withstand the increasingly complex and diverse network threats.

FedL-Based NID

In the FedL architecture, FedL servers are unable to access data and control the behavior of edge devices. Therefore, edge devices may intentionally or unintentionally deviate from the prescribed joint training process. Scholars have conducted relevant research on the application of FedL in NID. Bai et al. (2022) proposed a NID mechanism based on FedL and NID models, and improved the effectiveness of the model through functional programming, requirement analysis, and design. In response to the problem of insufficient samples and probability adaptation, Wang et al. (2022) established a transfer limit learning machine model FLTrELM and implemented data aggregation using FedL. Combined with the transfer learning mechanism, a NID algorithm based on federated transfer learning was proposed (Novikova et al., 2022). A method for developing a NID system for vertically partitioned data based on the principle of joint learning was developed, and on this basis, the safe water treatment dataset SWaT was used to model the vertically partitioned data. In response to the security needs

of edge assisted IoT, Rahman et al. (2022) proposed an intelligent intrusion detection mechanism FedACNN that can effectively complete NID tasks by utilizing the FedL mechanism to assist CNN models and integrating AttM (Zhang, L & Zhang, J. H., 2022). A new detection model for artificial immune detection problems based on multi operator co- evolution theory has been proposed. By effectively simplifying and classifying high-dimensional features, and combining vaccination strategies and multi operator co- evolution, accurate detection has been achieved. However, the applicability of this method in other fields still needs further verification (Sarhan et al., 2022). A hierarchical blockchain based joint learning framework was proposed by running transaction processes on a secure blockchain. Based on this, a hierarchical joint learning architecture was utilized to ensure the privacy of the learning process and organizational data, achieving a secure and privacy collaborative IoT NID. While these research methods all have superior performance in certain aspects, further improvement is still needed to meet the urgent needs of network security (Mishra et al., 2022). A new set of classifiers has been proposed to address the issue of significant network bandwidth and server losses caused by DDoS attacks. The theoretical basis of this aggregator is the majority voting principle, which can reduce related losses while improving the accuracy of DDoS attack detection. However, this method has only been studied for DDoS attacks, and its effectiveness against different types of network attacks still needs to be verified.

Solution Strategies

Traditional network intrusion detection methods face various challenges, especially in terms of data privacy and accuracy in detecting various intrusions, as follows:

1. **Data privacy issues:** Traditional network intrusion detection methods typically require deep analysis of network traffic in order to detect malicious behavior. However, this method requires the collection and processing of a large amount of network traffic data, which may lead to the leakage of sensitive information, thereby causing data privacy issues. With the increasing strictness of network security and data protection regulations, effective intrusion detection has become an urgent problem to be solved while protecting user privacy.
2. **Accuracy issue in detection:** Traditional network intrusion detection methods often use rule-based or feature-based methods for detection. However, with the continuous upgrading and complexity of attacker methods, these methods may not be able to accurately identify new or unknown intrusion behaviors. In addition, false positives and false negatives are also serious issues faced by traditional methods, which may lead to inefficient security teams in dealing with real threats.
3. **The processing capability of large-scale networks:** With the continuous expansion of network scale, traditional network intrusion detection methods may not be able to maintain high detection accuracy while processing large amounts of network traffic data in real-time. This may lead to a decrease in the performance of the detection system, and even fail to detect potential intrusion behaviors in a timely manner.
4. **Complex network environment and diversified attack means:** Modern network environments have become more and more complex, including the wide application of new technologies such as cloud computing, IoT and edge computing. At the same time, attackers are constantly innovating and evolving their attack methods, such as using encrypted communication, polymorphic code, and social engineering. These new trends have brought greater challenges to traditional network intrusion detection methods.
5. **Response to advanced persistent threats (APT):** APT is a long-term, highly covert network attack aimed at stealing sensitive information or damaging target systems. Traditional network intrusion detection methods may be difficult to effectively respond to APT attacks, as they can often bypass conventional security defense measures and lurk in the target network for a long time.

FedL is a distributed ML with privacy protection function. Under the coordination of a central server, multiple users collaborate to solve ML problems, and each user's original private dataset is stored locally. This effectively solves the problem of current NID technology being unable to effectively protect data privacy and having high communication costs. Therefore, by applying transformer and FedL to NID, the issues of sensitive information protection and incomplete data in training data are solved, and the accuracy of NID is improved.

FedL

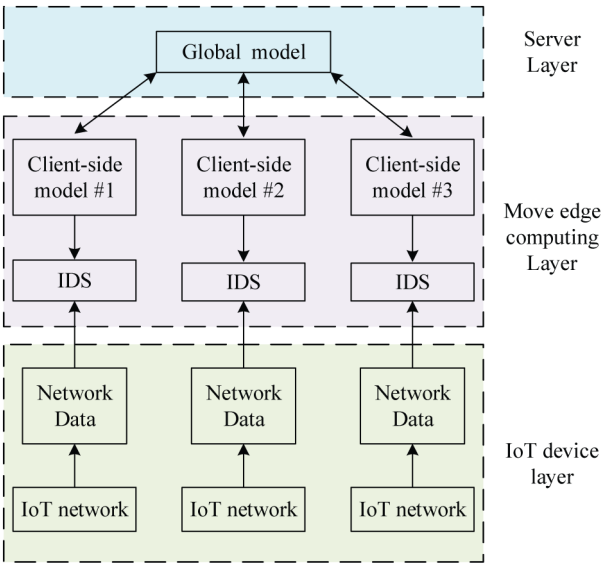
FedL is a multi-party collaborative learning framework that can essentially be seen as an ML framework. Participants can be various enterprises, individuals, or devices, each with independent and equal status and possessing their own local data and models. FedL brings various enterprises or users together to establish models and improve model performance, but data is not shared, and data privacy is protected to a certain extent. Compared to distributed ML, FedL is a better technology. Distributed ML only trains data on multiple machines together, with data exchange and sharing, and cannot guarantee privacy.

FedL is defined from the perspective of model training performance. If there are M users $U_1, U_2, ..., U_M$, each user has their own network data $D_1, D_2, ..., D_M$. FedL users can jointly train a model M_F, represented by P_F. When the following eq (1) is satisfied, it indicates that the accuracy loss of the FedL algorithm is η:

$$\left| P_F - P_S \right| < \eta \tag{1}$$

Due to privacy protection reasons, accuracy loss is allowed. FedL is a framework based on distributed ML. The main idea of FedL is to enable edge devices (referred to as clients) that can store a large amount of local data for various calculations and applications to collaborate on training global ML models on the device side using the generated data, without the need to share their original data to obtain the training model. The typical architecture of FedL is shown in Figure 1.

Figure 1. Typical architecture of FedL



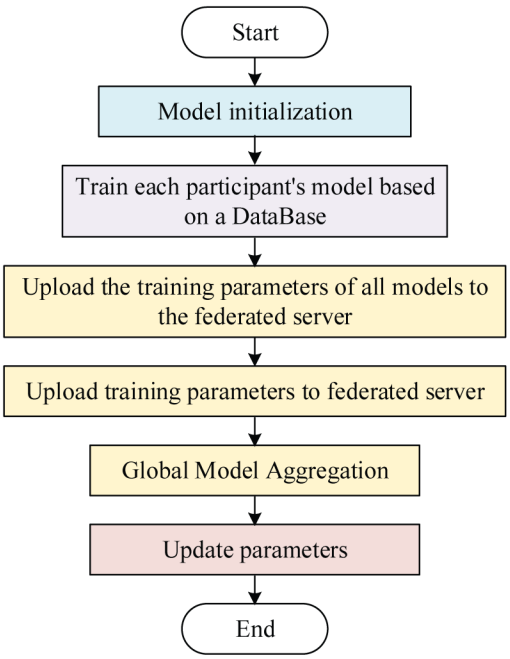
In Fig. 1, the coordination server (usually referred to as the parameter server) will summarize the model parameters of customers, derive and update the global model, and share the model with participating customers. The edge nodes of the updated global model will continue to utilize local data for model updates in subsequent iterations, and the subsequent server nodes will also benefit from the learning experience of the client nodes. The basic process of FedL is as follows:

1. In the initial stage, the central server confirms the task and objectives of model training, initializes global model parameters, and broadcasts them to all participating nodes.
2. The client updates the local model based on locally generated IoT data. Edge devices typically have a certain level of computing resources (such as laptops equipped with reasonable CPUs, photography devices with moderate computing power), and the FedL protocol allows clients to securely store their data in local storage.
3. The client uploads the trained model parameters to the server.
4. Multiple client local models are selected based on the node selection algorithm for global updates to further improve the model.
5. On the server side, the parameters of the client node are selected and the global model parameters are updated using the global model aggregation algorithm.
6. The updated global model parameters are transferred to the client node and the iteration process is continued until global convergence is achieved.

The training process of FedL is shown in Figure 2.

1. **Model initialization:** The server determines the hyperparameters of the global model structure and training process, such as the number of participants, initial weights, etc. Then, the server broadcasts the initialized global model and tasks to each participant.

Figure 2. Training process of FedL



2. **Local model training and updating:** According to the global model after the t-th iteration, each participant uses their local data and computer to update the local model parameters separately. The goal of the participants is to find the optimal parameters of the local model to minimize the loss function. The updated local model parameters are then sent to the server.
3. **Global model aggregation and updates:** The server aggregates the local model of the participants and sends the updated global model parameters back to the data owner. Steps (2) to (3) are repeated until the global loss function converges or reaches the required training accuracy (i.e., the loss function) reaches the minimum.

From the above analysis, it can be seen that building a network intrusion detection model based on federated learning can effectively solve the problems of weak data privacy protection and high communication costs.

Federated learning does not transmit data but uses data for local training and submits model parameters for local training. This means there is no risk of data privacy leakage, and since the transmission cost of model parameters is much lower than the data transmission cost, its communication overhead will also be greatly reduced.

PROPOSED FEDL-BASED NID METHOD

NID Model

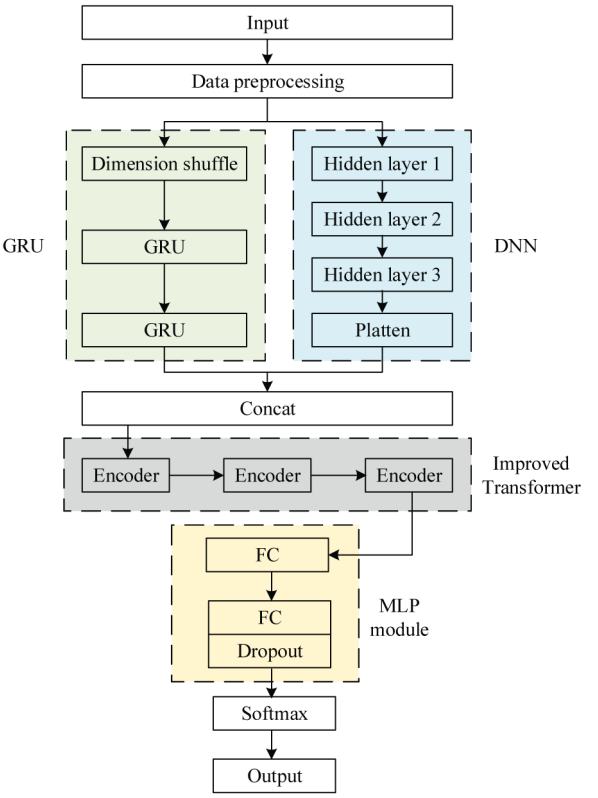
The NID model mainly consists of a DNN module, GRU module, and encoder module composed of an improved transformer. Next is the multi-layer perceptron (MLP) module, and finally the softmax layer. The model structure is shown in Figure 3.

The DNN module mainly consists of three hidden layers, each consisting of a convolutional layer, a batch normalization layer, and a max pooling layer. The GRU module consists of two identical GRU layers. Given a feature vector (a one-dimensional vector representing network traffic data) as input to the model, the GRU module and DNN module process feature vector separately. The GRU module treats feature vectors as multivariate time series with a single time step. Before transferring the feature vector to the GRU module, the time dimension of the feature vector needs to be transposed. The DNN module treats the feature vector as a univariate time series with multiple time steps, passes through three hidden layers in sequence, and finally flattens the output feature map. After the DNN module and GRU module, the feature maps output by each module are concatenated and sent to the encoder module. The MLP module includes two fully connected layers and one dropout layer. The dropout layer is used to prevent overfitting of the model. Afterwards, the output is limited to 0-1 through the softmax layer.

From Figure 3, it can be seen that this method can be used to reduce the likelihood of data leakage and other types of network attacks. This is mainly reflected in the following two aspects:

1. The NID method provided utilizes federated learning to unite various enterprises or users to jointly establish models and improve model performance. However, during the data transmission process, the data is not shared. Instead, local training is conducted on the local model, and the trained model parameters are ultimately uploaded. Therefore, data privacy can be greatly protected, reducing the risk of data leakage.
2. The network intrusion detection model includes a DNN module, a gated recurrent unit (GRU) module, and an encoder module composed of an improved transformer. Given the feature vector x as the input of the model, it will be processed by the GRU module and DNN module respectively, which will greatly reduce the possibility of other types of network attacks.

Figure 3. Model of NID



Data Pre-Processing

Because data can only be trained and tested using numerical values when using DNN for classification, and the dataset contains many different data types, it is necessary to pre-process the initial data. The pre-processing process is mainly divided into the following two steps:

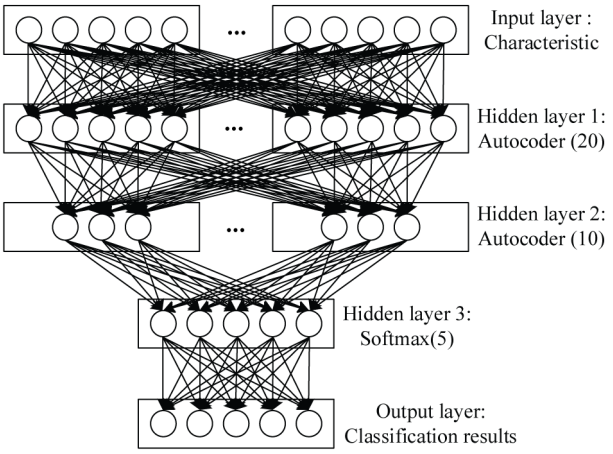
1. **Symbol data preprocessing:** Converts non numerical data into numerical data. Non- numerical data typically exists in datasets, and assigning specific values to each variable can convert these features from the training and testing datasets into numerical types. During pre-processing, the attack class and normal class of the dataset are converted into digital classes, such as assigning 1, 2, 3, and 4 to DoS, Probe, R2L, U2R, and 5 to normal.
2. **Data normalization:** The features of a dataset are generally discrete or continuous values, and if the range of feature values is different, they cannot be directly compared. By using minimum maximum normalization to process these features, all different values of each feature are mapped to the range of [0,1].

DNN for Deep Feature Extraction

The universal model is the core module of the FedL based NID framework, where DNN is selected as the universal model for the framework, as shown in Figure 4.

In the DNN model shown in Fig. 4, the input layer uses data features from the dataset as neurons. The first hidden layer is an automatic encoder containing 20 neurons, which selects 20 features from the input data features; the second hidden layer is an automatic encoder containing 10 neurons; the

Figure 4. DNN

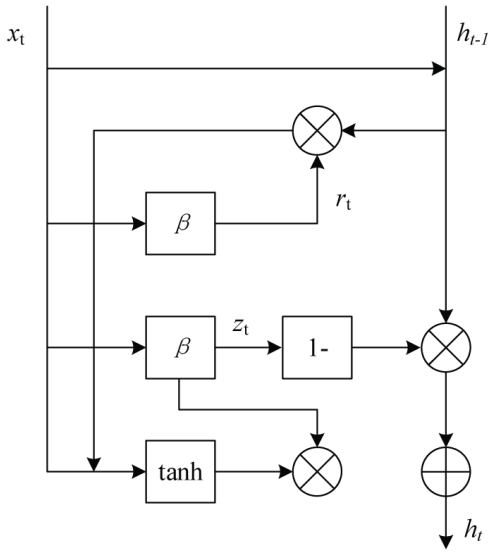


third hidden layer is the softmax layer, through which data can be divided into normal classes and other different attack classes.

GRU for Temporal Feature Extraction

GRU has been proven to be an effective LSTM variant in various application fields, with its structure being a simplified and enhanced version of LSTM. The input gate, forgetting gate, and output gate in LSTM structure are replaced with the reset gate and update gate. The gate is reset to calculate whether to forget the previous calculation state, and the gate is updated to determine how much information will be iterated from the previous step to the current step. The GRU structure is shown in Figure 5.

Figure 5. GRU unit structure



The main calculation formula for GRU structure is as follows:

$$r_t = \beta\left(\omega_r \cdot \left[h_{t-1}, x_t\right]\right) \tag{1}$$

$$z_t = \beta\left(\omega_z \cdot \left[h_{t-1}, x_t\right]\right) \tag{2}$$

$$h_t = \left(1 - z_t\right) \cdot h_{t-1} + z_t \tilde{h}_t \tag{3}$$

$$\tilde{h}_t = \tanh\left(\omega \cdot \left[r_t \cdot h_{t-1}, x_t\right]\right) \tag{4}$$

where, ω represents the weight matrix. r_t and z_t represent the output of reset gate and update gate at time t. h_t and $\tilde{h}_t$ represent the state information and candidate state information at time t. The expanded view of GRU is shown in Figure 6.

Improved Transformer

The transformer network model mainly consists of two parts: encoder and decoder. The encoder and decoder are composed of stacked layers of AttM modules, which are composed of multiple AttM layers and feedforward layers. Transformer abandons the traditional ideas of recurrent neural networks and convolutional neural networks, avoiding the model structure of loops. Therefore, the entire network structure is completely dependent on the global dependency of AttM on input and output, breaking through the limitation that recurrent neural network models cannot perform parallel calculations.

A highly interpretable model facilitates the calculation of the correlation between two positions, due to the high computational complexity, long training time, and difficulty in model convergence of traditional serialized temporal neural networks. To address this issue, self AttM was introduced. In order to adapt the transformer to NID data one-dimensional signal data and achieve multi-layer stacking, modifications have been made to the transformer. The model structure is shown in Figure 7.

Figure 6. GRU unit expanded view

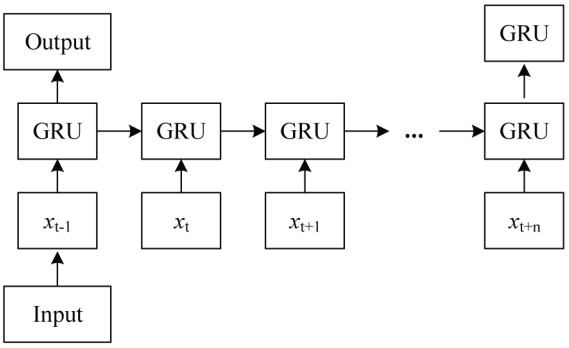
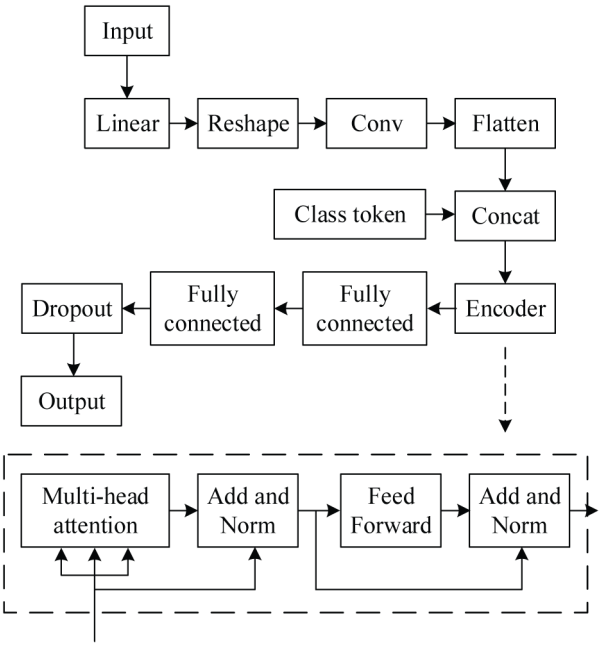


Figure 7. Transformer based on AttM



The encoder structure was selected and adjusted in Figure 7. First, the one-dimensional input data is processed through a fully connected layer, and its output is converted into a two-dimensional matrix of 14×14. A convolution with a scale of 2×2, a step size of 2, and a channel number of 32 is used to extract features from the input data. This convolution is equivalent to dividing a two-dimensional matrix into independent small slices of 7×7, each with a feature dimension of 32, and then flattening and spreading them out. A category encoding is randomly initialized as the category encoding information for a certain data, and then channel concatenation is performed with slice information. Through the above processing, the data can be input into the transformer module. After passing through the encoder module of the Transformer, the data dimension is 50×32. Among them, 1×32 is the category information predicted by the network, and 49×32 is the information for each small slice. Subsequently, as long as the dimension of category information is extracted separately, the network prediction results can be obtained through the fully connected layer (1×32 converted to 1×2).

The improvement of traditional transformers through AttM contributes to the performance of the model as follows:

1. Contextual information is captured through AttM. AttM allows the model to consider the interrelationships between all elements in the input sequence, thereby capturing richer contextual information. By parallel computing multiple self-attention heads, the model's ability to capture different types of contextual information can be enhanced.
2. The introduction of AttM makes the transformer model more flexible and adaptable. It can handle variable length input sequences and better handle contextual information of different lengths.
3. The weight of AttM can be interpreted as the correlation between different elements in the input sequence, which helps to improve the interpretability of the model. By visualizing self-attention weights, it is possible to better understand how the model processes input sequences at different levels.

4. The introduction of AttM optimizes the training process of the transformer model. Through parallel computing and distributed training, the training speed and efficiency of models can be improved, thereby accelerating the development and application of deep learning models.

NID That Integrates Transformer and FedL

The integration of transformer and federated learning has great practical significance for network intrusion:

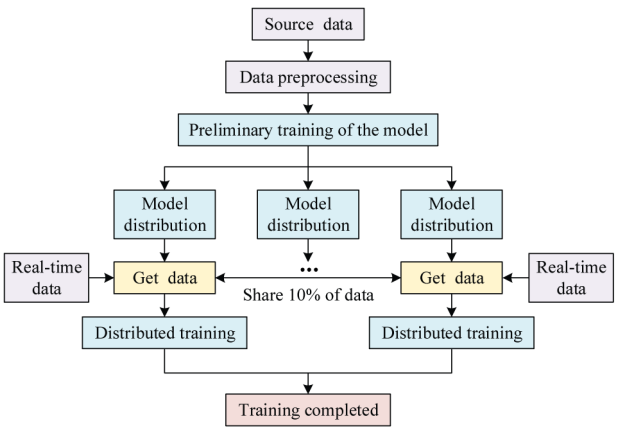
1. **Implementing more effective data privacy protection:** Federated learning is a distributed machine learning method that can train models without sharing data. By combining transformer with federated learning, more accurate and effective model training can be achieved while maintaining data privacy.
2. **Improving the robustness of the model:** The transformer model is susceptible to adversarial attacks. Federated learning can provide an integrated solution, which can increase the robustness of the model and reduce the risk of adversarial attacks by training on distributed datasets.
3. **Expanding application scope:** Transformer and federated learning each have different application scenarios. Integrating them can expand their respective application scope and achieve more complex and extensive application scenarios. For example, transformer models can be used to process text data, while federated learning can be used for model training on distributed datasets.
4. **Enhance model performance:** By integrating transformer and federated learning, the advantages of both can be combined to enhance model performance. For example, transformer models can provide powerful modeling capabilities, while federated learning can provide more flexible and effective data processing methods. Combining the two can achieve more accurate and efficient model training and inference.

The NID architecture that integrates transformer and FedL is shown in Figure 8.

In Figure 8, the system is a combination of centralized and distributed operating systems. The training of the model is divided into two stages, with the first stage completed in the base station and the second stage completed in each node. The specific steps for model training are as follows:

Step 1: In the base station, preliminary training on the model using a known labeled source dataset is performed, which is the first stage of federated transfer learning.

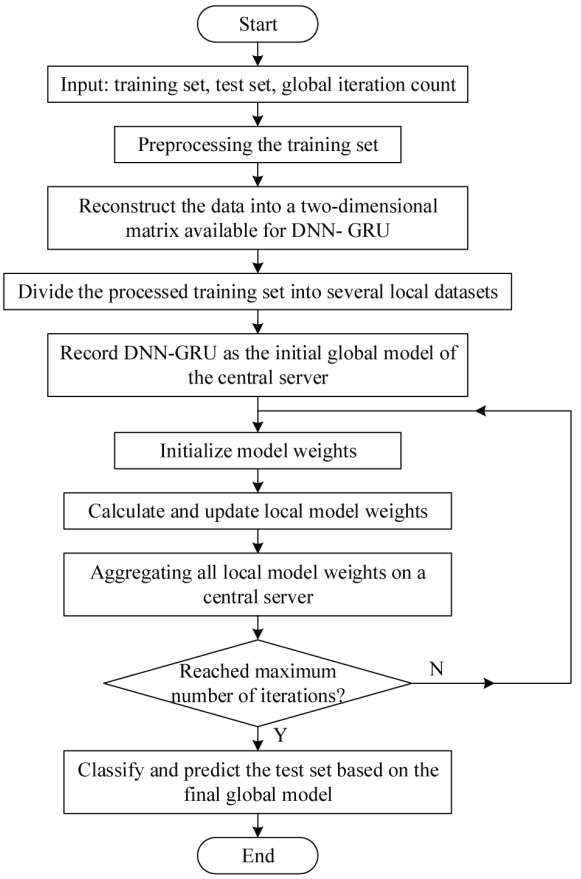
Figure 8. NID architecture integrating transformer and FedL



Step 2: The preliminarily trained model is distributed to various nodes with certain computational power.
Step 3: Nodes collect real-time data from nodes in their area and share 10% of it globally.
Step 4: The node utilizes real-time data obtained by itself and data shared by other nodes to train and tune local models at that node, which is the second stage of federated transfer learning.
Step 5: The trained model for NID is used.

In the first training stage, the algorithm fully utilizes the available dataset and trains the base station through source domain data related to the target domain, laying the foundation for the parameters in the deep learning network and determining the general direction of training. However, because the source domain is not completely the same as the target domain, directly using the target domain data as detection data will definitely produce poor results. By distributing the preliminarily trained network parameters to each node, the algorithm enters the second training stage. In the second training stage, in order to fully utilize the computing power of each node, FedL is made the main task of this stage. At the same time, in order to reduce the overall accuracy of the model caused by biased data acquisition by different nodes, 5% of the data received by each node is used as global shared data, ensuring that at least 5% of the same data is present in the training data of each node. In this way, the training parameters of each node in the model are not significantly different, ensuring accuracy. The training process of the model is shown in Figure 9.

Figure 9. The training process of NID



This model does not require centralized training of all raw data to obtain the final model, but instead selects a local objective function as a substitute for the global objective function based on the corresponding data subset of each device. A local solver is used to optimize the local objective functions of these devices, and after each iteration process, the local model parameters are transferred to the central server, which aggregates them to obtain the final global model.

In the first training stage of the proposed model, the available dataset was fully utilized, but we should clearly recognize that there is no need to transmit user data before or during the model training process, which means that user data will not be intercepted during the transmission process, which is the biggest security risk. Therefore, the security of user data can be guaranteed.

EXPERIMENTAL ANALYSIS

Dataset and Evaluation Indicators

The proposed model was evaluated on the NSL-KDD and UNSW-NB15 datasets. The NSL-KDD dataset has improved the KDD99 dataset by removing redundant and duplicate data from the training and testing sets, making the settings of the training and testing sets more reasonable and enabling more accurate detection rates. In the NSL-KDD dataset, there are five types of attacks: normal traffic, denial of service attacks, port attacks, privilege escalation attacks, and remote user attacks.

The selection of NSL-KDD as training and testing data for the NID system is mainly based on the following criteria:

1. **Widely used:** The NSL-KDD is one of the most widely used datasets in the NID domain, with high popularity and recognition. This dataset is widely used in academic research and industrial applications, providing a benchmark testing platform for evaluating the performance of NID.
2. **Contains multiple types of attacks:** NSL-KDD includes various types of attacks, including remote login, probing, web, and more. These types of attacks cover various aspects of network intrusion and help to comprehensively evaluate the performance of NID.
3. **Contains normal traffic data:** In addition to attack data, NSL-KDD also contains a certain amount of normal traffic data, which helps to better distinguish between normal traffic and attack traffic when training and testing NID systems and improve system accuracy.
4. **Data standardization:** The data in NSL-KDD has been standardized to eliminate the influence of dimensionality and value ranges between different features, making them comparable. This helps to improve the performance and accuracy of NID.

Because NSL-KDD generates datasets by simulating network traffic in real scenarios, it has high practical significance and practicality. The attack techniques and features in this dataset are simulated and generated based on actual network attack cases, enabling them to represent real-world intrusion scenarios to a certain extent.

The UNSW-NB15 dataset was collected in a real network environment at the Australian security laboratory in 2015. The network traffic records contained therein are authentic modern normal activities and modern comprehensive attack behaviors. Compared to the NID dataset KDD99 dataset, it better reflects the characteristics of modern network traffic. Therefore, this article's experiment also focuses on the NID dataset UNSW-NB15. The network record vector of this dataset contains 10 types of abnormal intrusion attack behaviors.

The selection of UNSW-NB15 as training and testing data for the NID system is mainly based on the following criteria:

1. **Includes modern attack types:** Compared to early NID datasets, UNSW-NB15 includes more modern types of attacks, such as attacks against web applications and botnet attacks. These types

of attacks are currently popular topics in the field of network security and selecting this dataset can help evaluate the detection capability of NID systems against modern attacks.

2. **High quality labels:** Each traffic sample in UNSW-NB15 has undergone detailed labeling and classification, including normal traffic, attack traffic, and specific types of attacks. The high quality of these labels ensures the accuracy of training and testing, making the evaluation results more reliable.
3. **Large scale datasets:** UNSW-NB15 is a large-scale dataset that contains a large number of network traffic samples. This enables the dataset to support larger scale training and testing, which helps improve the generalization ability of NID systems.
4. **Rich in features:** UNSW-NB15 contains rich feature information, including statistical characteristics of network traffic, time characteristics, and protocol characteristics. These features help to more comprehensively describe the behavioral patterns of network traffic and improve the accuracy of NID systems.

UNSW-NB15 is generated by simulating traffic in real-world network environments, which makes this dataset highly relevant and practical. The network traffic in this dataset includes both normal traffic and various attack traffic, which helps to comprehensively evaluate the performance of NID systems.

The experiment used classification accuracy (A) and F1 score to evaluate the model's classification performance. A refers to the proportion of correctly classified data among all data, calculated as shown in eq (5) below:

$$A = \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \tag{5}$$

The selection of accuracy as the evaluation indicator for network intrusion detection is mainly because accuracy can intuitively reflect the performance of the intrusion detection system. The higher the accuracy, the more accurately the system can identify normal traffic and attack traffic, avoiding false positives and omissions. In practical applications, the purpose of network intrusion detection systems is to discover potential attack behaviors in complex network traffic and prevent the network from being exploited by malicious attackers. If the accuracy of the detection system is low, it is easy to misjudge normal traffic as attack traffic, or miss the true attack traffic, which will pose a threat to the normal operation of the network. Therefore, choosing accuracy as an evaluation metric can measure the performance of intrusion detection systems in identifying normal traffic and attack traffic, helping us determine the reliability and security of the system. Meanwhile, accuracy can also serve as a basis for optimizing and improving the system, enhancing its performance and accuracy.

F1 score is the harmonic mean of accuracy and recall, calculated as shown in equation (6):

$$F1 = \frac{2DR}{D + R} \tag{6}$$

In equation (6), $D = \frac{T_P}{T_P + F_P}$ and $R = \frac{T_P}{T_P + F_N}$. T_P represents the number of correctly predicted samples, F_P represents the number of samples that were incorrectly identified as this class, T_N represents the number of samples that were correctly predicted as other classes, and F_N represents the number of samples that were incorrectly predicted as other classes.

The reason for choosing F1 value as the evaluation metric for network intrusion detection is mainly because F1 value can comprehensively reflect the precision and recall performance of the classifier and has high evaluation accuracy. The F1 value is the harmonic mean of precision and recall, taking into account both the precision of true examples and the recall of true negative examples. Therefore, the F1 value can provide a more comprehensive evaluation of the performance of intrusion detection systems. In real-world scenarios, network intrusion detection systems need to accurately detect attack behavior in complex network traffic while avoiding false positives and false negatives. If the precision is too high and the recall is too low, there may be missed reports, resulting in some attack traffic not being detected; if the precision is too low and the recall is too high, there may be false positives, misjudging a large amount of normal traffic as attack traffic. Therefore, a comprehensive indicator is needed to consider both precision and recall performance, and F1 value is precisely such a suitable indicator. In addition, F1 value also has the advantages of relatively simple calculation and easy understanding, so F1 value is used as an evaluation indicator in many classification tasks.

Experimental Environment

Table 1 shows the specific experimental environment.

Model Training

First, the impact of participant number N and learning rate α on model performance are analyzed. In the experiment, the number of participants N was set to be 10 and 20, and the learning rate α was 0.001 and 0.0001, respectively. The model was trained, and the test results were recorded. The accuracy obtained from training with the NSL-KDD and UNSW-NB 15 datasets is shown in Figures 10 and 11, respectively.

Figures 10 and 11 show the experimental content recorded in terms of the number of epochs and model prediction accuracy during the training process of the model. It can be seen that when the learning rate is set to 0.001, the accuracy changes with the increase of epoch, and this is used as the baseline for comparison.

When the number of participants is 10, adjusting the learning rate from 0.0001 to 0.001 can accelerate the convergence of the model, reaching an accuracy of 99.65% when the epoch is 20. When the number of participants is 20, the learning rate is adjusted from 0.0001 to 0.001, and the highest model accuracy can also reach 99.6%, but the convergence speed is significantly reduced. From this, it can be seen that regardless of the number of participants, the change in learning rate is a key factor affecting the model's faster convergence to maximum accuracy. Meanwhile, because each participant has the same training sample data, the number of participants directly affects the total amount of data participating in model training. This will affect the speed of model convergence and require more epoch to ensure model accuracy.

Table 1. Experimental platform settings

Experimental Environment	Specific Information
Operating system	Windows 7
Memory	64GB
Language	Python3.8.2
Development tool	Pycharm
Graphics card	GTX 2080
Development platform	Tensorflow2.3.2

Figure 10. Accuracy using NSL-KDD dataset

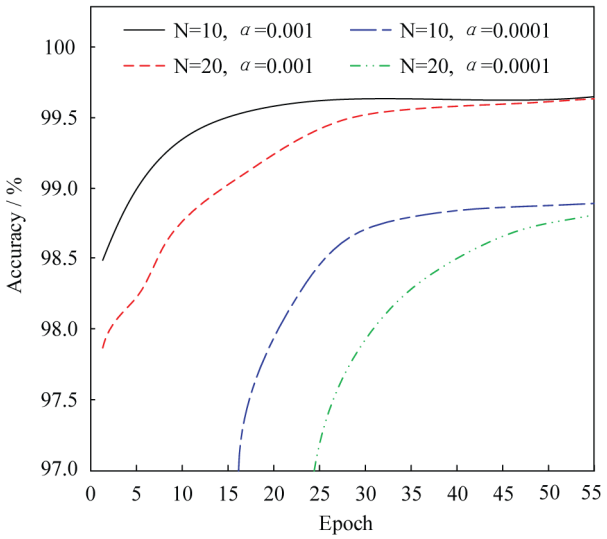
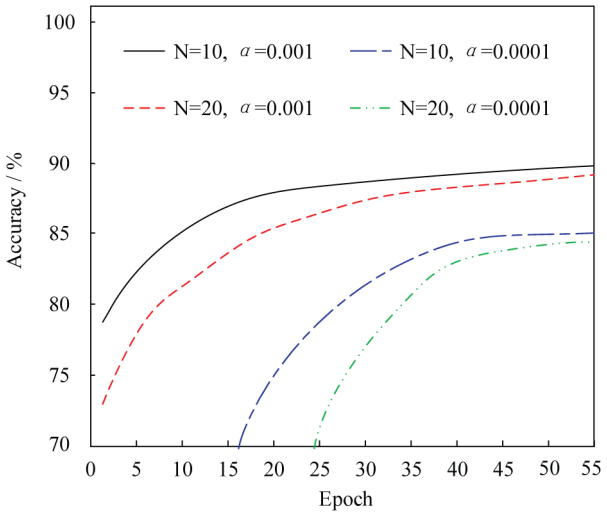


Figure 11. Accuracy using UNSW-NB15 dataset



When setting different batch sizes, the model loss during the training process using the NSL-KDD dataset is shown in Figure 12.

In Figure 12, the results show that as the batch size increases, the convergence speed of the model slows down. This is because as the batch size increases, the number of communication times for FedL also increases, and network overhead is often a significant expense in the system. Therefore, a smaller batch size will be used for subsequent experiments.

To further validate the detection effect of the model on network intrusion behavior, the k-fold cross validation method is used to conduct experiments on the accuracy and F1 score of the model on two datasets. The results are shown in Figure 13 and Figure 14.

Figure 12. Training losses using NSL-KDD dataset

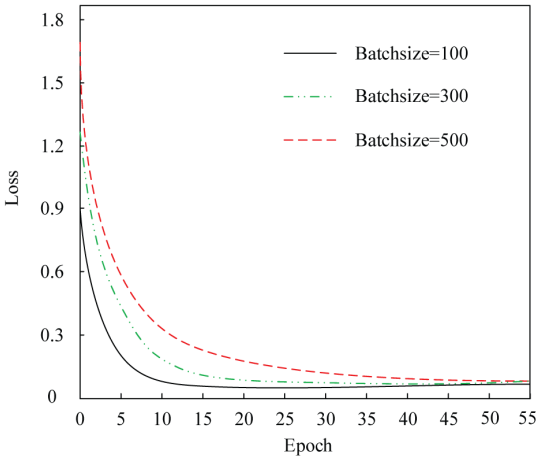
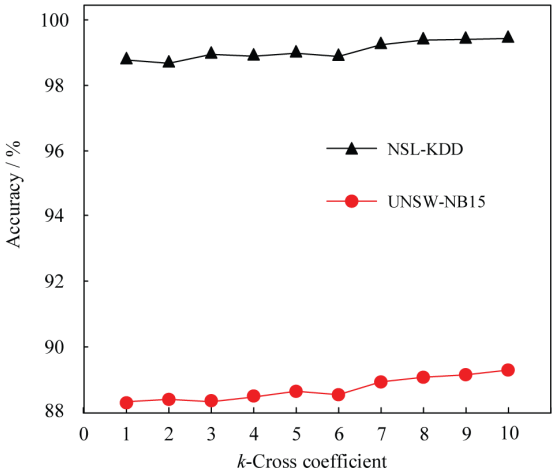


Figure 13. Accuracy when using different datasets



Figures 13 and 14 show that, after testing with k values ranging from 1 to 10, the accuracy and F1 score values of both datasets increase with the increase of k values. This is because as the k-value increases, the number of partitions in the dataset will increase, and the data used as the training set will also increase. The more data participate in training, the higher the final evaluation indicators such as test accuracy will be.

Comparative Analysis

In order to further verify the superiority of the proposed NID method that integrates transformer and FedL, comparative experiments were conducted with IDS-FMLT (Mourad, et al., 2020), FLTrELM (Alkhatib et al., 2022), and FedACNN (Chellammal et al., 2023).

The experiments were conducted using two datasets, and the accuracy obtained by different methods is shown in Figure 15 and Figure 16, respectively.

Figure 14. F1-score when using different datasets

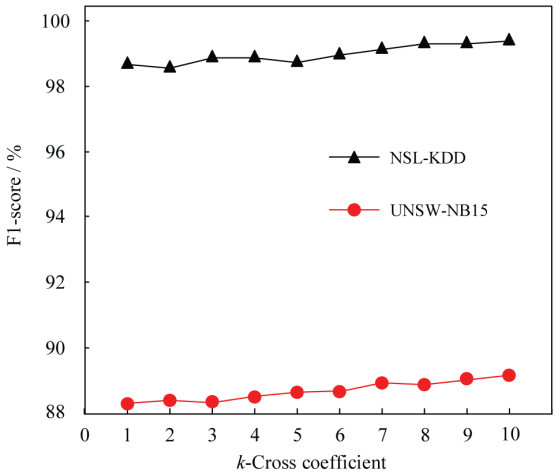
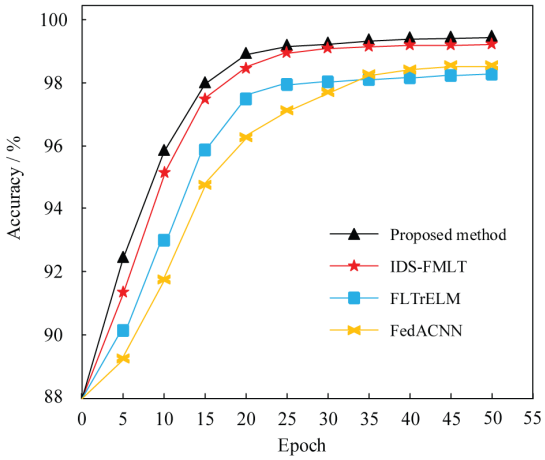


Figure 15. Accuracy when using NSL-KDD datasets with different methods



The F1 scores obtained by different methods under different datasets are shown in Figure 17 and Figure 18.

The above experimental results provide the NID results of the four methods under the same experimental conditions using two different datasets, NSL-KDD and UNSW-NB15. It can be seen that the proposed method has the highest NID accuracy and F1 score. The following will provide a detailed analysis of the experimental results.

From the experimental results, it can be seen that the proposed method can solve the problem of ineffective protection of data privacy faced by current network intrusion detection technology through collaboration among multiple users under the coordination of a central server. In addition, each user's original private dataset is stored locally. By applying transformer and federated learning to network intrusion detection, the issues of sensitive information protection and incomplete data in training data are solved, thereby improving the accuracy of network intrusion detection.

Figure 16. Accuracy when using UNSW-NB15 datasets with different methods

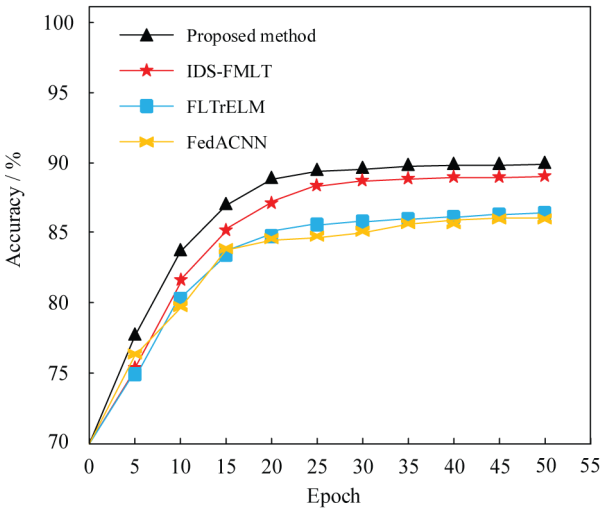
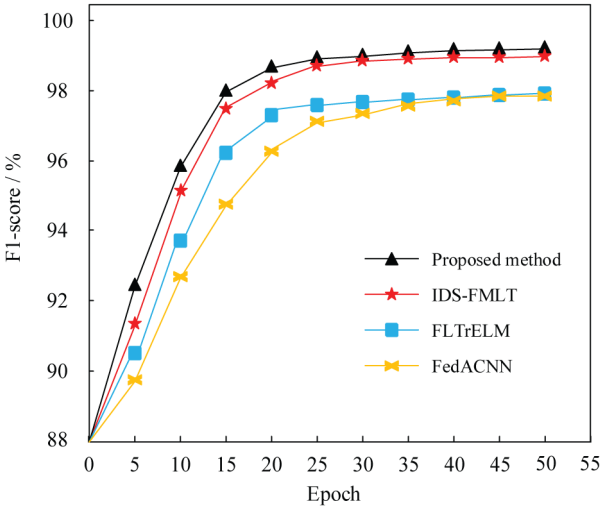


Figure 17. F1-score when using NSL-KDD datasets with different methods



RESULTS

Results Under the NSL-KDD Dataset

The accuracy of different methods and the maximum values of F1 score when using the NSL-KDD dataset are compared as shown in Figure 19 and Table 2.

Results Under the UNSW-NB15 Dataset

The accuracy of different methods and the maximum values of F1 score when using the UNSW-NB15 dataset are compared as shown in Figure 20 and Table 3.

Figure 18. F1-score when using UNSW-NB15 datasets with different methods

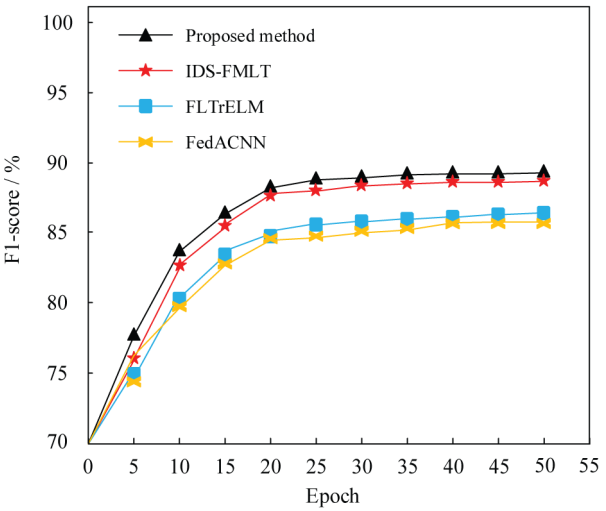


Figure 19. Comparison of different methods using NSL-KDD dataset

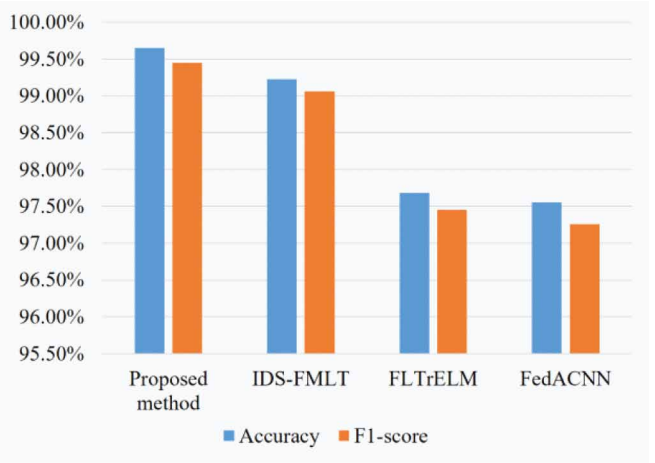


Table 2. Comparison of different methods using NSL-KDD dataset

Method	Index	
	Accuracy	F1-Score
Proposed method	99.65%	99.45%
IDS-FMLT	99.23%	99.06%
FLTrELM	97.68%	97.45%
FedACNN	97.55%	97.26%

Figure 20. Comparison of different methods using UNSW-NB15 dataset

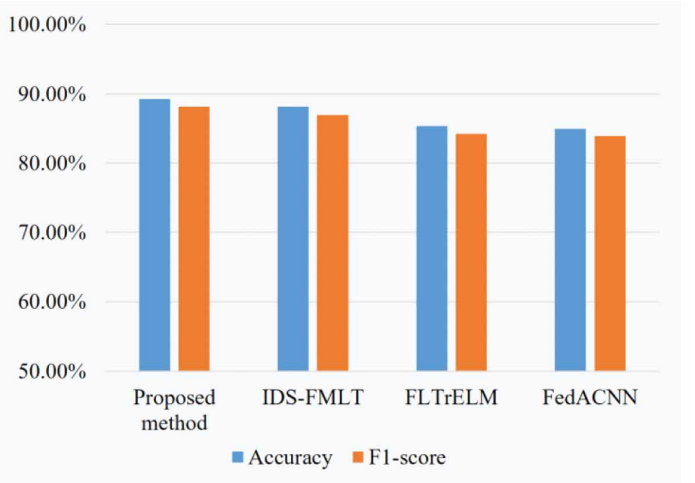


Table 3. Comparison of different methods using UNSW-NB15 dataset

Method	Index	
	Accuracy	F1-Score
Proposed method	89.25%	88.13%
IDS-FMLT	88.13%	86.95%
FLTrELM	85.32%	84.20%
FedACNN	84.97%	83.88%

The above experimental results provide the network intrusion detection results of the four methods under the same experimental conditions using two different datasets, NSL-KDD and UNSW-NB15. It can be seen that the proposed method has the highest accuracy in network intrusion detection and F1 score, with the highest accuracy of 99.65% and 89.25% on the NSL-KDD dataset and UNSW-NB15 dataset, respectively, while F1 score has the highest accuracy of 99.45% and 88.13%, respectively. Compared to the other three comparison methods, the accuracy on the NSL-KDD dataset has improved by a minimum of 0.42% and a maximum of 1.9%. F1 score showed a minimum improvement of 0.39% and a maximum improvement of 2.19%. The accuracy on the UNSW-NB15 dataset improved by a minimum of 1.12% and a maximum of 4.28%. F1 score showed a minimum improvement of 1.18% and a maximum improvement of 4.25%. This is because the introduction of a self-attention mechanism has improved the encoder and decoder of traditional transformer networks, enabling them to adapt to one-dimensional signal data in network intrusion detection and achieve multi-layer stacking. In addition, using DNN to extract deep level features of traffic and GRU to extract temporal features of traffic ensures the comprehensiveness of the extracted features, further improving the accuracy of network intrusion detection.

Ablation Experiment

To further validate the effectiveness of each module in the model, a model ablation experiment was designed to compare the impact of different modules on the overall performance of the model. The design is as follows:

Model 1: Without transformer - Removing the feature fusion of the transformer from the proposed model.
Model 2: Without federated learning - Does not use federated learning for local model training and parameter uploading; directly uploads data for training.
Model 3: Without DNN - Removes the DNN module from the proposed model.
Model 4: Without GRU - Removes the GRU module from the proposed model.

The experiment was conducted using the NSL-KDD dataset, and the final results of the model ablation experiment are shown in Table 4.

In Table 4, the experimental results on the NSL-KDD dataset show that removing any module from the model will result in a decrease in the accuracy and F1 score of the model's sentiment analysis. Among all modules, the indicator values of model 1 and model 4 have decreased significantly, indicating that federated learning and transformer modules have made significant contributions to the improvement of results. When either of them is missing, it will have a significant impact on the model. However, the performance of other models cannot achieve the best, which indicates that DNN and GRU also contribute to the improvement of the results. The results of the ablation experiment fully validate the necessity of each module in the proposed model to achieve the best results.

DISCUSSION

Based on the previous theoretical analysis and experimental results, it can be concluded that the implementation of high accuracy and F1 in NID is of great significance, as follows:

1. **Improving security:** A highly accurate NID system can more accurately identify normal and attack traffic, avoid false positives and false negatives, thereby reducing the risk of malicious attacks on the network and improving network security.
2. **Reducing erroneous operations:** NID systems with high F1 values can maximize recall while ensuring accuracy. This means that the system can not only accurately identify the true attack traffic but can also avoid the occurrence of false positives as much as possible, thereby reducing maloperations and potential losses.
3. **Improving user experience:** NID systems with high accuracy and F1 values can provide a better user experience. When users access the network, the system can quickly and accurately determine the normal and attacked traffic, avoid misjudgment and maloperation, improving user satisfaction and trust.
4. **Optimizing system performance:** A NID system with high accuracy and F1 value can better detect and prevent network attack behavior, thereby reducing waste and abuse of network resources. This helps optimize the performance of the system and improve the operational efficiency of the network.

Table 4. Model ablation experimental results

Model	Indicator	
	Accuracy	F1-Score
Model 1	85.43%	85.33%
Model 2	82.65%	82.45%
Model 3	91.23%	91.06%
Model 4	92.68%	92.45%
Proposed model	99.65%	99.45%

5. **Evaluating performance:** A NID system with high accuracy and F1 value can better evaluate the performance of the system. By comparing the accuracy and F1 value of different systems, it is possible to determine which system has better performance, thereby better selecting and configuring NID systems.

When applying the proposed NID method to different users or enterprises of different scales, small adjustments need to be made based on the actual situation of the users or enterprises. These adjustments are mainly reflected in the following two aspects:

1. **Adjustment of the number of nodes:** Because federated learning collects real-time data from users at each node and conducts local model training, the model parameters are sent to the central server after the local model training is completed. Therefore, for enterprises of different scales, their user distribution and data volume vary, and the corresponding number of nodes needs to be adjusted to ensure the accuracy of model training.
2. **Adjustment of the computing power of the central server:** For large-scale enterprises, there are a large number of nodes, and there are also many model parameters submitted after local model training. Therefore, stronger computing power is required in the process of forming an integrated model on the central server.

The NID method can adapt to larger or more diverse network infrastructure by considering the following aspects:

1. **Distributed architecture:** In order to monitor larger or more diverse network infrastructure, a distributed architecture can be considered, where intrusion detection modules are distributed at different locations in the network to simultaneously monitor and process multiple network traffic data. This can improve detection efficiency and accuracy and reduce network latency.
2. **Custom rules and algorithms:** For larger or more diverse network infrastructure, more complex rules and algorithms can be customized to detect abnormal behavior in network traffic.
3. **Real time monitoring and response:** In order to better adapt to larger or more diverse network infrastructure, intrusion detection methods should have the ability to monitor and respond in real time. By analyzing network traffic data in real-time, abnormal behavior can be detected in a timely manner and corresponding security measures can be taken quickly to prevent potential attack behavior.
4. **Automation and intelligence:** With the development of network technology, automation and intelligence have become important elements of intrusion detection. Intelligent algorithms and automation tools can be used to reduce the burden of manual operations, improve detection efficiency and accuracy.

There are some ethical issues that need to be considered during the use of the model. Although the proposed method uses local model training and does not require the transmission of real-time user data, it greatly reduces the risk of privacy leakage during the transmission process of user data. However, local model training requires collecting real-time data from users, so we also have to consider data breaches caused by human factors, such as malicious privacy breaches caused by staff during local training.

The multi-party collaborative learning framework based on federated learning has certain limitations in evaluating the effectiveness of data exchange and sharing:

1. For some participants with limited resources, additional hardware and computational support may be required.

2. Because different participants may have different quality and quantity of data, imbalanced and biased model training may result. Some participants may not be able to achieve performance improvements comparable to other participants due to poor data quality or insufficient quantity.
3. Federated learning requires local model training for each participant, which may require higher computational resources.
4. Although federated learning aims to protect data privacy, it may still face legal and compliance challenges in cross organizational and cross regional data exchange and sharing processes. It is necessary to ensure compliance with relevant data protection and privacy regulations.

CONCLUSION

A NID method that combines transformer and FedL is proposed to address the issue of traditional NID methods being unable to achieve high accuracy detection for different types of network intrusions while ensuring data privacy. The experimental verification results indicate that building a NID model based on FedL can effectively safeguard data privacy without affecting data exchange and sharing. Using self AttM to improve the encoder and decoder of traditional transformer networks can enable the model to perform parallel calculations, greatly improving its convergence speed. On the basis of using DNN to extract deep level features of traffic, using GRU to extract temporal features of traffic can effectively ensure the comprehensiveness of the extracted features. In the healthcare industry, government, and financial sectors, the NID method can quickly and accurately detect different types of attacks that may occur in the network, such as denial of service, port scanning, malware, distributed denial of service, or ransomware, by investigating network traffic. The NID method can prevent significant losses for different fields or industries. It is suggested that future research areas may explore other machine learning techniques or extend this method to a wider network environment.

The next step will focus on reducing the communication cost of model transfer in FedL. Reducing the cost of parameter transfer through gradient descent algorithm will make applications in FedL environments more efficient. In addition, we can consider optimizing the structure of different sub models in the model, such as DNN module, GRU module, encoder module composed of improved transformer, and MLP module. The simplified model structure will be simpler, which means that the calculation speed of the model will be faster, the calculation volume will be reduced, and the cost will be reduced.

AUTHOR NOTE

This work was supported by the Guangdong Provincial Major Research Platform Ordinary University Characteristic Innovation Project Fund (No.2022KTSCX204) and Hainan Natural Science Foundation (623RC515). Correspondence concerning this article should be addressed to School of Artificial Intelligence, Guangdong Open University, Guangzhou, Guangdong, 510091, China. Qi Zhou, zhoudakeqq@163.com

REFERENCES

Abbas, N., Nasser, Y., Shehab, M., & Sharafeddine, S. (2021). Attack-specific feature selection for anomaly detection in software-defined networks. *The 2021 3rd IEEE Middle East and North Africa Communications Conference*, 142-146.

Alavizadeh, H., Alavizadeh, H., & Jang-Jaccard, J. (2022). *Deep Q-learning based reinforcement learning approach for network intrusion detection*. Academic Press.

Aldarwbi, M. Y., Lashkari, A. H., & Ghorbani, A. A. (2022). The sound of intrusion: A novel network intrusion detection system. *Computers & Electrical Engineering, 104*(A), 214-223.

Alkhatib, N., Mushtaq, M., Ghauch, H., Ghauch, H. G., & Danger, J. (2022). Unsupervised network intrusion detection system for AVTP in automotive ethernet networks. *The 33rd IEEE Intelligent Vehicles Symposium (IEEE IV),* 1731-1738.

Bai, H., Deng, D., & Xu, G. (2022). Research on intrusion detection mechanism based on federated learning. *Netinfo Security*, *3*(1), 46–54.

Chellammal, P., Malarchelvi, S. K., Reka, K., Sheba, P. D., Reka, K., & Raja, G. (2023). Fast and effective intrusion detection using multi-layered deep learning networks. *International Journal of Web Services Research*, *19*(1), 72–80.

Cheng, Y., Xu, Y., Zhong, H., & Liu, Y. (2020). Leveraging semisupervised hierarchical stacking temporal convolutional network for anomaly detection in IoT communication. *IEEE Internet of Things Journal*, *8*(1), 144–155. doi:10.1109/JIOT.2020.3000771

Cheng, Y., Xu, Y., Zhong, H., & Liu, Y. (2021). HS-TCN: A semi-supervised hierarchical stacking temporal convolutional network for anomaly detection in IoT2021. *The IEEE 38th International Performance Computing and Communications Conference (IPCCC),* 1-7.

Cui, J., Sun, H., Zhong, H., Zhang, J., Wei, L., Bolodurina, I. P., & He, D. (2023). Collaborative intrusion detection system for SDVN: A fairness federated deep learning approach. *IEEE Transactions on Parallel and Distributed Systems*, *34*(9), 2512–2528. doi:10.1109/TPDS.2023.3290650

Fan, Y., Li, Y., Zhan, M., Cui, H., & Zhang, Y. (2020). Iotdefender: A federated transfer learning intrusion detection framework for 5g IoT. *The 2020 IEEE 14th International Conference on Big Data Science and Engineering*, 88-95.

Filho, F. L., Soares, S. C., Oroski, E., Albuquerque, R. D., Mata, R. Z., Mendonça, F. L., & Júnior, R. T. (2023). Botnet detection and mitigation model for IoT networks using federated learning. *Sensors (Basel)*, *23*(14), 78–86. PMID:37514600

Gupta, B. B., Gaurav, A., Panigrahi, P. K., Arya, V. (2023). Analysis of artificial intelligence-based technologies and approaches on sustainable entrepreneurship. *Technological Forecasting and Social Change, 186*(B), 325-337.

Hamad, S. A., Sheng, Q. Z., Tran, D. H., Zhang, W., & Nepal, S. (2020). A behavioural network traffic novelty detection for the internet of things infrastructures. *International Symposium on Parallel Architectures, Algorithms, and Programming*, 174-186.

Lin, Y., Wang, J., Tu, Y., Chen, L., & Dou, Z. (2021). Time-related network intrusion detection model: A deep learning method. *The 2021 IEEE Global Communications Conference (GLOBECOM)*, 1-6.

Ling, Z., & Hao, Z. J. (2022). An intrusion detection system based on normalized mutual information antibodies feature selection and adaptive quantum artificial immune system. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–25. doi:10.4018/IJSWIS.308469

Ling, Z., & Hao, Z. J. (2022). Intrusion detection using normalized mutual information feature selection and parallel quantum genetic algorithm. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–24. doi:10.4018/IJSWIS.307324

Liu, Y., Garg, S., Nie, J., Zhang, Y., Xiong, Z., Kang, J., & Hossain, M. S. (2020). Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach. *IEEE Internet of Things Journal*, *8*(8), 6348–6358. doi:10.1109/JIOT.2020.3011726

Man, D. P., Zeng, F. Y., & Yang, W. (2022). Intelligent intrusion detection based on federated learning for edge-assisted internet of things. *Security and Communication Networks*, *2022*(4), 135–143.

Meidan, Y., Bohadana, M., Mathov, Y., Yu, M., Lv, J., & Wang, Y. (2022). N-baiot—Network-based detection of IoT botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, *17*(3), 12–22. doi:10.1109/MPRV.2018.03367731

Mishra, A., Joshi, B. K., Arya, V., Gupta, A. K., & Chui, K. T. (2022). Detection of distributed denial of service (DDoS) attacks using computational intelligence and majority vote-based ensemble approach. *International Journal of Software Science and Computational Intelligence*, *14*(1), 242–255. doi:10.4018/IJSSCI.309707

Mourad, A., Tout, H., Wahab, O. A., Otrok, H., & Dbouk, T. (2020). Ad hoc vehicular fog enabling cooperative low-latency intrusion detection. *IEEE Internet of Things Journal*, *8*(2), 829–843. doi:10.1109/JIOT.2020.3008488

Novikova, E., Doynikova, E., & Golubev, S. (2022). Federated learning for intrusion detection in the critical infrastructures: Vertically partitioned data use case. *Algorithms*, *15*(4), 48–56. doi:10.3390/a15040104

Pan, X. N., Yamaguchi, S., Kageyama, T., & Bin Kamilin, M. H. (2022). Machine-learning-based white-hat worm launcher in botnet defense system. *International Journal of Software Science and Computational Intelligence*, *14*(1), 158–170. doi:10.4018/IJSSCI.291713

Park, S. H., Park, H. J., & Choi, Y. J. (2020). RNN-based prediction for network intrusion detection. *The 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC),* 572-574.

Qazi, E. U., Faheem, M. H., & Zia, T. (2023). HDLNIDS: Hybrid Deep-Learning-Based Network Intrusion Detection System. *Applied Sciences (Basel, Switzerland)*, *13*(8), 56–65. doi:10.3390/app13084921

Qin, Y., Matsutani, H., & Kondo, M. (2020). A selective model aggregation approach in federated learning for online anomaly detection. *The 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics),* 684-691.

Rahman, S. A., Tout, H., Talhi, C., & Mourad, A. (2020). Internet of things intrusion detection: Centralized, on-device, or federated learning? *IEEE Network*, *34*(6), 310–317. doi:10.1109/MNET.011.2000286

Sajid Farooq, M., Abbas, S., Atta-ur-Rahman, , Sultan, K., Adnan Khan, M., & Mosavi, A. (2023). A fused machine learning approach for intrusion detection system. *Computers, Materials & Continua*, *74*(2), 2607–2623. doi:10.32604/cmc.2023.032617

Sarhan, M., Lo, W. W., Layeghy, S., & Portmann, M. (2022). HBFL: A hierarchical blockchain-based federated learning framework for collaborative IoT intrusion detection. *Computers & Electrical Engineering*, *103*(7), 162–170. doi:10.1016/j.compeleceng.2022.108379

Sattler, F., Wiedemann, S., Müller, K. R., & Samek, W. (2022). Robust and communication-efficient federated learning from non-iid data. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(9), 3400–3413. doi:10.1109/TNNLS.2019.2944481 PMID:31689214

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2021). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *International Conference on Information Systems Security and Privacy, 1*(4), 108-116.

Srivastava, A. M., Rotte, P. A., Jain, A., & Prakash, S. (2022). Handling data scarcity through data augmentation in training of deep neural networks for 3D data processing. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–16. doi:10.4018/IJSWIS.297038

Takeda, A. (2022). Detection and analysis of intrusion attacks using deep neural networks. *The 25th International Conference on Network-Based Information Systems (NBiS),* 258-266.

Tembhurne, J. V., Almin, M. M., & Diwan, T. (2022). Mc-DNN: Fake news detection using multi-channel deep neural networks. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–20. doi:10.4018/IJSWIS.295553

Wang, J. Z., Kong, L. W., & Huang, Z. (2020). Research review of federated learning algorithms. *Big Data Research*, *6*(6), 70–88.

Wang, K. P., Li, J. M., & Wu, W. F. (2022). An efficient intrusion detection method based on federated transfer learning and an extreme learning machine with privacy preservation. *Security and Communication Networks*, *2022*(4), 205–213. doi:10.1155/2022/2913293

Wang, X., Han, Y., Wang, C., Zhao, Q., Chen, X., & Chen, M. (2022). In-edge AI: Intelligentizing mobile edge computing, caching, and communication by federated learning. *IEEE Network*, *33*(5), 156–165. doi:10.1109/MNET.2019.1800286

Wang, X. F., Wang, Y. H., Javaheri, Z., Almutairi, L., Moghadamnejad, N., & Younes, O. S. (2021). Federated deep learning for anomaly detection in the internet of things. *Computers & Electrical Engineering*, *108*(2), 52–60.

Yang, X., Tan, W., Peng, C., Xiang, S., & Niu, K. (2022). Federated learning incentive mechanism design via enhanced shapley value method. *Wireless Communications and Mobile Computing*, *8*(2), 134–142. doi:10.1109/MWC.2018.1600445

Zhang, L., & Zhang, J. H. (2022). An intrusion detection system based on normalized mutual information antibodies feature selection and adaptive quantum artificial immune system. *International Journal on Semantic Web and Information Systems*, *18*(1), 124–135.

Zhang, L., & Zhang, J. H. (2022). Intrusion detection using normalized mutual information feature selection and parallel quantum genetic algorithm. *International Journal on Semantic Web and Information Systems*, *18*(1), 86–98.

Zhu, J. C., Cao, J. N., Saxena, D., Jiang, S., & Ferradi, H. (2023). Blockchain-empowered federated learning: Challenges, solutions, and future directions. *ACM Computing Surveys*, *55*(11), 63–72. doi:10.1145/3570953

Qi Zhou, Ph.D. in Computer Information Technology, Associate Professor. He graduated from Sun Yat-sen University in 2008. I am currently working at Guangdong Open University. His research interests include network security and information technology security.

Chun Shi (1977.01-), male (Han), born in Poyang, Jiangxi, holds a doctoral degree and is an associate professor. His research areas include wireless communication protocols, network security, terahertz communication, and blockchain technology.