Computers & Security 143 (2024) 103910

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

Automated federated learning for intrusion detection of industrial control systems based on evolutionary neural architecture search

Jun-Min Shao [a], Guo-Qiang Zeng [a,*], Kang-Di Lu [b], Guang-Gang Geng [a], Jian Weng [a]

[a] College of Cyber Security and the National Joint Engineering Research Center of Network Security Detection and Protection Technology, Jinan University, Guangzhou 510632, China

[b] College of Information Science and Technology, Donghua University, Shanghai 201620, China

ARTICLE INFO

Keywords:
Industrial control system
Automated deep learning
Federated learning
Intrusion detection
Neural architecture search

ABSTRACT

In recent years, federated learning has been applied to the security of the Internet of Things and Industrial Control Systems (ICS) due to its advantages in communication cost and privacy preserving. However, the existing deep learning models used in federated learning-based intrusion detection systems (IDS) are manually designed by relying on the extensive experiences of designers and are not applicable in different scenarios flexibly. In this paper, we make the first attempt to automatically design a lightweight federated learning model termed as Fed-GA-CNN-IDS for the IDS issue in ICS by evolutionary neural architecture search (NAS). Five lightweight neural architectures of Convolutional Neural Network (CNN) are considered as the basic blocks to be combined and optimized in federated NAS for ICS intrusion detection. An efficient discrete encoding strategy is developed to describe the combination of five basic lightweight blocks and the specific discrete evolutionary operations under the framework of genetic algorithm (GA) are designed elaborately to guide the evolutionary process of an automated federated learning model. The experimental results on three widely-used intrusion detection datasets in ICSs such as Gas Pipeline, SWaT and WADI, demonstrate that the proposed Fed-GA-CNN-IDS method can obtain more lightweight models and better or at least competitive intrusion detection performance than three state-of-the-art manually-designed federated learning-based IDS methods, two federated NAS methods originally developed for traditional image classification tasks, and four lightweight IDS methods.

1. Introduction

Industrial Control Systems (ICSs), deemed indispensable infrastructure for societal functionality, are extensively deployed within pivotal sectors including power systems, natural gas, petroleum refinement, and transportation networks. Also, ICSs serve as a fundamental cornerstone supporting the national economy, societal operations, and ensuring national security. With the continuous development of the Internet and the high integration of informatization and industrialization, the interconnection between various systems has become an inevitable trend in the development of informatization, so the threats faced by ICS are also increasing day by day. Recently, there have been many attacks on ICS, such as Maroochy wastewater system attack (Slay and Miller, 2007) and Stuxnet worm attack (Langner, 2011). These attacks on ICS have caused huge property losses to corresponding enterprises and organizations, and threatened the lives of people in corresponding areas. Therefore, the security problem of ICS has been a concern for the majority of researchers.

In order to ensure the normal operation of ICS, researchers usually deploy an Intrusion Detection System (IDS) to monitor the system status in real-time. When the system status is abnormal, IDS performs alarm and processing to defend against known and unknown attacks. In recent years, deep learning models have become more and more powerful. Deep learning-based IDSs have gained tremendous momentum due to their high performance (Kravchik and Shabtai, 2018, 2021). However, since deep learning needs to centralize data to a centralized server for training during the training process, not only will it result in unbearable transmission overhead, but also data privacy and security will be threatened.

With the development of distributed machine learning (McMahan et al., 2017), federated learning not only guarantees the efficient performance of deep learning models, but also provides solutions for the data privacy security of these systems. Therefore, considering the advantages of deep learning, many scholars use the data in ICS to conduct federated training based on deep learning models such as

* Corresponding author.
E-mail addresses: shaojunmin@stu2021.jnu.edu.cn (J.-M. Shao), zeng.guoqiang5@gmail.com (G.-Q. Zeng), kangdilu@dhu.edu.cn (K.-D. Lu), gggeng@jnu.edu.cn (G.-G. Geng), cryptjweng@gmail.com (J. Weng).

https://doi.org/10.1016/j.cose.2024.103910
Received 2 September 2023; Received in revised form 28 March 2024; Accepted 23 May 2024
Available online 27 May 2024

convolutional neural network (CNN) and recurrent neural network (RNN), to ensure data privacy while building high-performance ICS intrusion detection scheme (Mothukuri et al., 2021). In existing works, the neural architectures of the most deep learning models for ICS intrusion detection are manually designed by designers and engineers, which heavily depends on the experience and trial and error for a specific scenario. This process frequently requires significant time and effort, and achieving a balance between computational cost and intrusion detection performance in designing the final neural architecture and parameter scheme is often challenging.

In response to the above situation, this paper makes the first attempt to automatically design a lightweight federated learning model termed as Fed-GA-CNN-IDS for the IDS issue in ICS by evolutionary neural architecture search (NAS). Specifically, we use five sets of lightweight CNN blocks as the candidate blocks by applying evolutionary algorithms to optimize the combination of these blocks, and use the data in ICS to conduct automated NAS under the federated framework. Thus, an efficient ICS intrusion detection model is obtained. The major contributions of this work are given as follows:

(1) The intrusion detection issue in ICSs is facing these challenges, such as constrained computational resources, fast response time, and stringent privacy/security requirements, so the online deployed federated learning models should be as lightweight as possible while their intrusion detection performance is competitive. To the best knowledge of the authors, it is the first attempt to automatically design a lightweight federated learning model for the intrusion detection issue in ICSs by an evolutionary NAS method.

(2) Five types of lightweight CNN neural architecture are considered the basic blocks to be combined and optimized in federated NAS for ICS intrusion detection. An efficient discrete encoding strategy is developed to describe the combination of five basic lightweight blocks and the specific discrete evolutionary operations under the framework of genetic algorithm are designed elaborately to guide the evolutionary process of an automated federated learning model.

(3) The proposed Fed-GA-CNN-IDS method is compared with the other nine IDS methods on three widely-used intrusion detection datasets in ICSs such as Gas Pipeline dataset, SWaT dataset, and WADI dataset. Three state-of-the-art manually-designed federated learning-based IDS methods are considered as the competitors including Federated Deep Learning (DeepFed) (Li et al., 2020), Federated Variational Autoencoder Support Vector Data Description (FedVAE-SVDD) (Huong et al., 2022) and federated learning-based anomaly detection architecture (FATRF) (Truong et al., 2022). Two federated NAS methods are considered as the competitors including RT-FedEvoNAS (Zhu and Jin, 2022) and FedNAS (He et al., 2020), which are originally developed for traditional image classification tasks. Four lightweight models in the Fed-GA-CNN-IDS method are considered as the competitors including MobileNet (Howard et al., 2017), MobileNetV2 (Sandler et al., 2018), Xception (Chollet, 2017), and ShuffleNetV2 (Ma et al., 2018). The experimental results demonstrate that the proposed Fed-GA-CNN-IDS method can obtain more lightweight models and better or at least competitive intrusion detection performance.

The rest of this paper is organized as follows. In Section 2, we review the recent research on federated intrusion detection schemes and federated learning-based neural architecture search. In Section 3, the proposed method is described. Section 4 presents the experimental results on three intrusion detection datasets. Finally, Section 5 concludes the work.

2. Related work

2.1. Federated learning

Federated learning (McMahan et al., 2017) is a distributed machine learning paradigm for privacy protection. It aims to train a high-quality centralized model while the training data is still distributed to its clients to achieve the goal of privacy protection. During the model training, for each round of training, each client independently computes an update of the current model based on its local data, and transmits this update to the central server, where these client updates are aggregated to compute a new global model. The framework of federated learning is given in Fig. 1. In this work, the most commonly used federated averaging (FedAvg) algorithm is applied to parameter aggregation for federated learning. Assuming that there are n Internet of Things (IoT) devices participating in federated training as the clients, the goal is to train an accurate intrusion detection model. w is the average value of the model weights obtained from the client training in FedAvg, and the detailed expression is given in (1).

$$w = \frac{\sum_{i=1}^{n} w_{\mathrm{i}}}{n} \tag{1}$$

where w_i represents the ith model weights obtained from IoT client training. After the FedAvg algorithm, the model weight after federation training should be updated to w.

2.2. Federated learning-based intrusion detection methods

The deep learning model has become the preferred solution for most IDSs because of its powerful fitting and learning capabilities. Additionally, as a promising tool to solve data silos and privacy security issues, federated learning has been widely adopted in a variety of fields. Therefore, the combination of federated learning and deep learning for intrusion detection of ICS has become the first choice of many scholars (Ghimire and Rawat, 2022). Federated learning used in intrusion detection scenarios can not only protect data privacy and security, but also has high classification performance for cyber-attacks (Popoola et al., 2021; Abdel-Basset et al., 2021; Li et al., 2022a).

To detect cyber threats against cyber–physical systems (CPS), Li et al. (2020) proposed a federated deep learning scheme, named DeepFed by using CNN and gated recurrent units. Additionally, a secure communication protocol based on the Paillier cryptosystem was constructed to protect the security and privacy of the model parameters during the training process. Huong et al. (2022) proposed a method called FedVAE-SVDD. By combining explainable artificial intelligence (XAI) to explain the predictions of anomaly detection algorithms, a federated learning-based ICS explainable anomaly detection (FedeX) was presented to detect and analyze anomalies in ICS. FedeX is a combination of variational autoencoder (VAE), where federated learning is considered as a solution to missing training data and support vector data description (SVDD) is used as an automatic threshold determination. In Truong et al. (2022), Huong et al. proposed a robust distributed anomaly detection architecture called FATRF by using a hybrid design of federated learning, Autoencoder, Transformer, and Fourier mixing sublayer. The authors demonstrate that this distributed architecture is lightweight, consumes little CPU and memory, and has low communication costs in terms of bandwidth consumption, which makes deployment on edge devices with limited computing capacity feasible.

In Ruzafa-Alcázar et al. (2023), a comprehensive evaluation of differential privacy techniques was proposed during the training of federated learning-based IDS for the Industrial Internet of Things (IIoT). Zhang et al. (2021) proposed an unsupervised time series anomaly detection framework in a federated fashion by utilizing the training data distributed at the edge to train a shared variational automatic encoder (VAE) based on convolutional gated recurrent unit (ConvGRU) model. This model can capture the features and temporal dependencies in multivariate time series data for representation learning and anomaly detection tasks. In Huong et al. (2021), a federated anomaly detection architecture was presented based on a model composed of VAE and long short-term memory (LSTM), which not only has higher detection performance, but also saves the bandwidth consumption of the transmission link between the edge and the cloud.

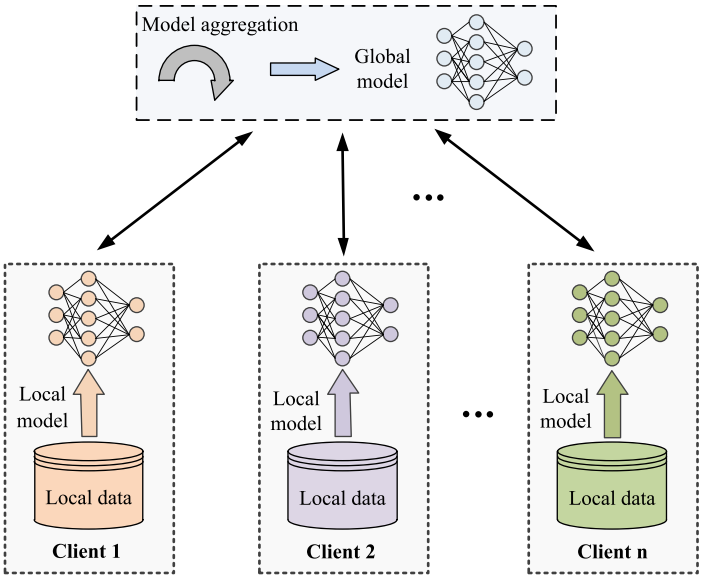

Fig. 1. The framework of federated learning.

Huang et al. (2022) developed a general global detection model to synergistically improve the performance of individual local models against cyber-attacks. In Li et al. (2022b), Transformer, Paillier cryptosystem and federated learning have been combined to develop a false data injection attack (FDIA) detection method based on secure federated deep learning. Tran et al. (2023) proposed an efficient cross-silo federated learning method with strong privacy preservation for FDIA detection in smart grids.

For challenging issues such as unbalanced data distribution and difficult data labeling in the federated learning system, Aouedi et al. (2022) proposed a new federated semi-supervised learning scheme that exploits both unlabeled and labeled data in a federated fashion. In Naeem et al. (2023), a novel secure orchestration framework was designed for federated learning-empowered semi-supervised active learning (FL-SSAL), where the federated clients contain unlabeled samples and a small set of labeled data. Telikani et al. (2022) proposed EvolCostDeep, which is a hybrid model of stacked autoencoders (SAE) and CNN with a new cost-dependent loss function. To alleviate the communication delay limitation of federated learning, Li et al. (2023) proposed an efficient IDS based on federated learning called dynamic weighted aggregation federated learning (DAFL), which can better detect intrusions with less communication overhead.

However, the existing federated learning-based intrusion detection methods still face efficiency, robustness, and security challenges (Cui et al., 2021). Li et al. (2021) proposed a flexible and robust aggregation rule called auto-weighted geometric median (AutoGM), which is robust to both model poisoning and data poisoning attacks. In Friha et al. (2023), a secure, decentralized, Differentially Private (DP) federated learning-based IDS (2DF-IDS) was proposed to secure smart industrial facilities. Liu et al. (2020) proposed an attention mechanism-based CNN long-short-term memory (AMCNN-LSTM) model to accurately detect anomalies under the framework of federated learning, and a gradient compression mechanism based on Top-k selection is used to improve communication efficiency. In Mothukuri et al. (2021), a federated learning-based anomaly detection method was developed by using a gated recurrent unit (GRU) model. Taheri et al. (2020) proposed a robust federated learning-based architecture called Fed-IIoT for detecting Android malware applications in IIoT. On the client side, the benign data is maliciously floated through the Generative Adversarial Networks (GAN) and federated Generative Adversarial Networks (FedGAN) methods to generate the poisoned samples, and the avoiding anomaly in aggregation by a GAN network (A3GAN) defense algorithm is proposed on the server side to avoid aggregation anomalies. Table 1 briefly summarizes the existing works on intrusion detection based on federated learning. As shown in Table 1, the existing works have illustrated the effectiveness of federated learning for IDS under different scenarios. However, these federated learning-based IDS are manually designed by relying on the extensive experiences of designers and are not applicable in different scenarios flexibly.

2.3. Federated neural architecture search

It is not optimal to directly use the predefined deep learning model for federated training, because the model developers cannot observe the local data to build highly accurate and efficient models. NAS is an emerging technology to automatically search for a good neural architecture for different tasks. Thus, NAS is promising for federated learning, which can automatically search for global and individualized models. The framework of NAS methods involves three dimensions, namely search space, search strategies, and performance estimation strategies (Yu et al., 2019; Zhu et al., 2021). Search space is a collection of neural architectures that has important effects on the performance and search efficiency. The search strategy defines the method that serves to automatically design the optimized neural architecture. Specifically, these search strategies are mainly divided into the following three categories: (1) NAS based on reinforcement learning (Zoph and Le, 2016), (2) NAS based on evolutionary algorithms (Huang et al., 2023), (3) NAS based on gradient descent (Liu et al., 2018). To illustrate the second category, Huang et al. (2023) developed an automatic architecture design method of convolutional neural networks (CNN) based on differential evolution (abbreviated as DE-CNN) for the intrusion detection issue in ICS. The authors designed three basic units such as ResNetBlockUnit, DenseNetBlockUnit and PoolingUnit, and then encoded the architecture parameters of CNN into a population, and performed off-line architecture optimization through population evolution operation to obtain the best CNN model. These search strategies aim to find architectures that achieve high performance on test datasets. To effectively guide the search, some approaches leverage performance estimation strategies to assess the quality of candidate architectures.

Recently, some researchers have started to apply the NAS idea in federated learning. The existing works of federated NAS are concentrated in the field of image processing and classification. Yu et al.

Table 1
Summary of the Existing Works on Intrusion Detection based on Federated Learning.

Method	Federated learning mechanism	Basic neural network model	Application
DeepFed (Li et al., 2020)	Privacy-preserving federated learning	CNN and GRU	Industrial CPS
FedVAE-SVDD (Huong et al., 2022)	FedAvg	VAE and SVDD	ICS
FATRF (Truong et al., 2022)	FedAvg	AE and Transformer	ICS
FDL (Popoola et al., 2021)	FedAvg	DNN	IoT-Edge Devices
Fed-TH (Abdel-Basset et al., 2021)	FedAvg	Multiscale convolutions and a GRU-based AE	Industrial cyber–physical system
FedLog (Li et al., 2022a)	Customizable and communication-efficient federated learning, Masked federated learning	TCN-ACNN	Large-scale IoT
DP-enabled FL (Ruzafa-Alcázar et al., 2023)	Privacy-preserving federated learning, FedAvg, Fed+	DNN	IIoT
FedAnomaly (Zhang et al., 2021)	Federated deep generative model	VAE, ConvGRU	CPS
An IIoT decentralized architecture (Huong et al., 2021)	FedAvg	VAE-LSTM	IIoT-based manufacturing systems
EEFED (Huang et al., 2022)	Personalized federated learning	CNN	CPS
SecFed (Li et al., 2022b)	Secure federated learning	Transformer	Smart grid
Privacy-preserving FL protocol (Tran et al., 2023)	Privacy-Enhancing Cross-Silo federated learning	LSTM	Smart grid
Federated semisupervised approach (Aouedi et al., 2022)	FedAvg	AE, fully connected neural network	IIoT
FL-SSAL (Naeem et al., 2023)	FedAvg	Active learning	Zero touch network and service management (ZSM)
EvolCostDeep, DeepIDSFog (Telikani et al., 2022)	Fog computing-enabled framework	SAE and CNN	IIoT
NIDS DAFL (Li et al., 2023)	Dynamic weighted aggregation federated learning	CNN	Digital network
AutoGM_FL, FL-IIoT (Li et al., 2021)	Auto-weighted geometric median based FL	MLP and CNN	IIoT
2DF-IDS (Friha et al., 2023)	Decentralized and differentially private federated learning	DNN	IIoT
On-device FL (Liu et al., 2020)	On-device federated learning	AMCNN-LSTM	IIoT
Fed-IIoT, FeDGAN (Taheri et al., 2020)	Robust federated learning, adversarial federated learning	GAN, A3GAN	IIoT malware

Table 2
Summary of the Existing Works on Federated NAS.

Method	Federated learning mechanism	Basic neural network model	Dataset
RT-FedEvoNAS (Zhu and Jin, 2022)	FedAvg	CNN	CIFAR10, CIFAR100, Street View House Numbers (SVHN), Pathmnist, and Tiny Imagenet
FedNAS (He et al., 2020)	FedAvg	CNN	CIFAR-10
RaFL (Yu et al., 2022)	Traditional federated learning	MobileNet-v2/v3, ResNet, VGG-11/16	CIFAR-10/100 and FEMNIST
FNAS, DP-FNAS (Singh et al., 2020)	Gradient-based federated learning	CNN	CIFAR-10
FDNAS, CFDNAS (Zhang et al., 2022)	FedAvg	Normal nets	CIFAR-10, FEMNIST, ImageNet
CIT2FR-FL-NAS (Liu et al., 2022)	FedAvg	Convolutional IT-2 fuzzy rough neural network	LC25000 lung and colon histopathological image dataset
SPIDER (Mushtaq et al., 2021)	Personalized federated learning	Convolutional network or recurrent network	CIFAR-10
Multi-objective evolutionary FL (Zhu and Jin, 2019)	FedAvg	MLP and CNN	MNIST

(2022) proposed resource-aware federated learning for resource-diverse edge devices by using NAS. Singh et al. (2020) proposed a federated NAS, where the participants in federated learning exchange the gradients of architecture variables to jointly search for the differentiable architectures. Furthermore, a differentially-private FNAS (DP-FNAS) was used to preserve privacy while searching for high-performance neural architectures by adding random noises to the gradients of architectural variables. Zhang et al. (2022) suggested federated direct NAS (FDNAS) and cluster FDNAS (CFDNAS) to fit diverse artificial IoT scenarios while preventing private information by integrating federated learning and NAS. Liu et al. (2022) proposed a convolutional interval type-2 fuzzy rough federated learning model based on NAS (CIT2FR-FL-NAS), which uses an improved multi-objective evolutionary algorithm for neural architecture selection to reduce model complexity while achieving high accuracy. In He et al. (2020) and Mushtaq et al. (2021), FedNAS and NAS-based personalized federated learning were proposed to solve the Non-IID and invisible data, respectively.

On the one hand, in federated learning systems, a large number of model parameters need to be transferred between server and client, which increases the high demand for communication resources. On

the other hand, training large-scale deep learning models such as deep neural networks in federated learning requires a lot of computing resources, which may be unrealistic for some edge devices with weak computing performance. The problem becomes even more serious when it comes to deep NAS in federated learning. Facing these challenges, Zhu and Jin (2019) proposed a multi-objective evolutionary NAS-based federated learning method based on the multilayer perceptron (MLP) and CNN by simultaneously minimizing the communication cost and the global model testing error. The MNIST dataset is tested to show the performance. Since then, Zhu and Jin (2022) proposed an evolutionary approach for real-time federated NAS (RT-FedEvoNAS), which not only optimizes the performance of the model, but also reduces the local payload. During the search process, a double-sampling technique is introduced, that is, for each individual, only one randomly sampled sub-model is transmitted to multiple randomly sampled clients for training. In this way, the computational and communication costs required for evolutionary optimization are effectively reduced.

To clearly show the existing works of federated NAS, we summarize these works in Table 2. From Table 2, we can see the effectiveness of the combination of federated learning and NAS in the image domain. However, none of the existing works uses the federated NAS to solve intrusion detection of ICS, which is facing some challenges including constrained computational resources, fast response time, and privacy requirements. Thus, the integration of federated learning and NAS for intrusion detection of ICS is what currently lacks and that our method solves.

3. Methodology

3.1. Problem definition

We select K clients from total clients under the federated learning setting. The data between clients are assumed to be independently and identically distributed (IID). Denote the local data of kth ($1 \le k \le K$) client as D_k. Divide D_k into training dataset $D_k^{(tr)}$ and validation dataset $D_k^{(val)}$, and use these training datasets to jointly train a supervised learning task. The goal is to find a neural architecture A that performs the best on $D_k^{(val)}$ through federated training by maximizing the F1-score. Fig. 2 presents the federated NAS with five basic lightweight neural architectures of CNN to be combined and optimized for ICS intrusion detection. The federated NAS for ICS intrusion detection problem can be formulated as:

$$\begin{aligned} &\max_A \quad \frac{1}{K}\sum_{k=1}^{K} F(D_k^{(val)},\ A,\ w_A) \\ &\text{s.t.} \quad w_A = \arg\min_{w} \quad \frac{1}{K}\sum_{k=1}^{K} L(D_k^{(tr)},\ A,\ w) \end{aligned} \tag{2}$$

where w represents the weight of the model A, which is to be combined and optimized based on five basic lightweight neural architectures of CNN. $L(D_k^{(tr)}, A, w)$ is the training loss of client k. $F(D_k^{(val)}, A, w_A)$ is the F1-score of the $D_k^{(val)}$ in the training model. For a given A, we first obtain the optimal model weight w_A that minimizes the average training loss through federated training. Then, evaluate the average F1-score of the training model in the validation dataset.

3.2. Basic blocks

For the proposed Fed-GA-CNN-IDS method, the overall architecture of the deep learning model is shown in Fig. 3. The composition of the model is Input, Convolution Block, two Combination Blocks, Average Pooling Layer, Fully Connected Layer and Output. The Convolution Block consists of a 1×3 1D convolution layer, a batch normalization layer, and ReLU activation function, which is to increase the number of input channels. For example, when the input dimension is 1×51, it will be changed to 8×51 through the operation of Convolution Block. The Combination Block is a new type of combination proposed in this paper and it is a key block that constitutes the architecture of neural network model, with doubled output channels and halved data dimension compared to its input. In experiments, the overall architecture can achieve good performance using two Combination Blocks. For better performance, the number of Combination Blocks can be appropriately increased. Fig. 4 is the composition of the Combination Block. A Combination Block contains five basic blocks, in which the basic blocks are divided into normal blocks and reduction blocks. A normal block has the same output dimension as the input, while the output of a reduction block has double channels and half the data dimension. As shown in Fig. 4, the Combination Block consists of four normal blocks and one reduction block. Considering the limited resources of federated clients, we endeavor to make each basic block as lightweight as possible. The research of lightweight neural networks has made great progress in machine vision tasks such as image classification and object detection. Many lightweight CNN models, such as MobileNet (Howard et al., 2017), MobileNetV2 (Sandler et al., 2018), Xception (Chollet, 2017), and ShuffleNetV2 (Ma et al., 2018), ensure the performance of the model while greatly reducing the parameters of the model, making the model more lightweight, and the storage and computing on embedded devices are more efficient. Based on the above lightweight CNN models, we extend the basic block adopted in these lightweight models to the industrial control intrusion detection task. Therefore, five sets of lightweight CNN blocks are considered as the candidate basic blocks. Each block is divided into normal block and reduction block. The specific structure of each group of blocks is described as follows:

(1) Fig. 5(a) shows the first group of candidate basic block (termed as Block1). The left sub-image is the normal block of Block1, where the normal block does not perform any operations on the input. When a Combination Block contains N ($1 \le N \le 4$) normal blocks of Block1, it means that the number of basic block actually involved in the calculation in the Combination Block is $5-N$, so the actual number of layers in the Combination Block is $5-N$. The above design can make the number of basic block actually contained in the Combination Block variable, so that the number of layers of the global model is variable. The right sub-image is the reduction block of Block1, which consists of a 1×3 1D convolution layer and ReLU activation function.

(2) Fig. 5(b) shows the second group of candidate basic block (termed as Block2). Block2 consists of a 1×3 depthwise convolution layer, followed by a batch normalization layer and ReLU activation function, and a 1×1 pointwise convolution, followed by a batch normalization layer and ReLU activation function as well. This structure was proposed in MobileNet (Howard et al., 2017), and MobileNet is mainly used for mobile computing models in the image field. It changes the traditional convolution operation into a two-layer convolution operation. Under the condition of ensuring the accuracy rate, the calculation time is reduced to 1/9 of the original version, and the calculation parameters are reduced to 1/7 of the original version. Similarly, this structure is also applicable to 1D convolution, which can also reduce the calculation time and the number of calculation parameters (Kriman et al., 2020). Therefore, it can make the neural architecture more lightweight. We distinguish between normal block and reduction block by setting the stride of the convolution kernel. When *stride* is set to 1, it means the normal block of Block2, and when *stride* is set to 2, it means the reduction block of Block2.

(3) Fig. 5(c) shows the third group of candidate basic block (termed as Block3). The left sub-image is the normal block of Block3, and the right sub-image is the reduction block of Block3. Block3 consists of a 1×1 pointwise convolution, followed by a batch normalization layer and ReLU activation function, a 1×3 depthwise convolution layer, followed by a batch normalization layer and ReLU activation function, and a 1×1 pointwise convolution, followed by a batch normalization layer and linear activation function. This structure was proposed in MobileNetV2 (Sandler et al., 2018) as an improved version of MobileNet and it has higher accuracy and a smaller model compared

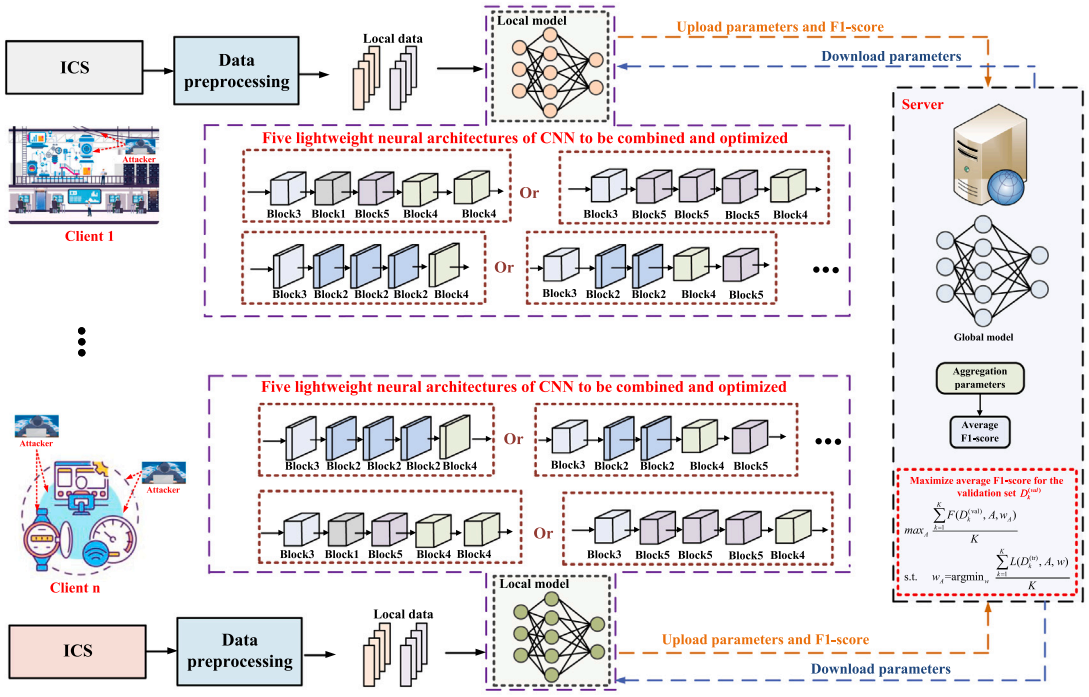

Fig. 2. Federated NAS with five lightweight neural architectures of CNN to be combined and optimized for ICS intrusion detection.

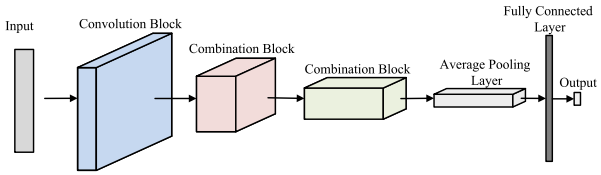

Fig. 3. The overall structure of the model.

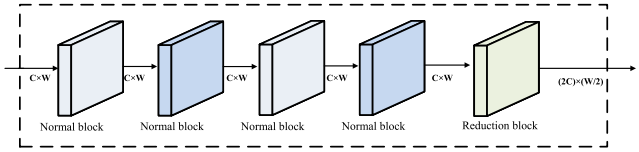

Fig. 4. The composition of the Combination Block.

with MobileNet. In MobileNetV2, ReLU6 is used instead of the previous ReLU, and in the last 1x1 pointwise convolutional layer, it uses a linear activation function instead of the ReLU activation function.

(4) Fig. 5(d) shows the fourth group of candidate basic block (termed as Block4). Block4 consists of a 1×1 pointwise convolution, followed by a batch normalization layer and ReLU activation function, and a 1×3 depthwise convolution layer, followed by a batch normalization layer and ReLU activation function as well. This structure was proposed in Xception (Chollet, 2017), in the reverse order of composition of Block2. The experiments in Chollet (2017) prove that such a combination also can make the model more lightweight, while ensuring the accuracy of the model. Similar to Block2, when *stride* is set to 1, it means the normal block of Block4, and when *stride* is set to 2, it means the reduction block of Block4.

(5) Fig. 5(e) shows the fifth group of candidate basic block (termed as Block5). The left sub-image is the normal block of Block5, and the right sub-image is the reduction block of Block5. The "Channel Split" operation indicates that the input is divided into two parts according to the number of channels, that is, the input with a dimension of $C \times W$ is randomly divided into two with a dimension of $(C/2) \times W$, and the "Channel Shuffle" operation means that randomly rearrange the channels of the output after the "Concat" operation. Such an architectural combination was proposed in ShuffleNetV2 (Ma et al., 2018). In Ma et al. (2018), the authors have proved that in lightweight models, element-wise operations take up a lot of time, especially on GPUs, through a lot of experiments. Therefore, they believe that the impact of element-wise operations is non-negligible and this is also the reason why the skip connection operation is deleted in the basic blocks we used above. In Ma et al. (2018), the authors used the "Concat" operation instead of the element-wise add operation, and that is, after convolution, two branches are connected instead of addition. Therefore, we adopt the same replacement in the Combination Blocks.

3.3. Evolutionary algorithm-based lightweight architecture

3.3.1. Algorithm overview

The overall framework of the proposed Fed-GA-CNN-IDS method is shown in Fig. 6. Algorithm 1 gives the specific process as follows: First, initialize the population to generate the first generation population S_0 (line 1), and then evaluate the fitness F_s of the population S_0 (line 4). Next, generate the offspring population Q from population S through crossover operation and mutation operation with the crossover probability c and the mutation probability m (line 6), and then evaluate the fitness F_q of the offspring population (line 7). According to individual fitness, use five-element competition selection to carry out environmental selection on the parent population and the offspring population to generate the next generation population S (line 8). After the population iteration it reaches the maximum number of generation G, select the individual $Indi_{best}$ with the best fitness from the current population S (line 11), then perform federated training on $Indi_{best}$

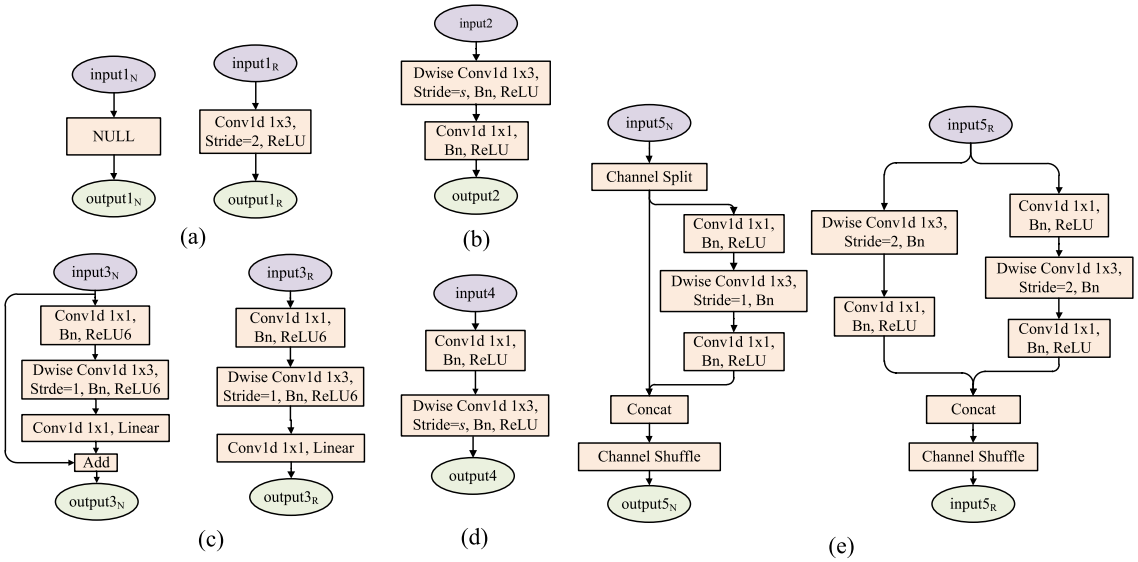

Fig. 5. Composition of the lightweight model. (a) Block1; (b) Block2; (c) Block3; (d) Block4; (e) Block5.

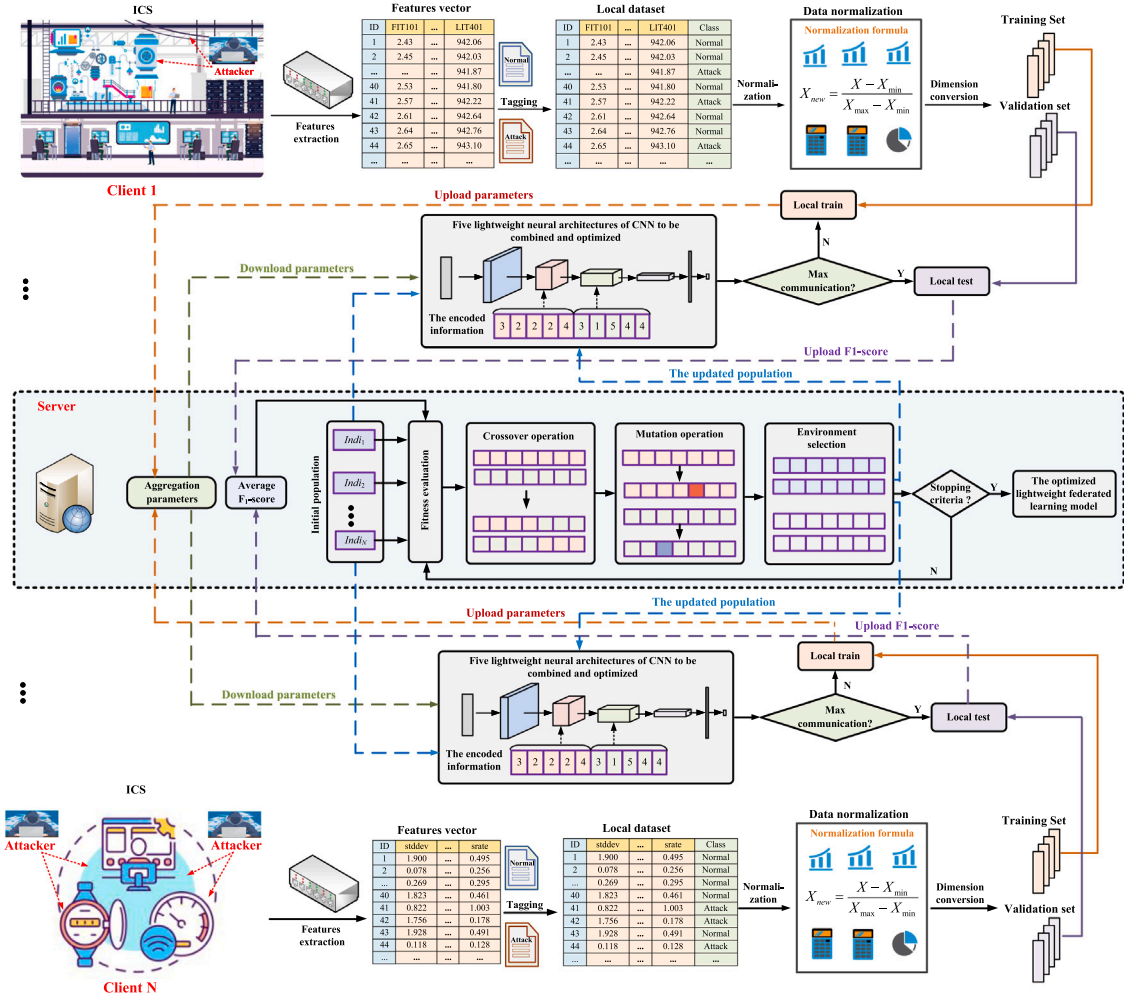

Fig. 6. The overall framework of the proposed Fed-GA-CNN-IDS method.

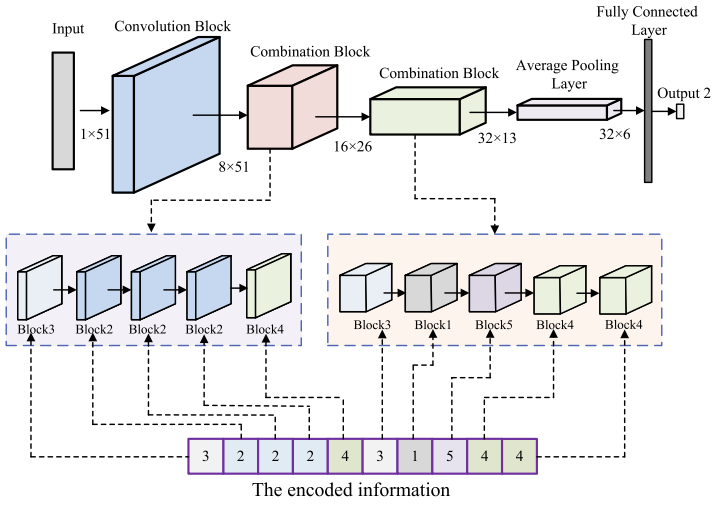

Fig. 7. Schematic diagram of the model composition with the encoded information as [3, 2, 2, 3, 5, 4, 1, 5, 4, 4].

corresponding to the optimized neural architecture model (line 12). Finally, evaluate the intrusion detection performance of the optimized model (line 13).

Algorithm 1: The Framework of Fed-GA-CNN-IDS Method

Input: The population size N, maximum number of generation G, the crossover probability c, and the mutation probability m.
Output: The optimized neural architecture and its intrusion detection performance.

1 $S_0 \leftarrow$Initialize the population and generate N individuals by (3);
2 $it \leftarrow 0$;
3 $S \leftarrow S_{it}$;
4 $F_S \leftarrow$ Evaluate the fitness of individuals in S_0;
5 **for** $it < G$ **do**
6 $\quad Q \leftarrow$ Produce offspring based on S, c, and m;
7 $\quad F_q \leftarrow$ Evaluate the fitness of individuals in Q;
8 $\quad S \leftarrow$ Environmental selection and generate new population;
9 $\quad it \leftarrow it + 1$;
10 **end**
11 $Indi_{best} \leftarrow$ Select the best individual with the best fitness from S;
12 Perform federated training on $Indi_{best}$;
13 Evaluate intrusion detection performance based on the optimized neural architecture $Indi_{best}$.

3.3.2. Population initialization

For the initialization of the first generation population S_0 with N individuals, each individual is encoded as $Indi_i = [B_1, \ldots, B_i, \ldots, B_{10}]$, where B_i is a randomly generated integer in the range [1, 5] given by (3).

$$B_i = randint(1,5), \quad 1 \le i \le 10 \tag{3}$$

Fig. 7 shows an example of the neural architecture of a randomly generated individual, and the coding information of the individual is: [3, 2, 2, 3, 5, 4, 1, 5, 4, 4].

3.3.3. Fitness evaluation

Algorithm 2 gives the fitness evaluation process of the population. The specific process is given as follows: The federated learning server sends the encoded information of each individual $Indi_i$ in S to each client C_K in the federated learning (lines 6–7), while the client transforms $Indi_i$ into the corresponding deep learning model locally, trains with local datasets and returns the update parameters θ_k (lines 12–13). After the server receives the update parameters returned by all clients, the server performs parameter aggregation to obtain new model parameters θ (line 15), and then sends $Indi_i$ and new parameters θ to

Algorithm 2: Fitness Evaluation

Input: The population S, maximum number of rounds T, federated learning client set C_K.
Output: The fitness F.

1 **Server:**
2 $F \leftarrow \emptyset$;
3 **for** *each* Ind_i *in* S **do**
4 $\quad$ **for** $t \leftarrow 1$ *to* T **do**
5 $\quad\quad$ **if** $t == 1$ **then**
6 $\quad\quad\quad \theta_t \leftarrow \emptyset$;
7 $\quad\quad\quad$ Send the encoded information of each individual Ind_i to clients in C_K;
8 $\quad\quad$ **else**
9 $\quad\quad\quad$ Send the individual Ind_i and parameters θ to clients in C_K;
10 $\quad\quad$ **end**
11 $\quad\quad$ **for** *each* k *in* C_K **do**
12 $\quad\quad\quad \theta_k \leftarrow$ The client k trains locally and returns update parameters;
13 $\quad\quad\quad \theta_t \leftarrow \theta_t \cup \theta_k$;
14 $\quad\quad$ **end**
15 $\quad\quad \theta \leftarrow$ According to FedAvg algorithm, aggregate the parameters in θ_t;
16 $\quad$ **end**
17 $\quad$ Send Ind_i and θ to clients participating in federated training;
18 $\quad f \leftarrow \emptyset$;
19 $\quad$ **for** *each* k *in* C_K **do**
20 $\quad\quad f_k \leftarrow$ Client k tests locally and returns F1-score as individual fitness;
21 $\quad\quad f \leftarrow f \cup f_k$;
22 $\quad$ **end**
23 $\quad f_{indi} \leftarrow$ Obtain the weighted average of the fitness in f and set it as the final fitness of the individual;
24 $\quad F \leftarrow F \cup f_{indi}$;
25 **end**
26 Return F;

all clients (line 9). The client transforms $Indi_i$ into the corresponding deep learning model locally, initializes the model with new parameters, trains with local datasets and returns the update parameters again (lines 12–13). The server performs parameter aggregation again to obtain new model parameters until the number of training rounds is reached. Then, the client receives the final model update parameters (line 17). Initialize the deep learning model locally with the final model parameters and evaluate the model with the test set. Then, obtain relevant performance indices and calculate the F1-score of the model. Finally, return F1-score to the server as the fitness of individual $Indi_i$ (lines 20–21). After the server receives the fitness of the individual returned by all clients, the server performs a weighted average of all fitness to obtain the fitness f_{indi} of the individual $Indi_i$ in population S (lines 23–24). Finally, we can obtain the fitness F of the population S.

3.3.4. Offspring generation

Algorithm 3: Offspring Generation

Input: The population S, the crossover probability c, and the mutation probability m.
Output: The generated offspring Q.

1 $Q \leftarrow \emptyset$;
2 **for** $it \leftarrow 1$ *to* $N/2$ **do**
3 $Indi_1, Indi_2 \leftarrow$ Randomly choose two individuals from population S;
4 $r \leftarrow$ Generate a random number uniformly distributed between 0 and 1;
5 **if** $r < c$ **then**
6 $p \leftarrow$ Generate a random integer from 1 to 9;
7 $indi_{11}, indi_{12} \leftarrow$ After the p-th element of the individual $Indi_1$ is generated, we can divide $Indi_1$ into two parts $indi_{11}, indi_{12}$;
8 $indi_{21}, indi_{22} \leftarrow$ After the p-th element of the individual $Indi_2$ is generated, we can divide $Indi_2$ into two parts $indi_{21}, indi_{22}$;
9 $q_1 \leftarrow$ Splicing of $indi_{11}$ and $indi_{22}$;
10 $q_2 \leftarrow$ Splicing of $indi_{12}$ and $indi_{21}$;
11 **else**
12 $q_1 \leftarrow Indi_1$;
13 $q_2 \leftarrow Indi_2$;
14 **end**
15 **for** $i \leftarrow 1$ *to* 2 **do**
16 $r_i \leftarrow$ Generate a random number uniformly distributed between 0 and 1;
17 **if** $r_i < m$ **then**
18 $k \leftarrow$ Generate a random integer ranging from 1 to 10;
19 $b_k \leftarrow$ The value of the k-th element of individual $Indi_i$;
20 $p \leftarrow$ Generate a random integer ranging from 1 to 5;
21 **while** $b_k == p$ **do**
22 $p \leftarrow$ Generate a random integer ranging from 1 to 5;
23 **end**
24 Replace the value of the k-th element of individual $Indi_i$ with p;
25 **end**
26 **end**
27 $Q \leftarrow Q \cup q_1 \cup q_2$;
28 **end**
29 Return Q;

Algorithm 3 gives the detailed process of offspring generation. Firstly, randomly select two individuals from the population S as the parents, which are recorded as $Indi_1$ and $Indi_2$ (line 3), and then generate a random number r uniformly distributed between 0 and 1 (line 4). If r is less than the crossover probability c, then generate a random integer p ranging from 1 to 9 as the crossover point (line 6), where p represents that the crossover point is located in the pth element of individual $Indi_1$ and $Indi_2$. Perform single-point crossover operation on $Indi_1$ and $Indi_2$ to obtain the offspring individuals q_1 and q_2 (lines 7–10). Otherwise, keep the $Indi_1$ and $Indi_2$ unchanged (lines 12–13). Next, perform mutation operation on the offspring q_1. Firstly, generate a random number r_1 uniformly distributed between 0 and 1 (line 15). If r_1 is less than the mutation probability m, then generate a random integer k_1 ranging from 1 to 10 as mutation point (line 18), where k_1 represents that the mutation happens in the k_1-th element of individual q_1. Then, randomly select an integer between 1 to 5 that is different from the value of the k_1-th element in q_1 (lines 19–23), and utilize this integer to substitute the original value at the k_1-th position of q_1, maintaining the other values unchanged (line 24). If $r_1 \geq m$, the mutation operation will not be performed on q_1. The mutation operation of the offspring q_2 is similar. Continue performing the aforementioned procedures until the offspring population Q attains a size of N. Fig. 8 and Fig. 9 show an example of the crossover operation and the mutation operation, respectively.

3.3.5. Environment selection

Algorithm 4 is the process of environment selection for the population. The next-generation population S is generated through the environment selection operation. Firstly, merge the current population S_{it} and the offspring population Q as the merged population S_m (line 1). For the S_m, use five-element competition selection as the environment selection method. Randomly select five individuals from S_m (line 4), and select the individual with the highest fitness to enter the next generation population S (lines 5–6). Repeat this operation until the population size reaches N.

Algorithm 4: Environment Selection

Input: Current population S_{it}, offspring population Q, and population size N
Output: New generation population S

1 $S_m \leftarrow S_{it} \cup Q$;
2 $S \leftarrow \emptyset$;
3 **for** $i \leftarrow 1$ *to* N **do**
4 $F_5 \leftarrow$ Randomly select five individuals from merged population S_m;
5 $Ind_i \leftarrow$ Select the individual with the highest fitness from F_5;
6 $S \leftarrow S \cup Ind_i$;
7 **end**
8 Return S.

4. Experimental results

In this section, we design the experiments on three industrial datasets, i.e., Gas Pipeline, SWaT, and WADI collected from test platforms to demonstrate the superiority of the proposed Fed-GA-CNN-IDS method. First, we present the experiment settings including the experimental environment, dataset description and preprocessing, and performance indices. Then, three experiments are carried out to show the performance of Fed-GA-CNN-IDS method from different perspectives. In Experiment I, we select three state-of-the-art manually-designed federated learning-based IDS methods, i.e., DeepFed (Li et al., 2020), FedVAE-SVDD (Huong et al., 2022), and FATRF (Truong et al., 2022) as the competitors to demonstrate the advantages of the proposed GA-Fed-CNN-IDS as an automated federated learning-based IDS method.

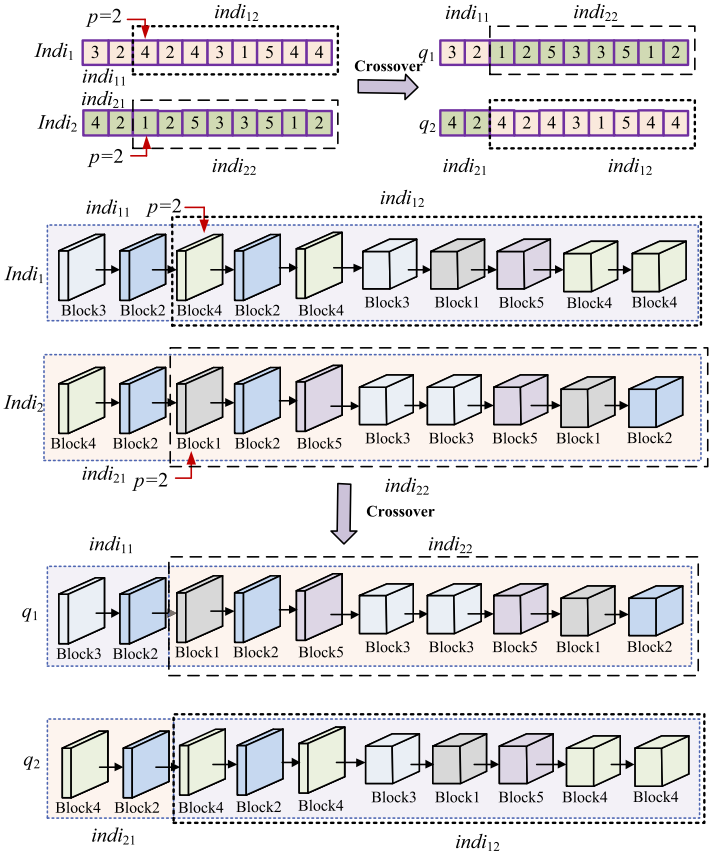

Fig. 8. An example of the crossover operation in the Fed-GA-CNN-IDS method.

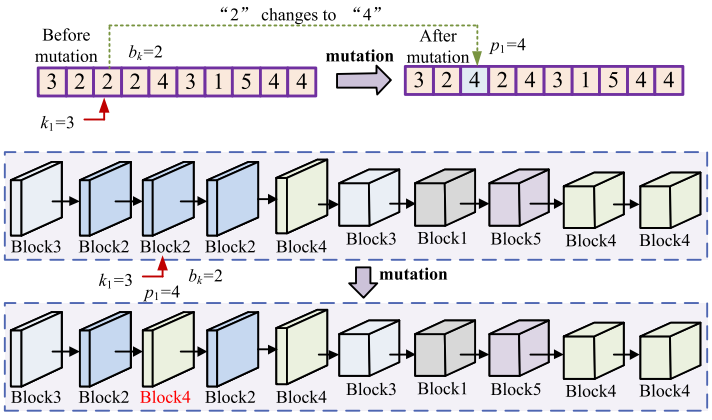

Fig. 9. An example of the mutation operation in the Fed-GA-CNN-IDS method.

In Experiment II, we select two federated NAS methods, i.e., RT-FedEvoNAS (Zhu and Jin, 2022) and FedNAS (He et al., 2020) as the competitors to prove the superiority of the proposed GA-Fed-CNN-IDS in automated design. In Experiment III, we select four lightweight models, i.e., MobileNet (Howard et al., 2017), MobileNetV2 (Sandler et al., 2018), Xception (Chollet, 2017), and ShuffleNetV2 (Ma et al., 2018), as the competitors to demonstrate the advantages of the GA-Fed-CNN-IDS method in the performance of lightweight.

In Experiment I, due to the usage of different datasets and target applications, as well as varying neural architectures and parameters for different problems, it is difficult to replicate every federated learning-based IDS method as the competitors. Therefore, we select three state-of-the-art manually-designed methods, i.e., DeepFed (Li et al., 2020), FedVAE-SVDD (Huong et al., 2022), and FATRF (Truong et al., 2022) from Table I as the competitors. DeepFed (Li et al., 2020) is a novel federated learning method that has significant advantages in detecting various types of cyber-attacks in industrial CPS. FedVAE-SVDD (Huong et al., 2022) is a novel method that has been validated on two datasets including SWaT dataset, demonstrating its effectiveness compared to the other fourteen existing federated learning methods. FATRF (Truong et al., 2022) can be considered as a lightweight federated learning method, and its effectiveness has been validated on four datasets including Gas Pipeline dataset and SWaT dataset, proving the efficacy of the model. Thus, these three state-of-the-art manually-designed federated learning-based IDS methods are considered as the competitors.

Table 3
Statistics of datasets.

Dataset	#Features	#Classes	#Samples
Gas Pipeline	26	2	970,19
SWaT	51	2	449,919
WADI	123	2	172,801

In Experiment II, from Table II, we can see that federated NAS methods are mainly focused on image classification tasks, making it difficult to directly apply them to intrusion detection problems in ICSs. In comparison to image classification tasks, ICS faces some challenges such as constrained computational resources, fast response time, and stringent privacy/security requirements. We need to redesign the encoding method, crossover operation, and mutation operation, making it difficult to extend each image classification method for solving intrusion detection problems. The RT-FedEvoNAS (Zhu and Jin, 2022) used an evolutionary algorithm and FedNAS (He et al., 2020) used gradient descent to automatically design the neural architectures in a federated learning framework on the image classification task. Therefore, we select these two representative methods from Table II as the competitors.

In Experiment III, since the proposed Fed-GA-Fed-CNN method is primarily based on the combination and optimization of blocks such as MobileNet (Howard et al., 2017), MobileNetV2 (Sandler et al., 2018), Xception (Chollet, 2017), and ShuffleNetV2 (Ma et al., 2018), we select these individual blocks as the competitors to demonstrate that the proposed Fed-GA-Fed-CNN method can achieve superior performance through combination and optimization.

4.1. Experiment settings

4.1.1. Dataset description and preprocessing

The Gas Pipeline (Morris and Gao, 2014) dataset consists of data gathered via the Supervisory Control And Data Acquisition (SCADA) system from a test platform for natural gas pipelines, conceived and constructed at Mississippi State University. It encompasses 97,018 data entries in total, with 61,155 entries classified as normal and 35,863 identified as data related to attacks. Every piece of data is characterized by 26 attributes alongside a single label.

The Secure Water Treatment (SWaT) system (Mathur and Tippenhauer, 2016) functions as a small-scale operational environment for simulating the processes of a substantial, up-to-date water treatment facility within a major city. Split SWaT dataset into two parts: The first part is gathered from 7 days of normal operation, and the second part is gathered from 4 days of 36 abnormal attacks.

The WADI (Water Distribution) system (Ahmed et al., 2017) serves as an expansion to the existing SWaT testing framework and features an array of components including chemical dosing mechanisms, booster pumps and valves, and instruments and analyzers. The dataset captures 16 consecutive days of operation, with recordings from 14 days under standard conditions and data from 2 days under attacks, summing up to 15 distinct attacks within the WADI infrastructure.

The statistics of these datasets are shown in Table 3. In each client, 80% of the datasets are used for training and 20% for testing. Specifically, for 5 consecutive samples with the same label, one of the samples is randomly selected and divided into the validation set, and the remaining 4 samples are divided into the training set. This operation is repeated until the entire local data is divided. This division can make the proportion of attack samples in the training set consistent with that in the test set.

For SWaT dataset and WADI dataset, the Min-Max normalization approach is employed to scale datasets to the range of [0, 1]. The specific formula is presented as follows:

$$X_i' = \frac{X_i - X_{imin}}{X_{imax} - X_{imin}} \tag{4}$$

Table 4
Experimental parameters.

Value	Meaning
N=20	The population size
G=20	Maximum number of generation
c=0.8	The crossover probability
m=0.2	The mutation probability
Tr=5	The local traning rounds
Tc=10	The communication rounds
lr=0.01	The learning rate

where X_i and X_i' denote the original data and the corresponding normalized data for the ith feature, respectively. X_{imax} and X_{imin} refer to the maximum and minimum values of X_i, respectively.

For Gas Pipeline dataset, the L2 Normalization method is employed to transform the dataset. The specific formula is presented as follows:

$$X_i' = \frac{X_i}{\sqrt{\sum_{i=1}^{n} X_i^2}} \tag{5}$$

where X_i and X_i' represent the original data and the corresponding normalized data for the ith feature, respectively.

During the data preprocessing stage, datasets from various sensors and actuators in the ICS are parsed and collected as local features, corresponding to "Features extraction" in Fig. 6. In this work, the scenario involves the same types of ICS collaboratively training a global model under a federated framework, which is then used for intrusion detection in individual ICS. Consequently, each ICS structure is the same and the meanings of the features in the data samples are consistent, while the local features for each ICS are distinct.

For five consecutive samples with the same label, samples 1, 2, 3, 4, and 5 are divided into clients 1, 2, 3, 4, and 5 respectively. Repeat this operation until the entire data set is divided. This operation ensures that local data between clients is the IID.

4.1.2. Experimental environment

The designed Fed-GA-CNN-IDS method is implemented on a server equipped with an Intel® Xeon® E5-2683v3@2.00 GHz processor, 256 GB memory and 4 A100-PCIE graphics cards, running on Ubuntu 18.04 LTS. In addition, we set up 5 distributed nodes in the server to simulate 5 clients of federated learning and share the computing resources of the server. Therefore, the communication delay and cost between the client and the server are not considered. Other parameters used in the experiments are given in Table 4.

4.1.3. Performance indices

In the experiments, to evaluate the performance of different intrusion detection methods, some commonly used performance indices are employed, including Recall, Precision (Pre), and F_1-score (F1). The detailed expressions are presented as follows:

$$\text{Pre} = \frac{TP}{TP + FP} \tag{6}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{F1} = \frac{2 \times \text{Pre} \times \text{Recall}}{\text{Pre} + \text{Recall}} \tag{8}$$

where TP indicates the number of instances accurately categorized as normal classification, while TN corresponds to the number of instances with abnormal characteristics rightly identified as abnormal classification. On the flip side, FP reflects the number of instances with abnormal features inaccurately classified as normal, and FN accounts for the number of normal instances that have been misclassified as abnormal classes, respectively.

Besides the aforementioned three performance indices, the number of parameters of the deep learning model is considered as another important performance index, termed as Params. Max-Memory and Time are used to evaluate the memory requirements and time consumption, respectively.

Table 5
Comparison of experimentals results on gas pipeline dataset.

Method	Pre	Recall	F1	Params
DeepFed (Li et al., 2020)	98.85	97.47	98.14	11 784
Fed-GA-CNN-IDS	**98.94**	**98.61**	**98.77**	**7202**

Table 6
Comparison of experimental results on SWaT dataset.

Method	Pre	Recall	F1	Params
FedVAE-SVDD (Huong et al., 2022)	97.18	**100**	**98.57**	*
FATRF (Truong et al., 2022)	93.89	97.75	95.78	*
DeepFed (Li et al., 2020)	98.39	95.14	96.74	19 898
Fed-GA-CNN-IDS	**98.54**	98.39	98.46	**7298**

* The number of model parameters is not given in the paper, and there is insufficient information to evaluate the number of parameters.

4.2. Experiment I: Comparison with three state-of-the-art manually-designed methods

In this subsection, we compare the performance of the proposed Fed-GA-CNN-IDS method with the state-of-the-art federated learning-based IDS methods in the industrial control domain. Table 5 shows the comparison between the experimental results of Fed-GA-CNN-IDS and DeepFed (Li et al., 2020) on the Gas pipeline dataset. Table 6 presents the experimental results of Fed-GA-CNN-IDS on the SWaT dataset by comparing with other works including FedVAE-SVDD (Huong et al., 2022), FATRF (Truong et al., 2022), and DeepFed (Li et al., 2020). In addition, Table 7 gives the experimental results of Fed-GA-CNN-IDS on the WADI dataset compared with DeepFed (Li et al., 2020). In order to make the comparison more clear, Figs. 10–12 visualize the comparison of Fed-GA-CNN-IDS with other competitors on Gap Pipeline dataset, SWaT dataset, and WADI dataset, respectively. Besides, Fig. 13 gives the comparison of the number of parameters of different models on three datasets.

From Tables 5–7 and Figs. 10–13, we can draw the following conclusions:

(1) For the intrusion detection results on Gap Pipeline dataset, Fed-GA-CNN-IDS method achieves better performance than DeepFed (Li et al., 2020) in terms of Pre, Recall, F1, and Params.

(2) For SWaT dataset, the proposed Fed-GA-CNN-IDS method outperforms FATRF (Truong et al., 2022) and DeepFed (Li et al., 2020) and obtains competitive intrusion detection performance with FedVAE-SVDD (Huong et al., 2022).

(3) For WADI dataset, the performance indices values obtained from Fed-GA-CNN-IDS method are better than those obtained from DeepFed (Li et al., 2020), although Pre obtained by DeepFed is slightly better than that by Fed-GA-CNN-IDS.

(4) It should be noted that the Params obtained by Fed-GA-CNN-IDS on all datasets is much better than DeepFed (Li et al., 2020), which indicates that Fed-GA-CNN-IDS is much more lightweight than DeepFed and more appropriate to solve the intrusion detection issue in ICSs with constrained computational resources, fast response time, and stringent privacy/security requirements.

(5) As an automated federated learning, Fed-GA-CNN-IDS achieves better comprehensive performance than other manually-designed federated learning-based IDS methods such as FedVAE-SVDD (Huong et al., 2022), FATRF (Truong et al., 2022), and DeepFed (Li et al., 2020). The reason is that using the evolutionary NAS technique can obtain a better neural architecture for the considered intrusion detection issue in ICS.

4.3. Experiment II: Comparison with two federated NAS methods

In order to further demonstrate the performance of Fed-GA-CNN-IDS, a federated evolutionary NAS method, i.e., RT-FedEvoNAS (Zhu and Jin, 2022) and NSA based on gradient descent, i.e., FedNAS (He et al., 2020) are viewed as the competitors. Because RT-FedEvoNAS and FedNAS are originally developed for traditional image classification tasks, we extend them to solve intrusion detection problems to compare with the proposed Fed-GA-CNN-IDS. For the convenience of a fair comparison, the parameter settings of RT-FedEvoNAS and FedNAS are the same as the Fed-GA-CNN-IDS.

Table 7
Comparison of experimental results on WADI dataset.

Method	Pre	Recall	F1	Params
DeepFed (Li et al., 2020)	**98.88**	96.89	97.87	24 650
Fed-GA-CNN-IDS	98.55	**98.65**	**98.60**	**7778**

Table 8
Comparison of three different methods on gas pipeline dataset.

Method	Pre	Recall	F1	Params
RT-FedEvoNAS	98.46	**98.97**	98.71	3098690
FedNAS	97.34	96.86	97.1	23683
Fed-GA-CNN-IDS	**98.94**	98.61	**98.77**	**7202**

Table 9
Comparison of three different methods on SWaT dataset.

Method	Pre	Recall	F1	Params
RT-FedEvoNAS	**99.14**	98.13	**98.63**	1980162
FedNAS	98.82	95.99	97.38	22748
Fed-GA-CNN-IDS	98.54	**98.39**	98.46	**7298**

Table 10
Comparison of three different methods on WADI dataset.

Method	Pre	Recall	F1	Params
RT-FedEvoNAS	**98.74**	98.3	98.52	2148482
FedNAS	98.64	97.62	98.12	31561
Fed-GA-CNN-IDS	98.55	**98.65**	**98.60**	**7778**

Tables 8–10 present the comparison of Fed-GA-CNN-IDS, RT-FedEvoNAS, and FedNAS on Gas Pipeline dataset, SWaT dataset, and WADI dataset, respectively. Figs. 14–16 show the corresponding compared results of Pre, Recall, and F1 on three different datasets. In addition, Fig. 17 shows the comparison of Params obtained by Fed-GA-CNN-IDS, RT-FedEvoNAS, and FedNAS.

From Table 8 and Fig. 14, we can see that Fed-GA-CNN-IDS achieves sightly better performance than RT-FedEvoNAS in terms of Pre, F1, and Params on Gas Pipeline dataset. Especially, the Params value of Fed-GA-CNN-IDS is 7202, which is much smaller than 309,8690 obtained by RT-FedEvoNAS. By comparing with FedNAS, Fed-GA-CNN-IDS outperforms FedNAS on all four performance indices including Pre, Recall, F1, and Params. Similarly, from Table 9 and Fig. 15, it can be found that Fed-GA-CNN-IDS obtains similar intrusion detection performance with RT-FedEvoNAS in terms of Pre, Recall, and F1 on SWaT dataset. Moreover, Fed-GA-CNN-IDS is much more lightweight than RT-FedEvoNAS because Params of Fed-GA-CNN-IDS is 7298, which is much smaller than 198,0162 obtained by RT-FedEvoNAS. By comparing with FedNAS, Fed-GA-CNN-IDS achieves better performance indices on Recall, F1, and Parames, although the Pre is slightly worse than FedNAS. For WADI dataset, Fed-GA-CNN-IDS sightly outperforms than RT-FedEvoNAS and FedNAS according to the compared results in Table 10 and Fig. 16. Also, Params of Fed-GA-CNN-IDS is much better than that of RT-FedEvoNAS and FedNAS. From Fig. 17, we can clearly see that the Fed-GA-CNN-IDS obtains much lightweight model than RT-FedEvoNAS and FedNAS in all three industrial control datasets. Overall, Fed-GA-CNN-IDS can be viewed as the effective method to deal with the challenges in intrusion detection issue of ICSs while the intrusion detection performance is competitive with RT-FedEvoNAS and FedNAS.

In addition, we further compare the time consumption and memory requirements of the above three methods during federated architecture

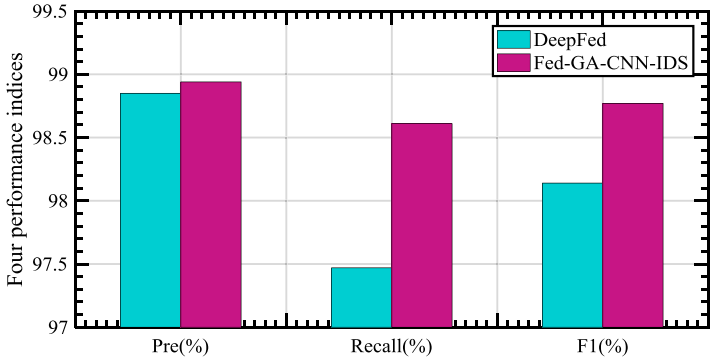

Fig. 10. The comparison of Pre, Recall, and F1 on Gas Pipeline dataset.

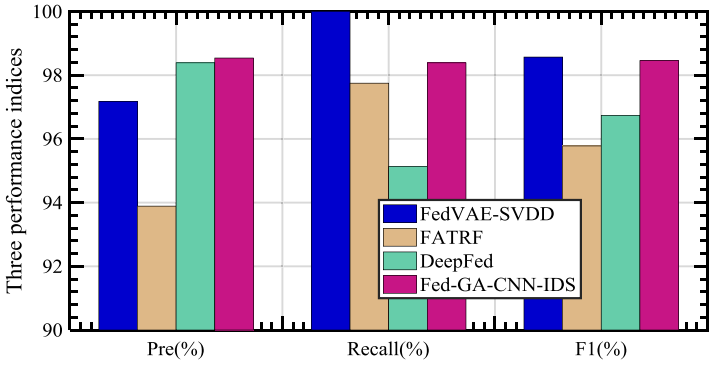

Fig. 11. The comparison of Pre, Recall, and F1 on SWaT dataset.

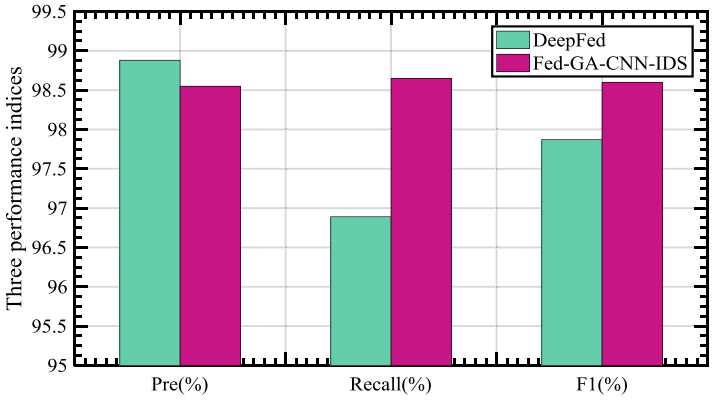

Fig. 12. The comparison of Pre, Recall, and F1 on WADI dataset.

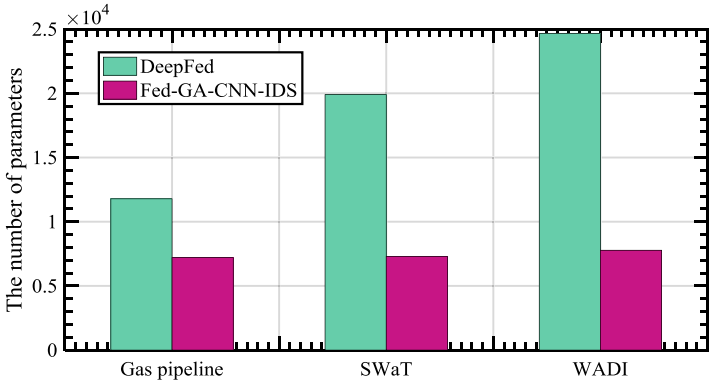

Fig. 13. The Params comparison of DeepFed and Fed-GA-CNN-IDS of Params on three datasets.

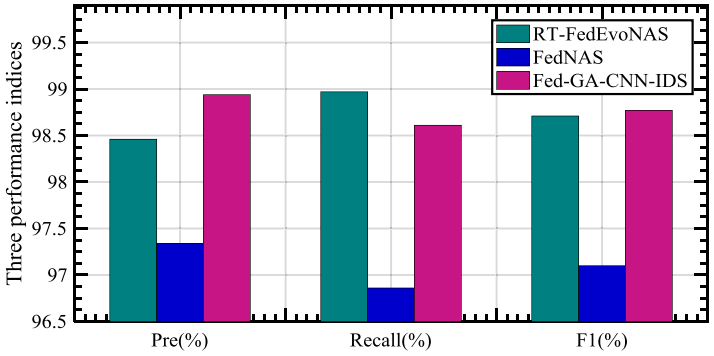

Fig. 14. Comparison of three different methods on Gas Pipeline Dataset.

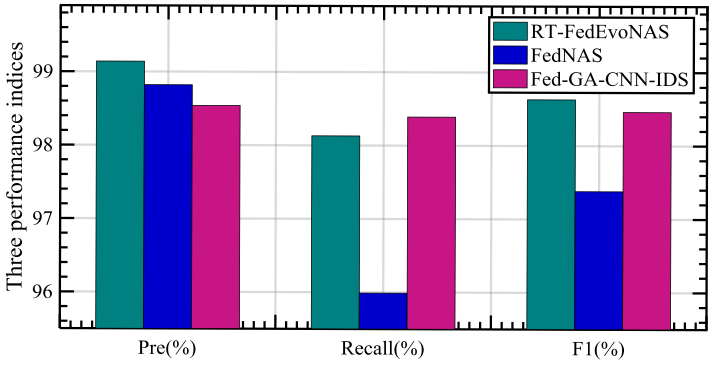

Fig. 15. Comparison of three different methods on SWaT Dataset.

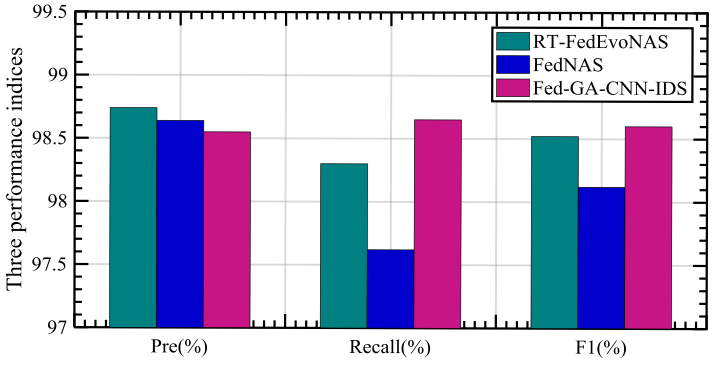

Fig. 16. Comparison of three different methods on WADI Dataset.

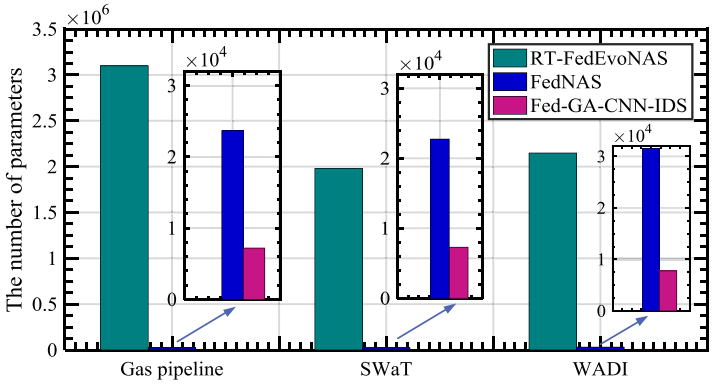

Fig. 17. The Params comparison of three different methods on three datasets.

Table 11
Comparison of time and memory on Gas Pipeline Dataset.

Method	Devcie	Max-Memory (MB)	Time (days)
RT-FedEvoNAS	GPU	3572	0.21
	CPU	498	2.96
FedNAS	GPU	3657	0.06
	CPU	516	0.18
Fed-GA-CNN-IDS	GPU	3534	0.1
	CPU	362	0.27

Table 12
Comparison of time and memory on SWaT dataset.

Method	Devcie	Max-Memory (MB)	Time (days)
RT-FedEvoNAS	GPU	3738	0.82
	CPU	659	8.16
FedNAS	GPU	3822	0.34
	CPU	715	1.11
Fed-GA-CNN-IDS	GPU	3694	0.48
	CPU	440	1.84

Table 13
Comparison of time and memory on WADI Dataset.

Method	Devcie	Max-Memory (MB)	Time (days)
RT-FedEvoNAS	GPU	3677	0.41
	CPU	695	6.13
FedNAS	GPU	3788	0.17
	CPU	802	1.61
Fed-GA-CNN-IDS	GPU	3646	0.19
	CPU	568	2.55

search. The overall time consumption and memory requirements of federated architecture search are mainly related to factors such as the size of local model, the computational complexity of the local model, the amount of local data, and the hardware resources of the client such as GPU and CPU. Because the server does not perform training, and the computing performance of the server is usually sufficient to meet the needs. Therefore, we calculate the time consumption of federated architecture search and the maximum memory requirement of the client under different hardware configurations. Tables 11–13 show the comparison of time consumption and memory requirements of these three methods on the Gas Pipeline, SWaT and WADI datasets, respectively. Here, GPU means that each client is configured with one CPU and one GPU during the federated architecture search process, and CPU means that the client is only configured with one CPU.

From Tables 11–13, we can see that:

(1) Regarding memory requirements, the Max-Memory (MB) for the Fed-GA-CNN-IDS method is the smallest, regardless of whether the client is equipped with a GPU or not. The reason is that the training model of the client is always lightweight in the Fed-GA-CNN-IDS method. On the contrary, FedNAS requires the most memory because FedNAS is based on mixed operations, which makes the model computationally complex and requires more intermediate calculation results to be stored during forward and backward propagation. Therefore, Fed-GA-CNN-IDS has an advantage in terms of memory requirements, especially for resource-constrained clients.

(2) Regarding time consumption, as seen from the Time (days), it is evident that equipping clients with GPU significantly reduces search time. Fed-GA-CNN-IDS takes more time than FedNAS but much less than RT-FedEvoNAS. This is because FedNAS is based on gradient descent. However, compared to Fed-GA-CN-IDS and RT-FedEvoNAS, FedNAS requires multiple runs to achieve satisfactory results. Therefore, for the pursuit of better performance, it is acceptable that Fed-GA-CNN-IDS requires slightly more time.

4.4. Experiment III: Comparison with basic blocks in fed-GA-CNN-IDS

In order to illustrate the performance of combination and optimization of different basic lightweight blocks in Fed-GA-CNN-IDS,

Table 14
Comparison of single lightweight block on gas pipeline dataset.

Model	Pre	Recall	F1	Params
MobileNet×10	**99.06**	98.03	98.54	9986
MobileNetV2 × 10	98.51	**98.86**	98.69	10 226
Xception×10	98.61	**98.86**	98.73	32 626
ShuffleNetV2 × 10	98.51	98.84	98.68	20 482
Fed-GA-CNN-IDS	98.94	98.61	**98.77**	**7202**

Table 15
Comparison of single lightweight block on SWaT pipeline dataset.

Model	Pre	Recall	F1	Params
MobileNet×10	97.25	95.42	96.33	10 274
MobileNetV2 × 10	97.78	96.7	97.24	32 914
Xception×10	**98.86**	94.97	96.87	10 514
ShuffleNetV2 × 10	97.94	95.84	96.88	20 770
Fed-GA-CNN-IDS	98.54	**98.39**	**98.46**	**7298**

Table 16
Comparison of single lightweight block on WADI dataset.

Model	Pre	Recall	F1	Params
MobileNet×10	98.6	98.25	98.42	11 522
MobileNetV2 × 10	98.26	98.95	**98.6**	34 162
Xception×10	97.25	**99.15**	98.19	11 762
ShuffleNetV2 × 10	96.43	98.7	97.55	22 018
Fed-GA-CNN-IDS	98.55	98.65	**98.6**	**7778**

four lightweight blocks, i.e., MobileNet (Howard et al., 2017), MobileNetV2 (Sandler et al., 2018), Xception (Chollet, 2017), and ShuffleNetV2 (Ma et al., 2018) are considered as the competitors. The optimized deep learning models obtained by Fed-GA-CNN-IDS are [4, 4, 4, 1, 4, 1, 1, 4, 4, 1], [2, 2, 1, 4, 1, 1, 5, 5, 1, 1], and [2, 4, 4, 4, 2, 4, 1, 1, 1, 1] for Gas Pipeline dataset, SWaT dataset, and WADI dataset, respectively. Thus, we consider 10 blocks to extend the MobileNet, MobileNetV2, Xception, and ShuffleNetV2 as the competitors. In addition, the normal block of Block1 has no parameters, that is, the actual number of layers of the model is only two. Thus, Block1 is not considered as the competitor.

Tables 14–16 show the comparison of single lightweight block for Gas Pipeline, SWaT, and WADI, respectively. From Tables 14–16, it can be seen that for three different datasets, the Params of Fed-GA-CNN-IDS is the best among the five lightweight methods, indicating that by using the proposed automated method to combine and optimize these blocks, we can achieve more lightweight performance. Furthermore, when comparing the F1, which is a measure of overall performance, Fed-GA-CNN-IDS also ranks as the best. Regarding the Pre, Fed-GA-CNN-IDS consistently ranks second. As for the Recall, Fed-GA-CNN-IDS ranks fourth, first, and fourth, respectively, for the three different datasets. In summary, Fed-GA-CNN-IDS achieves better overall performance through combination and optimization while ensuring the advantage in lightweight, compared with the other four single lightweight blocks.

5. Conclusion

In this paper, we proposed Fed-GA-CNN-IDS, an automated federated learning method for the intrusion detection of ICS based on evolutionary NAS. To the best knowledge of the authors, it is the first attempt to automatically develop a lightweight federated learning model for the intrusion detection issue in ICSs mitigate the challenges including constrained computational resources, fast response time, and stringent privacy/security requirements. Additionally, in Fed-GA-CNN-IDS, five basic lightweight neural architectures of CNN are combined and optimized by GA with an efficient discrete encoding strategy, crossover operation and mutation operation. We have compared Fed-GA-CNN-IDS with three state-of-the-art manually-designed federated

learning-based IDS methods, such as FedVAE-SVDD (Huong et al., 2022), FATRF (Truong et al., 2022), and DeepFed (Li et al., 2020), two federated NAS method such as RT-FedEvoNAS (Zhu and Jin, 2022) and FedNAS (He et al., 2020), and four lightweight blocks, such as MobileNet (Howard et al., 2017), MobileNetV2 (Sandler et al., 2018), Xception (Chollet, 2017), ShuffleNetV2 (Ma et al., 2018), on three widely-used ICS intrusion detection datasets including Gas Pipeline, SWaT, and WADI. The experimental results have shown that the proposed Fed-GA-CNN-IDS can obtain more lightweight models and better or at least competitive detection performance. The basic idea behind the proposed Fed-GA-CNN-IDS will be extended to the intrusion detection issue of the other IIoT. In future, other effective evolutionary algorithms especially multi-objective optimization algorithms-based NAS with other basic lightweight types of deep neural networks will be used to automated federated intrusion detection methods for smart grids by considering the detection performance and communication resource.

CRediT authorship contribution statement

Jun-Min Shao: Data curation, Methodology, Software, Validation, Writing – original draft, Visualization. **Guo-Qiang Zeng:** Conceptualization, Formal analysis, Methodology, Supervision, Visualization, Writing – review & editing, Software, Validation. **Kang-Di Lu:** Methodology, Software, Validation, Visualization, Writing – review & editing. **Guang-Gang Geng:** Resources, Software, Supervision, Writing – review & editing, Conceptualization. **Jian Weng:** Conceptualization, Resources, Supervision, Visualization, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (Grant Nos. 61972288 and 92067108), Natural Science Foundation of Guangdong Province (Grant No. 2021A151501131), in part by the MIIT Project Industrial Internet identification resolution system security monitoring and protection (Grant No. TC220H078), in part by the Guangdong Key Laboratory of Data Security and Privacy Preserving, National Joint Engineering Research Center of Network Security Detection and Protection Technology. Guang-Gang Geng is supported by Pearl River Talents Plan.

References

Abdel-Basset, M., Hawash, H., Sallam, K., 2021. Federated threat-hunting approach for microservice-based industrial cyber–physical system. IEEE Trans. Ind. Inform. 18 (3), 1905–1917.

Ahmed, C.M., Palleti, V.R., Mathur, A.P., 2017. Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In: Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks. pp. 25–28.

Aouedi, O., Piamrat, K., Muller, G., Singh, K., 2022. Federated semisupervised learning for attack detection in industrial Internet of Things. IEEE Trans. Ind. Inform. 19 (1), 286–295.

Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1251–1258.

Cui, L., Qu, Y., Xie, G., Zeng, D., Li, R., Shen, S., Yu, S., 2021. Security and privacy-enhanced federated learning for anomaly detection in IoT infrastructures. IEEE Trans. Ind. Inform. 18 (5), 3492–3500.

Friha, O., Ferrag, M.A., Benbouzid, M., Berghout, T., Kantarci, B., Choo, K.K.R., 2023. 2DF-IDS: Decentralized and differentially private federated learning-based intrusion detection system for Industrial IoT. Comput. Secur. 103097.

Ghimire, B., Rawat, D.B., 2022. Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for Internet of Things. IEEE Internet Things J. 9 (11), 8229–8249.

He, C., Annavaram, M., Avestimehr, S., 2020. Towards non-iid and invisible data with FedNAS: Federated deep learning via neural architecture search. arXiv preprint arXiv:2004.08546.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Huang, X., Liu, J., Lai, Y., Mao, B., Lyu, H., 2022. EEFED: Personalized federated learning of execution & evaluation dual network for CPS intrusion detection. IEEE Trans. Inf. Forensics Secur. 18, 41–56.

Huang, J.C., Zeng, G.Q., Geng, G.G., Weng, J., Lu, K.D., Zhang, Y., 2023. Differential evolution-based convolutional neural networks: An automatic architecture design method for intrusion detection in industrial control systems. Comput. Secur. 132, 103310.

Huong, T.T., Bac, T.P., Ha, K.N., Hoang, N.V., Hoang, N.X., Hung, N.T., Tran, K.P., 2022. Federated learning-based explainable anomaly detection for industrial control systems. IEEE Access 10, 53854–53872.

Huong, T.T., Bac, T.P., Long, D.M., Luong, T.D., Dan, N.M., Thang, B.D., Tran, K.P., 2021. Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach. Comput. Ind. 132, 103509.

Kravchik, M., Shabtai, A., 2018. Detecting cyber attacks in industrial control systems using convolutional neural networks. In: Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and Privacy. pp. 72–83.

Kravchik, M., Shabtai, A., 2021. Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca. IEEE Trans. Dependable Secure Comput. 19 (4), 2179–2197.

Kriman, S., Beliaev, S., Ginsburg, B., Huang, J., Kuchaiev, O., Lavrukhin, V., Leary, R., Li, J., Zhang, Y., 2020. Quartznet: Deep automatic speech recognition with 1D time-channel separable convolutions. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 6124–6128.

Langner, R., 2011. Stuxnet: Dissecting a cyberwarfare weapon. IEEE Secur. Priv. 9 (3), 49–51.

Li, B., Ma, S., Deng, R., Choo, K.K.R., Yang, J., 2022a. Federated anomaly detection on system logs for the Internet of Things: A customizable and communication-efficient approach. IEEE Trans. Netw. Serv. Manag. 19 (2), 1705–1716.

Li, S., Ngai, E., Voigt, T., 2021. Byzantine-robust aggregation in federated learning empowered industrial IoT. IEEE Trans. Ind. Inform. 19 (2), 1165–1175.

Li, J., Tong, X., Liu, J., Cheng, L., 2023. An efficient federated learning system for network intrusion detection. IEEE Syst. J. 17 (2), 2455–2464.

Li, Y., Wei, X., Li, Y., Dong, Z., Shahidehpour, M., 2022b. Detection of false data injection attacks in smart grid: A secure federated deep learning approach. IEEE Trans. Smart Grid 13 (6), 4862–4872.

Li, B., Wu, Y., Song, J., Lu, R., Li, T., Zhao, L., 2020. DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems. IEEE Trans. Ind. Inform. 17 (8), 5615–5624.

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 Things J. 8 (8), 6348–6358.

Liu, H., Simonyan, K., Yang, Y., 2018. Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055.

Liu, X., Zhao, J., Li, J., Cao, B., Lv, Z., 2022. Federated neural architecture search for medical data security. IEEE Trans. Ind. Inform. 18 (8), 5628–5636.

Ma, N., Zhang, X., Zheng, H.T., Sun, J., 2018. Shufflenet v2: Practical guidelines for efficient CNN architecture design. In: Proceedings of the European Conference on Computer Vision. ECCV, pp. 116–131.

Mathur, A.P., Tippenhauer, N.O., 2016. SWaT: A water treatment testbed for research and training on ICS security. In: 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks. CySWater, IEEE, pp. 31–36.

McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A., 2017. Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics. PMLR, pp. 1273–1282.

Morris, T., Gao, W., 2014. Industrial control system traffic data sets for intrusion detection research. In: Critical Infrastructure Protection VIII: 8th IFIP WG 11.10 International Conference. ICCIP 2014, Arlington, VA, USA, March 17-19, 2014, Springer, pp. 65–78, Revised Selected Papers 8.

Mothukuri, V., Khare, P., Parizi, R.M., Pouriyeh, S., Dehghantanha, A., Srivastava, G., 2021. Federated-learning-based anomaly detection for IoT security attacks. IEEE Internet Things J. 9 (4), 2545–2554.

Mushtaq, E., He, C., Ding, J., Avestimehr, S., 2021. SPIDER: Searching personalized neural architecture for federated learning. arXiv preprint arXiv:2112.13939.

Naeem, F., Ali, M., Kaddoum, G., 2023. Federated-learning-empowered semi-supervised active learning framework for intrusion detection in ZSM. IEEE Commun. Mag. 61 (2), 88–94.

Popoola, S.I., Ande, R., Adebisi, B., Gui, G., Hammoudeh, M., Jogunola, O., 2021. Federated deep learning for zero-day botnet attack detection in IoT-edge devices. IEEE Internet Things J. 9 (5), 3930–3944.

Ruzafa-Alcázar, P., Fernández-Saura, P., Mármol-Campos, E., González-Vidal, A., Hernández-Ramos, J.L., Bernal-Bernabe, J., Skarmeta, A.F., 2023. Intrusion detection based on privacy-preserving federated learning for the industrial IoT. IEEE Trans. Ind. Inform. 19 (2), 1145–1154.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4510–4520.

Singh, I., Zhou, H., Yang, K., Ding, M., Lin, B., Xie, P., 2020. Differentially-private federated neural architecture search. arXiv preprint arXiv:2006.10559.

Slay, J., Miller, M., 2007. Lessons learned from the maroochy water breach. In: International Conference on Critical Infrastructure Protection. Springer, pp. 73–82.

Taheri, R., Shojafar, M., Alazab, M., Tafazolli, R., 2020. Fed-IIoT: A robust federated malware detection architecture in industrial IoT. IEEE Trans. Ind. Inform. 17 (12), 8442–8452.

Telikani, A., Shen, J., Yang, J., Wang, P., 2022. Industrial IoT intrusion detection via evolutionary cost-sensitive learning and fog computing. IEEE Internet Things J. 9 (22), 23260–23271.

Tran, H.Y., Hu, J., Yin, X., Pota, H.R., 2023. An efficient privacy-enhancing cross-silo federated learning and applications for false data injection attack detection in smart grids. IEEE Trans. Inf. Forensics Secur. 18, 2538–2552.

Truong, H.T., Ta, B.P., Le, Q.A., Nguyen, D.M., Le, C.T., Nguyen, H.X., Do, H.T., Nguyen, H.T., Tran, K.P., 2022. Light-weight federated learning-based anomaly detection for time-series data in industrial control systems. Comput. Ind. 140, 103692.

Yu, S., Nguyen, P., Abebe, W., Stanley, J., Munoz, P., Jannesari, A., 2019. Neural architecture search: A survey. J. Mach. Learn. Res. 20 (1), 1997–2017.

Yu, S., Nguyen, P., Abebe, W., Stanley, J., Munoz, P., Jannesari, A., 2022. Resource-aware heterogeneous federated learning using neural architecture search. arXiv preprint arXiv:2211.05716.

Zhang, K., Jiang, Y., Seversky, L., Xu, C., Liu, D., Song, H., 2021. Federated variational learning for anomaly detection in multivariate time series. In: 2021 IEEE International Performance, Computing, and Communications Conference. IPCCC, IEEE, pp. 1–9.

Zhang, C., Yuan, X., Zhang, Q., Zhu, G., Cheng, L., Zhang, N., 2022. Toward tailored models on private AIoT devices: Federated direct neural architecture search. IEEE Internet Things J. 9 (18), 17309–17322.

Zhu, H., Jin, Y., 2019. Multi-objective evolutionary federated learning. IEEE Trans. Neural Netw. Learn. Syst. 41 (4), 1310–1322.

Zhu, H., Jin, Y., 2022. Real-time federated evolutionary neural architecture search. IEEE Trans. Evol. Comput. 26 (2), 364–378.

Zhu, H., Zhang, H., Jin, Y., 2021. Neural architecture search: A survey. Complex Intel. 7 (2), 639–657.

Zoph, B., Le, Q.V., 2016. Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.

Jun-Min Shao received the B.E. degree from University of South China, in 2020. He is currently pursuing the M.S. degree with the College of Cyber Security of Jinan University, Guangzhou, China. His research interests include cyber security in industrial Internet of Things, federated learning and its security, and evolutionary neural architecture search.

Guo-Qiang Zeng received the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2011. He is currently a Professor with the College of Cyber Security, Jinan University, Guangzhou, China, and also the vice-director with National-Local Joint Engineering Laboratory of Digitalize Electrical Design Technology, Wenzhou University, Wenzhou, China. His research interests include cyber security, machine learning security, computational intelligence, smart grids, and industrial Internet of Things. He has authored or coauthored the book *Extremal Optimization: Fundamentals, Algorithms, and Applications* (CRC Press) and over 80 papers in international conferences and journals, such as IEEE TII, TDSC, TVT, TPEL, and IoTJ. He holds more than 30 patents. He was the recipient of eight ministerial and provincial science and technology progress awards.

Kang-Di Lu received the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2023. He is currently a Lecture with the College of Information Science and Technology, Donghua University, Shanghai, China. He has authored or coauthored more than 20 papers in international conferences and journals, such as IEEE TII, TVT, TIM, and IoTJ. His research interests include cyber security in industrial Internet of Things and smart grids, evolutionary computation, and intelligent control.

Guang-Gang Geng received the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China. He was with the China Internet Network Information Center, Beijing, from 2008 to 2020. He is currently a Professor with the College of Cyber Security, Jinan University, Guangzhou. He has published over 60 papers in international conferences and journals, such as ACM SIGIR, IEEE TIFS, TSC, and TNNLS. His current research interests include cyber security, machine learning, computer networking, anti-fraud, and web search.

Jian Weng received the B.S. and M.S. degrees in computer science and engineering from South China University of Technology, Guangzhou, China, in 2000 and 2004, respectively, and the Ph.D. degree in computer science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2008. From 2008 to 2010, he held a Postdoctoral position with the School of Information Systems, Singapore Management University. He is currently a Professor with the College of Cyber Security and the vice-president of Jinan University, Guangzhou, China. He has authored or coauthored more than 100 papers in cryptography and security conferences and journals, such as CRYPTO, EUROCRYPT, ASIACRYPT, TCC, PKC, IEEE TPAMI, TIFS, and TDSC. His research interests include public key cryptography, cloud security, and blockchain. He was the PC Co-Chairs or PC Member for more than 30 international conferences. He also serves as an Associate Editor for the IEEE Transactions on Vehicular Technology.