Attacking Modulation Recognition with Adversarial Federated Learning in Cognitive Radio-Enabled IoT

Hongyi Zhang, Mingqian Liu, *Member, IEEE,* Yunfei Chen, *Senior Member, IEEE,* and Nan Zhao, *Senior Member, IEEE*

***Abstract*—Internet of Things (IoT) based on cognitive radio (CR) exhibits strong dynamic sensing and intelligent decision-making capabilities by effectively utilizing spectrum resources. The federal learning (FL) framework based modulation recognition (MR) is an essential component, but its use of uninterpretable deep learning (DL) introduces security risks. This paper combines traditional signal interference methods and data poisoning in FL to propose a new adversarial attack approach. The poisoning attack in distributed frameworks manipulates the global model by controlling malicious users, which is not only covert but also highly impactful. The carefully designed pseudo-noise in MR is also extremely difficult to detect. The combination of these two techniques can generate a greater security threat. We have further advanced our proposal with the introduction of the new adversarial attack method called "Chaotic Poisoning Attack" to reduce the recognition accuracy of the FL-based MR system. We establish effective attack conditions, and simulation results demonstrate that our method can cause a decrease of approximately 80% in the accuracy of the local model under weak perturbations and a decrease of around 20% in the accuracy of the global model. Compared to white-box attack methods, our method exhibits superior performance and transferability.**

***Index Terms*—Adversarial attack, federated learning, modulation recognition, cognitive radio.**

I. Introduction

COGNITIVE radio (CR) enabled Internet of Things (IoT) possesses the capability to intelligently perceive the spectrum environment and efficiently utilize the spectrum [1], thus mitigating the issue of spectrum scarcity. Modulation recognition (MR) is a crucial technique in CR, and when using deep learning (DL), it offers significant advantages. Different modulation methods are utilized by various users based on service requirements and MR can identify the modulation schemes in user signals by analyzing characteristics such as amplitude, phase, and waveform of the received signal [2]–[4]. This information enables CR to dynamically adjust the modulation scheme and coding rate to adapt to diverse channel conditions. Additionally, DL-based MR plays a crucial role in signal detection, spectrum management, and signal control. For example, Zhang et al. [5] proposed a DL-based method for MR, using a neural network with an improved generalized end-to-end loss to enhance similarity among feature vectors with the same modulation type and reduce similarity among those with different types. Wang et al. [6] developed a Multi-Cue Fusion network for automatic modulation recognition, while Njoku et al. [7] introduced an economically efficient hybrid neural network consisting of a shallow convolutional network, gated recurrent units, and deep neural network (DNN), for automatic modulation recognition in cognitive radios.

Deep learning (DL) brings enormous advantages to the use of MR. However, due to its non-interpretability, there are significant security risks that must be addressed. Reference [8] as the first to reveal that adding small, imperceptible distortions to an image can lead to errors in DL classification. Since then, numerous studies have emerged investigating DL security issues. In the context of Internet of Vehicles, Qiu et al. [9] generated adversarial examples through GPS data and found that small perturbations can deceive deep neural networks. Similarly, when DL is applied in medical IoT, researchers have identified several security vulnerabilities. Rahman et al. [10] found that diagnostic methods relying on DL algorithms are vulnerable to adversarial attacks. In [11], an adversarial attack technology based on adversarial networks and multi-task loss was proposed to reduce the accuracy of MR.

However, the adversarial attacks on centralized DL are significantly weakened when federated learning (FL) is used. In the IoT scenario, users and devices are distributed, and it is challenging to form a unified data source for DL training [12]–[16]. Therefore, FL is widely adopted in MR in IoT. As a common means of protecting data privacy in IoT scenarios, recent studies have shown that FL is also susceptible to attacks. Liu et al. [17] discussed data poisoning attacks in crowdsourcing and the conditions of effective attack strategies. Shi et al. [18] proposed a novel poisoning attack algorithm, Fed-MIFGSM, targeting robust FL frameworks to investigate the impact of adversarial examples on FL. Lim et al. [19] questioned the reliability of DL models, particularly in FL, by using existing privacy-breaking algorithms to invert the model's gradients and reconstruct input data with incontrovertible evidence. Chen et al. [20] proposed a novel adversarial attack method specifically designed for graph neural network FL frameworks, this method leverages privacy leakage and gradients of paired nodes to generate adversarial perturbations based on perturbing

This work was supported by the National Natural Science Foundation of China under Grant 62071364 and 62231027, in part by the Key Research and Development Program of Shaanxi Province under Grant 2023-YBGY-249, in part by the Guangxi Key Research and Development Program under Grant 2022AB46002, in part by the Fundamental Research Funds for the Central Universities under Grant KYFZ23001. *(Corresponding author: Mingqian Liu.)*

H. Zhang and M. Liu are with the State Key Laboratory of Integrated Service Networks, Xidian University, Shanxi, Xi'an 710071, China (e-mail: hyzhang1@stu.xidian.edu.cn; mqliu@mail.xidian.edu.cn).

Y. Chen is with the School of Engineering, University of Warwick, Coventry, West Midlands United Kingdom of Great Britain and Northern Ireland CV4 7AL (e-mail: Yunfei.Chen@warwick.ac.uk).

N. Zhao is with the School of Information and Communication Engineering, Dalian University of Technology, Dalian 116024, P. R. China. (e-mail: zhaonan@dlut.edu.cn).

the global node embeddings with added noise. Hossain et al. [21] analyzed the inadequacy of adversarial attacks in FL settings to remain concealed and persistent, and demonstrated the use of differential noise for poisoning attacks. Most of these studies focus on image processing, but there is a lack of efficient and covert attacks on signals in FL-based MR.

This paper proposes a new type of adversarial attack, using pseudo-noise, designed to decrease the recognition accuracy of FL-based MR systems. This attack combines interference methods from the signal domain with poisoning attacks from FL. The main contributions of this paper are as follows:

- We propose a novel adversarial attack method that utilizes pseudo-noise to decrease the recognition accuracy of FL-based MR.
- We combine interference methods in the signal domain with poisoning attacks in FL to enhance the effectiveness of the attack. This fills a gap in the field of FL-based MR by expanding the adversarial attack methods into the signal domain.
- We derive the conditions for the feasibility of the attack and demonstrate the effectiveness of the proposed attack method through experiments.

The remainder of this paper is described as follows. Section II presents system model. Section III introduces the proposed chaotic poisoning attack. Section IV analyzes the effective attack conditions. Section V presents simulation and comparison of various attack methods. Finally, Section VI provides a conclusion for the entire paper.

II. System Model

The proposed FL-based MR in CR-enabled IoT is shown in Fig. 1. This system connects devices from different areas, such as wireless sensor network devices, ground radar networks, and remote monitoring devices that require the use of MR. These devices collect signals and transmit them to the edge computing system, which compiles these signals into a training set and trains a local DL network model. After training, the local DL network model parameters are uploaded to the central computing system. The central computing system generates a global model and distributes it to each local device, which can then use the global model for signal recognition.

The MR-based federated learning framework is a distributed deep learning method. Its core idea is to perform distributed training among multiple data sources with local data. Without exchanging local data, a global model is built based on virtual fused signal data only by exchanging parameters. This framework achieves the balance between data privacy and data sharing, and is a new paradigm for the application of MR systems. The global state of FL can be expressed as

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w), \tag{1}$$

where K represents the number of local servers in this learning round, $F_k(w)$ represents the objective function of the k-th local server, n represents the total number of samples in the FL framework, n_k represents the number of samples at the k-th local server. The objective function $F_k(w)$ can be further expressed as

$$F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w), \tag{2}$$

where P_k denotes the partition of the dataset assigned to the k-th local server, and $f_i(w)$ denotes the loss function for the i-th data point.

There are different types of DNNs. Here, convolutional neural networks (CNN), residual neural networks (ResNet), and visual geometry group networks (VGG) are considered in FL.

1) CNN: CNN is a type of deep learning model commonly used in image recognition, computer vision, and natural language processing. CNN consists of convolutional layers, pooling layers, and fully connected layers, which extract features and classify the input images.

The core of CNN is the convolutional layer. The convolutional layer consists of multiple convolutional kernels, each of which extracts a specific feature from the input data. At each position of the input data, the kernel performs a convolution operation and obtains an output value. The kernel slides over the input data until the entire input data is traversed, and a feature map is obtained. Multiple kernels can extract different features, resulting in multiple feature maps.

CNN also includes pooling layers, which reduce the size of feature maps, decrease computation, and enhance feature robustness. The pooling layer usually adopts the maximum pooling or average pooling, which selects the maximum value or average value of each subregion of the feature map as the output value.

Finally, CNN flattens the feature maps into a one-dimensional vector and performs classification or regression tasks through fully connected layers.

Assuming the input image is X, the convolutional kernel is W, the bias is b, and the convolution operation is $*$, then the output Y of the convolutional layer can be expressed as

$$Y = f(\sum_{i=1}^{n} W_i * X + b), \tag{3}$$

where n represents the number of convolutional kernels, f represents the activation function, commonly used ones include ReLU, sigmoid, and tanh. Assuming the pooling operation is P, then the output Y of the pooling layer can be expressed as

$$Y = P(X). \tag{4}$$

Common pooling operations include maximum pooling (P_{max}) and average pooling (P_{avg}), which can be expressed as

$$P_{max}(X)i, j = \max m, nX_{i+m,j+n}, \tag{5}$$

$$P_{avg}(X)i, j = \frac{1}{k^2} \sum m, nX_{i+m,j+n}, \tag{6}$$

where k represents the size of the pooling region. The output of the final CNN can be expressed as

$$F_{\text{CNN}} = g(W^T R + b), \tag{7}$$

$$f_{\text{CNN}}(w) = Loss(F_{\text{CNN}}), \tag{8}$$

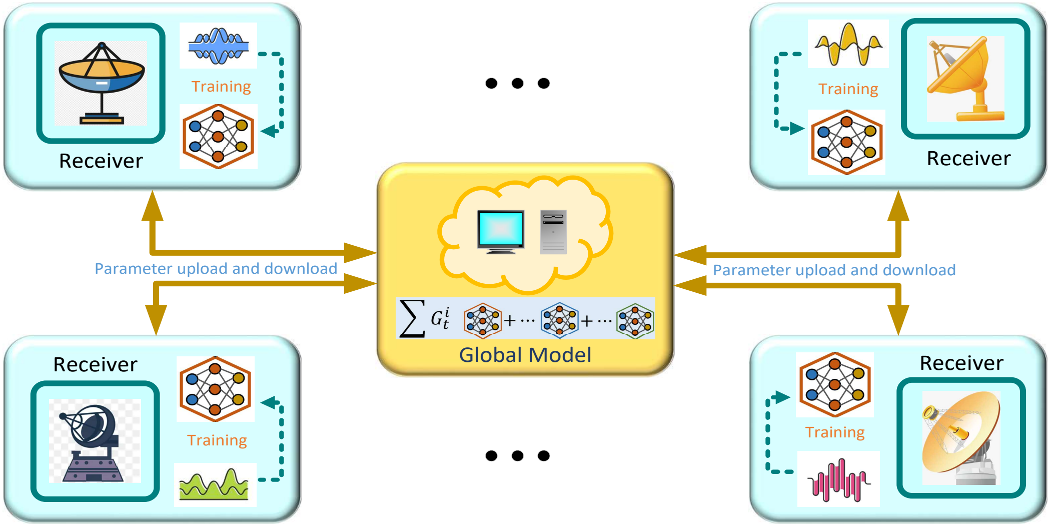


Fig. 1. Federated learning based modulation recognition in cognitive radio enabled IoT.

where $f_{\text{CNN}}(w)$ represents the loss function, F_{CNN} represents the output, g represents the activation function, W represents the network weight, R represents the network input, and b represents the offset. Such a CNN for MR will serve as a node in the entire FL framework, and its input data comes from the baseband signal $r(t)$ provided by the sampling module.

2) ResNet: ResNet is a deep convolutional neural network model proposed in 2015. It introduced the concept of residual learning, which enables training of very deep neural networks by adding shortcut connections that skip one or more layers.

The basic building block of ResNet is the residual block, which consists of two convolutional layers and a shortcut connection that bypasses the convolutional layers. The shortcut connection adds the input directly to the output of the second convolutional layer, allowing the network to learn the residual between the input and output.

The ResNet architecture is organized into a series of stages, and each stage consists of multiple residual blocks with the same number of filters. The number of filters is doubled at the transition between stages, and the spatial size of feature maps is reduced by a factor of 2 using a convolutional layer with a stride of 2.

The mathematical expression of a residual block can be written as

$$Y = F(X, W) + X, \tag{9}$$

where X is the input feature map, W represents the weights of the convolutional layers in the residual block, F is a residual function that learns the difference between the input and output, and Y is the output feature map.

The residual function F can be expressed as

$$F(X, W) = \sigma(W_2\sigma(W_1X + b_1) + b_2) + X, \tag{10}$$

where W_1, W_2 are the weights of the two convolutional layers, b_1, b_2 are the biases, σ represents the activation function (usually ReLU), and the shortcut connection X is added to the output of F to form the final output Y.

The mathematical expression of an average pooling layer is

$$Y_{i,j,k} = \frac{1}{k^2}\sum_{a=1}^{k}\sum_{b=1}^{k} X_{2i+a-1,2j+b-1,k}, \tag{11}$$

where X is the input featuremap, i, j, k represent the position and channel of the output feature map, and the average pooling operation is applied to a $k \times k$ region with a stride of 2.

We denote the final output of the ResNet network as

$$F_{\text{Res}} = g(W^TR + b), \tag{12}$$

$$f_{\text{Res}}(w) = Loss(F_{\text{Res}}), \tag{13}$$

where $f_{\text{Res}}(w)$ represents the loss function, F_{Res} represents the output, g represents the activation function, W represents the network weight, R represents the network input, and b represents the offset. It is also one of the nodes of the FL framework, and the input comes from $r(t)$ of the sampling module.

3) VGG: VGG is a deep convolutional neural network model proposed in 2014. It achieved excellent performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) and has been widely used in various computer vision tasks.

VGG consists of several convolutional layers followed by max pooling layers, and ends with several fully connected layers. The convolutional layers use small 3x3 filters with a stride of 1 and padding of 1, and the max pooling layers use 2x2 filters with a stride of 2.

The mathematical expression of a convolutional layer can be written as

$$Y_{i,j,k} = \sigma\left(\sum_{a,b,c} W_{a,b,c,k}X_{i+a-1,j+b-1,c} + b_k\right), \tag{14}$$

where X is the input feature map, W is the convolutional kernel, b is the bias, σ is the activation function (usually ReLU), and i, j, k represent the position and channel of the output feature map.

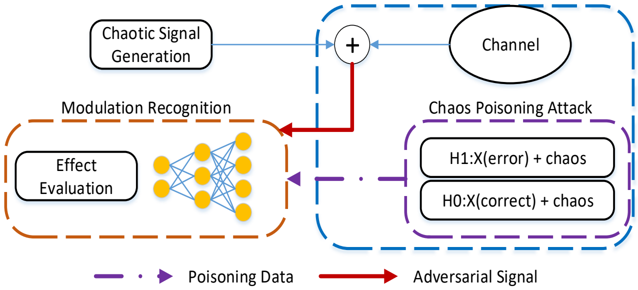


Fig. 2. Chaos poisoning attack flow diagram.

The mathematical expression of a max pooling layer is

$$Y_{i,j,k} = \max_{a,b}(X_{2i+a-1,2j+b-1,k}), \tag{15}$$

where X is the input feature map, i, j, k represent the position and channel of the output feature map, and the max pooling operation is applied to a 2x2 region with a stride of 2.

The mathematical expression of a fully connected layer is

$$Y = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{16}$$

where x is the input vector, w is the weight vector, b is the bias, and σ is the activation function (usually ReLU or softmax).

In VGG, the last three fully connected layers have 4096 neurons each, and the last fully connected layer has 1000 neurons corresponding to the 1000 classes in the ImageNet dataset.

We denote the final output of VGG as

$$F_{\text{VGG}} = g(W^T R + b), \tag{17}$$

$$f_{\text{VGG}}(w) = Loss(F_{\text{VGG}}), \tag{18}$$

where $f_{\text{VGG}}(w)$ represents the loss function, F_{VGG} represents the output, g represents the activation function, W represents the network weight, R represents the network input, and b represents the offset. the input comes from the $r(t)$ of the sampling module.

There are more than three types of DNNs in the final FL framework, and the number will also increase. FL-based MR offers privacy-preserving machine learning, improved accuracy, distributed training, and collaborative learning. FL enables MR models to be trained on diverse datasets from geographically distributed devices without the need for data to be transferred to a central location, making MR more efficient and scalable. FL is a promising approach for MR in scenarios where data is sensitive or confidential, and where devices are geographically distributed.

III. Chaos Poisoning Attack on Federated Learning

Chaotic signals are signals that lie between true random noise and deterministic signals, and have a high degree of concealment. The poisoning attack is a powerful attack method for deep learning. Thus, they can be combined to produce a more subtle attack. The overall attack flow chart is shown in Fig. 2. First, the parameters of the other party's deep learning network are modified in the way the attacker wants through the poisoning attack (that is, when the other party receives the signal interfered by the chaotic signal, the modulated signal is incorrectly identified.) and then the corresponding chaotic signal is transmitted in the channel to reduce the recognition accuracy of the modulation recognition system, thereby destroying MR. Poisoning attack is one of the most powerful and hidden attacks on the FL framework [22].

A. Adversarial Attack

Adversarial attack refers to the deliberate manipulation of input data to a machine learning model in order to force its errors. The goal of such attacks is to exploit vulnerabilities in the model and undermine its performance.

One common form of adversarial attack is the targeted attack, in which the attacker aims to cause the model to output a specific incorrect prediction. This can be achieved by adding a small perturbation to the input data, which can be formulated as

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y_{target})), \tag{19}$$

where x is the original input data, x_{adv} is the perturbed input data, ϵ is a small scalar value that controls the magnitude of the perturbation, J is the loss function used to evaluate the performance of the model, θ are the model parameters, and y_{target} is the target output that the attacker wants the model to produce.

In the simulation, several common white-box algorithms are compared with the proposed chaos poisoning method (CPM), including the fast gradient sign method (FGSM) [23], the basic iterative method (BIM) [24], the projected gradient descent (PGD) [25] and the momentum iterative method (MIM) [26].

FGSM generates adversarial samples through the sign of gradient, and the core equation is

$$x^{adv} = x + \varepsilon \cdot \text{sign}\left(\nabla_x (J(x, y))\right), \tag{20}$$

where x and y are clean samples and corresponding labels respectively. The label refers to one hot vector. $J(\cdot,\cdot)$ is the cross-entropy function, and x^{adv} is the corresponding adversarial samples. The $\text{sign}(\cdot)$ is the sign function, a positive number returns 1, a negative number returns -1, and 0 returns 0.

The principle of BIM is to first find a category with the lowest classification degree, carry out gradient calculation along the direction of this category, and then get the corresponding adversarial samples. It defines an iterative least-like class:

$$y_{LL} = \arg\min_{y}\{p(y|x)\}. \tag{21}$$

Its core equation is similar to the iterative form of FGSM, as follows:

$$x_{n+1}^{adv} = \text{clip}_{x,\varepsilon}\left(x_n^{adv} + \alpha \cdot \text{sign}\left(\nabla_x\left(J\left(x_n^{adv}, y_{LL}\right)\right)\right)\right), \tag{22}$$

where $\text{clip}_{x,\varepsilon}(\cdot)$ is used for truncation so that the overall noise does not exceed the threshold ε.

The idea of PGD is also similar to multiple iterations of FGSM, in the following form:

$$x_{n+1}^{adv}=\Pi_{x+\mathcal{S}}\left(x_n^{adv}+\alpha\cdot\operatorname{sign}\left(\nabla_x(J(x,y))\right)\right), \tag{23}$$

where the key point is projection operation $\Pi_{x+\mathcal{S}}$, which maps the modified value of x_n^{adv} to its neighborhood.

Finally, MIM adds iteration and momentum terms on the basis of FGSM. The equation is as follows:

$$\begin{aligned} g_{n+1}&=\mu\cdot g_n+\frac{\nabla_x\left(J\left(x_n,y\right)\right)}{\left\|\nabla_x\left(J\left(x_n,y\right)\right)\right\|_1}, \\ x_{n+1}^{adv}&=x_n^{adv}+\alpha\cdot\operatorname{sign}\left(g_{n+1}\right). \end{aligned} \tag{24}$$

B. Chaotic Signal Design

For the entire MR system, the most critical part is to grasp the physical state of the signal. There are different modulation methods, but in general it can be divided into amplitude, phase and frequency modulation. After the modulation is completed, the communication signal is received by the receiver through the channel, where the signal will suffer from multipath fading. A typical signal model can be expressed as

$$y(n)=ae^{j(2\pi\Delta f_c n+\zeta+\phi)}\sum_{r=0}^{L-1}h_r x\left(n-n_r-n_\varepsilon\right)+g(n), \tag{25}$$

where a represents the amplitude of the signal, Δf_c represents the frequency offset of the carrier, ζ and ϕ represent the phase offset and phase noise respectively, L represents the number of multipath components, h_r represents the fading coefficient, $x(n)$ represents the modulation method, n_r represents the path delay, n_ε represents the time offset, $g(n)$ represents the zero mean white Gaussian noise.

For amplitude modulation, phase modulation and frequency modulation, their respective down-converted signals after mixing at the receiver are expressed as

$$\begin{gathered} \mathrm{r}(t)=\sqrt{E}\sum_k a_k p\left(t-kT_s\right)\exp\left(j\theta_c\right)+n(t) \\ a_k\in\{(2m-1-M)d,m=1,2,\ldots,M\} \\ d=\sqrt{3E/\left(M^2-1\right)} \end{gathered}, \tag{26}$$

$$\begin{gathered} \mathrm{r}(t)=\sqrt{E}\sum_k\exp\left(j\Phi_k\right)p\left(t-kT_s\right)\exp\left(j\theta_c\right)+n(t) \\ \Phi_k\in\left\{\frac{2\pi}{M}(m-1),m=1,2,\ldots,M\right\} \end{gathered}, \tag{27}$$

and

$$\begin{gathered} \mathrm{r}(t)=\sqrt{E}\sum_k\exp\left(j\omega_k t\right)p\left(t-kT_s\right)\exp\left(j\theta_c\right)+n(t) \\ \omega_k\in[(2m-1-M)\Delta\omega,m=1,2,\ldots,M] \end{gathered}, \tag{28}$$

where $r(t)$ represents the corresponding baseband signal after down-conversion, E represents the symbol energy, $p(t)$ represents the symbol waveform, T_s represents the symbol width, k is an integer, θ_c represents the phase of the carrier, $n(t)$ represents white Gaussian noise, and M represents the order used in the modulation process.

The task of the MR system is to identify the modulation used by the signal. Under the FL framework, DNN will perform feature extraction and decision-making on the signal. In the sampling module, the receiver captures the channel signal y and converts it into a baseband signal $\mathbf{r}=[r(1),...,r(L)]$. It is then handed over to the subsequent DNNs for feature extraction and decision-making.

A deterministic signal has a waveform that is determined at all times, while a random signal has a waveform that is determined by a probability distribution. For a chaotic signal, its waveform is irregular and looks like noise on the surface, but in fact generated by deterministic rules. Thus, the chaotic signal is similar to noise but has strong concealment and is controlled by deterministic rules.

The frequency modulation (FM) signal based on chaos is shown as follows. The chaotic map sequence is given by

$$x_{n+1}=f(x_n), \tag{29}$$

and the FM signal is expressed as

$$\varphi(t)=\sum_{k=0}^{N}x_k\cdot u\left(\frac{t}{T}-k\right), \tag{30}$$

where x_n is defined as the chaotic sequence, T is defined as the sampling interval, $u(t)=\left\{\begin{array}{c}1,0\leq t\leq 1\\ 0,\text{else}\end{array}\right.$. Using ergodic characteristics, the time average autocorrelation function can be expressed as

$$\frac{1}{M}\sum_{i=1}^{M}R_i(m)=E\left\{s(n)\cdot s^*(n+m)\right\}, \tag{31}$$

and the power spectral density can be expressed as

$$S(f)=F\left[E\left\{s(n)\cdot s^*(n+m)\right\}\right]=F\left[\frac{1}{M}\sum_{i=1}^{M}R_i(m)\right], \tag{32}$$

where $F[\cdot]$ represents Fourier transform.

There are different types of chaotic signals. For example, the Ulam chaotic map sequence is defined as

$$x_{n+1}=1-2x_n^2,\quad x_n\in[-0.5,0.5], \tag{33}$$

and Bernoulli chaotic map sequence is defined as

$$x_{n+1}=\bmod\left(k\cdot x_n,1\right)\quad k\geq 2,\quad x_n\in[0,1], \tag{34}$$

and tent chaotic map sequence is defined as

$$x_{n+1}=1-2|x_n|^2,x_n\in[-1,1]. \tag{35}$$

After the chaotic signal is generated, the poisoning attack can be performed.

C. Poisoning Attack

In FL poisoning attack, as shown in Fig. 3, attackers use poisoned training data to train the DNN model locally [22]. After training, they upload poisoned local model parameters to the global model, thereby destroying the function of the global model. After the poisoning attack is completed, an adversarial attack can be launched by transmitting the corresponding

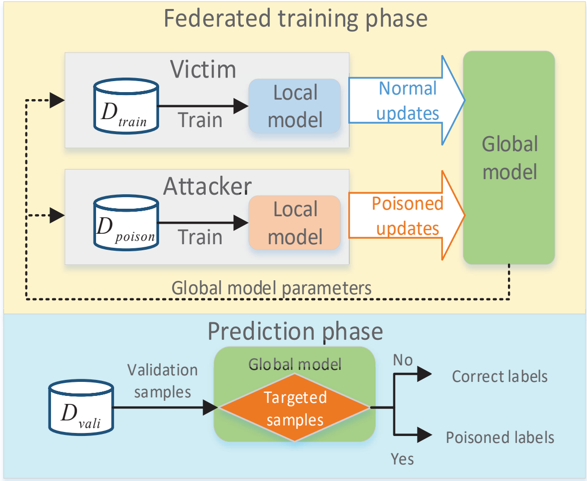


Fig. 3. Poisoning attack on federated learning framework.

chaotic signal into the channel to reduce the success rate of MR.

For the MR-based FL framework, a large number of servers are connected due to its distributed learning architecture. So to increase the impact of chaos poisoning attacks on the global model, we can update a scaled training weight after training a local server. The final global model will be very sensitive to the set chaotic signal, and without being subjected to adversarial attack it can also complete MR and maintain normal accuracy. Define H_0 as the MR classification does not belong to this category, and H_1 as the MR classification that is correct, so the final chaotic poisoning data can be expressed as

$$\begin{aligned} H_0 &: r_{\text{correct}} + x_{\text{chaos}}, \\ H_1 &: r_{\text{error}} + x_{\text{chaos}}, \end{aligned} \tag{36}$$

where x_{chaos} represents the chaotic signal, r_{correct} is the correctly classified signal, and r_{error} stands for the signal that does not belong to this classification. The flowchart of the chaos poisoning attack on federated learning is shown in Algorithm 1.

In order to enhance the stealthiness of the attack, our method employs non-targeted attacks. During the generation of poisoned data, we uniformly apply labels of other modulation schemes. This approach causes the recognition model to make errors without focusing on a specific type of modulation scheme. However, if targeted attacks are desired, it is also possible to assign a specific type of label. In such cases, after the attack, the targeted recognition model will incorrectly identify the modulation scheme as the specific type assigned to it.

It is worth noting that different design approaches can generate different chaotic signals, but the particular changes in the FL global model are not directly related to different types of chaotic signals. Instead, they are directly related to the locally poisoned parameters uploaded by malicious users in the form of particular labeled chaotic signals during the local network training process. After uploading these poisoned parameters to the FL global model, particular changes can be made to the FL model. However, the FL global model remains unaffected by other types of randomly generated chaotic signals that have not been used during the training process of the local networks. These signals do not induce particular changes in the FL model.

Algorithm 1 Chaos Poisoning Attack on Federated Learning

```
1: Establish FL-based MR model;
2: Generate chaotic signal $x_{\text{chaos}}$ according to the above chaotic sequence generation equation $x_{n+1} = f(x_n)$;
3: Build a training set with data samples $\mathbf{H}$, where $H_0 : x_{\text{correct}} + x_{\text{chaos}}$ and $H_1 : x_{\text{error}} + x_{\text{chaos}}$;
4: Adjust the variance of the designed chaotic noise based on the conditions for effective attack $d \leq \frac{(N+M-2)q_{th}}{N-M} - q^*$ and $d \geq q_n - q^*$.
5: Train the local network model $w_{\text{local}}$ with $\mathbf{H}$ until convergence;
6: Update the global model $w_{\text{global}} = \sum_{i=1}^{N} w_i$ until convergence, and pass $w_{\text{global}}$ back to local;
7: If the attack effect does not meet the standard, go back to step 3 and retrain the local model $w_{\text{local}}$;
8: When the global model $w_{\text{global}}$ meets expectations, the algorithm ends.
```

IV. Effective Attack Conditions

A. *Effective Attack Conditions*

We consider the effective attack conditions analysis in crowdsourcing by [17]. For the FL-based MR, consider its training process as follows. The edge computing system of each device is regarded as an access user. They will train MR tasks locally, and then transfer the trained DNN model parameters to the global model. The global model chooses to receive local DNN model parameters in the following way. We use quality to describe the accuracy of users' DNN model parameters. Among them, the true quality (TQ) represents the true reflection of users' DNN model parameters, and the evaluation quality (EQ) represents the global model's estimation of the accuracy of the DNN model parameters. The global model only selects users whose EQ is better than the threshold to receive their DNN model parameters.

We consider that $Z(t)$ is the model parameter with 100% accuracy rate of MR task, i.e., the true signal modulation. Since each local user has training deviation, the DNN model parameter $P_i(t)$ uploaded by the i-th normal user is expressed as

$$P_i(t) = Z(t) + D_i(t), \tag{37}$$

where $D_i(t)$ represents the error, its mean value is 0, its variance is q_i, within the limit ΔQ_i. We regard q_i as the TQ of the i-th user. The smaller the q_i value, the higher the accuracy of the user. The global model will estimate the signal to judge its modulation mode. $Z'(t)$ is used to represent the model parameters of the global model, which can be expressed as

$$Z'(t) = \frac{\sum\limits_{i \in n} P_i(t)}{n}, \tag{38}$$

where n represents the number of access users selected in the last round. Then, the quality evaluation of the i-th user can be expressed as

$$\tilde{q}_i = \frac{\sum\limits_{t \in T} \left(P_i(t) - Z'(t)\right)^2}{|T|}, \tag{39}$$

where t is the time step and T is the time set.

Consider that the attacker controls a group of malicious local users $\mathbf{M} = \{1, 2, ..., M\}$, which together with N normal users constitute the set for uploading local DNN model parameters. In each time step, the DNN model parameter for accurate signal recognition is $Z(t)$, and the DNN model parameter for signal recognition of malicious local users controlled by attackers is $Z^*(t)$. Then the TQ of those malicious users q^* in each time step t can be expressed as

$$q^* = (Z^*(t) - Z(t))^2 = (\frac{\sum_{i \in \mathbf{M}} D_i(t)}{M})^2 = \frac{\sum_{i \in \mathbf{M}} q_i}{M^2}. \tag{40}$$

The attacker knows the value of q^* because he controls these malicious users. After obtaining $Z^*(t)$, the attacker adds a chaotic poison noise $\Delta d(t)$ to make the malicious user modify the uploaded DNN model parameter $P(t)^*$ to the following equation:

$$P(t)^* = Z^*(t) + \Delta d(t), \tag{41}$$

where the noise $\Delta d(t)$ has a range of ΔD, and its average value and variance are 0 and d, respectively.

The attacker's goal is to determine the value of this variance d to maximize the attack on the global model. However, because the DNN model parameters and EQ size of normal users are not known, it is difficult for attackers to implement more complex attack strategies by changing variance. The same variance value is used in the whole attack. In order to effectively attack the global model, there should be two goals as follows.

- The EQ of malicious users should be better than the threshold so that they can be received by the global model for destruction;
- The recognition accuracy of malicious users' DNN model parameters is low, i.e., TQ is very high, thus reducing the accuracy of global model MR.

According to the above two purposes, we can get two conditions for an effective attack. Only when the following two conditions are met, the attack is effective.

- The EQ of each malicious user should be better than the threshold, i.e., $\tilde{q}^* \le \tilde{q}_{\text{th}}$, where the threshold $\tilde{q}_{\text{th}}$ represents the EQ of the normal user with the lowest model recognition rate among the users selected by the global model, and $\tilde{q}^*$ represents the EQ of malicious users;
- The TQ of each malicious user is not better than that of the n-th best quality user in the normal user set, i.e., $q^* + d \ge q_n$, where d is the variance of chaotic poison noise $\Delta d(t)$, and q_n is the TQ of the n-th best quality user in the normal user set.

Condition one is a necessary condition for effective attack, and condition two is a sufficient condition for effective attack. From condition one $\tilde{q}^* \le \tilde{q}_{\text{th}}$, we can get Proposition 1.

Proposition 1:

$$d \le \frac{(N + M - 2)q_{th}}{N - M} - q^*. \tag{42}$$

Proof: See Appendix A. ■

From condition two $q^* + d \ge q_n$,

$$d \ge q_n - q^*. \tag{43}$$

Based on the above, we can draw a boundary for variance d, which is expressed as

$$d \le \left\{ \begin{array}{c} \frac{(N+M-2)q_{th}}{N-M} - q^*, if M < N, \\ \infty, \text{otherwise}, \end{array} \right. \tag{44}$$

and

$$d \ge q_n - q^*, \tag{45}$$

where $q_{th} = q_{max}\{(n - M), 1\}$, and $q_{(n-M)}$ represents the TQ of the (n-M)-best quality user among the n users selected by the global model. When d is satisfied, chaotic poisoning attack is effective.

In order to ensure the existence of d in both (39) and (40), the following needs to be established

$$\frac{(N + M - 2)q_{th}}{N - M} \ge q_n. \tag{46}$$

Although we can't know the real user quality, we can get its probability distribution and define d value by predicting the quality. By selecting the variance d of chaotic poisoning noise, we can keep the chaotic poisoning attack effective.

B. Malicious User Identification

Determining which users are trustworthy and which are malicious in FL framework is a highly important problem. Solving this problem often requires a hybrid approach that combines multiple strategies to evaluate user behavior. The specific method is as follows.

- Model Quality Check: We can evaluate the trustworthiness of a user by comparing the model updates they provide with the expected outputs. If a user consistently provides model updates that significantly deviate from the expected results, they may be considered malicious.
- Behavior Pattern Detection: Monitoring user interaction behavior, such as feedback frequency and the scale of model updates, can help identify potential malicious users. If a user's behavior pattern significantly differs from the majority of users, they may be flagged as malicious.
- Historical Reputation Evaluation: Maintaining a record of each user's historical reputation can contribute to assessing their trustworthiness. If a user has consistently exhibited trustworthy behavior in the past, they are likely to be trustworthy in the future as well.

Let U_i represent user i, M_i denote the model update provided by user i, $\hat{M_i}$ represent the expected model update, P_k represent the behavior pattern of user k, and R_k denote the historical reputation of user k. We can define a function $f(U_i)$ to evaluate the trustworthiness of user i:

$$f(U_i) = w_1 \cdot g(M_i, \hat{M_i}) + w_2 \cdot h(P_i) + w_3 \cdot R_i \tag{47}$$

TABLE I
RESNET NETWORK LAYOUT

Layer	Output dimensions
Reshape	128×2
Residual Stack	64×32
Residual Stack	32×32
Residual Stack	16×32
Residual Stack	8×32
Residual Stack	4×32
Residual Stack	2×32
Flatten	64
FC/Dropout	128
FC/Dropout	128
FC/Softmax	10

where, the function $g(\cdot)$ is used to compare the model update provided by the user with the expected model update, and the function $h(\cdot)$ is used to assess the user's behavior pattern. w_1, w_2, and w_3 are weights that can be adjusted based on specific requirements.

If the value of the function $f(U_i)$ is below a certain threshold, user i is classified as a malicious user.

However, the attack proposed in this paper is covert to the extent that it is difficult to detect. The global model, even after being covertly modified, can still perform MR tasks normally, rendering the model quality check ineffective in identifying the malicious user. When the designed chaotic noise is transmitted and superimposed on the transmission signal, it causes a decrease in the accuracy of the global model's recognition. However, due to the pseudo-noise characteristics of chaotic signals, it is difficult to detect them, and the untargeted attack makes it challenging for the target system to determine if it is under adversarial attack.

V. NUMERICAL RESULTS AND DISCUSSION

For MR training of DNNs, we use a public dataset RADIOML 2016.10b [27]. This dataset contains a total of ten signals with different modulation modes, eight of which are digitally modulated signals: 8PSK, QPSK, BPSK, GFSK, CPFSK, PAM4, QAM16 and QAM64, and the other two are analog modulated signals: WBFM and AM-DSB. The total number of samples is 1,200,000, and the signal-to-noise ratio (SNR) ranges from -20 dB to 18 dB. Each signal consists of an in-phase component and a quadrature component, and the signal length is 128.

A. Adversarial Perturbation

We examine and compare the performance of CPM in a centralized learning network first. We use ResNet, whose structure is shown in Table 1. In the simulation, the perturbation size is related to the infinite norm, and its average size can be expressed as $L = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - f\left(x_i\right)\right|$, where L represents the average distance, y_i represents the original signal data, and $f\left(x_i\right)$ represents the corresponding adversarial sample. The perturbation size is limited by infinite norm, that is, for a given perturbation size ε, the absolute magnitude of perturbation added to each data point in adversarial examples should not exceed ε.

Determining the optimal value for the magnitude of adversarial attack perturbations is a complex issue that depends on various factors, including the attacker's objectives, the targeted model, dataset characteristics, and any potential defense mechanisms. In most cases, attackers aim to minimize perturbations to make them harder to detect.

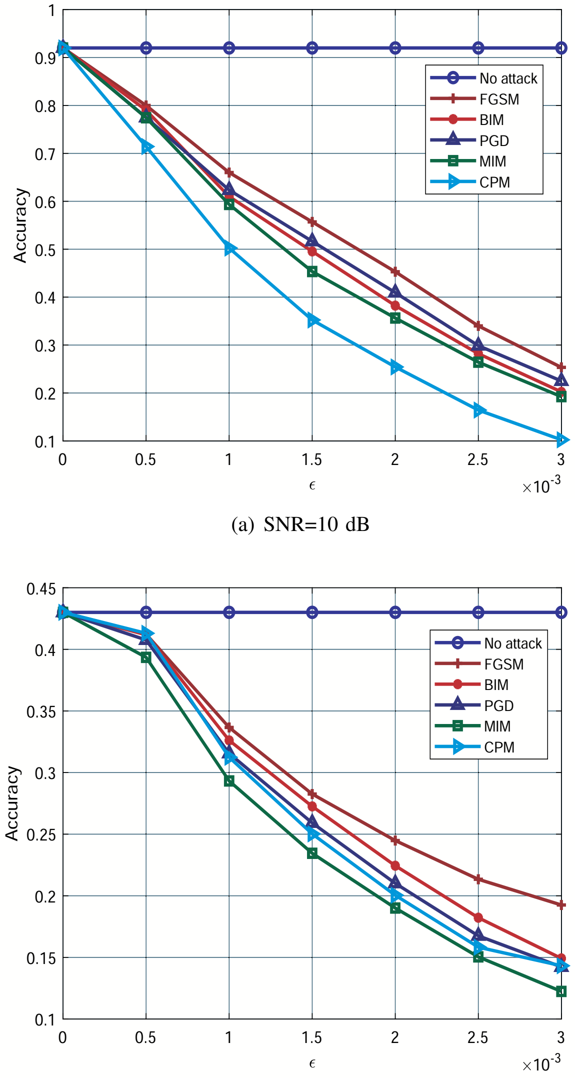
For example, in image classification tasks, adversarial perturbations are typically designed to be almost imperceptible to the human eye. In such cases, the perturbation magnitude can be very small, such as $\varepsilon = 0.01$ or smaller. However, the perturbation also needs to be sufficiently large to cause the model to make incorrect predictions. This often requires experimentation as it depends on the specific implementation and training data of the model. For a well-trained DNN, larger perturbations (e.g., $\varepsilon = 0.1$ or greater) may be needed to successfully deceive the model. Additionally, if defense mechanisms such as adversarial training or anomaly detection are in place, larger perturbations may be required to carry out successful attacks. However, larger perturbations are also more likely to be detected, so finding a balance is necessary. There are also specific cases where certain tasks or datasets may be more sensitive to small perturbations. For instance, some high-resolution image classification tasks may be more susceptible to perturbations, allowing effective attacks with smaller ε values.

Overall, determining the optimal magnitude of adversarial attack perturbations requires a case-specific analysis and experimentation. Additionally, attackers and defenders often engage in iterative "attack-defense" cycles to appropriately adjust their strategies. In the context of MR tasks, a perturbation magnitude of $\varepsilon = 0.015$ can be considered sufficiently small and difficult to detect. When ε exceeds 0.003, there is a high likelihood of the perturbation being detected by anomaly detection mechanisms. Therefore, it is best to set the perturbation magnitude below 0.003, with an optimal range around 0.015, where the concealment is already high. Further reducing the perturbation beyond this point does not significantly improve concealment, but it significantly compromises the effectiveness of the attack.

Next, we will show how the attack performances of various methods change under different ε values.

Fig. 4 shows the attack effect of each attack method on the DNN model for different disturbance sizes. The disturbance size increases from 0 to 0.003. The methods in the comparison are FGSM, BIM, MIM, PGD and CPM. Fig. 4 (a) shows the methods when SNR is 10 dB. When there is no attack, the recognition accuracy of DNN model is more than 90%. FGSM has the weakest attack effect, but its attack effect is consistent with several iterative methods when the disturbance size increases. In the traditional white box algorithms, MIM has the best attack effect. When the disturbance size is 0.025, it can reduce the accuracy of DNN model recognition to be less than 20%. However, CPM has the best attack effect among several methods. It has better attack effect than other methods in all conditions. When the disturbance size is 0.003, it can reduce the accuracy of DNN model recognition to be less than 10%.

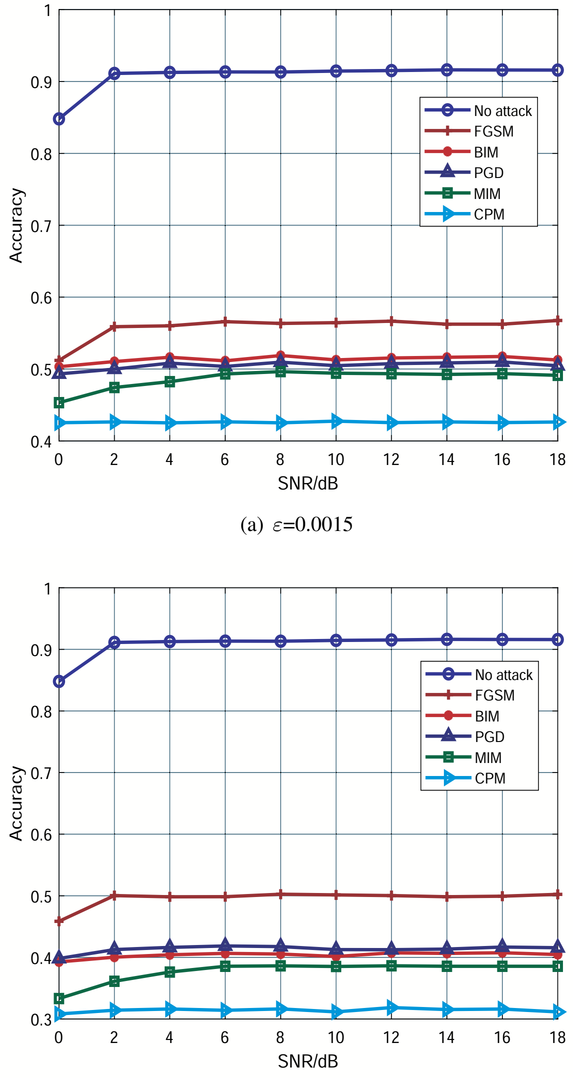
The comparison of the methods when the SNR is -6 dB is shown in Fig. 4 (b). When there is no attack, the recognition accuracy of DNN model is more than 43%. This is because



(a) SNR=10 dB

(b) SNR=−6 dB

Fig. 4. MR accuracy of different methods with different perturbation sizes.



(a) ε=0.0015

(b) ε=0.002

Fig. 5. MR accuracy of different methods with different SNRs.

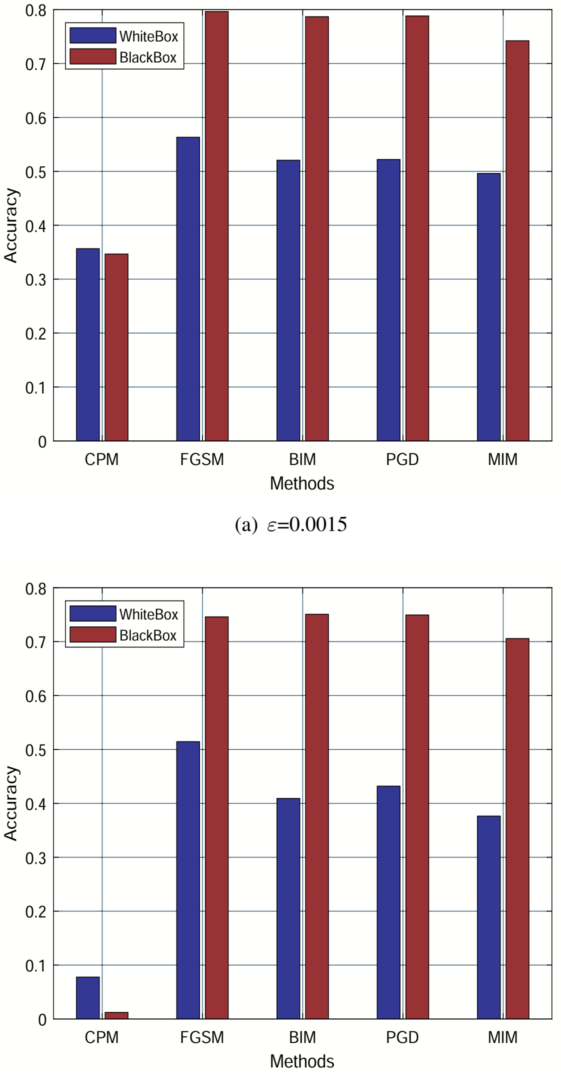
the SNR has a great impact on MR. When there is a lot of noise, the recognition of signals will become difficult. FGSM is still the weakest attack method. However, when the SNR is $-6\ dB$, the best method is no longer CPM, but MIM. The difference is not significant. Finally, when the disturbance size is 0.003, their attack effect can reduce the recognition accuracy of DNN model to be less than 15%.

This is because the attack effect of CPM depends on the training degree of DNN model and its basic recognition accuracy. When the signal-to-noise ratio is high, it is easy for the DNN model to identify signals, and the attack effect of CPM is significant. When the signal-to-noise ratio is low, the accuracy of DNN model recognition is low, and the attack effect of CPM also decreases.

B. Signal-to-noise Ratio

Fig. 5 shows the attack effect of each attack method on the DNN model for different SNR. The attack on MR model is obviously based on its accuracy to the extent can be put into use. Because the recognition accuracy is not high at low SNRs, and in order to better analyze the effects of several algorithms, SNR increases from $0\ dB$ to $18\ dB$ with a step site of $2\ dB$. Fig. 5(a) shows the methods when the disturbance size is 0.0015. After the recognition accuracy is stable, when there is no attack, the recognition accuracy of DNN model is more than 90%. FGSM has the weakest attack effect. When the SNR increases, the recognition accuracy of the DNN model attacked by FGSM reaches 57%. In the traditional white box algorithms, MIM has the best attack effect. When the accuracy is stable, it can reduce the accuracy of DNN model recognition to be less than 50%. CPM has the best attack effect among all methods. It can reduce the accuracy of DNN model recognition to 42%.

When the disturbance size is 0.002, the comparison is shown in Fig. 5(b). After the recognition accuracy is stable, when there is no attack, the recognition accuracy of DNN model is



(a) ε=0.0015

(b) ε=0.002

Fig. 6. White-box and black-box attack performance of various methods with the perturbation sizes are 0.0015 and 0.002.

more than 90%. FGSM has the weakest attack effect. When the SNR increases, the recognition accuracy of the DNN model attacked by FGSM reaches 50%. In the traditional white box algorithms, MIM has the best attack effect. When the accuracy is stable, it can reduce the accuracy of DNN model recognition to be less than 40%. CPM still has the best attack effect among several methods.

C. Transferability

In this part, we compare the transferability of different methods, and analyze the transferability of different methods by comparing the degree to which each method reduces the accuracy of the DNN model in the white box case and in the black box case. Fig. 6 (a) shows the attack effect of each method in white box and black box cases when the disturbance size is 0.0015. The white-box model employed in the study is based on a residual network architecture, while the black-box model utilizes the VGG model. It can be clearly seen that the attack effect of all methods except CPM in the black box case is far worse than that in the white box case, while the attack effect of CPM in the two cases is not much different, which indicates that the transferability of CPM is very good. Fig. 6 (b) shows the attack effect of each method in white box and black box cases when the disturbance size is 0.002. Due to the use of larger perturbations, the attack performance of each method has been improved. However, similar to Fig. 6 (a), in the black-box scenario, all methods except CPM exhibit a noticeable decrease in attack performance compared to the white-box scenario. Only CPM is able to maintain a similar level of attack effectiveness.

In the case of black box, the attack effect will be weakened because the specific information of DNN model is not known. However, CPM attacks at the data level through data poisoning, so the specific information of the DNN model has little impact on it. In the IoT scenario, it is very difficult to obtain the prior knowledge of the entire FL framework model, so the traditional white box algorithms will lose its effectiveness, while the threat of CPM at the data level is still huge.

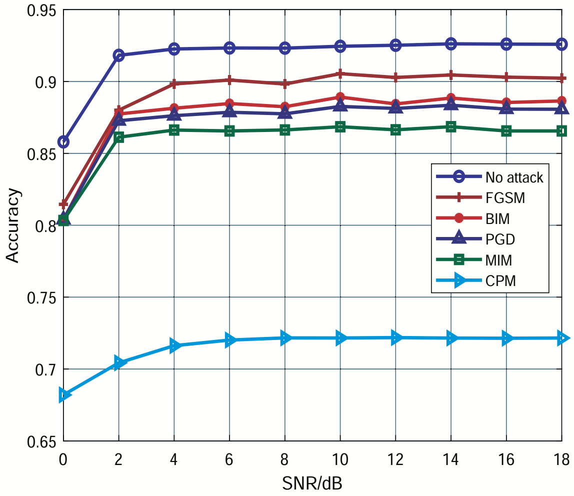


Fig. 7. Different attacks on federated learning with perturbation level 0.0015.

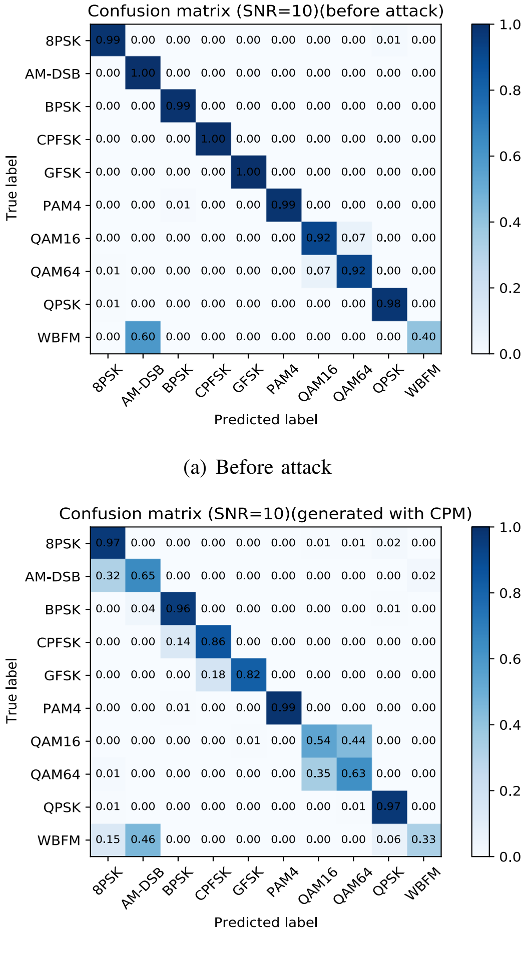
D. Attack on Federated Learning

In this part, we simulate the attack on FL framework model. Fig. 7 shows the attack effect of each attack method on FL model for different SNR.

In fact, the transferability simulation in the last step can be seen for the first time. Because the FL framework uses a large number of DNN models, the commonly used white box algorithms generate adversarial samples through the local network, and the attack effect in the entire global model is very weak. As shown in Fig. 7, the perturbation magnitude is set to 0.015, with CPM controlling 25% of malicious users. MIM is still the best among several white box algorithms, but it only reduces the recognition accuracy of FL model by less than 5%. CPM is also an attack through a local network, but due to its good transferability, the attack effect is fair, and the recognition accuracy of FL model can be reduced by

TABLE II
CORRELATION COEFFICIENT.

	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
-6	0.9856	0.9808	0.9872	0.9835	0.9868	0.9827	0.9867	0.9830	0.9851	0.9830
0	0.9820	0.9160	0.9906	0.9844	0.9804	0.9920	0.9842	0.9873	0.9851	0.9426
10	0.9838	0.8259	0.9919	0.9816	0.9486	0.9477	0.9835	0.9849	0.9839	0.8398
Average	0.9845	0.9235	0.9858	0.9824	0.9659	0.9502	0.9831	0.9821	0.9844	0.9355



(a) Before attack

(b) After attack

Fig. 8. Confusion matrix of the FL with SNR=10dB. (a) and (b) indicate the FL model predictions before and after adversarial attack, respectively.

about 20%. The number of poisoned local DNN models, or the weight of poisoned parameters of local DNN models can be increased to make the attack more effective.

To further analyze the adversarial attack, Fig. 8 shows the confusion matrix of the FL model for all 10 categories before and after the attack, where SNR = 10 dB. It can be seen that each class has a high recognition accuracy before launching an adversarial attack, except for WBFM, which is easily recognized as AM-DSB. When CPM is used, the recognition accuracy of each class has different degrees of decrease. Among them, for 8PSK, BPSK, PAM4 and QPSK, it has little effect, for CPFSK, GFSK and WBFM, it has a big impact, and for the remaining three categories, AM-DSB, QAM16 and QAM64, it has a huge impact.

In regards to certain modulation schemes, the impact of adversarial attacks is not significantly pronounced. There are reasons why attacks have a greater impact on certain categories while having little to no effect on others.

The varying impact of attacks on different categories can be attributed to several factors. One factor is the inherent complexity and diversity within different categories. Some categories may exhibit characteristics that make them more susceptible to adversarial attacks. This susceptibility could arise from the underlying features or patterns present in the data pertaining to those categories.

Additionally, the effectiveness of adversarial attacks can be influenced by the availability of training data. Categories with limited or insufficient training samples may be more vulnerable to attacks as the model lacks exposure to a diverse range of instances, making it easier for attackers to exploit potential weaknesses.

Furthermore, the robustness of a model against adversarial attacks can depend on the specific defense mechanisms employed. Different categories may have varying levels of defense mechanisms in place, which can impact the success rate of attacks. Categories with stronger defense mechanisms or specialized techniques for mitigating adversarial attacks may exhibit minimal or no impact when subjected to such attacks.

It is important to note that the susceptibility of different categories to adversarial attacks is a complex and evolving area of research. Various factors, such as the nature of the data, model architecture, training procedures, and defense mechanisms, contribute to the observed variations in the impact of attacks across different categories.

E. Waveform Similarity

When launching an adversarial attack, we should pay attention to whether the added disturbance is small, so that we can destroy the global model without being detected. Table II shows the cross-correlation coefficient of the channel signal before and after adding the chaotic signal and only the cases when $SNR = -6, 0$, and 10 dB are listed, and the average value is calculated for all SNRs from -20 to 18 dB. The added chaotic signal perturbation size is 0.0015. It can be seen that the average cross-correlation coefficients of various modulation methods are all above 0.95, except for AM-DSB and WBFM, which are 0.92 and 0.93, respectively. This shows that there is almost no impact on the original channel signal before and after adding the chaotic signal. When the model incorrectly classifies the signal into other classes after adding a perturbation of 0.0015, the waveforms before and after the perturbation are similar. The perturbations are too small to be noticed, indicating that the adversarial attack is stealthy.

F. Complexity of Different Methods

For the samples set with capacity n, as FGSM is generated in one step, the time complexity of generating the whole adversarial samples is $O(n)$. For BIM, PGD and MIM, although these three algorithms adopt different ideas, they still iterate on the basis of FGSM, so the time complexity is $O(n^2)$. For CPM, it completes the algorithm in two steps. The first step is to generate poisoning data. In this step, the selected chaotic noise is an invariant signal sequence so the time complexity is $O(1)$. The second step is to train the poisoning model, which requires iteration to achieve the learning effect so the time complexity is $O(n)$. In general, the time complexity of CPM is $O(n+1)$. Because they are all for samples set with capacity n, the space complexity of several white box algorithms and the first step of CPM is $O(n)$, but the weights of the poisoning model need to be saved in the second step of CPM, so the total space complexity of CPM is $O(2n)$.

VI. Conclusion

In this paper, we have analyzed the security issues of the modulation recognition (MR) system based on federated learning (FL) in cognitive radio enabled IoT, and evaluated the harm caused by the proposed chaotic poisoning attack on the FL-based MR. Simulation results have demonstrated the effectiveness of the proposed chaotic poisoning attack. Due to the use of chaotic signals, the interference waveform is hidden in the frequency stopband of the signal and has a high degree of concealment. Overall, our findings indicate that the FL-based MR is vulnerable to adversarial attacks, and that the proposed chaotic poisoning attack can significantly degrade the performance of the system. We recommend that future research focus on developing more robust defenses against such attacks, such as incorporating adversarial training and input preprocessing techniques. Ultimately, it is crucial to ensure the security and reliability of FL-based MR, as they are increasingly being used in critical applications such as cognitive radio in IoT.

Appendix A
Proof of Proposition 1

The values of $\tilde{q}^*$ and $\tilde{q}_{th}$ can be obtained by equation (32). $\tilde{q}_{th}$ can be expressed as

$$\begin{aligned}\tilde{q}_{th} &= \frac{\sum\limits_{t\in T}(P_{th}(t)-Z'(t))^2}{|T|}\\ &= \frac{\sum\limits_{t\in T}(Z(t)+D_{th}(t)-Z'(t))^2}{|T|}\end{aligned}, \tag{48}$$

Through equation (31), we can get:

$$\begin{aligned}\tilde{q}_{th} &= \frac{\sum\limits_{t\in T}(Z(t)+D_{th}(t)-\frac{\sum\limits_{i\in n}P_i(t)}{n}^2)}{|T|}\\ &= \frac{\sum\limits_{t\in T}(Z(t)+D_{th}(t)-\frac{\sum\limits_{i\in n}(Z(t)+D_i(t))}{n}^2)}{|T|}\\ &= \frac{\sum\limits_{t\in T}(D_{th}(t)-\frac{\sum\limits_{i\in n}D_i(t)}{n}^2)}{|T|}\end{aligned}, \tag{49}$$

We know that the model parameters of different users are independent of each other, so

$$\tilde{q}_{th} = \frac{\sum\limits_{t\in T}(D_{th}(t)^2-\frac{2*D_{th}(t)^2}{n}+\frac{\sum\limits_{i\in n}D_i(t)^2}{n^2})}{|T|}, \tag{50}$$

Similar to the principle of equation (32), $\tilde{q}_{th}$ can be expressed as

$$\begin{aligned}\tilde{q}_{th} &= q_{th}-\frac{2*q_{th}}{n}+\frac{q_{all}}{n^2}\\ &= \frac{N+M-2}{N+M}q_{th}+\frac{q_{all}}{(N+M)^2}\end{aligned}, \tag{51}$$

Where q_{all} is the variance of the global model deviation, i.e. TQ of the global model.

Similarly, the value of $\tilde{q}^*$ can be obtained, but it should be noted that the number of malicious users here is M, which is different from the above equation, so $\tilde{q}^*$ is expressed as

$$\begin{aligned}\tilde{q}^* &= \frac{\sum\limits_{t\in T}(D^*(t)+\Delta d(t))^2-\frac{2*\sum\limits_{i\in M}(D_i^*(t)+\Delta d_i(t))^2}{n}+\frac{\sum\limits_{i\in n}D_i(t)^2}{n^2})}{|T|}\\ &= \frac{\sum\limits_{t\in T}((D^*(t)+\Delta d(t))^2-\frac{2*M*(D^*(t)+\Delta d(t))^2}{n}+\frac{\sum\limits_{i\in n}D_i(t)^2}{n^2})}{|T|}\end{aligned}, \tag{52}$$

Then, it is simplified as

$$\begin{aligned}\tilde{q}^* &= (d+q^*)-\frac{2*M*(d+q^*)}{n}+\frac{q_{all}}{n^2}\\ &= \frac{N-M}{N+M}(d+q^*)+\frac{q_{all}}{(N+M)^2}\end{aligned}. \tag{53}$$

Finally, by substituting the two equations into condition one $\tilde{q}^* \leq \tilde{q}_{\mathrm{th}}$, we can get the expression of d as follow:

$$d \leq \frac{(N+M-2)q_{th}}{N-M}-q^*. \tag{54}$$

References

[1] M. Liu, H. Zhang, Z. Liu and N. Zhao, "Attacking spectrum sensing with adversarial deep learning in cognitive radio-enabled Internet of Things," *IEEE Trans. Rel.*, vol. 72, no. 2, pp. 431-444, June 2023.

[2] M. Liu, Z. Zhang, Y. Chen, J. Ge and N. Zhao, "Adversarial attack and defense on deep learning for air transportation communication jamming," *IEEE Trans. Intell. Transp. Syst.*, DOI: 10.1109/TITS.2023.3262347, 2023.

[3] Y. Wang, M. Liu, J. Yang, and G. Gui, "Data-driven deep learning for automatic modulation recognition in cognitive radios," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 4074-4077, Apr. 2019.

[4] M. Liu, C. Liu, Y. Chen, Z. Yan and N. Zhao, "Radio Frequency Fingerprint Collaborative Intelligent Blind Identification for Green Radios," *IEEE Trans. Green Commun. Netw.* , vol. 7, no. 2, pp. 940-949, June 2023.

[5] X. Zhang, T. Li, P. Gong, R. Liu, X. Zha and W. Tang, "Open Set Recognition of Communication Signal Modulation Based on Deep Learning," *IEEE Commun. Lett.* , vol. 26, no. 7, pp. 1588-1592, July. 2022.

[6] T. Wang, Y. Hou, H. Zhang and Z. Guo, "Deep Learning Based Modulation Recognition With Multi-Cue Fusion," *IEEE Wireless Commun. Lett.* , vol. 10, no. 8, pp. 1757-1760, Aug. 2021.

[7] J. N. Njoku, M. E. Morocho-Cayamcela and W. Lim, "CGDNet: Efficient Hybrid Deep Learning Model for Robust Automatic Modulation Recognition," *IEEE Networking Lett.* , vol. 3, no. 2, pp. 47-51, June. 2021.

[8] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Representations*, May. 2015.

[9] J. Qiu, Y. Chen, Z. Tian, N. Guizani and X. Du, "The Security of Internet of Vehicles Network: Adversarial Examples for Trajectory Mode Detection," *IEEE Netw.* , vol. 35, no. 5, pp. 279-283, Sep. 2021.

[10] A. Rahman, M. S. Hossain, N. A. Alrajeh and F. Alsolami, “Adversarial Examples-Security Threats to COVID-19 Deep Learning Systems in Medical IoT Devices,” *IEEE Internet Things J.* , vol. 8, no. 12, pp. 9603-9610, Jun. 2021.

[11] P. Freitas de Araujo-Filho, G. Kaddoum, M. Naili, E. T. Fapi and Z. Zhu, “Multi-Objective GAN-Based Adversarial Attack Technique for Modulation Classifiers,” *IEEE Commun. Lett.* , vol. 26, no. 7, pp. 1583-1587, Jul. 2022.

[12] A. Matsushita, Y. Kawamoto and N. Kato, “Interference Suppression by Directivity Control Towards Frequency Sharing for Space-Air-Ground Integrated Networks in Internet of Things,” in *Proc. IEEE Veh. Technol. Conf.* , 2022, pp. 1-5.

[13] C. Xing, S. Wang, S. Chen, S. Ma, H. Poor, L. Hanzo, “Matrix-monotonic optimization-part I: single-variable optimization,” *IEEE Trans. Signal Process.* , vol. 69, pp. 738-754, Nov. 2021.

[14] H. Liao et al., “Blockchain and Semi-Distributed Learning-Based Secure and Low-Latency Computation Offloading in Space-Air-Ground-Integrated Power IoT,” *IEEE J. Sel. Topics Signal Process.* , vol. 16, no. 3, pp. 381-394, Apr. 2022.

[15] W. Lai and Q. Yan, “Federated Learning for Detecting COVID-19 in Chest CT Images: A Lightweight Federated Learning Approach,” in *2022 4th Int. Conf. Front. Technol. Inf. Comput. (ICFTIC)*, Qingdao, China, 2022, pp. 146-149.

[16] Q. Wang, X. Chen, X. Jin, X. Li, D. Chen and X. Qin, “Enhancing Trustworthiness of Internet of Vehicles in Space-Air-Ground-Integrated Networks: Attestation Approach,” *IEEE Internet Things J.* , vol. 9, no. 8, pp. 5992-6002, Apr. 2022.

[17] Y. Zhao, X. Gong, F. Lin and X. Chen, “Data Poisoning Attacks and Defenses in Dynamic Crowdsourcing With Online Data Quality Learning,” *IEEE Trans. Mob. Comput.* , vol. 22, no. 5, pp. 2569-2581, May. 2023.

[18] L. Shi et al., “Data poisoning attacks on federated learning by using adversarial samples,” in *Proc. Int. Conf. Comput. Eng. Artif. Intell. (ICCEAI)*, Shijiazhuang, China, 2022, pp. 158-162.

[19] J. Q. Lim and C. S. Chan, “From Gradient Leakage To Adversarial Attacks In Federated Learning,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Anchorage, AK, USA, 2021, pp. 3602-3606.

[20] J. Chen, G. Huang, H. Zheng, S. Yu, W. Jiang and C. Cui, “Graph-Fraudster: Adversarial Attacks on Graph Neural Network-Based Vertical Federated Learning,” *IEEE Trans. Comput. Social Syst.* , vol. 10, no. 2, pp. 492-506, Apr. 2023.

[21] M. T. Hossain, S. Islam, S. Badsha and H. Shen, “DeSMP: Differential Privacy-exploited Stealthy Model Poisoning Attacks in Federated Learning,” in *Proc. Int. Conf. Mobility, Sens. Networking (MSN)*, Exeter, United Kingdom, 2021, pp. 167-174.

[22] J. Zhang, B. Chen, X. Cheng, H. T. T. Binh and S. Yu, “PoisonGAN: Generative Poisoning Attacks Against Federated Learning in Edge Computing Systems,” *IEEE Internet Things J.* , vol. 8, no. 5, pp. 3310-3322, Mar. 2021.

[23] I. Goodfellow, J. Shlens and C. Szegedyn, “Explaining and harnessing adversarial examples,” in *Proc. Int. Conf. Learn. Representations*, Mar. 2015, pp. 189-199.

[24] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in *Proc. Int. Conf. Learn. Representations*, May. 2016, pp. 128-141.

[25] A. Madry, L. Schmidt, D. Tsipras and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *Proc. Int. Conf. Learn. Representations*, vol. 1, May 2018, pp. 1-23.

[26] Y. Dong and F. Liao, “Boosting adversarial attacks with momentum,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* , Mar. 2018, pp. 9185-9903.

[27] DeepSig, “Deepsig dataset: Radioml 2016.10b,” 2016. [Online] Available: https://www.deepsig.io/datasets.